3-1-2022

Spectral Clustering of Mixed-Type Data

Felix Mbuga
San Jose State University

Cristina Tortora
San Jose State University, cristina.tortora@sjsu.edu

Recommended Citation

Felix Mbuga and Cristina Tortora. "Spectral Clustering of Mixed-Type Data" *Stats* (2022): 1-11. https://doi.org/10.3390/stats5010001

stats

MDPI

Article

Spectral Clustering of Mixed-Type Data

Felix Mbuga † and Cristina Tortora *,†

Department of Mathematics and Statistics, San José State University, San Jose, CA 95116, USA; felix.mbuga@sjsu.edu

* Correspondence: cristina.tortora@sjsu.edu

† These authors contributed equally to this work.

Abstract: Cluster analysis seeks to assign objects with similar characteristics into groups called clusters so that objects within a group are similar to each other and dissimilar to objects in other groups. Spectral clustering has been shown to perform well in different scenarios on continuous data: it can detect convex and non-convex clusters, and can detect overlapping clusters. However, the constraint on continuous data can be limiting in real applications where data are often of mixed-type, i.e., data that contains both continuous and categorical features. This paper looks at extending spectral clustering to mixed-type data. The new method replaces the Euclidean-based similarity distance used in conventional spectral clustering with different dissimilarity measures for continuous and categorical variables. A global dissimilarity measure is than computed using a weighted sum, and a Gaussian kernel is used to convert the dissimilarity matrix into a similarity matrix. The new method includes an automatic tuning of the variable weight and kernel parameter. The performance of spectral clustering in different scenarios is compared with that of two state-of-the-art mixed-type data clustering methods, *k*-prototypes and KAMILA, using several simulated and real data sets.

Keywords: cluster analysis; spectral clustering; mixed-type data

Citation: Mbuga, F.; Tortora, C. Spectral Clustering of Mixed-Type Data. *Stats* **2022**, *5*, 1–11. https://doi.org/10.3390/stats5010001

Academic Editors: Marta Nai Ruscone and Daniel Fernández

Received: 9 October 2021
Accepted: 27 November 2021
Published: 23 December 2021

1. Introduction

Clustering or cluster analysis seeks to assign objects with similar characteristics into groups called clusters such that objects within a group are similar to each other and dissimilar to objects in other groups. Different definitions of similarity and dissimilarity can be used, and each definition can lead to a different clustering technique. In this paper, we focus on partitioning clustering techniques, i.e., the algorithm finds an optimal partition of the n units in k clusters. Clustering partitioning techniques include model-based clustering (e.g., Gaussian mixture models [1]), partition-based clustering (e.g., k-means clustering [2]), graph-based clustering (e.g., spectral clustering [3]), and density-based clustering (e.g., Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [4]). Although there is not a single clustering technique that is always preferred to the others, spectral clustering has the advantage of detecting clusters of arbitrary shapes, such as non-convex clusters, and of giving good results when clusters overlap. Spectral clustering does not always give the best performance but it is always among the better techniques [5]. One of the main limitations of spectral clustering in practical applications is that it requires continuous data, whereas many data sets have mixed-type data, i.e., continuous and categorical variables. SpectralCaT [6] was proposed to deal with this problem, it first automatically transforms the data into categorical values and then applies a dimension reduction version of spectral clustering. Another strategy to cluster mixed-type data is to use a specific distance measure for mixed-type data, for example, Gower's dissimilarity [7]. Gower's dissimilarity is a weighted sum of individual dissimilarities for continuous and categorical variables. For categorical variables, the simple matching coefficient is often used: assigning a value of 1 when the two objects being compared have the same value and 0 otherwise. For continuous variables, the Manhattan distance is often used. Several dissimilarity measures exist [8], and several mixed-type data clustering algorithms have

been developed using the idea of a different dissimilarity measure for each type of variable. One of the most used is *k*-prototypes [9], that extends *k*-means clustering for mixed-type data. KAy-means for MIxed LArge data (KAMILA) [10] is a further semi-parametric extension of *k*-means. Many other algorithms exist, see for example [11]. For a review of mixed-type data clustering techniques, see [12–14]. In this paper we propose an extension of spectral clustering for mixed-type data based on different dissimilarity measures per type of variable. We also introduce an automatic tool to tune the weight of the continuous vs. categorical variables. The method is compared with KAMILA and *k*-prototypes on simulated and real data.

2. Background: Spectral Clustering

Spectral clustering frames the clustering problem as one of partitioning a graph [15]. The $n \times p$ data matrix X is first used to construct a similarity graph $G(V, E)$ of the data where the vertices/nodes V represent the n objects to be clustered and the edges E connect the vertices based on some measure of similarity between the n points. The clustering problem is then seen as finding a partition of the graph such that the edges between different groups have very low weights and edges within a group have high weights. The most intuitive cut, i.e. the cut that minimizes the sum of the edges, leads to isolate points. Two commonly used criteria are called RatioCut [16] and the normalized cut Ncut [17]. The approximate solution can be obtained by doing an eigendecomposition of a graph Laplatian matrix L. The solution obtained is a real valued solution matrix, see Figure 1, that can be converted into a discrete partition using a standard clustering method, such as *k*-means clustering, on the normalized eigenvectors.

To built the graph $G(V, E)$ starting from the data matrix X with p continuous columns, it is common to first calculate pairwise dissimilarities/distances and then use these to calculate similarities. On the similarity matrix, several spectral clustering algorithms can be used and each of them has some advantages [18]. A commonly used one is the NJW algorithm by Ng, Jordan and Weiss [19]. The first step of the NJW algorithm is the calculation of all pairwise squared Euclidean distances between the n objects. The distances can be conveniently represented in a symmetric $n \times n$ matrix S whose diagonal entries S_{ii} are all zero and the entries $S_{ij} = S_{ji}$ are the pairwise distances between objects i and j. The affinity matrix A is then obtained as a weighted negative exponential of S after which the diagonal entries A_{ii} are set to zero. The affinity matrix A is, therefore, a similarity matrix as required by all spectral clustering algorithms.

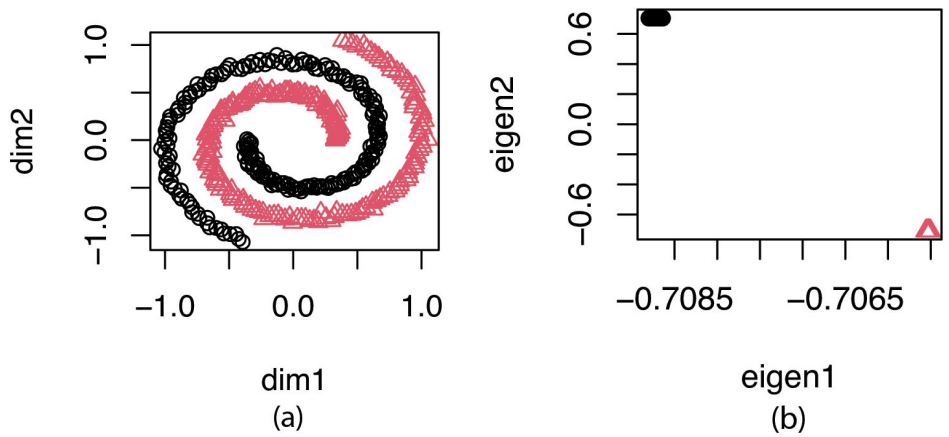

Figure 1. Simulated bivariate data set in the original space (**a**) and space obtained with the eigendecomposition of L (**b**), colors and shapes represent the partition obtained with spectral clustering.

The next matrix to construct is the diagonal matrix D with D_{ii} equal to the sum of the elements of row i of A. The graph Laplacian matrix L can then be calculated from A and D as $L = D^{-1/2}AD^{-1/2}$. Spectral decomposition of L gives its eigenvalues and corresponding

eigenvectors. The eigenvectors corresponding to the k largest eigenvalues are stacked as columns to form the $n \times k$ matrix Y'. Each row of Y' is re-normalized to unit length to give Y. The matrix Y is a real valued solution of the clustering problem, the number of columns of Y, k, is equal to the number of clusters. A simple clustering algorithm can now be used to detect the clusters. Figure 1 shows a data set in the original space and in the space obtained with the eigendecomposition of L where k-means clustering is performed, colors and shapes represent the partition. Commonly, Y and k are the inputs for k-means clustering [20]. The cluster assignments for the n rows of Y are the cluster assignments for the original corresponding n objects. Figure 2 shows examples of challenging data sets on which spectral clustering is able to detect the correct clustering structure. For more examples and comparisons, see [5].

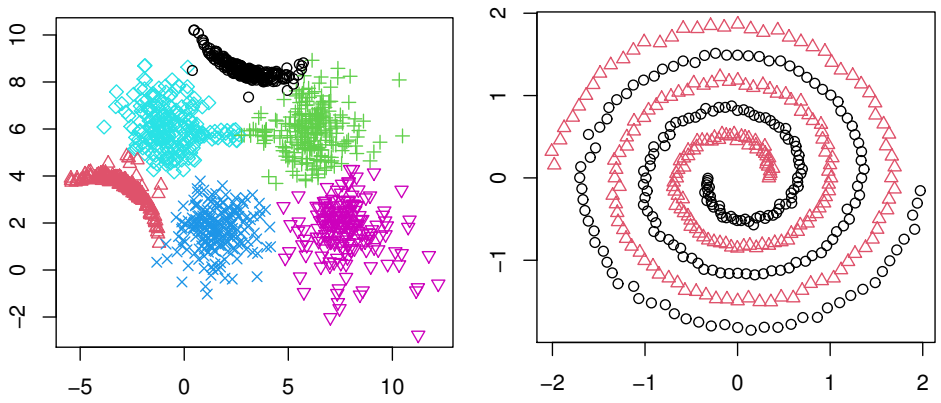

Figure 2. Simulated bivariate data sets, colors, and shapes represent the partition obtained with spectral clustering.

3. Spectral Clustering for Mixed Type Data

We extended the technique discussed in Section 2 for data of mixed-type. To apply spectral clustering we need to compute the affinity matrix A, whose entries are the pairwise similarities between the n objects. The original data matrix needs to be separated in two matrices: matrix Z containing all the continuous columns of X, and the matrix Y containing all the categorical columns of X. Both Z and Y have the same number n of rows as X. Binary variables are considered a special case of categorical variables.

From the matrix Z containing the continuous variables of X, the $n \times n$ data matrix $\hat{Z}$ is calculated, where the entries of $\hat{Z}$ are all the pairwise Euclidean distances between the n rows. The matrix $\hat{Z}$ is then scaled to be between 0 and 1 by dividing each entry by the largest entry of $\hat{Z}$.

From Y, the $n \times n$ data matrix $\hat{Y}$ is obtained, where the entries of $\hat{Y}$ are all the pairwise Hamming distances between the n rows of Y scaled to be between 0 and 1. An entry is 0 only if the two rows being compared take on the same value for all categorical variables. An entry is 1 only if the two rows being compared never take the same value for any categorical variable. When the two rows being compared take on the same value for some categorical variables but not others, the corresponding $\hat{Y}$ entry will be greater than 0 but less than 1.

Both $\hat{Z}$ and $\hat{Y}$ are, therefore, $n \times n$ matrices of pairwise dissimilarities with 0 on the diagonal and all other entries between 0 and 1. A single dissimilarity matrix J is then calculated as a simple weighted sum $J = (1 - w)\hat{Z} + w\hat{Y}$, with the weight w being a tunable parameter between 0 and 1. Having w equal 1 is equivalent to ignoring the contribution of the continuous variables and having w equal 0 is equivalent to ignoring the contribution of the categorical variables.

The affinity matrix A is then calculated from J as $A = \exp(-J^2/(2\sigma^2))$ where σ is a tunable parameter. Roughly speaking, smaller values of σ correspond to each object being connected to only closer nearest neighbors in the graph. Once σ is small enough, the objects

are no longer connected to each other. Conversely, increasing σ connects each object to further and further neighbors until eventually for large enough σ, each object is connected to all $n-1$ other objects. Notice that we started with a dissimilarity matrix J and ended up with a similarity matrix A because of the negative exponential.

The NJW algorithm then changes the diagonal of A from all 1 to all 0. The next step of our mixed-data spectral clustering algorithm then forms the graph Laplacian matrix L from A and D as $L = D^{-1/2}AD^{-1/2}$. Recall from Section 2 that D is the diagonal matrix with D_{ii} equal to the sum of the elements of row i of A.

We then calculate the k eigenvectors corresponding to the k largest eigenvalues of L, with k being the number of desired clusters. These eigenvectors are then stacked as columns to form the $n \times k$ matrix V'. Each row of V' is re-normalized to unit length to give V, which is then used as the input for k-means clustering. The cluster assignments for the n rows of V are the cluster assignments for the original corresponding n objects.

We implemented the spectral clustering algorithm for mixed-type data as a function in R [21]. The function takes as input the $n \times p$ data matrix X, where the rows are the n objects to be clustered and the p columns are the continuous and categorical attributes to be used for clustering; for instance n patients with p observations/measurements on them such as age, height, weight, gender, pre-existing conditions, etc.

The `daisy` command in the `cluster` package [22] is used to calculate the matrix $\hat{Y}$ whereas the function `dist` is used to compute $\hat{Z}$.

The `eigs_sym` function from the `RSpectra` package [23] is used for the spectral decomposition of L to give its k largest eigenvalues and the corresponding eigenvectors. The function `kmeans`, part of the `stats` package is used to perform k-means clustering. Values for the two free parameters w and σ are chosen by running the algorithm over a grid of w-σ values and picking the value pair that gives the highest ratio of between-to-within sum-of-squares at the k-means step. The tuning is obtained at the k-means step in the space obtained with an eigendecomposition of L; in this space, clusters are spherical, as shown in Figure 1, and, therefore, a ratio based on the between-to-within sum-of-squares is appropriate.

4. Application

4.1. Competitors

Several methods for clustering mixed-type data sets exist [13]. In our study, we compare spectral clustering of mixed-type data to two particular methods: k-prototypes [9,24] and KAMILA [4,10]. Those techniques are among the most commonly-used techniques for clustering mixed-type data and that have shown good performance [25]. The k-prototypes algorithm is an adaptation of the k-means algorithm to mixed-type data. The differences are that the 'distances' used for cluster assignments are no longer Euclidean since each object to be clustered has continuous as well as categorical attributes. Instead, a dissimilarity measure that is the weighted sum of the squared Euclidean distance for the continuous variables and the simple matching coefficient for the categorical variables is used. There is no universal method of determining the 'correct' value of the weights. The second difference is that the centroids (known as prototypes) are the means for the continuous variables data, and modes for categorical variables data. KAMILA is a hybrid of the non-parametric k-means clustering and the parametric model-based Gaussian-multinomial mixture models. It has the advantage of the user not having to specify the weight for the continuous variables with respect to the categorical variables, similar to Gaussian-multinomial mixture models.

4.2. Simulation Design

The simulation design is divided in two parts: two-cluster data sets and four-cluster data sets. For two-clusters, a full-factorial approach was used for the simulation. Data were simulated in two clusters looking at the effect of four factors on the performance of three mixed-type data clustering algorithms: k-prototypes, KAMILA and spectral clustering. The four factors are:

1. The degree of overlap in the variables: high–high, high–low, low–high and low–low (continuous-categorical);
2. The number of continuous-categorical variables: 2-2, 1-3, and 3-1;
3. The number of levels in the categorical variables: 3 and 5;
4. Whether or not the clusters were balanced (number of points per cluster): 200-200 vs. 320-80.

This gives 48 unique combinations of the four factors. For each combination, 100 data sets were generated.

Data were simulated in four clusters looking at the effect of two factors:

1. Cluster shape (convex vs. non-convex);
2. The degree of overlap in the variables.

All data sets had two continuous and two categorical variables. Two of the clusters had 320 data points each and the other two had 80 data points each.

The continuous variables were simulated from a multivariate normal distribution using the `rmvnorm` function of the `mvtnorm` package [26]. For each of the two clusters, this required p means and a $p \times p$ covariance matrix, where p is the number of continuous variables desired for the data set. The p means were sampled from random continuous uniform distributions. The `rcorrmatrix` function of the `clusterGeneration` package [27] was used to generate the two random $p \times p$ covariance matrices, one for each cluster. Overlap between clusters was varied by specifying the width of the random uniform from which the p means were sampled: selecting a narrow range resulted in higher overlap, whereas selecting a wide range resulted in low overlap.

The categorical variables were simulated independent of both the continuous variables and of each other. The starting point was two continuous unit uniform distributions—one for each cluster. Overlap between clusters was varied by specifying the width of the two unit uniforms: for instance zero overlap would have the two unit uniforms as (0, 1) and (1, 2), whereas complete overlap would have the two unit uniforms identical as (0, 1). Binning the continuous output of the unit uniforms into three or five buckets gave the desired three-level and five-level categorical variables.

For the four-cluster data sets, for the convex clusters, the continuous data were pseudo-randomly generated from a multivariate generalized hyperbolic distribution with $\omega = 1$, $\lambda = 0.5$, randomly varying the mean, the covariance matrix, and the skewness. The random means and random covariance matrices were generated as previously described and together with ω and λ, used as inputs to the `rGHD` function of the `MixGHD` package [28] to generate the continuous variables.

To obtain non-convex clusters, the continuous variables were generated using bivariate Von-Mises distributions varying the θ parameters among clusters. This is carried out using the `rmovMF` function in the `movMF` package [29].

For both the convex and non-convex clusters in the four-cluster data sets, the categorical variables were generated independent of each other and of the continuous variables. This was carried out by sampling from multinomial distributions with five levels, as all categorical variables were set to have five levels. Overlap between categorical variables was controlled by making all five levels equi-probable for maximum overlap, and making one level more probable in a given cluster than the others to reduce overlap. An example of continuous variables for the three scenarios with four clusters, i.e., non-convex, convex-low overlap, and convex-high overlap is displayed in Figure 3.

Clustering performance was measured using the adjusted Rand index (ARI) [30]. The ARI can be obtained using the `ARI` function of the `MixGHD` package [28]. The ARI corrects the Rand index [31] for chance; it has an expected value equal to zero under random classification, and is equal to one when there is a perfect class agreement. The ARI of the clustering output for the three algorithms was calculated along with the median of the results used to compare performance.

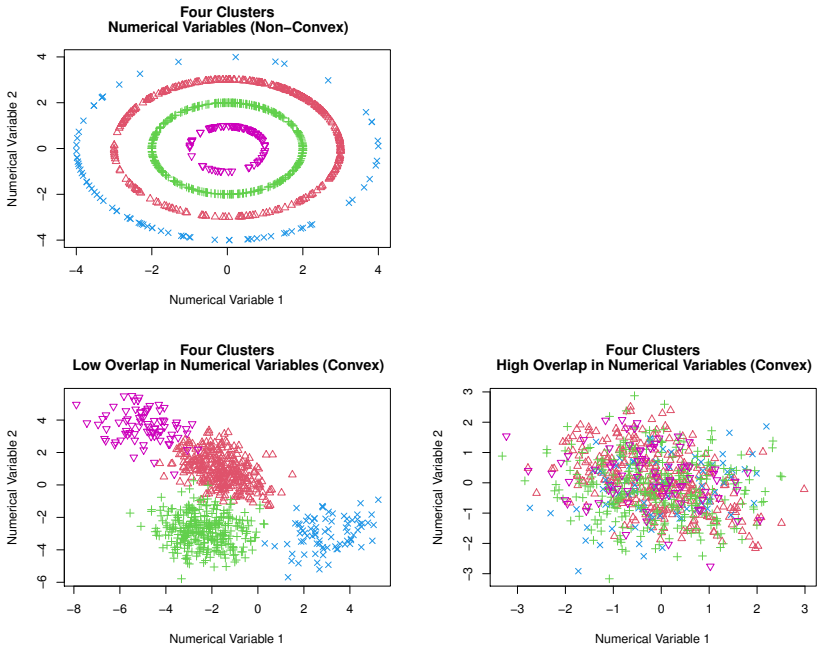

Figure 3. Example of continuous variables for the three scenarios with four clusters

4.3. Simulation Results

4.3.1. Results for 2-Cluster Data Sets

The results for the four overlap types described in Section 4.2 (low–low, low–high, high–low and high–high) are presented in Tables 1–4, respectively. For more detailed results, please see the Supplementary Material. The 'size' column refers to whether the 100 data sets had two clusters of equal size ('eq') 200 objects each, or non-equal size ('neq'), one cluster with 320 objects and the other with 80 objects. The 'con.' and 'cat.' columns refer to the number of continuous variables and categorical variables, respectively. The 'levs' column refers to the number of levels that the categorical variables had. The 'KAMILA', '*k*-proto' and 'spectral' columns are the median ARI for the 100 data sets clustered for the variable settings of the corresponding row. The 12 rows correspond to the 12 possible combinations of the levels for the three remaining factors under investigation, i.e., number of continuous-categorical variables, number of levels in the categorical variables, and whether or not the clusters were balanced.

Table 1. Median ARI for low overlap in continuous variables and low overlap in categorical variables, $n = 100$ two-cluster data sets.

Size	Con.	Cat.	Levs	KAMILA	*k*-Proto	Spectral
eq	1	3	3	0.980	0.970	0.980
eq	2	2	3	0.931	0.680	0.941
eq	3	1	3	0.359	0.341	0.680
eq	1	3	5	0.970	0.341	0.990
eq	2	2	5	0.921	0.285	0.926
eq	3	1	5	0.365	0.222	0.647
neq	1	3	3	0.108	0.966	0.988
neq	2	2	3	0.102	0.036	0.965
neq	3	1	3	0.082	0.026	0.524
neq	1	3	5	0.110	0.010	1.000
neq	2	2	5	0.078	0.018	0.965
neq	3	1	5	0.062	0.027	−0.018

Table 2. Median ARI for low overlap in continuous variables and high overlap in categorical variables, $n = 100$ two-cluster data sets.

Size	Con.	Cat.	Levs	KAMILA	*k*-Proto	Spectral
eq	1	3	3	0.931	0.921	0.951
eq	2	2	3	0.860	0.615	0.865
eq	3	1	3	0.358	0.335	0.503
eq	1	3	5	0.931	0.133	0.951
eq	2	2	5	0.765	0.160	0.823
eq	3	1	5	0.238	0.134	0.482
neq	1	3	3	0.108	0.880	0.977
neq	2	2	3	0.101	0.082	0.896
neq	3	1	3	0.047	0.018	0.291
neq	1	3	5	0.087	0.044	0.965
neq	2	2	5	0.042	0.022	0.908
neq	3	1	5	0.056	0.032	0.236

Table 3. Median ARI for high overlap in continuous variables and low overlap in categorical variables, $n = 100$ two-cluster data sets.

Size	Con.	Cat.	Levs	KAMILA	*k*-Proto	Spectral
eq	1	3	3	0.002	0.990	0.990
eq	2	2	3	0.016	0.898	0.931
eq	3	1	3	0.004	0.060	0.639
eq	1	3	5	0.002	0.194	0.990
eq	2	2	5	0.007	0.050	0.921
eq	3	1	5	0.002	0.008	0.301
neq	1	3	3	0.001	0.977	0.988
neq	2	2	3	0.003	−0.001	0.965
neq	3	1	3	0.000	−0.001	0.449
neq	1	3	5	0.000	−0.001	0.988
neq	2	2	5	−0.001	−0.001	0.954
neq	3	1	5	−0.002	−0.002	−0.024

Table 4. Median ARI for high overlap in continuous variables and high overlap in categorical variables, $n = 100$ two-cluster data sets.

Size	Con.	Cat.	Levs	KAMILA	*k*-Proto	Spectral
eq	1	3	3	0.001	0.941	0.941
eq	2	2	3	0.010	0.243	0.828
eq	3	1	3	0.002	0.040	0.499
eq	1	3	5	0.002	0.032	0.941
eq	2	2	5	0.004	0.005	0.828
eq	3	1	5	0.001	0.000	0.005
neq	1	3	3	0.001	0.921	0.965
neq	2	2	3	0.002	−0.001	0.908
neq	3	1	3	0.000	−0.001	0.265
neq	1	3	5	0.000	0.001	0.944
neq	2	2	5	0.003	−0.001	0.885
neq	3	1	5	−0.001	−0.001	0.007

Table 1 has the median ARI results for the data sets with low overlap in both continuous and categorical variables. Spectral clustering is the best performer for all settings except the last row where all three clustering methods perform poorly. KAMILA is the second-best performer for most settings with *k*-prototypes performing worst. Spectral clustering

performs similarly for balanced ('eq') and unbalanced ('neq') clusters. The exception is the last row where none of the three methods performs well. Spectral clustering especially outperforms for the data sets with three continuous variables and one categorical variable. For all three clustering methods, the ARI drops as the number of continuous-categorical variables goes from 1-3 to 2-2 to 3-1. Spectral clustering performs comparably whether the categorical variables have three or five levels (again with the exception of the last-row data sets).

Table 2 has the median ARI results for the data sets with low overlap in the continuous variables but high overlap in categorical variables. Spectral clustering is the best performer for all settings. Spectral clustering especially outperforms KAMILA and *k*-prototypes for the data sets with three continuous variables and one categorical variable. KAMILA is the second-best performer for most settings with *k*-prototypes performing worst. Spectral clustering performs similarly for balanced ('eq') and unbalanced ('neq') clusters. For all three clustering methods, the ARI tends to drop as the number of continuous-categorical variables goes from 1-3 to 2-2 to 3-1. Spectral clustering performs comparably whether the categorical variables have three or five levels.

Table 3 has the median ARI results for the data sets with high overlap in the continuous variables but low overlap in categorical variables. Spectral clustering is the best performer for all settings except the last row where all three clustering methods perform poorly. For nine of the eleven settings where it is the best performer, the out-performance is much larger than previously seen for the low–low and low–high overlap settings. Also in contrast to observations in Tables 1 and 2, KAMILA tends to perform worse than *k*-prototypes for these data sets. Spectral clustering performs similarly for balanced ('eq') and unbalanced ('neq') clusters. The exception is the data sets with three continuous variables and one categorical variable where performance is better for the data sets with balanced clusters. Similar to low–low and low–high overlap data sets, the ARI tends to drop as the number of continuous-categorical variables goes from 1-3 to 2-2 to 3-1. Spectral clustering performs comparably whether the categorical variables have three or five levels, with the exception of the data sets with three continuous variables and one categorical variable.

Table 4 has the median ARI results for the data sets with high overlap in both the continuous and the categorical variables. Spectral clustering is the best performer for all settings, in most cases outperforming KAMILA and *k*-prototypes by large margins. For nine of the eleven settings where it is the best performer, the out-performance is much larger than previously seen for the low–low and low–high overlap settings. Similar to Table 3, KAMILA tends to perform worse than *k*-prototypes for these data sets. Spectral clustering performs similarly for balanced ('eq') and unbalanced ('neq') clusters. As seen in previous tables, the ARI tends to drop as the number of continuous-categorical variables goes from 1-3 to 2-2 to 3-1. Spectral clustering performs comparably whether the categorical variables have three or five levels, again with the exception of the data sets with three continuous variables and one categorical variable where performance is better with three levels than five.

4.3.2. Results for Four-Cluster Data Sets

For the four-cluster data sets, when the clusters are non-convex in the continuous variables (Table 5), spectral clustering outperforms both *k*-prototypes and KAMILA by a large margin when the degree of overlap is either high or low for the categorical variables. When the overlap is medium, the three methods all perform poorly.

For four-cluster data sets where the clusters are convex in the continuous variables (Table 6), spectral clustering performs comparably to both *k*-prototypes and KAMILA when continuous-categorical overlap is either low–high or high–low. When overlap is medium–medium, it performs worse than both methods.

Table 5. Median ARI for different levels of overlap in categorical variables, $n = 10$ four-cluster data sets with non-convex clusters for continuous variables.

Categorical	*k*-Prototypes	KAMILA	Spectral
high	0.017	0.020	0.864
medium	0.098	0.025	0.073
low	0.508	0.151	0.896

Table 6. Median ARI for different levels of overlap in continuous and categorical variables, $n = 10$ four-cluster data sets with convex clusters for continuous variables.

Continuous	Categorical	*k*-Prototypes	KAMILA	Spectral
low	high	0.216	0.330	0.389
medium	medium	0.450	0.251	0.208
high	low	0.720	0.359	0.778

4.4. Real Data—Diamonds Data Set

The three clustering algorithms (*k*-prototypes, KAMILA, and spectral clustering) were tested on the diamonds data set freely available on Kaggle [32]. The data set contains the prices and other attributes of almost 54,000 diamonds. We used seven of the nine variables for the cluster analysis. The categorical ones are cut (quality of the cut: Fair, Good, Very Good, Premium, Ideal), color, and clarity. The continuous variables are table (width of top of diamond relative to widest point (43–95)), carat (weight of the diamond (0.2–5.01), x length in mm (0–10.74), and depth (total depth percentage = $z/mean(x, y) = 2z/(x + y)(43 - 79)$). The price variable was not used in clustering but was used to evaluate the clustering by converting it into a categorical variable with five levels: $\leq$1000, 1000–2500, 2500–5000, 5000–10,000, and $>$10,000.

Five random samples were used to select 800 of the 53,940 instances for clustering by the three algorithms. The median resulting ARI for each method is 0.337 for *k*-prototypes, 0.496 for KAMILA, and 0.481 for spectral clustering. Spectral clustering outperforms *k*-prototypes and performs comparably to KAMILA. It did, however, take significantly more time to run spectral clustering than KAMILA (about 10 minutes vs. a few seconds). Moreover, the best possible ARI with spectral clustering is obtained using a weight for categorical variables different from the one chosen by our automatic tuning. The best median ARI is 0.520, making spectral clustering possibly the best performer. This emphasizes the potential of the method and the need for more research in improving the tuning technique and in speeding up the algorithm.

5. Conclusions

This paper extends spectral clustering to mixed-type data by replacing the Euclidean-based similarity distance used in conventional spectral clustering with different dissimilarity measures for continuous and categorical variables. A global dissimilarity measure is than computed using a weighted sum, and a Gaussian kernel is used to convert the dissimilarity matrix into a similarity matrix. The new method includes an automatic tuning of the variable weight and kernel parameter. The method is compared with the *k*-prototype and KAMILA, two state-of-art clustering techniques for mixed-type data.

For two-cluster simulated data sets, spectral clustering performs better or comparably to *k*-prototypes and KAMILA based on ARI. Spectral clustering especially outperforms when there is high overlap in both continuous and categorical variables

For four-cluster simulated data sets, spectral clustering performs comparably to *k*-prototypes and KAMILA when: clusters are convex in the continuous variables, or when continuous-categorical overlap is either low–high or high–low. When overlap is medium-medium (convex), spectral clustering under-performs *k*-prototypes and KAMILA. Spectral clustering outperforms *k*-prototypes and KAMILA when: clusters are non-convex in the

continuous variables, or when degree of overlap is either high or low for the categorical variables. When overlap is medium (non-convex), all three methods perform poorly.

Tested on the diamonds data set, spectral clustering outperforms k-prototypes and performs comparably to KAMILA. It did, however, take significantly more time to run spectral clustering than KAMILA emphasizing the need for computational improvements. The reduced speed of Spectral clustering is partially due to the grid search for the tuning parameters, w and σ. Future research will focus on the study of the effect of reducing the grid size to improve the algorithm running time. However, the main bottleneck is the decomposition of the $n \times n$ graph Laplacian. Our implementation of mixed-type data spectral clustering using the `RSpectra` package [23] for the decomposition is a significant improvement of the native R decomposition using the `eigen` command. For instance, for a data set of 3200 objects and three clusters, the run-time is reduced by a factor of 150. A further improvement to be investigated could be multi-threading the decomposition.

Future research will also include extending the current algorithm for ordinal and binary data. This can be achieved including specific distances for those kind of variables and an extra set of weights.

Supplementary Materials: The following are available at https://www.mdpi.com/article/10.3390/stats5010001/s1.

Author Contributions: Conceptualization, C.T. and F.M.; methodology, C.T. and F.M.; Software, F.M.; writing, C.T. and F.M.; supervision, C.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: This paper did not involve human subjects or animals.

Informed Consent Statement: Not applicable.

Data Availability Statement: The Diamonds data set is freely available on Kaggle, the code used for the simulations will be released upon request.

Acknowledgments: The authors want to thank the two anonymous reviewers.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. McLachlan, G.J.; Basford, K.E. *Mixture Models: Inference and Applications to Clustering*; Marcel Dekker Inc.: New York, NY, USA, 1988; p. 253.
2. MacQueen, J. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium, Berkeley, CA, USA, 21 June–28July 1965; Volume 1, pp. 281–297.
3. Schubert, E.; Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst. (TODS)* **2017**, *42*, 1–21. [CrossRef]
4. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference of Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; Volume 96, pp. 226–231.
5. Murugesan, N.; Cho, I.; Tortora, C. Benchmarking in Cluster Analysis: A Study on Spectral Clustering, DBSCAN, and K-Means. In Proceedings of the Conference of the International Federation of Classification Societies, Thesssaloniki, Greece, 26–29 August 2019; Springer: Cham, Switzerland, 2019, pp. 175–185.
6. David, G.; Averbuch, A. SpectralCAT: Categorical spectral clustering of numerical and nominal data. *Pattern Recognit.* **2012**, *45*, 416–433. [CrossRef]
7. Gower, J.C. A general coefficient of similarity and some of its properties. *Biometrics* **1971**, *27*, 857–871. [CrossRef]
8. Foss, A.H.; Markatou, M.; Ray, B. Distance metrics and clustering methods for mixed-type data. *Int. Stat. Rev.* **2019**, *87*, 80–109. [CrossRef]
9. Huang, Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov.* **1998**, *2*, 283–304. [CrossRef]
10. Foss, A.H.; Markatou, M. KAMILA: Clustering mixed-type data in R and Hadoop. *J. Stat. Softw.* **2018**, *83*, 1–44. [CrossRef]
11. McParland, D.; Gormley, I.C. Model based clustering for mixed data: ClustMD. *Adv. Data Anal. Classif.* **2016**, *10*, 155–169. [CrossRef]
12. Hunt, L.; Jorgensen, M. Clustering mixed data. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2011**, *1*, 352–361. [CrossRef]

13. Ahmad, A.; Khan, S.S. Survey of state-of-the-art mixed data clustering algorithms. *IEEE Access* **2019**, *7*, 31883–31902. [CrossRef]
14. van de Velden, M.; Iodice D'Enza, A.; Markos, A. Distance-based clustering of mixed data. *Wiley Interdiscip. Rev. Comput. Stat.* **2019**, *11*, e1456. [CrossRef]
15. Von Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416. [CrossRef]
16. Hagen, L.; Kahng, A.B. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **1992**, *11*, 1074–1085. [CrossRef]
17. Shi, J.; Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.
18. Priebe, C.E.; Park, Y.; Vogelstein, J.T.; Conroy, J.M.; Lyzinski, V.; Tang, M.; Athreya, A.; Cape, J.; Bridgeford, E. On a two-truths phenomenon in spectral graph clustering. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 5995–6000. [CrossRef] [PubMed]
19. Ng, A.Y.; Jordan, M.I.; Weiss, Y. On Spectral Clustering: Analysis and an Algorithm. In Advances in Neural Information Processing Systems. 2002. Available online: https://www.bibsonomy.org/bibtex/2e9c06dab81a9a2e06123cd9b31d3d83f/mhwombat (accessed on 9 October 2021).
20. Hartigan, J.A.; Wong, M.A. A *K*-means clustering algorithm. *Appl. Stat.* **1979**, *28*, 100–108. [CrossRef]
21. R Core Team. *R: A Language and Environment for Statistical Computing*; R Foundation for Statistical Computing: Vienna, Austria, 2019.
22. Maechler, M.; Rousseeuw, P.; Struyf, A.; Hubert, M.; Hornik, K. *Cluster: Cluster Analysis Basics and Extensions*; R Package Version 2.1.0; MIT Press: Cambridge, MA, USA, 2019.
23. Qiu, Y.; Mei, J. RSpectra: Solvers for Large-Scale Eigenvalue and SVD Problems; R Package Version 0.16-0. 2019. Available online: https://cran.r-project.org/package=RSpectra (accessed on 9 October 2021).
24. Szepannek, G. clustMixType: User-Friendly Clustering of Mixed-Type Data in R. *R J.* **2018**, *42*, 200–208. [CrossRef]
25. Jimeno, J.; Roy, M.; Tortora, C. Clustering Mixed-Type Data: A Benchmark Study on KAMILA and K-Prototypes. In Proceedings of the Conference of the International Federation of Classification Societies, Thessaloniki, Greece, 26–29 August 2019; Springer: Cham, Switzerland, 2019; pp. 83–91.
26. Genz, A.; Bretz, F.; Miwa, T.; Mi, X.; Leisch, F.; Scheipl, F.; Hothorn, T. Mvtnorm: Multivariate Normal and t Distributions; R Package Version 1.1-1. 2020. Available online: https://mran.microsoft.com/snapshot/2017-02-04/web/packages/mvtnorm/index.html (accessed on 9 October 2021).
27. Qiu, W.; Joe, H. ClusterGeneration: Random Cluster Generation (with Specified Degree of Separation); R Package Version 1.3.4. 2015. Available online: https://cran.r-project.org/web/packages/clusterGeneration/index.html (accessed on 9 October 2021).
28. Tortora, C.; Browne, R.P.; ElSherbiny, A.; Franczak, B.C.; McNicholas, P.D. Model-Based Clustering, Classification, and Discriminant Analysis Using the Generalized Hyperbolic Distribution: MixGHD R package. R package version 2.3.3. *J. Stat. Softw.* **2021**, *98*, 1–24. [CrossRef]
29. Hornik, K.; Grün, B. movMF: An R Package for Fitting Mixtures of von Mises-Fisher Distributions. R package version 0.2.4. *J. Stat. Softw.* **2014**, *58*, 1–31. [CrossRef]
30. Hubert, L.; Arabie, P. Comparing partitions. *J. Classif.* **1985**, *2*, 193–218. [CrossRef]
31. Rand, W.M. Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850. [CrossRef]
32. Agrawal, S. Diamonds Data Set. Kaggle. 2017. Available online: https://www.kaggle.com/shivam2503/diamonds (accessed on 9 April 2021).