Advances in Data Analysis and Classification
https://doi.org/10.1007/s11634-025-00642-9

REGULAR ARTICLE

Improving the computational performance of TCLUST through ensemble initialization

Pedro C. Álvarez-Esteban[1] · **Luis A. García-Escudero**[1] · **Agustín Mayo-Iscar**[1] · **Javier Crespo-Guerrero**[2]

Received: 6 February 2024 / Revised: 28 February 2025 / Accepted: 24 April 2025

Abstract
Outliers are known to be detrimental to widely used clustering techniques. Robust clustering alternatives have been introduced to better resist outlying observations. Among these, robust clustering methods based on trimming have proven effective by allowing the removal of a fraction of observations where outliers are likely to be found, with TCLUST being one of the most popular for handling elliptically contoured clusters. The algorithm for applying TCLUST can be seen as an extension of the concentration steps used in the fast-MCD algorithm for computing the Minimum Covariance Determinant. However, obtaining good initializations for these concentration steps in TCLUST is more complex than in MCD. This initialization task is particularly challenging unless both the number of clusters and the dimensionality are small. To address this, a new ensemble initialization procedure for TCLUST will be presented, which takes advantage of partially correct information from all iterated random initializations rather than focusing solely on the best individual one found. Initial experiments suggest that this methodology could improve the computational performance of the standard TCLUST algorithm.

Keywords Cluster analysis · Robustness · Trimming · Model-based clustering

Mathematics Subject Classification 62H30 · 62H11 · 62G35

✉ Pedro C. Álvarez-Esteban
pedrocesar.alvarez@uva.es

Luis A. García-Escudero
lagarcia@uva.es

Agustín Mayo-Iscar
agustin.mayo.iscar@uva.es

[1] Departamento de Estadística e Investigación Operativa and IMUVA, Universidad de Valladolid, Paseo de Belén s/n, 47011 Valladolid, Spain

[2] Valladolid, Spain

Published online: 22 May 2025

1 Introduction

Modifying a small proportion of observations can severely affect the result of many clustering methods. Consequently, the clustering results may be negatively affected by including a certain fraction of anomalous or incorrect measurements. Accordingly, several robust clustering approaches have been proposed trying to make the clustering results less affected by outliers (García-Escudero et al. 2010; Banerjee and Dave 2012; Ritter 2014; García-Escudero et al. 2016; García-Escudero and Mayo-Iscar 2024). In this work, we focus exclusively on a single trimming-based approach, specifically the TCLUST method introduced in García-Escudero et al. (2008). TCLUST employs an "impartial" trimming approach, where the term impartial means that the dataset itself determines which observations to trim.

Robust clustering methods based on impartial trimming provide a partition of the n observations into G clusters but allowing a fraction $\alpha \in [0, 1)$ of observations to be left unassigned or, in other words, trimmed. The idea was firstly introduced in Cuesta-Albertos et al. (1997), through the trimmed k-means as a trimmed extension of the classical k-means. However, trimmed k-means inherits from classical k-means its preference for spherical and equally scattered clusters. TCLUST is an extension of trimmed k-means which uses a trimmed classification likelihood approach where, apart from discarding a fixed fraction α of observations, G multivariate normally distributed components are assumed. Consequently, given a sample $\{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, TCLUST seeks location vectors $m_1, \ldots, m_G$ in $\mathbb{R}^p$, symmetric positive definite $p \times p$ matrices $S_1, \ldots, S_G$, weights $p_1, \ldots, p_G$ with $\sum_{g=1}^{G} p_g = 1$, and a partition $\{R_0, R_1, \ldots, R_G\}$ with $\#R_0 = [n\alpha]$, aiming to maximize

$$\sum_{g=1}^{G} \sum_{i \in R_g} \log(p_g \phi(x_i; m_g, S_g)), \tag{1}$$

where $\phi(\cdot; \mu, \Sigma)$ is the density function of a p-variate normal with location vector μ and scatter matrix Σ. Another important aspect of TCLUST is the restriction on the scatter matrices $S_1, \ldots, S_G$ to ensure that the constrained maximization of (1) is a mathematically well-defined problem and to avoid detecting uninteresting or spurious components with scatter matrices whose determinants $|S_g|$ are close to 0. These restrictions are expressed in terms of constraining the relative size of the eigenvalues of the scatter matrices as follows: if $\{\lambda_j(S_g)\}_{j=1}^{p}$ denote the p eigenvalues of the scatter matrix S_g, then it is enforced that

$$\frac{\max_{g=1,\ldots,G; j=1,\ldots,p} \lambda_j(\Sigma_g)}{\min_{g=1,\ldots,G; j=1,\ldots,p} \lambda_j(\Sigma_g)} \leq c, \tag{2}$$

for a fixed tuning constant $c \geq 1$. These spurious components can be seen as another source of lack of robustness (Ingrassia and Rocci 2007; García-Escudero et al. 2018). Some appealing robustness properties of TCLUST have been demonstrated in Ruwet et al. (2012) and Ruwet et al. (2013). The target function in (1) can be modified to deal

with trimmed mixture likelihoods (Neykov et al. 2007; García-Escudero et al. 2014). Additionally, more sophisticated types of constraints could also be imposed, such as those introduced in García-Escudero et al. (2022).

From a computational perspective, the TCLUST problem involves solving a complex combinatorial problem because, in principle, all possible partitions of the n observations into $G+1$ subsets are considered, with one of these subsets containing the $[n\alpha]$ observations to be trimmed. Like other robust (impartial) trimming-based methods, the algorithm for applying TCLUST relies on so-called "concentration steps", which must be properly initialized to effectively explore the parameter space. These initializations are commonly obtained from small subsamples drawn from the dataset. In this work, we propose a new procedure for generating useful initializations for TCLUST based on a novel ensemble initialization approach. This procedure takes advantage of partially correct information from multiple iterated random initializations rather than focusing solely on the best individual iteration. To the best of our knowledge, these ensemble initialization concepts have never been applied within this framework of robust methods relying on random initializations, nor specifically in the context of TCLUST.

The outline of the manuscript is as follows. Section 2 briefly reviews the basics of the algorithm used for the implementation of TCLUST and describes the challenges in achieving proper initialization within that algorithm, particularly as the dimension p or the number of clusters G increases. Section 3 introduces the new ensemble initialization proposal to assist with this task. An illustrative example of the proposal and a simulation study are presented in Sect. 4. The possibility of combining subsampling and ensemble initialization methods will be discussed in Sect. 5. The effectiveness of the ensemble initialization strategy in achieving higher values in the constrained maximization of (1) is demonstrated with a real dataset in Sect. 6. Finally, Sect. 7 concludes the manuscript and briefly summarizes some open research directions.

2 Computational issues in TCLUST

The impartial trimming principles mentioned in Sect. 1 are also found in widely used robust procedures such as LTS (Least Trimmed Squares) and MCD (Minimum Covariance Determinant) (Rousseeuw and Leroy 1987), where "concentration steps" are central to the algorithms used for their implementation (Rousseeuw and van Driessen 1999, 2000). These concentration steps are somewhat analogous to the iterative steps in Lloyd's classical k-means clustering algorithm (Lloyd 1982), which aims to identify regions of the sample space where observations are "concentrated" to locate the optimal k-means centers. Similarly, it could be said that the classification EM algorithm (Celeux and Govaert 1992), which includes the k-means method as a special case, also relies on concentration steps.

Therefore, it makes sense to combine impartial trimming techniques with classification EM-type algorithms at a computational level. Building on this idea, an algorithm for implementing TCLUST was proposed in García-Escudero et al. (2008) and later improved in Fritz et al. (2013) through a more efficient application of the eigenvalue ratio constraints.

The basic algorithm for TCLUST can be described as follows:

1. Consider `nstart` random initializations, where each initialization is obtained by selecting $G \times (p+1)$ random observations from $x_1, \ldots, x_n$ to define G initial location vectors $m_1^0, \ldots, m_G^0$ and G initial scatter matrices $S_1^0, \ldots, S_G^0$ (details provided later). Random weights $p_1^0, \ldots, p_G^0$ are chosen, summing to 1 (these weights can initially be set as $p_1^0 = \ldots = p_G^0 = 1/G$). These scatter matrices $S_1^0, \ldots, S_G^0$ do not necessarily satisfy the eigenvalue ratio constraints, but these constraints can be imposed in later steps.
   1.1 From the location vectors $\{m_g^{l-1}\}_{g=1}^G$, the scatter matrices $\{S_g^{l-1}\}_{g=1}^G$, and the weights $\{p_g^{l-1}\}_{g=1}^G$ from the previous step, compute
   $$d_i = \max_{g=1,\ldots,G} p_g^{l-1} \phi(x_i; m_g^{l-1}, S_g^{l-1}),$$
   and sort them as $d_{(1)} \leq d_{(2)} \leq \ldots \leq d_{(n)}$ to define $H = \{i : d_i \geq d_{([n\alpha]+1)}\}$. This subset H of indices is then partitioned into $H = H_1 \cup \ldots \cup H_G$, where
   $$H_g = \{i \in H \text{ such that } p_g^{l-1} \phi(x_i; m_g^{l-1}, S_g^{l-1}) = \max_{\underline{g}=1,\ldots,G} p_{\underline{g}}^{l-1} \phi(x_i; m_{\underline{g}}^{l-1}, S_{\underline{g}}^{l-1})\}.$$
   1.2 Parameters are updated with $p_g^l = n_g/(n - [n\alpha])]$ to update weights, where n_g is the number of observations with indices in H_g, with the sample mean of the observations with indices in H_g to update the locations in m_g^l and with the sample covariance of the observations with indices in H_g to update the scatter matrices in S_g^l. The required constraints on the new S_g^l matrices are also imposed in this step using the "eigenvalue truncation operator" introduced in Fritz et al. (2013).
   1.3 Step 1.1 and 1.2 are alternated for `niter` times ($l = 1, \ldots,$ `niter`) or until a stopping criterion is met (for instance, by monitoring changes in the target function (1)).
2. Return as output from the algorithm the weights, location, and scatter matrices corresponding to the maximum value found for the target function (1) among all the iterated `nstart` initializations.

The algorithm monotonically increases the value of the target function (1) at each iteration while maintaining the eigenvalue-ratio constraint to identify local constrained maxima. However, achieving the global constrained maximum requires multiple random initializations due to the presence of numerous local extrema (maxima) in the target function, similar to simpler methods such as MCD or k-means. Therefore, the key components of the TCLUST algorithm are the random initializations and the iterative concentration steps. We use the notation `nstart` to denote the number of random initializations and `niter` to indicate the maximum number of iterative concentration steps considered for each random initialization. This algorithm has led to

the `tclust` package in R (Fritz et al. 2012) and the implementation of TCLUST in the FSDA Matlab toolbox (Riani et al. 2012).

As mentioned in Step 1 of this algorithm, the random initializations are obtained by randomly selecting $G \times (p+1)$ observations from the available dataset. In robust statistical literature, a subset containing the minimum number of observations required to initialize parameters is often referred to as an "elemental set". Specifically, $p+1$ observations in general position are needed to define a scatter matrix, so sets of size $p+1$ were used as elemental sets in the fast-MCD algorithm described in Rousseeuw and van Driessen (1999). The mean of these $p+1$ observations also serves as the initialization for the location vector. Since G location vectors and G scatter matrices are needed in TCLUST, at least $G \times (p+1)$ random observations must be considered to initialize TCLUST's concentration steps.

The TCLUST basic algorithm works quite well for problems with lower dimensions and when searching for a relatively small number of clusters G. For instance, in our experience, using `nstart=500` and `niter=20` is generally effective for most examples we have tested with dimensions p less than or equal to 5 or 6, and with G less than 3 or 4. Unfortunately, its performance clearly deteriorates as either p or G increases, necessitating larger values of `nstart` and `niter`. Note that the TCLUST algorithm reduces to the fast-MCD algorithm when $G = 1$ (with a very large value of c chosen in (2)), and it is known that the number `nstart` of random initializations (each based on elemental sets with $p+1$ observations) required for a successful application of the fast-MCD algorithm increases with dimension p. In fact, Hubert et al. (2012) demonstrated that `nstart=500` is insufficient at high levels of contamination when p exceeds 10, regardless of n. It is therefore reasonable to expect even greater challenges in achieving good initializations for TCLUST when $G > 1$.

It is reasonable to assume that the TCLUST algorithm is more likely to find the constrained global maximum of (1) by selecting $G \times (p+1)$ observations without outliers. Furthermore, it would be clearly beneficial if these $G \times (p+1)$ observations were ordered such that the first $p+1$ observations belong to one of the underlying cluster components, the next $p+1$ observations to another cluster component, and so on. This specific ordering of the $G \times (p+1)$ randomly selected observations becomes increasingly unlikely as p or G grows.

Given the challenge of obtaining good initializations as G or p increases, it may not be practical to iterate through all `nstart` initializations. A more effective strategy could be to fully iterate only the most "promising" initializations while exploring a larger number of initializations `nstart`. Therefore, it is advisable to use a higher `nstart` but with a minimal number of concentration steps for each, denoted as `niter1` (e.g., `niter1` could be set to 3 or 5). Among these minimally iterated initializations, we should focus on a small proportion that shows the best preliminary results based on their associated target function values. These promising initializations are then fully iterated until convergence or until reaching a maximum number of concentration steps, `niter2`. This approach was originally considered in the fast-MCD algorithm Rousseeuw and van Driessen (1999) and has been incorporated into the **tclust** package since version 2.0-1. Similarly, the "small EM" strategy described in Biernacki et al. (2003) could be adapted for this purpose. Another valuable strategy could be to complement random initializations with carefully selected

initializations derived from simpler yet robust clustering methods, such as trimmed k-means. This approach aligns with the philosophy behind the "deterministic-MCD" method described in Hubert et al. (2012). Instead of exploring these strategies in more detail, the remainder of this manuscript will focus on an "ensemble initialization" approach, which has shown promising results in our preliminary experiments.

3 Ensemble initialization

The standard final output of the algorithm presented in Sect. 2 is the TCLUST partition corresponding to the "best" iterated random initialization, i.e. the iterated solution with the largest value for (1). However, this approach implies that all "partially correct" information from other concentration steps, which do not conduct to this best value found in the target function, is completely neglected.

Different "ensemble clustering" approaches are quite popular in Cluster Analysis nowadays. To name a few, we refer to proposals in Fred and Jain (2002); Strehl and Ghosh (2002); Fred and Jain (2005) or Lipor et al. (2021). The main idea behind these approaches is that combining information from different clustering partitions can often lead to improved partitions of the data. In this work, we start with the same principle and extend it to the initialization of robust clustering procedures, particularly the TCLUST algorithm. We consider that the combination (ensemble) of all iterated random initializations could be useful for defining a new ensemble-type initialization to be further refined. This approach aims to leverage the partially correct information resulting from the concentration steps of all `nstart` random initializations.

To be more precise, the proposal is as follows, outlined in steps A.1 to A.5:

A.1 Consider $\{\mathcal{C}_b\}_{b=1}^{\texttt{nstart}}$, which represents the `nstart` partitions of the indices $\{1, 2, \ldots, n\}$ obtained after `niter1` iterations (steps 1.1 to 1.3 of the algorithm in Sect. 2) from `nstart` random initializations of the TCLUST algorithm (Step 1 of this algorithm), with G clusters and an α trimming level. For the b-th random initialization, the resulting partition after these `niter1` iterations is denoted as

$$\mathcal{C}_b = \{H_0^b, H_1^b, \ldots, H_G^b\},$$

where

$$H_0^b \cup H_1^b \cup \ldots \cup H_G^b = \{1, 2, \ldots, n\},\ H_g^b \cap H_{g'}^b = \emptyset \text{ for } g \neq g',\ \text{and } \#\{H_0^b\} = [n\alpha].$$

Here, $H_1^b, \ldots, H_G^b$ are the indices of the G clusters obtained, and H_0^b represents the indices of the trimmed observations, resulting from the iterated b-th random initialization.

A.2 Obtain an affinity matrix A $(n \times n)$ with terms defined as

$$A_{ii'} = \frac{1}{\texttt{nstart}} \#\{b : x_i \text{ and } x_{i'} \text{ are co-clustered (and not trimmed) in } \mathcal{C}_b\}.$$

In other words,

$$A_{ii'} = \frac{1}{\texttt{nstart}} \#\{b : \{i, i'\} \subset H_g^b \text{ for any } H_g^b \text{ with } g \neq 0\}.$$

A.3 Compute

$$A_i = \sum_{i'=1}^{n} A_{ii'} \text{ for } i = 1, \ldots, n, \tag{3}$$

and sort these A_i values in increasing order $A_{(1)} \leq A_{(2)} \leq \cdots \leq A_{(n)}$. For the ensemble initialization, we initially considered as trimmed observations those x_i with $i \in H_0$, where $H_0 = \{i : A_{(i)} \leq A_{([\alpha n])}\}$, and the non-trimmed observations are those with indices $H = \{1, 2, \ldots, n\} \setminus H_0$.

A.4 A hierarchical clustering algorithm is applied to the $n - [n\alpha]$ observations $\{x_i : i \in H\}$. In this work, we propose using the "Ward criterion" with dissimilarities for observation x_i and $x_{i'}$ given by $1 - A_{ii'}$. The resulting dendrogram is cut into G clusters. Other hierarchical clustering approaches could have been similarly considered, or even "spectral clustering" methods starting from the affinity matrix A. Ward's hierarchical clustering produces a partition of the indices in H as $H = H_1 \cup \ldots \cup H_G$. The partition of $\{1, 2, \ldots, n\}$ given by $H_0 \cup H_1 \cup \ldots \cup H_G$ is used as the "ensemble initialization".

A.5 Starting from the result of Step A.4, further `niter2` concentration steps are applied beginning with Step 1.2 in Sect. 2 by using the $\{H_g\}_{g=1}^{G}$ sets from the obtained ensemble initialization. The desired constraints on the eigenvalue-ratio are enforced during these `niter2` additional concentration steps.

By defining the terms $A_{ii'}$ in the affinity matrix A in this manner, we assign higher affinities $A_{ii'}$ to pairs of observations x_i and $x_{i'}$ that in more iterated random initializations end up in the same cluster (even if they do not always end up co-clustered together). On the other hand, outlying observations are less likely to be co-clustered with non-outlying ones, as they will be trimmed in several random initializations. This results in outliers being associated with lower values in $A_{ii'}$ and, consequently, with smaller values in A_i in (3).

In cases where the ensemble initialization process does not increase the value of the target function compared to the best of the individual initializations, of course, it makes sense to simply return the best iterated result from the individual initializations.

For simplicity, we will use the exact steps A.1-A.5 described above in the remaining material. However, there are many other open possibilities to explore, which are briefly outlined here as remarks:

Remark 1 One possibility to explore is using a different number of clusters, G^0, than G when generating the partitions $\{\mathcal{C}_b\}_{b=1}^{\texttt{nstart}}$ for computing the $A_{ii'}$ terms (i.e., by obtaining `nstart` partitions $\mathcal{C}_b = \{H_0^b, H_1^b, \ldots, H_{G^0}^b\}$ in Step A.1). It is important to note that we still recover the desired number of clusters G by cutting the resulting

dendrogram into G clusters in Step A.4. Additionally, during the `niter2` final concentration steps in Step A.5, we reapply the initial TCLUST parameters c, G, and α. We could experiment with different G^0 values, apply the `niter2` concentration steps to the different ensemble initializations (one for each G^0), and finally select the one corresponding to the G^0 resulting in the highest value of (1). Considering an initially larger number of clusters, $G^0 > G$, is not a new concept in ensemble clustering, as evidenced by the references cited earlier. This idea will be briefly examined in the real data example presented in Sect. 6.

Remark 2 Although we simply suggest using Ward's criterion, other hierarchical clustering techniques could be applied. In fact, several ensemble initializations can be derived from the matrix A by adopting different clustering strategies, and all of them could subsequently be iterated through Step A.5. In this case, the final output of our TCLUST algorithm would be the one that achieves the highest value of (1) after the final `niter2` iterations. Additionally, robust hierarchical clustering techniques based on the affinity matrix A could be explored. Some relevant approaches include those proposed by Li et al. (2007) and Balcan et al. (2014). Spectral clustering methods with appropriate thresholding (Lipor et al. 2021) could also be a viable option for extracting information from the A matrix. Finally, another approach would be to return clustering results directly obtained from the matrix A without applying Step A.5. This might be useful in cases where deviations from normal distribution assumptions occur and more general clustering structures are required. However, this option is not considered in this work, as the concentration steps in Step A.5 are designed to increase the target function (1), aligning with our goal of implementing TCLUST to achieve the highest possible target function value.

Remark 3 The ensemble initialization method can become problematic when n is large because the matrix A contains $n(n+1)/2$ elements that need to be stored in memory. Additionally, applying hierarchical clustering techniques to a large number of observations, $n - [n\alpha]$, can be problematic when n is large. However, this difficulty may be mitigated by using subsampling. Specifically, we can consider a random subsample of size n_0, with $n_0 < n$, to derive an ensemble initialization from this smaller subsample, which can then be refined using the full dataset in a modified Step A.5. This approach aligns with the "nested extensions" suggested in Section 3.3 of Rousseeuw and van Driessen (1999) and with more sophisticated methods like those described in De Ketelaere et al. (2020) for applying MCD to large sample sizes. Section 5 provides an example of how combining subsampling and ensemble initialization can be effective in situations where the sample size n is moderately large.

4 Illustrative example and simulation study

4.1 Illustrative example

We begin by demonstrating the potential benefits of the proposed ensemble initialization approach with an illustrative example. This example uses $n = 400$ observations

in $p = 10$ dimensions, with $G = 6$ elliptical clusters and 10% contamination added (more details on how the dataset is generated will be provided in Sect. 4.2). The relevant information about clusters and contamination is included in the first two variables. Figure 1 shows a pairwise plot of the first four variables out of the 10 variables.

The TCLUST algorithm described in Sect. 2 is first applied using the correct parameter values $G = 6, \alpha = 0.1$, and $c = 81$, with `nstart=100` random initializations and `niter=20` concentration steps. Figure 2 shows the values of the target function (1) resulting from the concentration steps applied to the `nstart` random initializations. The highest value of the target function found is -6381.048, marked with a red point.

The partition associated with that "best" initialization is represented in Fig. 3, showing only the first two variables. We see that this solution is deficient, with a

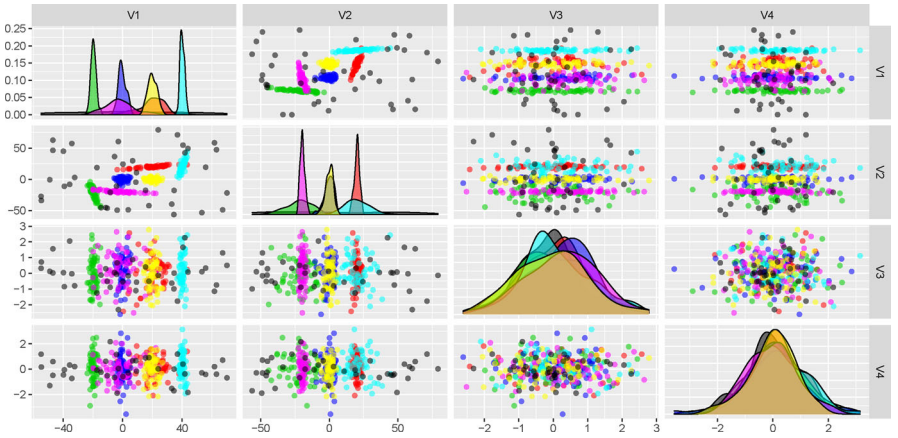


Fig. 1 The first 4 of the 10 variables for the dataset used in the illustrative example, with different colors representing the 6 clusters and black representing the contaminating observations (color figure online)

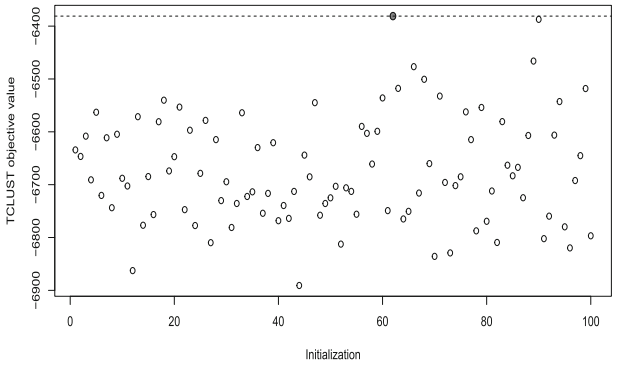


Fig. 2 Values of the target function from `nstart=100` random initializations and `niter=20` concentration steps, marking in red the highest value found (color figure online)

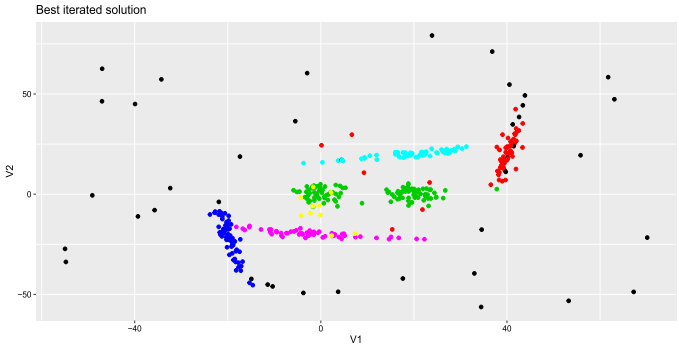


Fig. 3 Partition associated with that best initialization in the first two variables

misclassification rate of 20.75% since, for instance, two clear differentiated clusters are mixed together. Perhaps `nstart=100` was not a sufficient number of initializations for the complexity of the problem (p and G). The computational time spent in performing these initializations and concentration steps was 1.518 s.

Figure 4 shows the results after `niter=20` concentration steps for the first 4 random initializations (out of the `nstart=100` tried). As expected, none of them is perfect, but all four exhibit some "partial success" in detecting parts of the cluster structure and much of the contamination. This partially correct information is exactly what we aim to exploit through the ensemble initialization proposal.

The partially correct information in the `nstart=100` iterated random initialization with `niter=20` is being gathered in the affinity matrix A by following the procedure described in Sect. 3, represented in the heatmap graph in Fig. 5. We can see

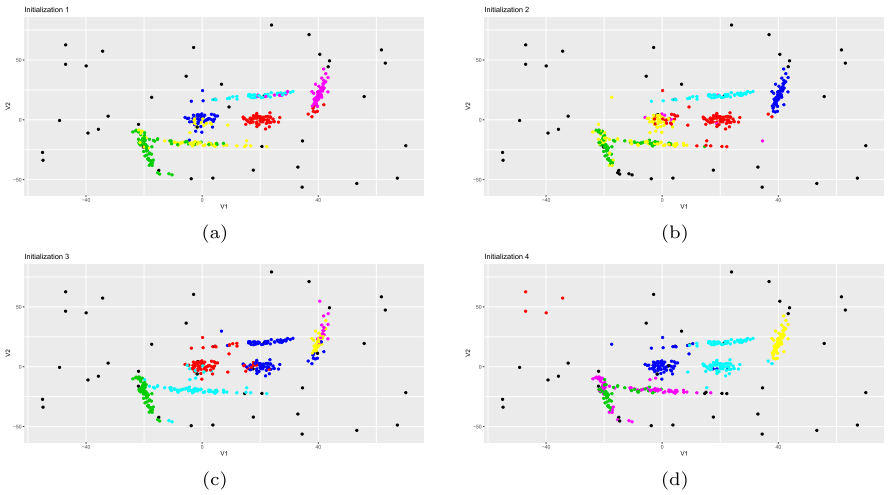


Fig. 4 Result of the concentration steps for the first 4 random initializations tried in the first two variables

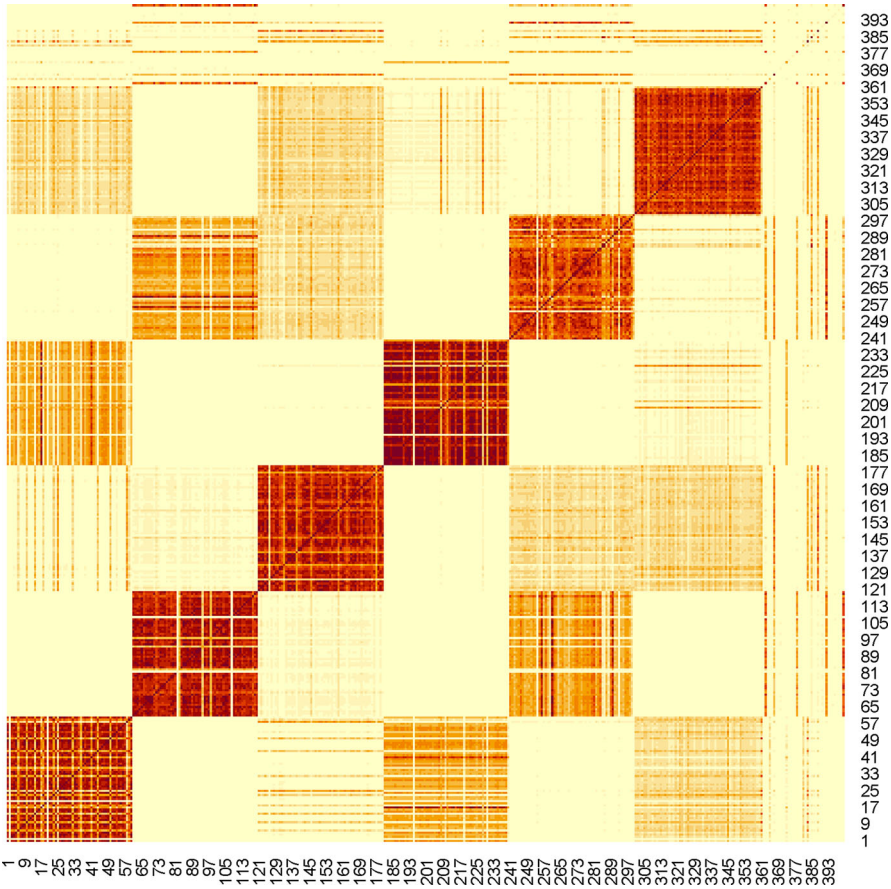


Fig. 5 Heatmap of the affinity matrix A obtained from the `nstart=100` random initializations

there that, near the diagonal, a structure of 6 boxes associated with higher affinities are clearly seen. In fact, the A matrix is in clear correspondence with the underlying clusters structure (no reordering of rows and columns has been done in the heatmap and the data generation process has followed the clusters sequential order). Figure 5 also displays several small $A_{ii'}$ affinities in its upper and right sections, which correspond to the portion of the dataset with the 10% added contamination.

The steps A.3 to A.5 described in Sect. 3 are applied with `niter2=10`, by starting from this matrix A. We obtain the robust clustering partition shown in Fig. 6 (only the first two variables shown). We can see that the obtained results are now more satisfactory, with an improved classification error rate 5.5% and the value of the target function has risen to -6352.729. The additional time required to achieve this improved solution is approximately 0.324 s.

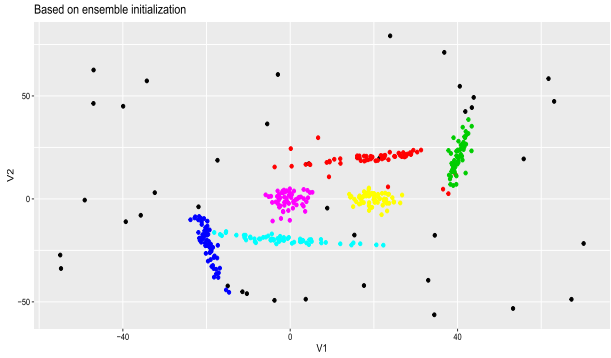


Fig. 6 Partition resulting from applying ensemble initialization (only the first two variables are shown)

4.2 Simulation study

The proposed simulation study is based on generating $n = \texttt{n}$ random observations ($\texttt{n} = 200, 500$ or 1000) by following three different types of scenarios (`scenario=1`, 2 or 3) with $G = \texttt{G}$ clusters ($\texttt{G} = 3$ or 6) and including a fixed fraction 10% of contaminating observations. These contaminating observations are uniformly distributed in the range for the non-outlying part of the data in a "symmetric" type of contamination (`symmetry=1`) or more concentrated in one corner of the dataset in an "asymmetric" type of contamination (`symmetry=2`). The datasets are generated in dimension $p = \texttt{p}$ ($\texttt{p} = 4$ and 10), but only the first two variables include relevant information about clusters and outliers while the remaining $\texttt{p}-2$ variables are "noise" variables each following independent standard normal distributions.

In `scenario=1`, we generate `G` spherically and equally scattered clusters with fairly close location vectors. `scenario=2` corresponds to `G` elliptical clusters with different orientations and some overlap between clusters. Finally, `scenario=3` corresponds to a case analogous to `scenario=1` but with three more distant locations vectors. The three scenarios have been generated with normal components whose maximal eigenvalues-ratio for their scatter matrices satisfies the eigenvalues restriction in (2) for $c = 1$ in the `scenario=1` and `scenario=3`, and $c = 9^2 = 81$ for `scenario=2`. We considered a `balanced=1` case that generates groups with the same sizes (all clusters including 30% of the observations for `G=3` and all clusters including 15% of the observations for `G=6`). The `balanced=2` case creates groups of different sizes (groups of sizes 25%, 30% and 35% for `G=3` and two groups with the 12.5%, two with 15% and two with 17.5% for `G=6`). The specific generation of these data sets follows from the application of a `simula_tclust(n,p,G,scenario,balanced,symmetry)` function which is provided in the supplementary material file. The dataset in Sect. 4.1 was the result of the application of `simula_tclust(n=400,p=10,G=6,scenario=2,balanced=1,symmetry=1)`. Figure 7 shows some examples of the generated data sets when `n=400` in the three

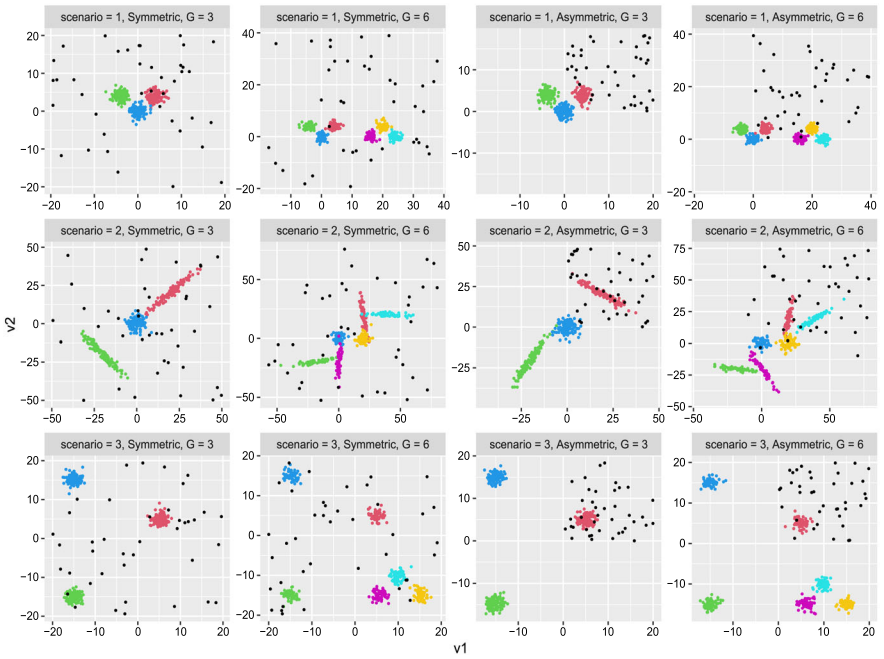


Fig. 7 Examples of the generated datasets in the simulation study, for different combinations of parameters (rows and columns) in the `balanced=1` case when `p=2`. Datasets have been standardized to make their scales more comparable. Different colors are used for the clusters, with black representing the contamination (color figure online)

scenarios and the two types of contamination for different numbers of clusters only in the `balanced=1` case when `p=2`.

For the simulation study, we have considered 100 random realizations of `simula_tclust()` for all possible parameter combinations. For each of these generated data sets, we apply the following three implementations of the TCLUST with appropriate G, α and c parameters (i.e., coinciding with the ones when generating the datasets):

Method 1 The use of the TCLUST algorithm in Sect. 2 with `nstart` random initializations (equal to 100 and 400) and `niter=20`.

Method 2 The ensemble initialization procedure in Sect. 3, starting from the same iterated `nstart` random initializations, and using `niter2=20` in Step A.5.

Method 3 The use of the TCLUST algorithm in Sect. 2 with $5\times$`nstart` random initializations and `niter=20`, where `nstart` is the number of random initializations considered for *Method 1* and *Method 2*.

Figure 8 shows boxplots for the TCLUST target values (the larger the better), Fig. 9 shows boxplots for the misclassifications rates (the smaller the better) and Fig. 10 the values of the computing times (the smaller the better) in a logarithmic scale. Only the results when `balance=1` and `symmetry=1`, i.e. the balanced and symmetric

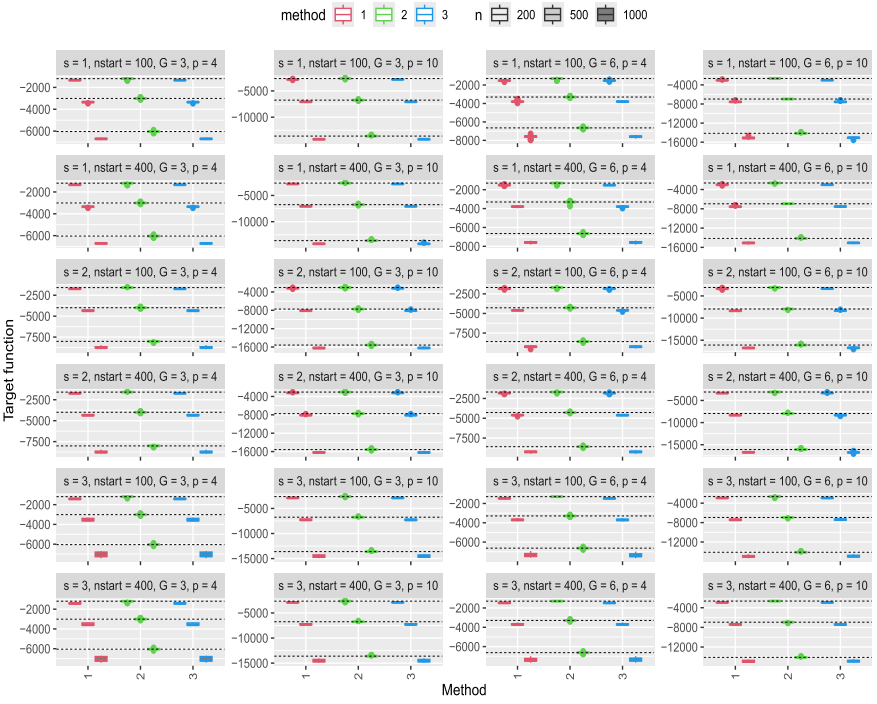


Fig. 8 Target function values for different combinations of parameters in the simulation study in the `balance=1` and `symmetry=1` case. *Method 2* (green color) shows the results for the proposed ensemble initialization and *Method 3* (blue color) the results when using *Method 1* (red color) with four times more random initializations. To better illustrate these values, horizontal dashed lines are plotted at the median of the target values obtained for *Method 2*. `s=1`, `s=2` and `s=3` refer to the 3 scenarios (`scenario=1`, `2` and `3`) (color figure online)

contamination case, are shown in these figures but the results are analogous in the other cases. The results for other values of `balance` and `symmetry` are given in the supplementary material file.

The boxplots in Fig. 8 show improvements in the target function in (1) for the proposed *Method 2* (in green) compared to *Method 1* (in red) and even *Method 3* (in blue). We have highlighted these improvements by adding horizontal dashed lines at the median target function values obtained for *Method 2*, aiming to make it easier to see the advantage provided by the ensemble initialization approach. In fact, it can be observed that the number of times that the ensemble initialization in *Method 2* results in higher target function values than *Method 1* is almost 100% or extremely close to 100% in all the situations considered in this simulation study. This advantage, in this particular case, translates into a reduction in misclassification rates, as shown in Fig. 9. The misclassification rate is considered as the fraction of non-trimmed observations that are not correctly assigned to their respective clusters (a reordering of cluster labels can be applied to address label switching issues), and a misclassified observation is also counted if a contaminating data point is wrongly assigned to any cluster. The

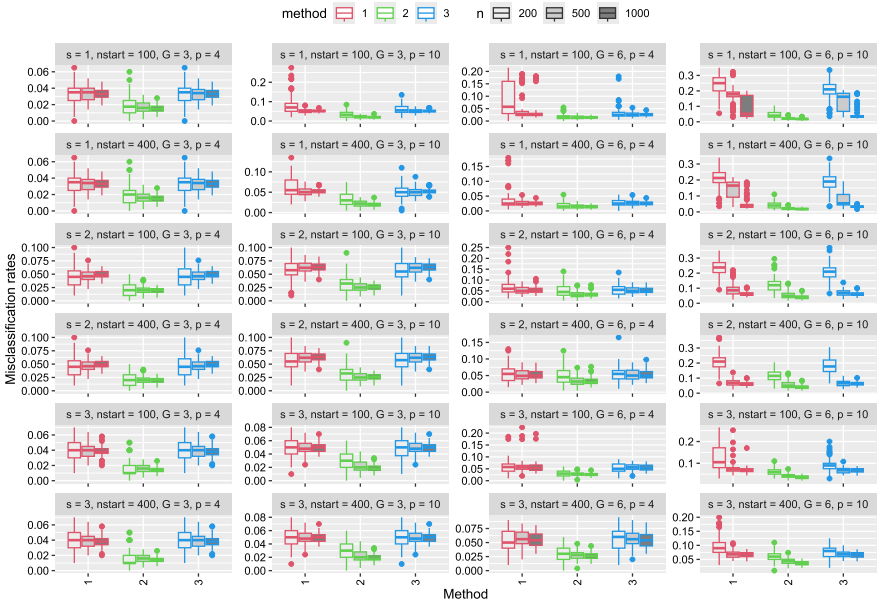


Fig. 9 Misclassification rates for different combinations of parameters in the simulation study in the `balance=1` and `symmetry=1` case. *Method 2* (green color) shows the results for the proposed ensemble initialization and *Method 3* the results when using *Method 1* with four times more random initializations (color figure online)

advantages of *Method 2* in terms of target function values and misclassification rates are not due to a significant increase in computing times, as shown in Fig. 10. The computing times never exceed those of using four times more random initializations, as done in *Method 3*, even though those four times more random initializations do not manage to significantly improve the target function values and misclassification rates as much as the proposed *Method 2*.

5 Combining subsampling and ensemble initialization

As mentioned in Remark 3, large sample sizes n could pose a challenge for the proposed ensemble initialization approach due to the large size ($n \times n$) of the affinity matrix A. However, we have identified a very simple yet seemingly effective approach based on "subsampling" in such cases. The idea is to randomly extract a subsample of size n_0, with $n_0 < n$ but such that $n_0 \times n_0$ is not an issue for applying an ensemble initialization approach. Let $\{i_1, \ldots, i_{n_0}\}$ be a random subsample of $\{1, \ldots, n\}$ and apply steps A.1 to A.4 to the subsample $\{x_{i_1}, \ldots, x_{i_{n_0}}\}$ to get the ensemble initialization $H_0 \cup H_1 \cup \cdots \cup H_G$ for that subsample. However, the final concentration steps in A.5 are performed to the "whole" data set $\{x_1, \ldots, x_n\}$ where initial parameters (instead of one initial partition) for these final concentration steps are computed from the sample

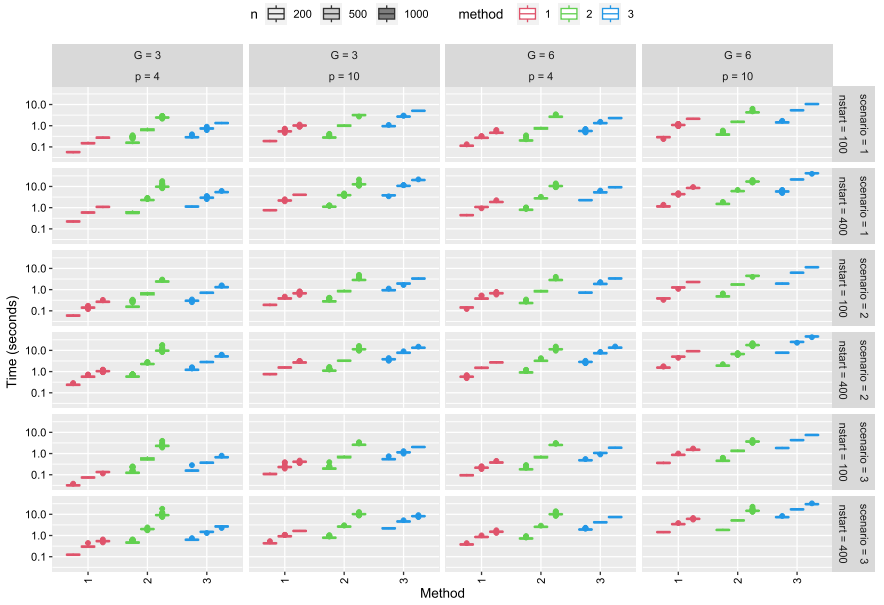
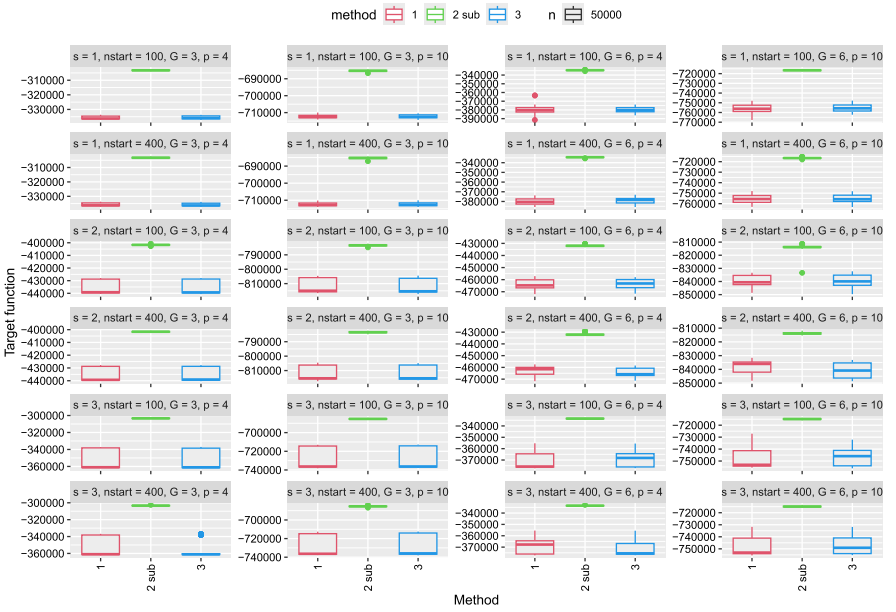


Fig. 10 Computing times in seconds (in logarithmical scale) for different combinations of parameters in the simulation study in the `balance=1` and `symmetry=1` case. *Method 2* (green color) shows the results for the proposed ensemble initialization and *Method 3* the results when using *Method 1* with four times more random initializations (color figure online)

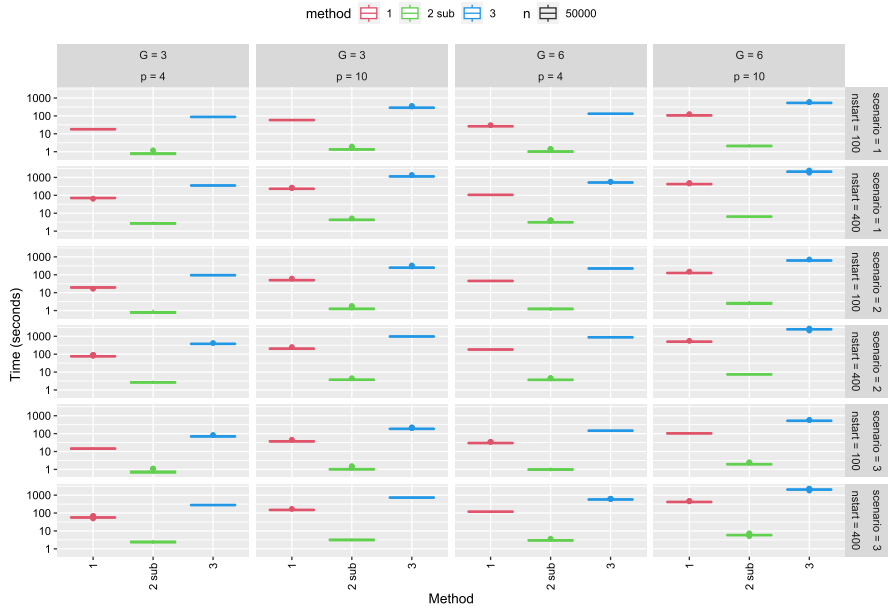
means and the sample covariances of the observations in each H_g, $g = 1, \ldots, G$, and weights initialized as $\#H_g/(n_0 - [n_0\alpha])$.

We have found that this simple approach works quite well in preliminary experiments. For instance, consider 100 datasets generated by `simula_tclust(n=50000,p,G,scenario,balanced=1,symmetry=1)` using the same values for `G`, `p` and `scenario` as in the simulation study in Sect. 4.2. Note that we are generating relatively large datasets, each with $n = 50000$ observations. We apply the described subsampling strategy with $n_0 = 400(<< 50000)$ and `niter2=20` for two different values of `nstart` (100 and 400). The results, in terms of target values and computing times (on a logarithmic scale), are presented in Fig. 11 and compared with the results of applying *Method 1* and *Method 3*, as introduced in Sect. 4.2, to these larger datasets.

We can clearly see that this combination of subsampling with ensemble initializations (*Method 2 sub*) produces the highest values for the target function. On the other hand, the computing time is significantly reduced compared to directly handling the dataset with $n = 50000$ using *Method 1*. This reduction in computing time is even more pronounced when $5 \times$ `nstart` random initializations are considered in *Method 3*, which does not significantly increase the target function values but results in highly demanding computing times.



(a)



(b)

Fig. 11 Target values in (**a**) and computing times (on logarithmic scale) in (**b**) when comparing the subsampling combined with ensemble initialization (denoted as *Method 2 sub*) described in Sect. 5 with subsampling size $n_0 = 400$ (green color) with respect *Method 1* (in red color) and *Method 3* (blue color) for simulated datasets of size $n = 50000$ (color figure online)

Another example with an even larger sample size, $n = 200000$, is provided in the supplementary materials file.

6 Real data example: olive oil data

In this example, we will see the benefits, in terms of achieving higher values in the target function, when using the proposed ensemble initialization methodology on a well-known real dataset while applying TCLUST for different combinations of G and α. With that purpose, we use the "olive oil" dataset (Forina et al. 1983), which is available, for instance, from the `pgmm` package at CRAN (McNicholas et al. 2015) and contains $p = 8$ chemical measurements on the acid components of $n = 572$ olive oil specimen produced in various regions in Italy. We apply TCLUST with $G = 7$ and $G = 9$ (the "true" number of regions is 9, but $G = 7$ has been also considered in the literature as can be seen in Cerioli et al. 2018), $\alpha = 0.05$ and $\alpha = 0.1$, and when $c = 15$ is fixed. The following three methods are considered:

Method 1 The TCLUST algorithm in Sect. 2 is used with `nstart=1000` and different values for `niter1`.

Method 2 The proposed ensemble initialization procedure described in Sect. 3, starting from the same number of random initializations `nstart=1000` and the same number of initial concentration steps `niter1`.

Method 3 The use of the TCLUST algorithm as in *Method 1* but with `nstart=5000` (i.e., four times more random initializations than *Method 1*).

Figure 12 shows boxplots corresponding to the values achieved in the target function (the larger the better) after running 100 times *Method 1* with same values of `niter1`, equal to 5, 10, 20 and 50.

We can see how, in this example, the ensemble initialization strategy in *Method 2* consistently yield larger values in the target function values than *Method 1* and *Method 3*. Moreover, the values attained in the target function when using the ensemble initialization approach seem to exhibit less variability and lower dependence on the particular choice of `niter1`. Due to the fact that $n = 572$ is not particularly large, the extra computing time to compute the affinity matrix and performing the hierarchical clustering never exceeded 5% of the total computing time.

We also take advantage of this example to briefly investigate the possibility of considering different values of G^0, not necessarily equal to G, when computing the affinity matrix A, as mentioned in Remark 1. In this example, we focus on the application of TCLUST when $G = 9$, $\alpha = 0.05$ and $c = 15$ by considering *Method 1*, *Method 2* and *Method 3*, again, applied 100 times with different values of `niter1` (5, 10, 20 and 50). However, when applying *Method 2*, we have considered different values of G^0 for obtaining the partitions $\{\mathcal{C}_b\}_{b=1}^{\texttt{nstart}}$ with $\mathcal{C}_b = \{H_0^b, H_1^b, \ldots, H_{G^0}^b\}$ in Step A.1. Apart from $G^0 = G = 9$, we have considered a value of $G^0 = 7$, smaller than 9, and values $G^0 = 12$, 18 and 24, larger than 9. Recall, that after applying steps A.3, A.4 and A.5 of the proposal, we still return a TCLUST-type partition with $G = 9$ clusters and an $\alpha = 0.05$ fraction of trimmed observations, regardless of the value G^0 used to define the affinity matrix.

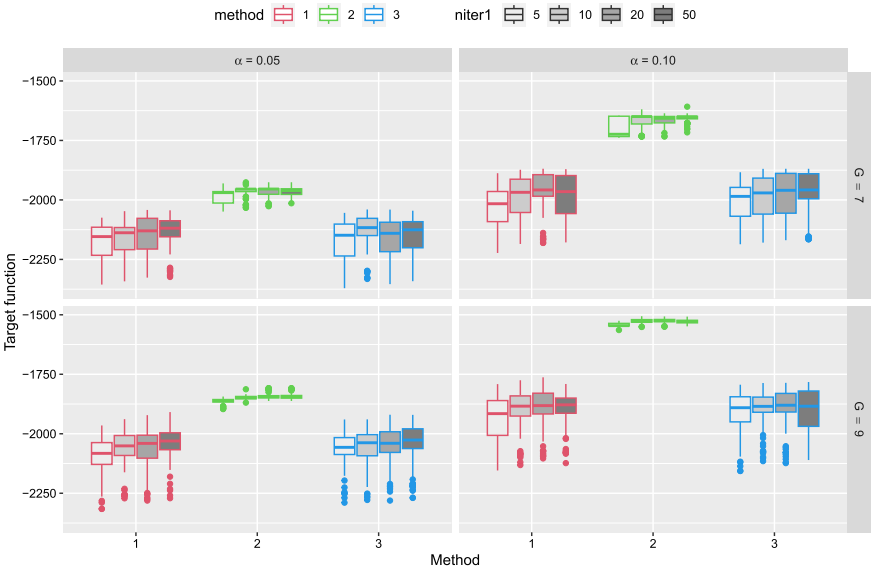


Fig. 12 Boxplots for the target function values achieved in 100 applications of *Method 1* (red color), *Method 2* (green color) and *Method 3* (blue color), as described in Sect. 6, when applying TCLUST for $G = 7$ and 9 and $\alpha = 0.05$ and 0.1 with $c = 15$ in the olive oil dataset. Different values of `niter1` are considered (color figure online)

Figure 13 shows the boxplots for the target function values achieved. We see that sometimes better values in the target function can be attained with G^0 values larger than $G = 9$. Anywhere, we also see that larger values of the target functions, combined with less variability, seems to be consistently attained in this example regardless of the choice of G^0 through ensemble initialization.

It is important to note that a higher value of the target function (1) does not necessarily guarantee a smaller misclassification rate when a "true" labeling is available. Figure 14 shows boxplots for the misclassification rates associated to the results of the three methods (*Method 1*, *Method 2* and *Method 3*) displayed in Fig. 13. These misclassification rates are computed for the non-trimmed observations based on their assignments to the 9 "true" regions, after the best possible permutation of the cluster labels. Note that, in this specific example, higher values of G^0 sometimes appear to perform better in terms of misclassification rates. However, we cannot rely on the supposedly unknown "true" labels to determine the appropriate G^0. Once again, we emphasize that achieving a higher value of (1), depending on the chosen c, is our sole criterion when using TCLUST for fixed values of the parameters G, α, and c.

Similar experiments have been conducted for other values of c, as shown in the supplementary material file. We observe that higher values of the target function are consistently obtained when using this ensemble initialization approach, but its effect on the misclassification rate is more difficult to evaluate, in this particular case, at least for the values of G and α considered.

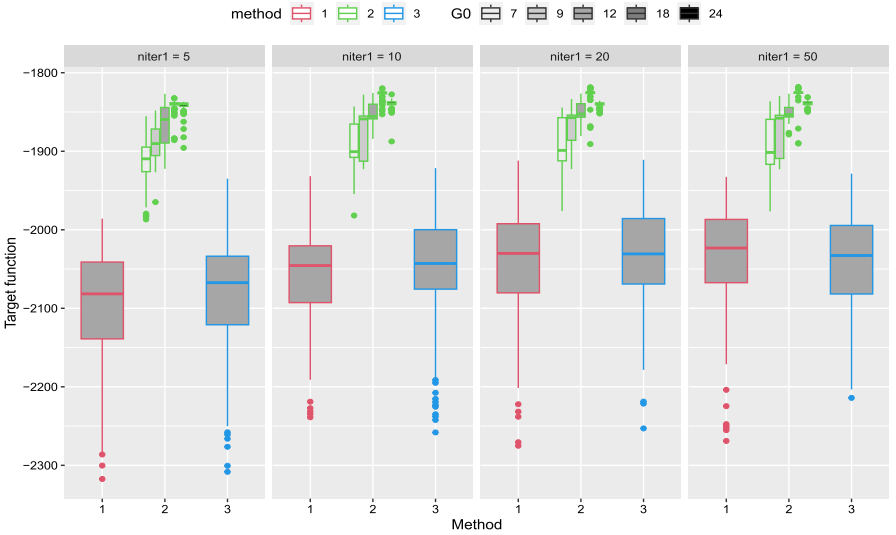


Fig. 13 Boxplots for the target function values achieved in 100 applications of *Method 1* (red color), *Method 2* (green color) and *Method 3* (blue color), with different values of `niter1`, when applying TCLUST for $G = 9$ and $\alpha = 0.05$ with $c = 15$ in the olive oil dataset. Different numbers of clusters G^0 in the partitions in Step A.1 are considered (color figure online)

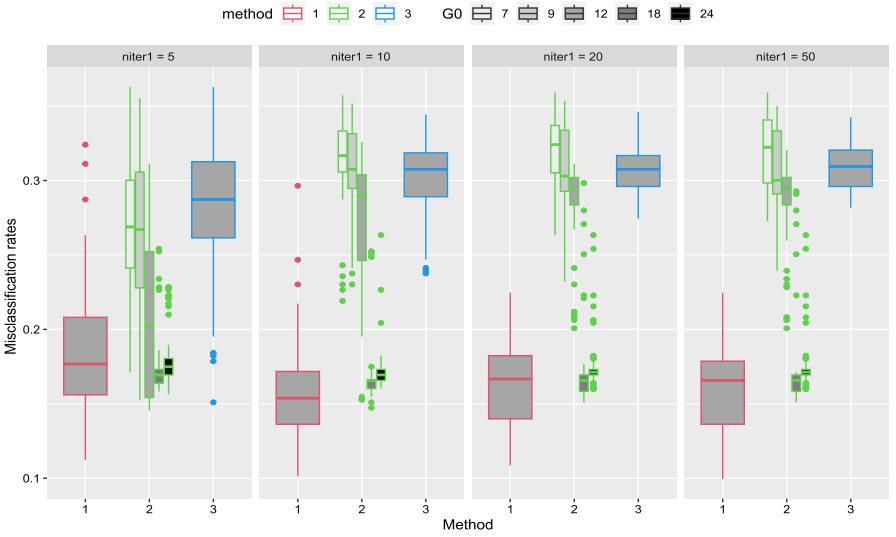


Fig. 14 Boxplots for the misclassification rates associated to the results of the three methods displayed in Fig. 13

7 Conclusion and open research lines

TCLUST is a robust clustering method based on trimming, which involves an algorithm that requires random initializations and concentration steps for its implementation. While the algorithm can be successfully applied in low dimensions p and with a small number of clusters G, its performance deteriorates as these conditions change if an extremely large number of random initializations is not considered. A novel partial remedy for this drawback has been introduced in this work, based on a new ensemble initialization approach. The initial experiments have shown promise, and we are confident that this type of ensemble initialization, or suitable modifications of it, can be genuinely helpful in practice. Of course, this proposal does not attempt to completely overlook the significant challenges associated with implementing TCLUST in cases with very high values of p or G.

The need for correct initializations is a common requirement for many other methods in robust clustering, such as trimmed k-means, the robust linear grouping algorithm (García-Escudero et al. 2009), OTRIMLE (Coretto and Hennig 2016), trimmed likelihoods (Neykov et al. 2007), and possibly other methods mentioned in the references provided in Sect. 1. We believe that an ensemble-type initialization approach, or tailored adaptations of it, could also be beneficial for these other robust clustering methods.

We have assumed in this work that G, α and c are known parameters for TCLUST. Admittedly, in most of the cases, we deal with unknown parameters to be determined from the dataset at hand and from the user's ultimate goals. Anyway, being able to apply TCLUST for fixed G and α in a computationally efficient way is clearly a first step towards the derivation of sensible parameter values. All the procedures already available in the literature to guide the user in determining input parameters in TCLUST require the careful monitoring of changes due to different combinations of parameter values (García-Escudero et al. 2011; Dotto et al. 2018; Cerioli et al. 2018). Furthermore, relevant information extracted from the affinity matrix A could result in new useful tools to choose parameter values in TCLUST.

The material presented here clearly warrants further research. In this work, we have made a simple and not overly elaborate use of the ideas behind the ensemble initialization proposal. Despite this initial application, we have observed noticeable advantages in achieving higher values in the constrained maximization that TCLUST aims to perform. We expect that more careful and refined approaches will likely yield even better results. Several open research directions and potential functionalities to explore are outlined in the remarks in Sect. 3. Once tested, our ultimate goal is to incorporate them into ready-to-use software.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11634-025-00642-9.

Acknowledgements Research partially supported by grant PID2021-128314NB-I00 funded by MCIN/AEI/10.13039/501100011033/FEDER and Junta Castilla y León grant VA064G24. The authors also thank the associate editor and two anonymous referees for their constructive comments, which have been very helpful in improving the manuscript.

Author contributions All authors contributed equally to this work.

Funding Open access funding provided by FEDER European Funds and the Junta de Castilla y León under the Research and Innovation Strategy for Smart Specialization (RIS3) of Castilla y León 2021-2027. Open access funding provided CRUE-CSIC Alliance and Universidad de Valladolid. This research has been partially supported by grant PID2021-128314NB-I00 funded by MCIN/AEI/10.13039/501100011033/FEDER and Junta Castilla y León grant VA064G24.

Data availability All data analyzed in this study are available and referenced in this published article.

Declarations

Conflict of interest All authors declare that they have no conflict of interest.

Ethics approval Not applicable.

References

Balcan M, Liang Y, Gupta P (2014) Robust hierarchical clustering. J Mach Learn Res 15:3831–3871

Banerjee A, Dave RN (2012) Robust clustering. Wiley Interdiscip Rev Data Min Knowl Discov 2:29–59

Biernacki C, Celeux G, Govaert G (2003) Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models. Comput Stat Data Anal 41:561–575

Celeux G, Govaert A (1992) A classification EM algorithm for clustering and two stochastic versions. Comput Stat Data Anal 14:315–332

Cerioli A, García-Escudero L, Mayo-Iscar A et al (2018) Finding the number of normal groups in model-based clustering via constrained likelihoods. J Comput Graph Stat 27:404–416

Coretto P, Hennig C (2016) Robust improper maximum likelihood: tuning, computation, and a comparison with other methods for robust Gaussian clustering. J Am Stat Assoc 111:1648–1659

Cuesta-Albertos J, Gordaliza A, Matrán C (1997) Trimmed k-means: an attempt to robustify quantizers. Ann Stat 25:553–576

De Ketelaere B, Hubert M, Raymaekers J et al (2020) Real-time outlier detection for large datasets by RT-DetMCD. Chemom Intell Lab Syst 199:103957

Dotto F, Farcomeni A, García-Escudero L et al (2018) A reweighting approach to robust clustering. Stat Comput 28:477–493

Forina M, Armanino C, Lanteri S et al (1983) Classification of olive oils from their fatty acid composition. In: Martens M, Russwurm HJ (eds) Food research and data Analysis. Applied Science Publishers, London, pp 189–214

Fred A, Jain A (2002) Data clustering using evidence accumulation. In: 2002 International Conference on Pattern Recognition, vol 4. IEEE, Quebec City, QC, Canada, pp 276–280

Fred A, Jain A (2005) Combining multiple clusterings using evidence accumulation. IEEE Trans Pattern Anal Mach Intell 27(6):835–850

Fritz H, García-Escudero L, Mayo-Iscar A (2012) tclust: an R package for a trimming approach to cluster analysis. J Stat Softw 47(12)

Fritz H, García-Escudero L, Mayo-Iscar A (2013) A fast algorithm for robust constrained clustering. Comput Stat Data Anal 61:124–136

García-Escudero L, Mayo-Iscar A (2024) Robust clustering based on trimming. Wiley Interdiscip Rev Comput Stat 16(4):e1658

García-Escudero L, Gordaliza A, Matrán C et al (2008) A general trimming approach to robust cluster analysis. Ann Stat 36:1324–1345
García-Escudero L, Gordaliza A, San Martín R et al (2009) Robust linear clustering. J R Stat Ser B Stat Methodol 71:301–318
García-Escudero L, Gordaliza A, Matrán C et al (2010) A review of robust clustering methods. Adv Data Anal Classiff 4:89–109
García-Escudero L, Gordaliza A, Matrán C et al (2011) Exploring the number of groups in robust model-based clustering. Stat Comput 21:585–599
García-Escudero L, Gordaliza A, Mayo-Iscar A (2014) A constrained robust proposal for mixture modeling avoiding spurious solutions. Adv Data Anal Classiff 8:27–43
García-Escudero L, Gordaliza A, Matrán C, et al (2016) Robustness and outliers. In: C. Hennig FM. Meila, R. Rocci (eds) Handbook of cluster analysis. Serie Chapman & Hall/CRC Handbooks of Modern Statistical Methods, Boca Raton, p 653–678
García-Escudero L, Mayo-Iscar A, Riani M (2022) Constrained parsimonious model-based clustering. Stat Comput 32:1–15
García-Escudero LA, Gordaliza A, Greselin F et al (2018) Eigenvalues and constraints in mixture modeling: geometric and computational issues. Adv Data Anal Classif 12:203–233
Hubert M, Rousseeuw P, Verdonck T (2012) A deterministic algorithm for robust location and scatter. J Comput Graph Stat 21:618–637
Ingrassia S, Rocci R (2007) Constrained monotone EM algorithms for finite mixture of multivariate Gaussians. Comput Stat Data Anal 51:5339–5351
Li Z, Liu J, Chen S, et al (2007) Noise robust spectral clustering. In: 2007 IEEE 11th international conference on computer vision, IEEE, pp 1–8
Lipor J, Hong D, Tan Y et al (2021) Subspace clustering using ensembles of K-subspaces. Inf Infer J IMA 10:73–107
Lloyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28:129–137
McNicholas P, ElSherbiny A, Jampani K, et al (2015) pgmm: parsimonious gaussian mixture models. Available https://cran.r-project.org/web/packages/pgmm/
Neykov N, Filzmoser P, Dimova R et al (2007) Robust fitting of mixtures using the trimmed likelihood estimator. Comput Stat Data Anal 4:299–308
Riani M, Perrotta D, Torti F (2012) FSDA: a MATLAB toolbox for robust analysis and interactive data exploration. Chemom Intell Lab Syst 116:17–32
Ritter G (2014) Cluster analysis and variable selection. CRC Press, Boca Raton
Rousseeuw P, van Driessen K (1999) A fast algorithm for the minimum covariance determinant estimator. Technometrics 41:212–223
Rousseeuw P, van Driessen K (2000) An algorithm for positive-breakdown regression based on concentration steps. Data Analysis. Springer, Berlin, Heidelberg, pp 335–346
Rousseeuw P, Leroy A (1987) Robust regression and outlier detection. Wiley-Interscience, New York
Ruwet C, García-Escudero L, Gordaliza A et al (2012) The influence function of the TCLUST robust clustering procedure. Adv Data Anal Classif 2:107–130
Ruwet C, García-Escudero L, Gordaliza A et al (2013) On the breakdown behavior of the tclust clustering procedure. TEST 22:466–487
Strehl A, Ghosh J (2002) Cluster ensembles-a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617