applied sciences

MDPI

Article

AutoElbow: An Automatic Elbow Detection Method for Estimating the Number of Clusters in a Dataset

Adeiza James Onumanyi [1,*], Daisy Nkele Molokomme [1], Sherrin John Isaac [1] and Adnan M. Abu-Mahfouz [1,2]

[1] Next Generation Enterprises and Institutions, Council for Scientific and Industrial Research, Pretoria 0001, South Africa; dmolokomme@csir.co.za (D.N.M.); sisaac@csir.co.za (S.J.I.); a.abumahfouz@ieee.org (A.M.A.-M.)

[2] Department of Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg 2006, South Africa

* Correspondence: aonumanyi@csir.co.za

Abstract: The elbow technique is a well-known method for estimating the number of clusters required as a starting parameter in the K-means algorithm and certain other unsupervised machine-learning algorithms. However, due to the graphical output nature of the method, human assessment is necessary to determine the location of the elbow and, consequently, the number of data clusters. This article presents a simple method for estimating the elbow point, thus, enabling the K-means algorithm to be readily automated. First, the elbow-based graph is normalized using the graph's minimum and maximum values along the ordinate and abscissa coordinates. Then, the distance between each point on the graph to the minimum (i.e., the origin) and maximum reference points, and the "heel" of the graph are calculated. The estimated elbow location is, thus, the point that maximizes the ratio of these distances, which corresponds to an approximate number of clusters in the dataset. We demonstrate that the strategy is effective, stable, and adaptable over different types of datasets characterized by small and large clusters, different cluster shapes, high dimensionality, and unbalanced distributions. We provide the clustering community with a description of the method and present comparative results against other well-known methods in the prior state of the art.

Keywords: automatic; clustering; elbow method; K-means; unsupervised

Citation: Onumanyi, A.J.; Molokomme, D.N; Isaac, S.J.; Abu-Mahfouz, A.M. AutoElbow: An Automatic Elbow Detection Method for Estimating the Number of Clusters in a Dataset. *Appl. Sci.* **2022**, *12*, 7515. https://doi.org/10.3390/app12157515

Academic Editor: Subhas Mukhopadhyay

Received: 1 July 2022
Accepted: 24 July 2022
Published: 26 July 2022

1. Introduction

Clustering is an important activity in many application areas, including image processing [1], smart grids [2], life sciences [3], molecular analysis [4], market and customer segmentation [5], pattern recognition [6], social network analysis [7], to mention a few. It entails the process of discovering groupings in data without prior training and is therefore classified as an unsupervised machine-learning approach. Estimating the appropriate number of clusters in any specified dataset is often the primary challenge in cluster analysis. This is due to the fact that many clustering methods, particularly the well-known K-means algorithm, require prior information about the appropriate number of clusters as a starting parameter. Consequently, the estimation of the ideal number of clusters in a given dataset has spawned an entire field of research devoted to the development of various approaches for resolving this challenge.

In this regard, there are a variety of well-known approaches for determining the ideal number of clusters in a dataset. The silhouette and elbow techniques are two noteworthy instances of direct approaches, while the gap statistic approach comes across as a statistical testing method. The silhouette technique assesses the quality of a clustering outcome by calculating the average silhouette of the data for varying k values [8]. The ideal number of clusters, k^*, is one that maximizes the average silhouette across a range of feasible k values.

A large average silhouette width is indicative of a dense population. Although successful and having a score confined between -1 (incorrect clustering) and +1 (very dense clustering),

the silhouette method produces a coefficient that is typically greater for convex clusters than other cluster concepts, thus, making it unreliable in such instances. In contrast, the gap statistic compares the total within-cluster variation for different values of k to their anticipated values under a null reference distribution of the data (normally the uniform distribution) [9]. The ideal cluster estimate will, thus, be the value that maximizes the gap statistic. This indicates that the clustering structure is significantly distinct from the uniform random distribution of points. The fact that the gap statistic is not always a convex or monotonic function makes it more difficult to determine the best point. In addition, as a "relative" statistic, it may not determine whether or not there are meaningful clusters in the data.

The elbow, or "knee of a curve," approach is the most common and simplest means of determining the appropriate cluster number prior to running clustering algorithms, suc has the K-means algorithm. The elbow method entails running the clustering algorithm (often the K-means algorithm) on the dataset repeatedly across a range of k values, i.e., $k = 1, 2, \ldots, K$, where K is the total number of clusters to be iterated. For each value of k, the average score for all clusters is calculated.

This score, called the within-cluster dispersion (WCD) is typically calculated by adding the squared distance between each point and its related centroid. We note that, apart from the WCD, there are other metrics that may be used to assess the number of clusters in a dataset, such as the Calinski–Harabasz score, Davies–Bouldin, and Dunn value [10]. The best k value, i.e., k^*, may then be discovered by graphing any of these scores against each cluster number; k^* is often located where the graph resembles an arm (i.e., the "elbow"—the point of inflection of the curve) as will be shown in Section 3.

The primary problem with the elbow approach is that it is interpretation is dependent on the visual analysis of an elbow-based graph, which necessitates manual human inspection in order to determine the elbow point and, as a result, can produce very subjective results. Furthermore, when the approach produces a graph with a smooth curve, then estimating the value of k^* becomes difficult since the actual elbow point becomes harder to distinguish. Most notably, the outcome of the elbow graph is typically a function of the number of K iterations.

Consequently, the elbow point would often vary dependent on the value of K, a circumstance for which many approaches are not readily adaptable. Nonetheless, there are a number of notable methods for automating the identification of the elbow point in the pursuit of building automated K-means or other K-dependent clustering algorithms, such as the elbow strength approach (ESA) [11], knee-point algorithm (KPA) [12], Kneedle algorithm [13], slope changing ratio method (SCRM) [14], and the angle-based discriminant method (ADM) [15].

A closer examination reveals that both the KPA and the SCRM work upon an identical concept of computing the intersection point that minimizes the error point of both best lines of fit. They are both similarly comparable to the L-method of Salvador et al. [16]. The ADM technique, on the other hand, is considered to be comparable to the angle-based approach of Zhao et al. [17].

While the aforementioned methods have yielded satisfactory results in a variety of application areas; nevertheless, they can be limited in situations where an elbow-based graph returns as a smooth curve [15]. Furthermore, a few of their algorithmic procedures, such as in the KPA and its variants, can be computationally costly. In addition, several of these methods, such as the KPA, ADM, and ESA methods, require a minimum of three graph points in order to estimate the elbow point, which limits their use in the case of sparse graphs.

Consequently, in this article, we present a simple yet effective method for automatically estimating the elbow point and, as a result, the approximate number of clusters in the dataset. The proposed approach yields a convex function based on the ratio of the squared Euclidean distance between each point on the graph and three reference points, namely the origin (minimum), the furthest (maximum) point, and the "heel" of the normalized graph.

The maximum value of this function inherently returns the optimum elbow point, and consequently the approximate number of clusters in the dataset. We note that the

current method does not inform users whether or not the K-means algorithm is the best tool to use for clustering. Instead, it only helps to automate the elbow method towards determining the appropriate number of clusters in the dataset to be clustered. As a result, we present clustering researchers with a description of the suggested approach as well as the pseudocode that implements the solution.

Therefore, the contributions of the present article are highlighted as follows:

1. Based on the elbow graph, we propose a new method for automatically determining the potential number of clusters in a data set. The AutoElbow method provides an accurate estimation of the elbow for different K values and is stable compared to other approaches across different datasets having small and large clusters, different cluster shapes, high dimensionality, and unbalanced distributions.
2. The proposed method can be used to easily automate the K-means algorithm or any other unsupervised clustering algorithm that requires the selection of K beforehand.

The rest of the article is structured as follows: Section 2 presents a summary of the related work, Section 3 details the AutoElbow method and its corresponding pseudocodes, Section 4 presents and discusses the results obtained, and our conclusions are drawn in Section 5.

2. Related Work

In this section, we discuss notable methods from the prior art for automatically determining the ideal elbow point of an elbow-based graph. We classify these approaches according to the properties of the elbow-based graph from which they were derived, namely the line fitting methods [12,14,16], the angle-based [15,17], derivative-based [11], and the distance-based methods [13].

The L-method is an early line fitting method that was proposed in [16] to detect the elbow-point automatically. It works by fitting two best lines to the graph points on either side of the point of interest. Then, the best lines that produce the smallest root mean square error (RMSE) are fitted to the graph, and the point where the two lines intersect is deemed to be the optimal knee point. On a wide variety of datasets, the L-method outperformed the gap statistics and permutation tests approaches.

Additionally, the L-method has inspired other variations, such as the MATLAB implementation version in [12] and the modified method in [14]. In particular, the research in Diao et al. [14] applied the method of line fitting for elbow point detection to the capacity fade curves of lithium-ion cells. All of these implementations have demonstrated the line-fitting method's efficacy and its capacity to improve performance in various application domains.

The angle-based technique of Zhao et al. [17] computes the Bayesian Information Criterion (BIC) measure of any clustering algorithm under consideration and then computes the first order difference of the BIC measure. The point with the greatest BIC angle is estimated to be the knee point. By conducting experiments with the K-means algorithm, the authors demonstrated that their technique outperformed other existing index metrics as well as using the BIC measure alone.

Similarly, the authors of the more recent article in [15] developed an angle-based discriminant technique. Unlike the method of Zhao et al. [17], the discriminant method computes the angles created by each point of the elbow-based graph instead of the BIC graph. Multiple datasets were then used to demonstrate that the discriminant method outperformed the gap statistics approach.

On the other hand, due to the variances that may occur along the elbow-based graph, derivative-based approaches are not widely utilized in the literature. Nonetheless, one notable example was proposed in [11] using an elbow strength algorithm (ESA). Here, the ESA determines the elbow point by computing the difference between the first- and second-order differences of the elbow-based graph.

The algorithm then considers the point with the largest difference (referred to as the elbow strength) to be the ideal location of the elbow. In contrast, the Kneedle algorithm proposed in [13] was used to implement a distance-based strategy. In this instance, the

Kneedle algorithm computes the squared Euclidean distance between each point of the elbow-based graph and a straight line that encloses the graph's endpoints. The point with the largest distance is considered to be the ideal elbow point.

Despite the fact that the approaches described above have yielded significant results in a variety of application areas, there is always room for improvement or a better approach. Improved approaches are needed, for example, when the elbow graph returns as a smooth and extremely curved graph, when the graph is significantly perturbed, or when the tail of the graph is either too short or too long. These are some of the challenging circumstances that prompted our proposed approach, which we will describe in the next section.

3. AutoElbow Method

This section presents the basic concept of the proposed method including a pseudocode of the algorithm. Figure 1 depicts a typical elbow-based graph with a smooth curve, and our objective is to find the "elbow" or "knee" point P of the graph. To achieve this objective, we first define the elbow-based graph as $G \in \mathbb{R}^2$, which is represented as $G = [g_1, g_2, \ldots, g_k, \ldots, g_K]^T$, where $g_k = (x_k, y_k)$ is a single point along G with coordinates (x, y), and G is a collection of these points describing the graph. Here, x represents the abscissa denoting the total number of clusters used to execute the clustering method (i.e., the K-means algorithm), whereas y corresponds to the evaluation measure plotted along the ordinate.

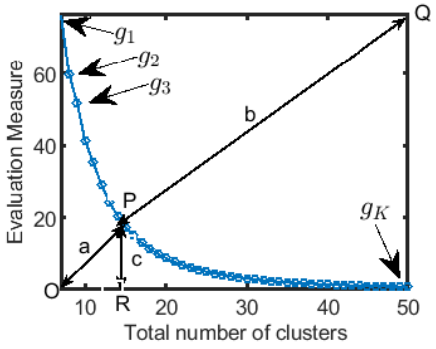


Figure 1. Representation of the parameters of the AutoElbow method.

3.1. Description of the Algorithm

In the AutoElbow method, we adopt the total WSD (TWSD) as the evaluation measure, which is obtained as the sum of the squared Euclidean distances between each point in a cluster and its representative centroid. After each run of the K-means algorithm, the value of TWSD is computed for different cluster numbers $k = 1, 2, \ldots, K$, where K is the maximum number of clusters anticipated in the dataset. The method for determining the TWSD using K-means is described in [15]. Next, we present a detailed explanation of how the AutoElbow technique operates:

1. Let the input data to be clustered be $Z = \{z_i\}_{i=1}^{N}$, where N is the total number of data points in $Z \in \mathbb{R}^D$, and D is dimension of Z.
2. The K-means algorithm is executed repeatedly for different cluster values k ranging from 1 to K. The K-means algorithm can be implemented as described in [18].
3. The TWSD, i.e., y for K clusters is computed as

$$y_k = \sum_{k=1}^{K} \sum_{i=1}^{N_k} ||z_i^{(k)} - c_k||^2, \tag{1}$$

for $k = 1, 2, \ldots, K$, where k represents a single cluster, K is the maximum number of clusters to be iterated upon, i is the index of a single data point z_i within a single cluster k, N_k is the total number of data points in cluster k and c_k represents each cluster's centroid.

4. The elbow-based graph G is obtained as $G = \{(x_k, y_k)\}_{k=1}^{K}$, where $\{x_1 = 1, x_2 = 2, \ldots, x_k = K\}$ are the actual total number of clusters considered in the iteration process.
5. The graph G is then inputted to the AutoElbow algorithm, which proceeds to compute the elbow or knee point P (i.e., g_{k^*}), where k^* is the optimal cluster number, as follows:
   a. First, G is normalized to the interval [0,1]. This is done to ensure that the computed squared Euclidean distances between graph points are same in magnitude. Therefore, the normalized values are denoted as

$$G^{norm} = \{(x_k^{norm}, y_k^{norm})\}_{k=1}^{K}. \tag{2}$$

   where

$$x_k^{norm} = \frac{x_k - \min\{x_k\}_{k=1}^{K}}{\max\{x_k\}_{k=1}^{K} - \min\{x_k\}_{k=1}^{K}}, \text{for } k = 1, 2, \ldots, K \tag{3}$$

$$y_k^{norm} = \frac{y_k - \min\{y_k\}_{k=1}^{K}}{\max\{y_k\}_{k=1}^{K} - \min\{y_k\}_{k=1}^{K}}, \text{for } k = 1, 2, \ldots, K. \tag{4}$$

   b. As illustrated in Figure 2, the algorithm then checks for the concavity structure of the graph G in order to identify the proper locations of O, R, and Q. Figure 2 depicts the relative positions of O and Q for each of the four forms of concavity, whereas R is determined as the equivalent x_k^{norm} value per graph point along the abscissa. The AutoElbow algorithm is designed to identify these distinct graph topologies, thus, making it applicable to both elbow- and knee-based graphs. We introduce a simple test for the concavity form of G^{norm} as follows:
      (i) First, we compute the slope of the straight line L that connects the first and last element in G^{norm} as

$$m = \frac{y_1^{norm} - y_K^{norm}}{x_1^{norm} - x_K^{norm}} \tag{5}$$

      where $x_1^{norm}, x_K^{norm}, y_1^{norm}$, and y_K^{norm} are the first and last elements in x^{norm} and y^{norm}, respectively.
      (ii) Then, a check point $l_{K/2}$ is computed at the mid-point of L (i.e., at $K/2$) using

$$l_{K/2} = y_1^{norm} - m \times (x_1^{norm} - x_2^{norm}) \tag{6}$$

      The midpoint (i.e., $K/2$) was selected as the checkpoint for the concavity test since it is the point furthest from the boundary line L. This choice ensures that the concavity test detects no misleading variations at the edges of the elbow- or knee-based graph G.
      (iii) Consequently, the concavity test $\mathcal{C}$ of G^{norm} is established as follows

$$\mathcal{C} = \begin{cases} 1 \text{ (i.e., Left elbow)} & \text{if } m < 0 \text{ and } l_2 - y_2^{norm} > 0 \\ 2 \text{ (i.e., Right elbow)} & \text{if } m > 0 \text{ and } l_2 - y_2^{norm} > 0 \\ 3 \text{ (i.e., Left knee)} & \text{if } m > 0 \text{ and } l_2 - y_2^{norm} < 0 \\ 4 \text{ (i.e., Right knee)} & \text{if } m < 0 \text{ and } l_2 - y_2^{norm} < 0 \end{cases} \tag{7}$$

   c. In this article, the TWSD is used as the evaluation metric, and thus the graph G is inevitably a left elbow-based graph, i.e., concave up with a decreasing slope. This is due to the fact that the TWSD decreases as the number of clusters increases [10]. Thus, under this condition, the AutoElbow algorithm proceeds by ensuring that the slope of the normalized graph is kept continuously decreasing via a graph cleaning process. This is done in order to ensure that all misleading variations are smoothed out from G as follows:

(a) For either a right knee- or left elbow-based graph, smoothing is achieved via

$$y_k^{norm} = \begin{cases} y_{k+1}^{norm} & \text{if } y_k^{norm} > y_{k-1}^{norm} \\ y_k^{norm} & \text{if } y_k^{norm} \leq y_{k-1}^{norm} \end{cases} \tag{8}$$

for $k = 1, 2, \ldots, K$.

(b) For either left knee- or right elbow-based graph, smoothing is achieved via

$$y_k^{norm} = \begin{cases} y_{k+1}^{norm} & \text{if } y_k^{norm} < y_{k-1}^{norm} \\ y_k^{norm} & \text{if } y_k^{norm} \geq y_{k-1}^{norm} \end{cases} \tag{9}$$

for $k = 1, 2, \ldots, K$.

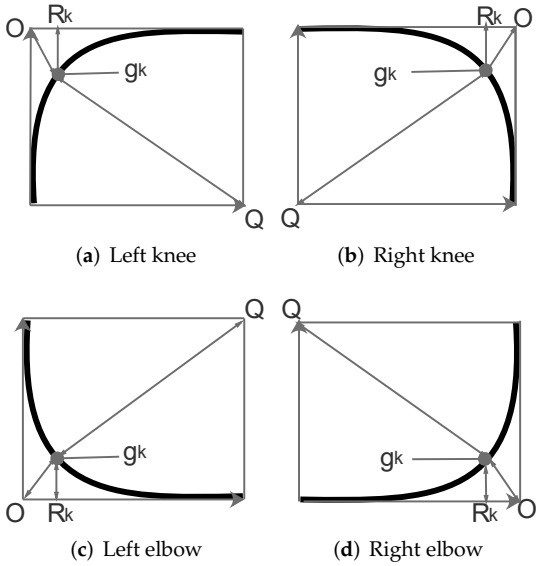


Figure 2. Types of concave graphs corresponding to either elbow- or knee-based graphs: (**a**) Left knee: Concave down with increasing slope, (**b**) Right knee: Concave down with decreasing slope, (**c**) Left elbow: Concave up with decreasing slope, and (**d**) Right elbow: Concave up with increasing slope.

6. Then, by considering both Figures 1 and 2, we propose the AutoElbow function f as

$$f_k = \frac{b_k}{a_k + c_k}, \text{for } k = 1, 2, \ldots, K \tag{10}$$

where b_k is the squared Euclidean distance between any point g_k on the graph and Q (which is the maximum reference distance), a_k represents the squared Euclidean distance between point O (minimum reference distance) and g_k along the graph, and c_k represents the squared Euclidean distance between points g_k and R_k. Furthermore, the coordinates of O, Q, and R for computing (10) are defined as follows:

(a) For a left knee-based graph (i.e., Figure 2a): The coordinates for O, Q, and R are given as $O = (0, 1)$, $Q = (1, 0)$, whereas $R = (x_k^{norm}, 1)$, and the squared Euclidean distances a_k, b_k, and c_k are computed as

$$a_k = (x_k^{norm})^2 + (y_k^{norm} - 1)^2, \text{for } k = 1, 2, \ldots, K \tag{11}$$

$$b_k = (x_k^{norm} - 1)^2 + (y_k^{norm})^2, \text{for } k = 1, 2, \ldots, K \tag{12}$$

$$c_k = (y_k^{norm} - 1)^2, \text{for } k = 1, 2, \ldots, K \tag{13}$$

(b) For a right knee-based graph (i.e., Figure 2b): $O = (1,1)$, $Q = (0,0)$, whereas $R = (x_k^{norm}, 1)$, and the squared Euclidean distances a_k, b_k, and c_k are computed as

$$a_k = (x_k^{norm} - 1)^2 + (y_k^{norm} - 1)^2, \text{for } k = 1, 2, \ldots, K \quad (14)$$

$$b_k = (x_k^{norm})^2 + (y_k^{norm})^2, \text{for } k = 1, 2, \ldots, K \quad (15)$$

$$c_k = (y_k^{norm} - 1)^2, \text{for } k = 1, 2, \ldots, K \quad (16)$$

(c) For a left elbow-based graph (i.e., Figure 2c): $O = (0,0)$, $Q = (1,1)$, whereas $R = (x_k^{norm}, 0)$, and the squared Euclidean distances a_k, b_k, and c_k are computed as

$$a_k = (x_k^{norm})^2 + (y_k^{norm})^2, \text{for } k = 1, 2, \ldots, K \quad (17)$$

$$b_k = (x_k^{norm} - 1)^2 + (y_k^{norm} - 1)^2, \text{for } k = 1, 2, \ldots, K \quad (18)$$

$$c_k = (y_k^{norm})^2, \text{for } k = 1, 2, \ldots, K \quad (19)$$

(d) For a right elbow-based graph (i.e., Figure 2d): $O = (1,0)$, $Q = (0,1)$, whereas $R = (x_k^{norm}, 0)$, and the squared Euclidean distances a_k, b_k, and c_k are computed as

$$a_k = (x_k^{norm} - 1)^2 + (y_k^{norm})^2, \text{for } k = 1, 2, \ldots, K \quad (20)$$

$$b_k = (x_k^{norm})^2 + (y_k^{norm} - 1)^2, \text{for } k = 1, 2, \ldots, K \quad (21)$$

$$c_k = (y_k^{norm})^2, \text{for } k = 1, 2, \ldots, K \quad (22)$$

7. After computing a_k, b_k, and c_k for any concavity form $\mathcal{C}$ of G^{norm}, the optimal elbow point k^* corresponding to the optimal cluster number is obtained as

$$k^* = \arg\max_{1 \le k \le K}(f_k), \quad (23)$$

In light of the preceding steps, the AutoElbow algorithm yields k^*, which is the proposed optimal number of clusters obtained based on the elbow or knee-based graph G. The method is summarized in the pseudocodes of Algorithms 1 and 2. Algorithm 1 is used to build the elbow- or knee-based graph G, whereas Algorithm 2 is used to estimate k^* based on G.

3.2. Time Complexity

In this section, we provide the time complexity (TC) of both the AutoElbow algorithm and the iterated K-means algorithm, based on the pseudocodes of Algorithms 1 and 2, respectively. First, it should be noted that Algorithm 1 implements the iteration of the K-means algorithm towards computing the elbow-based graph. In order to accomplish this, Algorithm 1 calls a single *for*-loop between steps 2 and 7, thus, yielding a TC of $O(K)$, where K is the total number of anticipated clusters in the dataset. However, because the basic K-means algorithm has a TC of $O(N^2)$ [19], where N is the total number of data points, the overall TC of Algorithm 1, thus, reduces to $O(KN^2)$.

The AutoElbow method in Algorithm 2 also adopts a single *for*-loop based on the outcome of the conditional concavity test of step 3. As a result, the single *for*-loop is executed towards computing the parameters of Equation (10) as seen in steps 7–10, 15–18, 23–26, and 31–34 of Algorithm 2. Hence, the TC of the AutoElbow algorithm approaches $O(K)$ asymptotically. Consequently, when considered independently of Algorithm 1, the AutoElbow method is indeed a fast algorithm, as K is often considered a small number (K < 100) based on the application area. However, when both the iteration of the K-means algorithm and the AutoElbow technique are executed sequentially, we obtain an overall TC of $O(KN^2) + O(K)$.

Algorithm 1 Generate elbow graph G

Input Data Z, The maximum number of potential clusters K;
Output Elbow graph, G;

$G \leftarrow [\,]$ Initialize G as an empty set
for $k = 1$ to K **do**
  $[z_i^{(k)}, c_k] \leftarrow$ **K-means(**Z, k**)** Run K-means algorithm with k as number of clusters
  Compute y_k (i.e., TWSD) using (1)
  $x_k = k$
  $g_k \leftarrow [x_k, y_k]$
  $G \leftarrow [G, g_k]$ Append (i.e., save) graph points g_k into G
end for
Return G

Algorithm 2 AutoElbow method

Input Graph G;
Output Estimated number of clusters k^*;

$K \leftarrow$ length(G)
Normalize G as G^{norm} (see (2)) using (3) and (4)
Check for the concavity $\mathcal{C}$ of G^{norm} using (7)
if $\mathcal{C} == 1$ **then**
  — left elbow graph —
  Clean up G^{norm} using (8)
  for $k = 1$ to K **do**
    Compute a_k, b_k, and c_k using (17), (18), and (19), respectively
    Compute f_k using (10)
  end for
  Obtain k^* using (23)
else if $\mathcal{C} == 2$ **then**
  — right elbow graph —
  Clean up G^{norm} using (9)
  for $k = 1$ to K **do**
    Compute a_k, b_k, and c_k using (20), (21), and (22), respectively
    Compute f_k using (10)
  end for
  Obtain k^* using (23)
else if $\mathcal{C} == 3$ **then**
  — left knee graph —
  Clean up G^{norm} using (9)
  for $k = 1$ to K **do**
    Compute a_k, b_k, and c_k using (11), (12), and (13), respectively
    Compute f_k using (10)
  end for
  Obtain k^* using (23)
else if $\mathcal{C} == 4$ **then**
  — right knee graph —
  Clean up G^{norm} using (8)
  for $k = 1$ to K **do**
    Compute a_k, b_k, and c_k using (14), (15), and (16), respectively
    Compute f_k using (10)
  end for
  Obtain k^* using (23)
end if
Return k^*

4. Results and Discussion

We considered four notable case studies to evaluate the performance of the proposed method: the iris (real) dataset (with three clusters), a seven-cluster [20], a forty-cluster synthetic dataset [21], and the Cleveland heart disease dataset [22]. The iris dataset with three clusters was considered suitable for assessing the different approaches for a small cluster number, whereas the seven- and forty-cluster datasets were considered for analyzing datasets with both medium and high cluster numbers, while the Cleveland dataset sufficed for a high dimensional and unbalanced dataset.

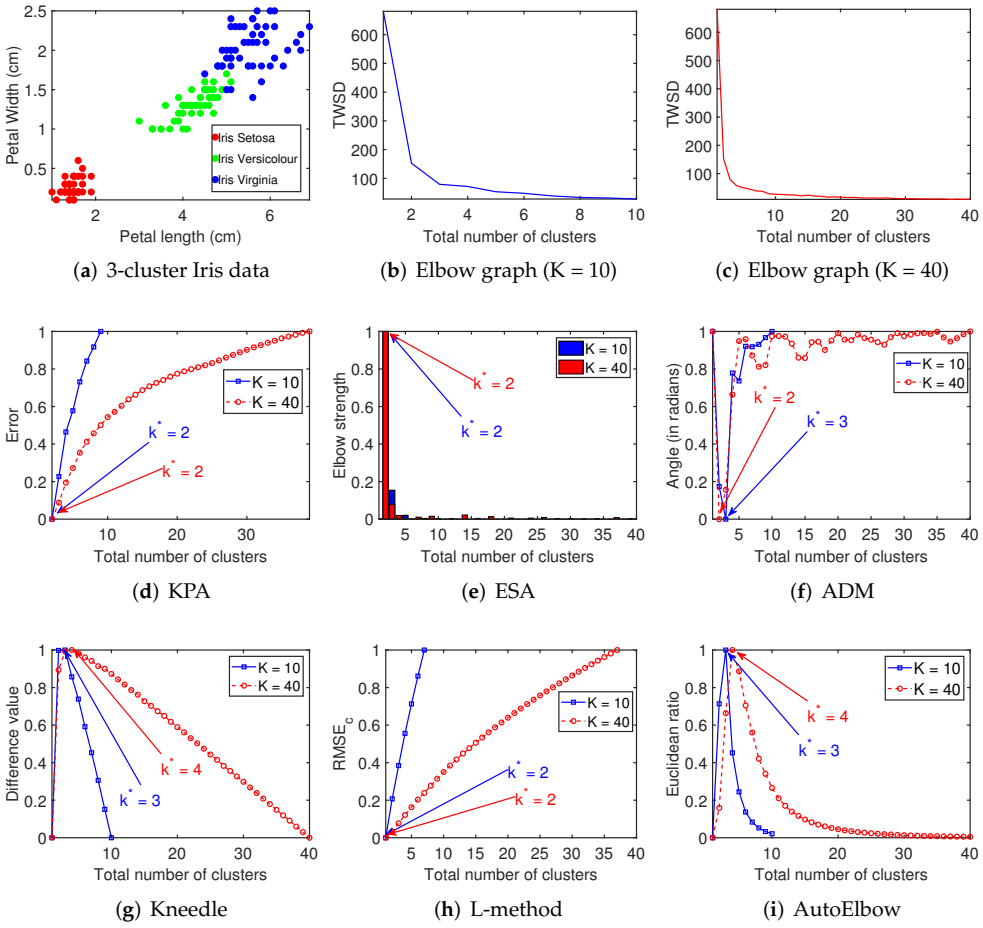
The idea of experimenting with different K values to generate the elbow graph was motivated by the fact that K typically determines the shape of the elbow graph and also because the actual number of clusters in the dataset is not always known in advance. Consequently, small K values typically generate graphs with short tails and may not correspond to the actual number of clusters in datasets, particularly datasets with many clusters. Large K values, on the other hand, may account for datasets with many actual cluster numbers but will generate long-tailed graphs, which will ultimately shift the potential location of the elbow k^*.

Hence, it is essential to evaluate the performance of the various methods for varying K values. As a result, we evaluated the different approaches with $K = 10$ and $K = 40$ for the various datasets, and the results are discussed in the following subsections. In addition, because the K-means algorithm is based on statistics, 1000 independent Monte Carlo simulations were conducted for each dataset. The average results for each method are displayed in the graphs below. We should also mention that, because the essence of each method is to determine k^* along the x-axis of the graph, each method, thus, produces different magnitudes along the y-axis based on their specific measure of interest. To ensure consistency of interpretation, we normalized these measures between 0 and 1 as can be seen in the different performance graphs.

4.1. Three-Cluster Dataset

Figure 3a depicts a 2D representation of the Iris dataset based on the petal width and petal length. This dataset consists of three ideal (i.e., groundtruth) clusters (i.e., the Iris Setosa, Iris Versicolour, and Iris Virginia species). Nevertheless, without prior knowledge of these clusters, it is common to consider the dataset as containing either two clusters (with the Setosa and Virginia species constituting one cluster) or four clusters (i.e., the Iris Setosa and Iris Versicolour as separate clusters, whereas the Iris Virginia as comprising two clusters). Consequently, the automation of this subjective process should result in some errors. However, these errors must fall within reasonable bounds, which we consider for the Iris dataset as between two and four clusters.

Figure 3b,c shows the corresponding elbow-based graphs for $K = 10$ and $K = 40$, respectively. By comparing the two graphs, it is clear that the elbow in Figure 3b is more pronounced than the smoother and longer-tailed graph in Figure 3c. Such variations in the structure of these elbow graphs are deemed useful in evaluating the performance of the various approaches even for the same dataset. As a result, Figure 3b–i shows the corresponding performance graphs for both $K = 10$ and $K = 40$ for the various approaches.



(**a**) 3-cluster Iris data (**b**) Elbow graph (K = 10) (**c**) Elbow graph (K = 40)

(**d**) KPA (**e**) ESA (**f**) ADM

(**g**) Kneedle (**h**) L-method (**i**) AutoElbow

Figure 3. Small cluster number case study: three clusters.

Figure 3d depicts the results of the KPA method. Typically, the KPA method computes an error (i.e., the RMSE) between two best lines that fit the graph, and the minimum value over a range of k is considered the optimal k^* value. In this case, the KPA returns the same value (i.e., $k^* = 2$) for the $K = 10$ and $K = 40$ graphs. Although this may appear appropriate for the small cluster Iris dataset, visual analysis shows that the elbow point in Figure 3c shifted slightly due to the longer tail effect of the graph.

Despite missing this shift, the KPA's performance falls within the expected bounds, and it is, thus, deemed adequate. A similar argument to the KPA can be made for the ESA and L-methods as can be seen from their results depicted in Figure 3e,h, respectively. This is due to the fact that both the KPA and the L-method utilize two best lines of fit for the elbow graph.

Figure 3f,g,i, on the other hand, show that the ADM, Kneedle, and the proposed AutoElbow methods were able to detect the actual cluster number as $k^* = 3$ for $K = 10$. For $K = 40$, the proposed AutoElbow and Kneedle methods also demonstrated a shift in the detected elbow point (i.e., $k^* = 4$), which is expected due to the long tail effect of the graph. However, for $K = 40$, the ADM detected a lower elbow point (i.e., $k^* = 2$), demonstrating that the method may not necessarily adapt to the graph length and elbow position but rather to variations along the curve.

the For instance, because the $K = 40$ graph was more circularly curved and smoother than anticipated, the ADM and ESA approaches did not adjust as expected, because the ADM method seeks to detect graph points with acute angles, which was absent in this case at the ideal elbow point (i.e., $k^* = 3$). It can, thus, be concluded that the proposed method and other competing methods typically performed well for datasets with small clusters and for smaller K values used to generate the elbow graph (e.g., $K = 10$). However, this may

not always be the case, and the consequences of other potential conditions are discussed in the following subsections.

4.2. Seven-Cluster Dataset

The various methods were tested on a synthetic dataset with seven clusters, as shown in Figure 4a. The dataset was deemed instructive because it included a mix of small and large, convex-shaped, and connected clusters. As a result, any clustering algorithm may be susceptible to subjective errors, which may be reflected in the apparent elbow position within the elbow-based graph. Consequently, we ran the K-means algorithm repeatedly for $K = 10$ and $K = 40$.

We note that using $K = 10$ for a dataset with a groundtruth of seven clusters may obstruct the precise location of the true elbow because it will be close to the graph's edge. As a result, instead of an ideal sharp elbow, the graph may suffice as a curve, as shown in Figure 4b. This exemplifies why a large K value is frequently preferred, particularly in applications where the expected cluster number in the dataset is unknown. As a result, we attempted $K = 40$, which introduces the long-tail effect, as shown in Figure 4c and which may eventually influence the ideal elbow position.

In addition, since it is well-known that the K-means algorithm performs better with spherical clusters and poorly with irregularly shaped clusters [9], we specified the error bound to be between six and eight clusters in this instance. This error bound was chosen because the K-means algorithm may estimate the convex-shaped cluster in the dataset as two separate clusters, whereas the connected clusters may be viewed as a single cluster.

The KPA method detected its elbow point at $k^* = 3$ for $K = 10$ and $k^* = 4$ for $K = 40$, as shown in Figure 4d. In essence, the KPA, ESA, and L-method all estimated k^* values that were outside of the error bound for both $K = 10$ and $K = 40$, while the poor performance for $K = 10$ can be explained by the groundtruth's proximity to the graph edge (i.e., $k^* = 7$ being close to $K = 10$); nevertheless, it is expected that when the tail was extended to $K = 40$, these algorithms would have adapted. However, this was not the case because both curves (i.e., for $K = 10$ and $K = 40$) had a high curvature, thus, making the true estimate of the elbow point difficult for these methods.

Although the ADM, Kneedle, and proposed AutoElbow method all underestimated the true k^* value as $k^* = 3$ for the $K = 10$ graph, as shown in Figure 4f,g,i; nevertheless, they all adapted appropriately when K was increased to $K = 40$. As a result, our proposed algorithm, along with the ADM and Kneedle methods, achieved an accuracy of $k^* = 6$, which was within the error bounds. These results are consistent with what a human observer might have estimated in the absence of prior knowledge of the true cluster number. By visually examining Figure 4a, for instance, an expert may localize the elbow point k^* to be between 4 and 6, thus, justifying the low k^* value estimated by the various algorithms for $K = 10$.

To summarize, all of the algorithms failed to identify the correct cluster number, owing to the poor structure of the elbow graph, which was caused by the low iteration value used in this case. As a result, this experiment validates the need for large K iteration values to be used, for which our proposed algorithm performed well and within appropriate bounds.

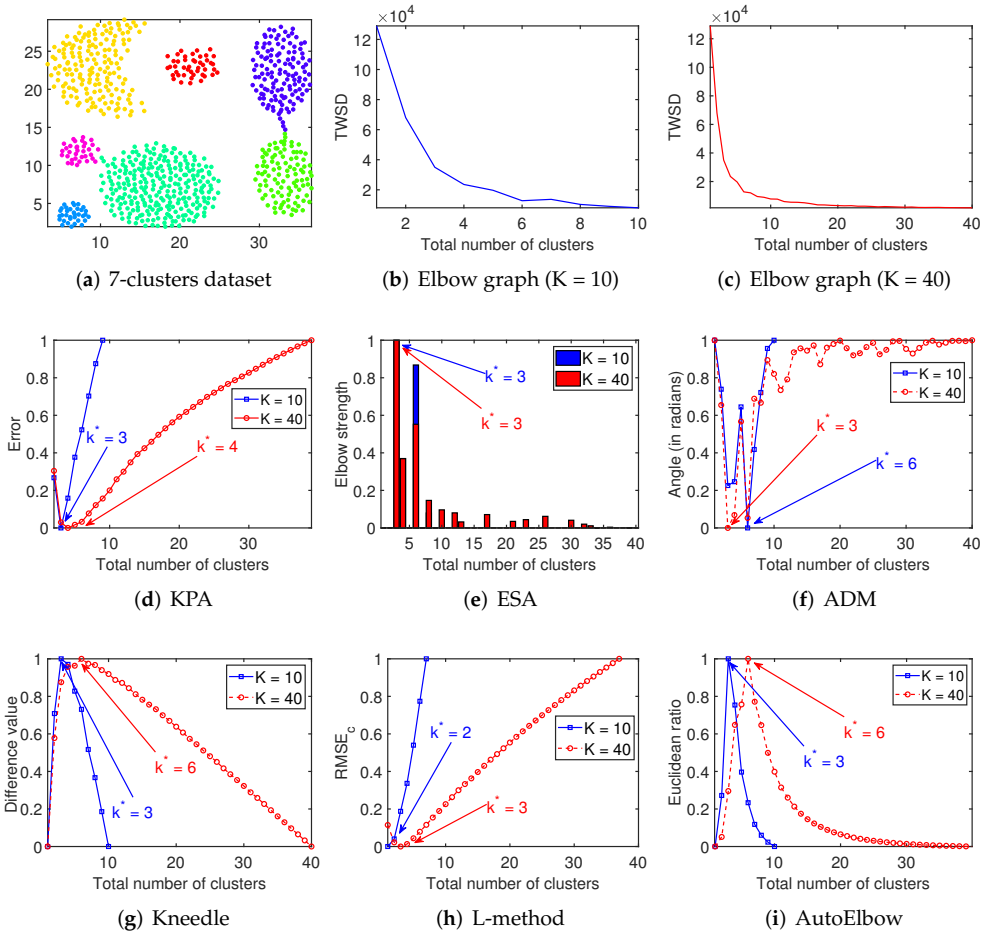
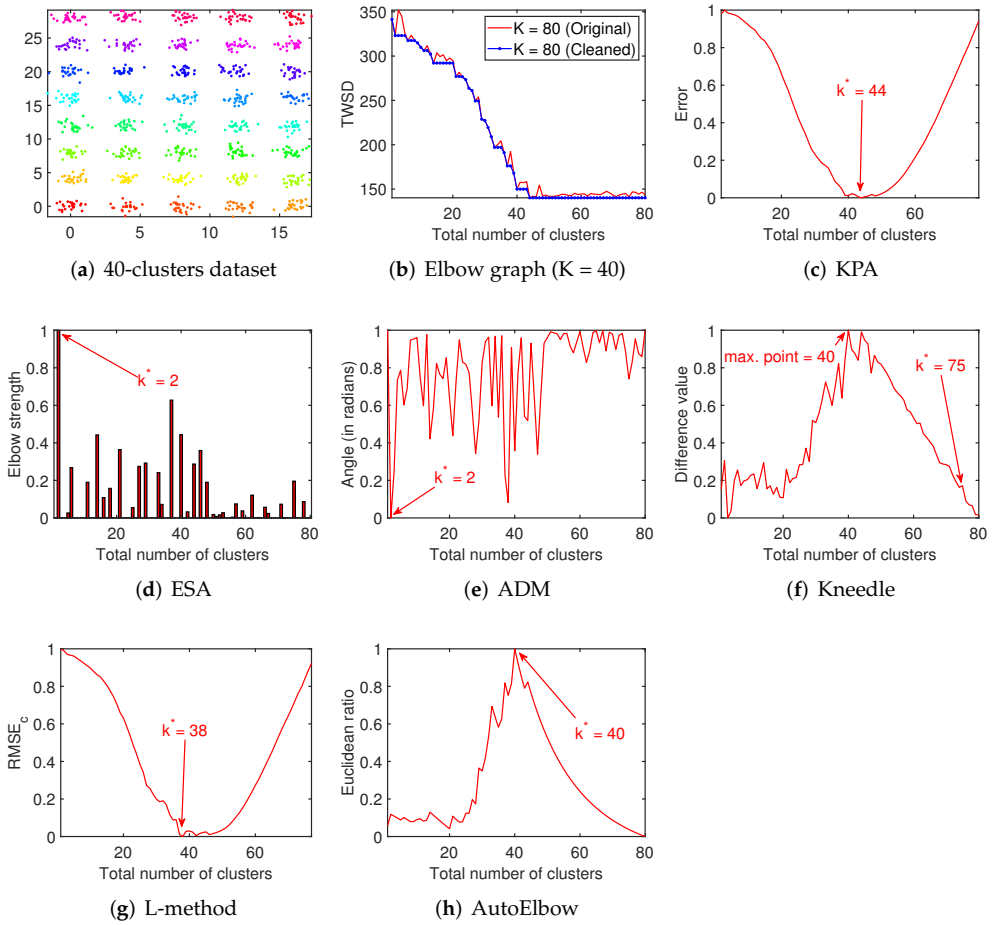


Figure 4. Medium cluster number case study: seven clusters.

4.3. Forty-Cluster Dataset

We tested the algorithms using a dataset with 40 clusters, as shown in Figure 5a, demonstrating the need for larger K values. As a result, we considered the case of $K = 80$, resulting in the elbow graph shown in Figure 5b. This experiment was valuable because it presented the algorithms with an elbow graph with fluctuating graph points. This allowed us to examine the concepts of the various methods using such a perturbed graph. The red graph in Figure 5a represents the original fluctuating elbow graph obtained after executing Algorithm 1, whereas the blue graph was obtained after applying the cleaning function (i.e., Equation (8)) in the AutoElbow method. Furthermore, because we previously used a 10% error bound around the groundtruth as the accuracy limit, we considered an error bound of 36 to 44 for the current 40-cluster dataset.



(**a**) 40-clusters dataset (**b**) Elbow graph (K = 40) (**c**) KPA

(**d**) ESA (**e**) ADM (**f**) Kneedle

(**g**) L-method (**h**) AutoElbow

Figure 5. Large cluster number case study: 40 clusters.

As illustrated in Figure 5d,e, the ESA and ADM methods failed to locate the correct elbow point due to the disturbances in the initial elbow graph of Figure 5b. We note that the ADM technique works by computing the angles formed at each graph point. However, as a result of the oscillations in the original graph, and thus many acute angles were created, which resulted in the chaotic nature of the ADM's output function as depicted in Figure 5e.

The ESA method, on the other hand, works by calculating the difference between the graph's second- and first-order differences. However, because of the steep slope at the beginning of the graph, the ESA's output function resulted in a high elbow strength, which misled the algorithm's judgment of the elbow point. Therefore, neither technique can be guaranteed to yield consistent results unless some smoothing or approximation functions are applied to the graph prior to using these methods.

On the other hand, Figure 5c,f–h shows that the KPA, Kneedle, L-method, and our proposed AutoElbow method all produced estimates that fell within the error bounds for this dataset. However, the KPA method overestimated the number of clusters given $k^* = 44$, which can be permitted based on the fact that the minimum point, as depicted in Figure 5c, falls within a broad minimum range spanning $k = 40$ to 48. This can be further verified visually in Figure 5b, where the elbow appears to be between $k^* = 40$ and 44. We note that Figure 5c,g shows that the L-method works similarly to the KPA method, as evidenced by the similar function shapes of their respective graphs. Consequently, the explanation provided for the KPA also applies to the L-method.

For the Kneedle algorithm, although it can be observed that the maximum point of the graph in Figure 5h is at $k^* = 40$; nevertheless, the Kneedle algorithm outputted the elbow point as $k^* = 75$. This is due to the fact that the Kneedle algorithm uses a threshold method to determine the true elbow point (see [13] p. 4), and when the threshold function

was used, the Kneedle algorithm incorrectly outputted $k^* = 75$ as the elbow point. This can be explained noting that the Kneedle algorithm's threshold function looks for the last local peak in the output function to determine the elbow point. However, due to the noisy nature of the function in Figure 5f, the algorithm detected an erroneous value of $k^* = 75$.

Our AutoElbow algorithm, on the other hand, estimated an accurate elbow point, as shown in Figure 5h. This can be explained by the fact that the AutoElbow method produces a smoother function than the Kneedle algorithm (see Figure 5h), which demonstrates the AutoElbow method's advantage. Furthermore, we note that even though the AutoElbow method correctly identifies the elbow point as the maximum value, which could have been applied to the Kneedle algorithm as well, nevertheless we can see that the AutoElbow's final peak localizes at $k^* = 44$, which, if taken, would still fall within the error bounds established for this dataset.

We demonstrated the stability of the AutoElbow approach by analyzing its performance across a range of K values for small, medium, and large cluster datasets. Stability is demonstrated by the method's accurate estimations of the ideal elbow location for graphs with smooth, strongly curved, and acute elbows. In addition, it displays adaptability by conforming to the length of the elbow graph, i.e., by following both short- and long-tailed graphs. The AutoElbow method's output is a smoother and convex-based function, which further ensures its stability in locating a single distinct elbow point in elbow-based graphs. These results have, thus, demonstrated the viability of the AutoElbow approach.

4.4. The Cleveland (Heart Disease) Dataset

The performance of our method was evaluated for the case of a high dimensional and unbalanced real dataset based on the well-known Cleveland heart disease dataset [22]. The Cleveland dataset is well-known and widely used as a benchmark for systems that detect heart disease [22]. The dataset is divided into five classes (i.e., five clusters in our case), each represented by an integer value ranging from 0 (no presence of heart disease) to 4 (presence of heart disease).

We used the processed file (i.e., "processed.cleveland.data") from the database, which contains 303 instances with 13 medical attributes (i.e., the dimensions). The class attribute, which is the 14th dimension, was removed since it only serves as the groundtruth for the number of clusters. Figure 6 depicts a summary of the dataset, which provides the number of attributes, their respective terms, and the class (i.e., cluster) bar chart showing the unbalanced distribution of the dataset.

Since the dataset consists of five clusters (groundtruth); thus, we tested each algorithm using $K = 10$. We note that, using $K = 10$ permits for some allowance in the tail of the elbow graph, which we considered sufficient to detect the elbow point. The obtained results are depicted in Figure 7a–g. The AutoElbow and L-method both correctly detected the presence of five clusters in the Cleveland dataset. The KPA method detected four clusters, which falls within the error tolerance range of four to six clusters; however, the Kneedle, ESA, and ADM methods fell outside of this range, making them the lowest performers.

Although the maximum point of the Kneedle response graph (see Figure 7e) corresponds to the actual cluster number; nevertheless, the use of the inbuilt threshold approach of the Kneedle algorithm resulted in an erroneous value of $k^* = 8$. Nonetheless, it is worth noting that the elbow graph in Figure 7a shows that an elbow appears at both $K = 5$ and $K = 8$, which may explain why the ESA, ADM, and Kneedle algorithms detected the cluster number at $k^* = 8$ for the Cleveland dataset. On the other hand, our AutoElbow algorithm clearly produced a smoother response curve than the Kneedle algorithm, thus, demonstrating its stability.

Another noteworthy feature of our proposed AutoElbow method is its ability to effectively smoothen the perturbed elbow graph, as illustrated in Figure 7a. By smoothing the curve, the AutoElbow algorithm is able to accurately find the elbow point and, by extension, the exact number of clusters in the dataset.

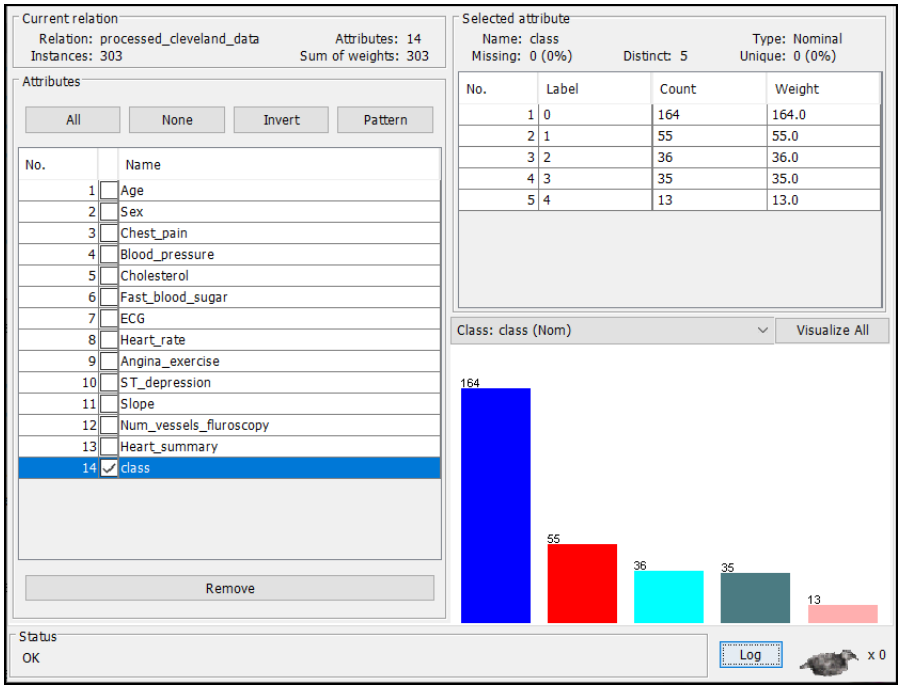


Figure 6. The Cleveland dataset characteristics: number, name of attributes, and distribution per cluster.

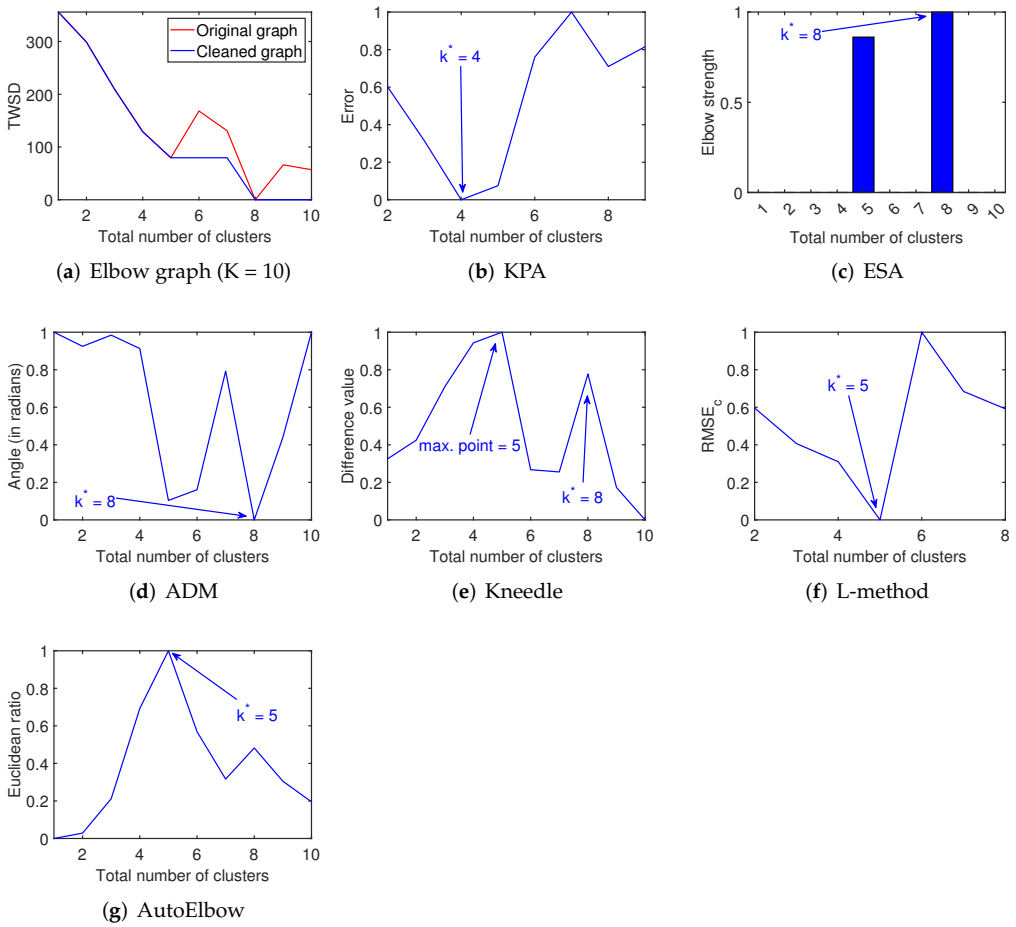


Figure 7. Performance on the high dimension Cleveland data with five clusters.

Summarily, we demonstrated that the AutoElbow method can be used to automatically determine the optimal cluster number from an elbow graph generated by iterating the K-means algorithm over a high-dimensional, unbalanced dataset. However, we note that the effects of higher dimensional and cluster imbalanced datasets may indeed have an effect on the K-means performance; nevertheless, as long as an elbow point exists in the elbow graph, our method is capable of detecting it accurately. Thus, prior to actually employing the K-means method or any other unsupervised clustering approach for the purposes of clustering, the AutoElbow method can be utilized to automatically obtain an estimate of the number of clusters in a dataset.

5. Conclusions

In this paper, we presented a new method for determining the optimal number of clusters in a dataset by accurately detecting the elbow point of an elbow-based graph. Our AutoElbow method, which works for both elbow- and knee-based graphs, can be used to easily automate the K-means algorithm and any other unsupervised clustering approach. The AutoElbow algorithm produced a more convex and smoother function than the Kneedle algorithm, thus, allowing it to be used on highly perturbed elbow- or knee-based graphs.

By comparing the AutoElbow method to other well-known methods in the prior art, we demonstrated the relative accuracy, stability, and adaptability of the AutoElbow method for different K values as well as for small, medium, and large cluster datasets. The performance of the AutoElbow method, thus, contributes towards improving the accuracy of clustering methods in determining the ideal number of clusters in a dataset. This will also allow clustering methods to be deployed more efficiently and effectively in a wide range of application areas.

However, because the proposed method is based on the generated elbow-based graph, it is limited by how accurately the graph depicts a sharp elbow point and, thus, the number of clusters in the dataset. In future works, the AutoElbow method may yet be further improved by examining other evaluation metrics and making better use of the output characteristics of the AutoElbow algorithm.

Author Contributions: These authors A.J.O., D.N.M., S.J.I. and A.M.A.-M. contributed equally to this work. Conceptualization, A.J.O., D.N.M., S.J.I. and A.M.A.-M.; methodology, A.J.O. and D.N.M.; writing—original draft preparation, A.J.O.; writing—review and editing, D.N.M., S.J.I. and A.M.A.-M.; supervision, S.J.I. and A.M.A.-M.; funding acquisition, S.J.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Council for Scientific and Industrial Research (CSIR) and The APC was funded by project number 05400 054AT KR3EEMG.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Li, C.; Kulwa, F.; Zhang, J.; Li, Z.; Xu, H.; Zhao, X. A Review of Clustering Methods in Microorganism Image Analysis. In *Advances in Intelligent Systems and Computing*; Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 13–25. [CrossRef]
2. Azaza, M.; Wallin, F. Smart meter data clustering using consumption indicators: responsibility factor and consumption variability. *Energy Procedia* **2017**, *142*, 2236–2242. [CrossRef]
3. Hayatu, H.I.; Mohammed, A.; Isma'eel, A.B. Big Data Clustering Techniques: Recent Advances and Survey. In *Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics*; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 57–79. [CrossRef]

4. Kaptan, S.; Vattulainen, I. Machine learning in the analysis of biomolecular simulations. *Adv. Phys. X* **2022**, *7*, 2006080. [CrossRef]
5. Torkzadeh, L.; Jalilian, H.; Zolfagharian, M.; Torkzadeh, H.; Bakhshi, M.; Khodayari-Zarnaq, R. Market segmentation in the health tourism industry: A systematic review of approach and criteria. *J. Policy Res. Tour. Leis. Events* **2021**, 1–20. [CrossRef]
6. Ghosal, A.; Nandy, A.; Das, A.K.; Goswami, S.; Panday, M. A Short Review on Different Clustering Techniques and Their Applications. In *Advances in Intelligent Systems and Computing*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 69–83. [CrossRef]
7. Yassine, S.; Kadry, S.; Sicilia, M.A. Detecting communities using social network analysis in online learning environments: Systematic literature review. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2022**, *12*, e1431. [CrossRef]
8. Dinh, D.T.; Fujinami, T.; Huynh, V.N. Estimating the optimal number of clusters in categorical data clustering by silhouette coefficient. In *International Symposium on Knowledge and Systems Sciences*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 1–17.
9. Tibshirani, R.; Walther, G.; Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *J. R. Stat. Soc. Ser. Stat. Methodol.* **2001**, *63*, 411–423. [CrossRef]
10. Khachumov, M. Distances, metrics and cluster analysis. *Sci. Tech. Inf. Process.* **2012**, *39*, 310–316. [CrossRef]
11. Granville, V. How to Automatically Determine the Number of Clusters in Your Data—And More. 2019. Available online: https://www.datasciencecentral.com/how-to-automatically-determine-the-number-of-clusters-in-your-dat/ (accessed on 26 July 2022).
12. Kaplan, D. Knee Point. Software, 2022. MATLAB Central File Exchange. Available online: https://www.mathworks.com/matlabcentral/fileexchange/35094-knee-point (accessed on 26 July 2022)
13. Satopaa, V.; Albrecht, J.; Irwin, D.; Raghavan, B. Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In Proceedings of the 2011 31st International Conference on Distributed Computing Systems Workshops (ICDCSW), Minneapolis, MN, USA, 20–24 June 2011; pp. 166–171.
14. Diao, W.; Saxena, S.; Han, B.; Pecht, M. Algorithm to Determine the Knee Point on Capacity Fade Curves of Lithium-Ion Cells. *Energies* **2019**, *12*, 2910. [CrossRef]
15. Shi, C.; Wei, B.; Wei, S.; Wang, W.; Liu, H.; Liu, J. A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm. *EURASIP J. Wirel. Commun. Netw.* **2021**, *2021*, 1–16. [CrossRef]
16. Salvador, S.; Chan, P. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, Boca Raton, FL, USA, 15–17 November 2004; pp. 576–584.
17. Zhao, Q.; Hautamaki, V.; Fränti, P. Knee point detection in BIC for detecting the number of clusters. In *International Conference on Advanced Concepts for Intelligent Vision Systems*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 664–673.
18. Singh, A.; Yadav, A.; Rana, A. K-means with Three different Distance Metrics. *Int. J. Comput. Appl.* **2013**, *67*, 13–17. [CrossRef]
19. Pakhira, M.K. A linear time-complexity k-means algorithm using cluster shifting. In Proceedings of the 2014 International Conference on Computational Intelligence and Communication Networks, Bhopal, India, 14–16 November 2014; pp. 1047–1051.
20. Gionis, A.; Mannila, H.; Tsaparas, P. Clustering Aggregation. In Proceedings of the 21st International Conference on Data Engineering (ICDE'05), Tokoyo, Japan, 5–8 April 2005; Number 2375-026X, pp. 341–352. [CrossRef]
21. Barton, T. Clustering Benchmarks. Available online: https://github.com/deric/clustering-benchmark (accessed on 8 June 2022).
22. Janosi, A.; Steinbrunn, W.; Pfisterer, M.; Detrano, R. Available online: https://archive.ics.uci.edu/ml/datasets/heart+disease (accessed on 19 July 2022).