Stop using the elbow criterion for k-means

and how to choose the number of clusters instead

Erich Schubert
TU Dortmund University
44221 Dortmund, Germany
erich.schubert@tu-dortmund.de

ABSTRACT

A major challenge when using k-means clustering often is how to choose the parameter k, the number of clusters. In this letter, we want to point out that it is very easy to draw poor conclusions from a common heuristic, the "elbow method". Better alternatives have been known in literature for a long time, and we want to draw attention to some of these easy to use options, that often perform better. This letter is a call to stop using the elbow method altogether, because it severely lacks theoretic support, and we want to encourage educators to discuss the problems of the method – if introducing it in class at all – and teach alternatives instead, while researchers and reviewers should reject conclusions drawn from the elbow method.

1. INTRODUCTION

Cluster analysis aims at identifying subgroups in the data that have high similarity within the group, while they also differ from the remainder of the data set. No single "best" definition of a cluster exists. Bonner [5] noted that "none of the many specific definitions [of clusters] seems 'best' in any general sense", and Estivill-Castro [12] argued that it cannot exist. each data set and use case may call for different properties to be desirable, which in turn leads to different algorithms to find the "best" solution. Hence, a large number of clustering methods were developed over the last decades, based on concepts such as finding a hierarchical structure (akin to phylogenetic trees), quantization and compression, parametric modeling, or identifying dense areas.

Despite the many different concepts of clusters and the wide variety of clustering algorithms available, one method currently is the most used and most taught clustering method: k-means clustering. One of the main reasons may be the simplicity of the standard algorithm: assigning each point to the nearest center, then recomputing all the cluster centers until nothing changes – this algorithm can be easily described in a single sentence. At the same time, this algorithm runs very fast, and it will always produce a result with exactly k clusters, giving a (false) suggestion of success. A key problem then with applying this method to data is often the need to choose the number of clusters k, although users should first consider whether k-means is even the right choice for their problem at all, and pay more attention to data preprocessing, too. It makes no sense to search for the "optimum" k if k-means is not solving the problem.

2. K-MEANS CLUSTERING

Formally, k-means clustering is a least-squares optimization problem. We can best view it as a data quantization technique, where we want to approximate the data set of N objects in a continuous, d-dimensional vector space $\mathbb{R}^d$ using k centers. The quantization error for a data set X and a set C of centers then is called inertia, the within-cluster sum of squares (WCSS), or the sum of squared errors (SSE):

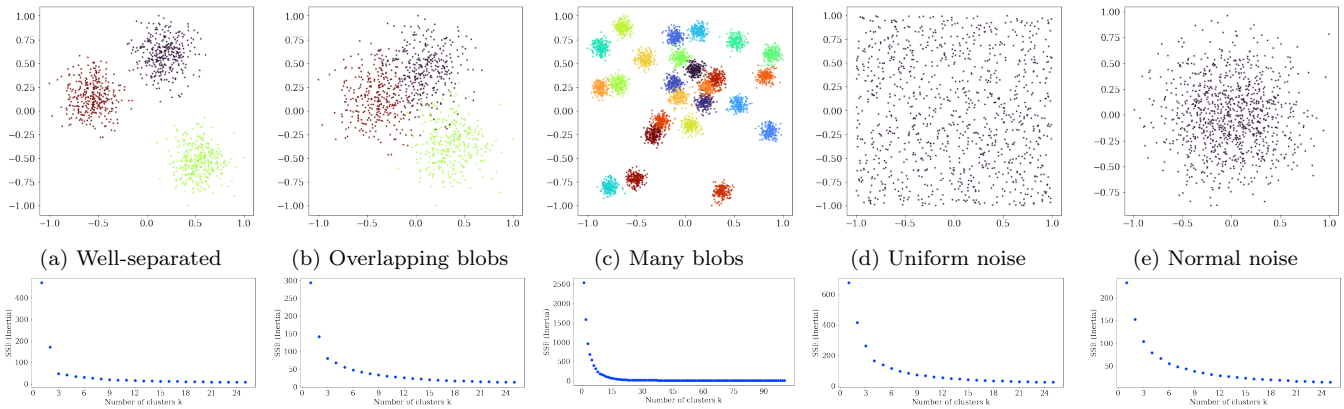
$$\mathrm{SSE}(X, C) = \sum_{x \in X} \min_{c \in C} \|x - c\|^2 . \quad (1)$$

While it is easy to optimize this for a single cluster center by taking the arithmetic average in each dimension, i.e., the data set centroid, the problem is NP-hard for multiple clusters and higher dimensionality [22; 1].

Several methods to optimize this objective exist, but because of the hardness, most heuristics will only find a local fixpoint. Because of the very common least-squares objective, the standard algorithm has likely been invented several times independently, as discussed in the overview of Bock [4]. The standard heuristic for k-means is an alternating optimization, which first assigns each point to the nearest current cluster center, then updates each cluster center position with the centroid of the points assigned to it. If we keep assignments unchanged whenever distances are identical, the algorithm will eventually not find any changes and stop (because both steps may never worsen the objective function, and there exists only a finite number of possible cluster assignments). The standard algorithm has a complexity of $O(Nkdi)$, where i is the number of iterations (which theoretically could be very high, but usually is small in practice). This makes it one of the fastest clustering methods we have available, compared to $O(N^3 + N^2 d)$ for the standard algorithm for hierarchical clustering, or $O(N^2 d)$ for DBSCAN without index acceleration. Many improvements have been proposed that avoid repeated computations in the standard algorithm, nevertheless, this very basic form has become quite popular again with the rise of parallel processing and GPUs: it is embarrassingly parallel, and hence very easy to implement both in clusters as well as GPUs. For this letter, it does not matter which variant of the algorithm we use.

3. THE ELBOW CRITERION

When the number of clusters k is not already given by the application, we have to choose this value; and it turns out this can be rather tricky. The elbow plot is a chart plotting the approximation error SSE on the y-axis over a range of values for k on the x-axis. The motivation of the elbow criterion is the concept of diminishing returns: as we increase



(a) Well-separated (b) Overlapping blobs (c) Many blobs (d) Uniform noise (e) Normal noise

(f) Well-separated, $k \leq 25$ (g) Overlapping, $k \leq 25$ (h) Many blobs, $k \leq 100$ (i) Uniform noise, $k \leq 25$ (j) Normal noise, $k \leq 25$

Figure 1: Toy data sets and resulting – very similar – elbow plots.

the number of clusters, the approximation error decreases.[1] In a data set with very well-separated clusters, we expect to initially see a sharp drop until some "optimum" number of clusters, afterwards we are splitting "true" clusters, which leads to much smaller gains. And indeed, on certain toy data sets, this appears to work well. In Figure 1a we have a data set with three well-separated clusters, that is easily clustered by k-means. Figure 1f is the corresponding elbow plot, with a clear inflection at the desired $k = 3$. But the other examples in Figure 1 show that the plot always looks similar, even on uniform data or when the data contains a single normal distribution.

The elbow method is attributed to Thorndike [39], albeit he notes "The curves do not provide much support for the intuitive specification of the number of clusters", and concludes with doubt: "At this point I can sense the bubbling up of doubts and questions: 'But what about your *units*?'".

There are several problems associated with the elbow plot, that statisticians know too well from the scree plot. Because the axes of the plot have very different meanings, we cannot compare them well. We do not have a meaningful measurement of angle, and changing the scaling of the axes (and, e.g., the parameter range of k) may well change the human interpretation of an "elbow".

3.1 Elbow Detection

Several attempts to formalize the notion of an "elbow" can be found in software and literature. We present only an excerpt in the following, largely to illustrate how *heuristic* and *visual* the machine learning community currently approaches this problem, instead of improving theory.

Sugar et al.[38] propose a "jump method", finding the maximum of $\mathrm{SSE}_k^{-Y} - \mathrm{SSE}_{k-1}^{-Y}$ where Y is a power parameter suggested to be half the dimensionality.

Salvador et al. [30] propose the L-method, which fits linear functions to the points before and after the break; choosing the breaking point where this piecewise-linear approximation fits best. But as discussed above, we will often see an exponential curve, and such a linear approximation often does not fit this curve at all. To improve this, the authors also suggest an iterative approach where they truncate the plot to the first $2 \cdot k$ values if k is the best solution found.

Satopää et al. [31] in their Kneedle algorithm want to measure the curvature. For this they fit a smoothing spline to the data, normalize it to 0 to 1, and compute the difference to the diagonal. The last maximum before a parameterizable stopping threshold is chosen.

Zhang et al.[42] note that the standard curvature definition is not independent of rescaling the data, and propose to choose the maximum of a modified curvature:

$$\mathrm{Curvature}_k := \frac{\mathrm{SSE}_{k-1} - \mathrm{SSE}_k}{\mathrm{SSE}_k - \mathrm{SSE}_{k+1}} - 1$$

The pyclustering library [25] defines an elbow length:

$$\mathrm{ElbowLen}_k := \frac{(y_0 - y_1)\, x_k + (x_1 - x_0)\, y_k + (x_0 y_1 - x_1 y_0)}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}}$$

where x_0, x_1, y_0, y_1 denote the minima and maxima of the graph; intended to measure the length when approximating the curve with the elbow point. There is no literature given for this approach, the given reference to Thorndike [39] does not entail this equation, which again appears to depend very much on the scaling of the plot.

Shi et al. [37] note that "experienced analysts cannot clearly identify the elbow point", and suggest applying a min-max scaling to the range 0 to 10 instead, then computing angles between triples of adjacent values. The performance of this approach depends much on this weighting factor.

Onumanyi et al. [26] propose AutoElbow, for which they min-max scale the elbow plot to 0 to 1, then propose a geometrically-motivated measure of how close a point is to the bottom left corner and bottom line. The result of this objective changes substantially when increasing the candidate range of k, and hence likely should not be used at all.

$$\mathrm{AutoElbow}_k := \frac{(x_k - 1)^2 + (y_k - 1)^2}{x_k^2 + 2 \cdot y_k^2}$$

[1] Given a k-means solution for some k, we can trivially construct a solution with $k + 1$ that is better (unless the error already is zero) by simply adding any point with non-zero error as an additional center.

Table 1: "Optimum" k chosen by different heuristics on the toy data sets. † indicates the result can still be recognized as a poor result by the score value or by visual inspection. ‡ indicates results that fluctuate with random seeds.

		well-sep.		overlapping		many blobs		uniform		normal	
true k		3		3		25		1		1	
max k		10	25	10	25	50	100	10	25	10	25
Elbow-based											
Jump	[38]	3	3	3	3	23	23	4	4	6	21
L-Method	[30]	3	3	3	4	7	9	4	5	4	5
L-Method (iter.)		-	3	-	4	-	6	4	4	4	5
Kneedle	[31]	3	3	3	5	8	10	4	5	4	6
Curvature	[42]	3	3	3	3	38	38	4	4	3	21
Pyclustering	[25]	3	3	3	5	8	10	4	5	4	6
Shi angles	[37]	3	3	3	3	3	3	4	4	4	3
AutoElbow	[26]	3	3	3	6	9	11	4	6	4	7
Variance-based											
Marriot	[23]	3	3	3	3	25	25	$9^{\dagger}$	$17^{\dagger}$	$2^{\dagger}$	$2^{\dagger}$
VRC	[6]	3	3	3	3	25	25	$9^{\dagger}$	$20^{\dagger}$	$4^{\dagger}$	$4^{\dagger}$
K-L-Index	[20]	7	7	4	10	35	62	5	21	8	8
Pham	[28]	3	3	2	2	8	8	4	4	10	21
Max reduction		3	3	3	3	8	8	$4^{\dagger}$	$12^{\dagger}$	$1^{\dagger}$	$1^{\dagger}$
Last reduction		3	3	3	3	25	25	$9^{\dagger}$	$20^{\dagger}$	$1^{\dagger}$	$1^{\dagger}$
Information-theory-based											
BIC	[27]	3	3	3	3	25	100	9	25	1	1
BIC (fixed)	[13]	3	3	3	3	25	25	$1^{\dagger}$	$1^{\dagger}$	$1^{\dagger}$	$1^{\dagger}$
Distance-based											
Dunn	[10]	3	3	8	17	18	18	7	20	3	24
DB	[7]	3	3	3	3	21	21	4	4	10	22
Silhouette	[29]	3	3	3	3	21	21	4	4	3	3
Simpl. Silhouette		3	3	2	2	21	21	4	4	3	3
Simulation-based											
Gap	[40]	3	3	3	3	$30^{\ddagger}$	$30^{\ddagger}$	$14^{\ddagger}$	$21^{\ddagger}$	1	1

3.2 Detection performance

Several of these measures are sensitive to the range of k that we analyze, even if the additional values of k perform poorly. They are heuristics based on the geometric idea of an elbow point, but not taking the process causing the measured data into account. In Table 1 we give the results obtained for the toy data sets of Figure 1 using several heuristics proposed in the literature. Because some methods are very sensitive to the range of k included, we tested two limits, one rather conservative, and one that is much larger. We observe that all methods were able to recognize the best solution on the well-separated data set, and even on the more overlapping version, they all worked – for a small enough maximum k. For the data set with many clusters as well as the uniform data set, all the elbow-based methods failed. The methods based on variance – which we will discuss below – worked much better, but when applied naïvely will still cluster the uniform data. Only when using additional thresholds (or visually inspecting the score plot), the uniform data is recognizable as not clustered. For the normal data, our method indicates a single cluster, while the classic variance-ratio criterion also discussed below has a maximum at six clusters.

First of all, the quantity measured, the sum of squared deviations, is a *squared* value. It would make much more sense to analyze the square root of this value, and if we also take the number of points into account, the root-mean-squared-deviation (RMSD), which corresponds to a standard deviation of each point to the nearest center. How meaningful are "angles", "distances", "elbows", and "slopes" on a graph that compares k to SSE, two quantities of different scales? If we scale the entire data set by a factor of α, the SSE will change by α^2, and the "optimum" found by most of the geometric methods changes, while it is clear that it should not. Secondly, increasing the parameter range of k analyzed must not change the decision once the optimum k is included. Normalizing to the observed minimum and maximum values (often even starting with $k = 2$, not $k = 1$) seems inappropriate. In particular, even without running the algorithm, we know that for $k = N$ we will be able to get an approximation error of 0, so we likely should always consider N to be the maximum x coordinate, and 0 to be the minimum y coordinate. A meaningful normalization should preserve 0.

Third, we know that even on random data we obtain a descending curve, and hence we should try to remove this expected behavior from our measure.

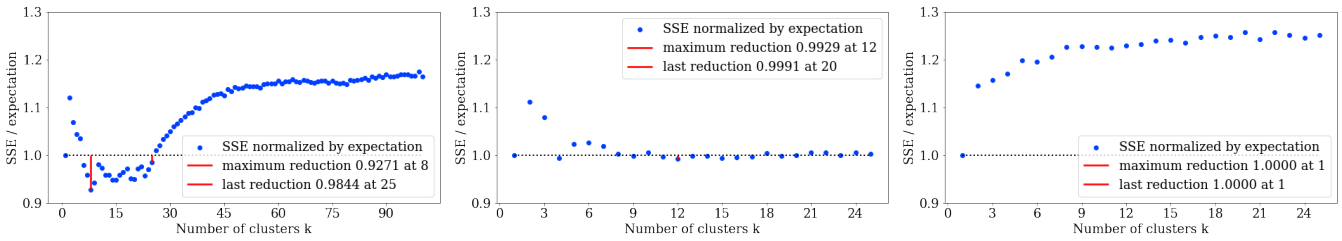
Fourth, the method should be able to choose $k = 1$ for data that does not contain any meaningful clusters.

3.3 Expected behavior of SSE

Instead of proposing heuristic visual approaches to formalize an imaginary "elbow", we need to first better understand the quantity that we are working with. The sum of squared errors closely resembles the variance of the data set. Because our cluster centers are derived from the data, we should be using a form of sample variance. Simply diving the SSE by N will be a biased estimate, and we postulate that $\mathrm{SSE}/(N-k)$ is a more suitable estimate in this context. But since usually $k \ll N$, this will not make much of a difference yet. Instead of working with the squared quantity, we then may want to apply the square root instead, i.e., use $\sqrt{\mathrm{SSE}/(N-k)}$ to have the intuition of a *standard deviation from the nearest center*. Still, the plot obtained this way will look similar to what we started with, and because the square root is a monotone function on the outside, it will not affect the ordering of results – it only serves to make the quantity more interpretable, because ideally, the domain expert should judge whether this is sufficiently small.

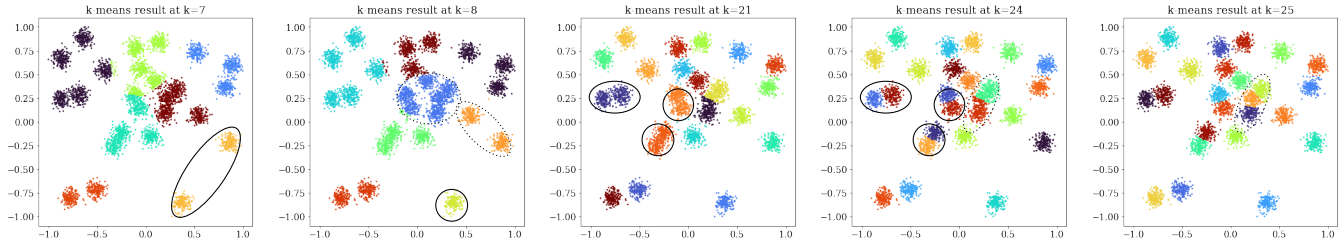
As a baseline "expected" behavior, we will for simplicity assume the input data to be uniformly distributed in a single dimension, but with the variance of the input data set. The variance of a uniform interval of length b is $\mathrm{Var}([0;b]) = \frac{1}{12}b^2$. If we slice this into k slices of equal length, each of these has $\mathrm{Var}([0;b/k]) = \frac{1}{12}b^2/k^2$, and we obtain for the resulting total variance $k \cdot \mathrm{Var}([0;b/k]) = \frac{1}{k}\mathrm{Var}([0;b])$. Because of this observation, we propose to use the naïve estimate SSE_1/k as normalization factor. But Krzanowski and Lai [20] suggest that $\mathrm{SSE}/k^{\frac{2}{d}}$ may be more appropriate than our naïve estimate. We should further include the $N-k$ factor discussed above. If there is more than one good parameter k (e.g., because there are substructures in the data), we may also want to compare the solution with the best found so far, e.g., using:

$$\widehat{\mathrm{SSE}}_k := \frac{N-k}{k} \min_{j=1\ldots k-1} \frac{j}{N-j}\,\mathrm{SSE}_j \tag{2}$$



(a) Many blobs (b) Uniform distribution (c) Normal distribution

Figure 2: Reduction in $\sqrt{SSE}$ over the estimate $\sqrt{\widehat{\text{SSE}}_k}$.



(a) $k = 7$ (b) $k = 8$ (c) $k = 21$ (d) $k = 24$ (e) $k = 25$

Figure 3: Clustering results on the "many blobs" data set.

We can now generate a standard deviation reduction plot, comparing the observed with the estimated values:

$$\frac{\sqrt{\text{SSE}_k/(N-k)}}{\sqrt{\widehat{\text{SSE}}_k/(N-k)}} = \sqrt{\frac{\text{SSE}_k}{\widehat{\text{SSE}}_k}} \tag{3}$$

Note that because $\widehat{\text{SSE}}_k$ will tend to 0 as we increase k, eventually this will become unstable for too large k. Depending on our objective, either the smallest value or the last value below 1 (or below a suitable threshold such as 0.99) can be chosen as "best" k. Figure 2 plots this score for some of the above data sets. The normal distribution never scores below 1, and the uniform distribution remains very close to 1; hence these data sets can be recognized as unclustered. For the many blobs data set, the largest reduction is obtained for $k = 8$, and the last reduction is at $k = 25$, the number of generated clusters in this data set. To better understand why both of these solutions are of interest, Figure 3 shows that when going from $k = 7$ to $k = 8$ we observe structural changes that substantially improved the clustering result (such as separating the far cluster in the bottom, but also improving the clustering in the center), whereas the last improvement from $k = 24$ to $k = 25$ only affected three overlapping blobs in the center, that previously were split into two and now correctly into three clusters. For $k = 8$ we have the best structural improvement, but for $k = 25$ we get the finest clustering; both may have their use cases. The solution $k = 21$ is preferred by many distance-based measures, it is worth noting that well-separated blobs are separated, but touching blobs are still joined at this k.

Instead of using the Elbow heuristic, most "intrinsic" cluster evaluation criteria can be used. An extensive survey was published by Arbelaitz et al. [2], we only discuss a few examples of particular interest here. This involves some of the best-performing indexes in this study, namely the Silhouette, the VRC, and the DB-Index.

3.4 Variance-based criteria

So have we found a new, better method to choose the number of clusters k? The above "novel" approach is very similar to the **Variance Ratio Criterion (VRC)** published already in the mid-70s by Calinski and Harabasz [6]:

$$\text{VRC} := \frac{\text{SSE}_1 - \text{SSE}_k}{k-1} \Big/ \frac{\text{SSE}_k}{n-k} \tag{4}$$

(using the fact that $\text{BGSS} = \text{TSS} - \text{WGSS} = \text{SSE}_1 - \text{SSE}_k$). As they noted, this is analogous to the F-statistic used when testing for the significance of a difference in means, but we must not use such a significance test here, because we optimized the means (which makes such a test on the difference in means invalid). There are many more methods discussed in the 70s and 80s literature overlooked by many machine learning scientists of today, that we do not have the space to discuss here, we only briefly highlight some starting points. Marriott [23] (and before, Friedman and Rubin [14]) analyze the determinant of the variance-covariance matrix $|W|$, containing the within-class scatter, also known as generalized variance, instead of the regular variance $\text{tr}(W)$, and discuss that the expected change when partitioning into k clusters is a reduction by $1/k^2$. They argue this approach is superior because it takes correlations into account. Krzanowski and Lai [20] depart from Marriott and return to using the trace again. They argue that the variance is expected to decrease by $k^{\frac{2}{d}}$, define the successive difference as $\text{Diff}_k = (k-1)^{\frac{2}{p}}\,\text{SSE}_{k-1} - k^{\frac{2}{p}}\,\text{SSE}_k$, and then find the maximum of $KL(k) := |\text{Diff}_k/\text{Diff}_{k+1}|$. When Diff_{k+1} becomes small, this can become unstable, explaining the poor performance in our experiments.

Pham et al. [28] propose a scoring function for $k \geq 2$ based on $\text{SSE}_k/(\alpha_k\,\text{SSE}_{k-1})$, where the weights $\alpha_2 = 1 - \frac{3}{4d}$ and $\alpha_k = \frac{5}{6}\alpha_{k-1} + \frac{1}{6}$ model an expected change on a uniform distribution.

3.5 Distance-based criteria

The Dunn [10] index compares the diameter of clusters to the cluster separation. It exists in several variations, in the most basic form it is defined as the ratio of the smallest cluster separation to the largest cluster diameter,

$$\text{Dunn} := \frac{\min_i \min_{j\neq i} \min_{x\in C_i} \min_{y\in C_y} d(x,y)}{\max_i \max_{x\in C_i} \max_{y\in C_i} d(x,y)} .$$

In this basic (original) version, only the smallest cross-cluster distance and the largest inter-cluster distance are taken into account, but we might also consider averages instead [2]. The Davies-Bouldin-Index [7] compares the distance to the nearest other cluster with the radius of the two clusters. This is then averaged over all clusters:

$$\text{DB} := \frac{1}{k}\sum_i \max_{j\neq i} \frac{S_i + S_j}{M_{ij}}$$

where S_i is the (arithmetic, or root-mean-square) average distance of points to their cluster center (and hence a kind of radius), and M_{ij} is the distance between the cluster centers. When using the root-mean-square averages, S_i is the average distance of points within the cluster, and M_{ij} is the average distance between points in different clusters.

One of the most used distance-based criteria is the average silhouette width measure [29], which compares the average distance of each point to its own cluster to the average distance to the nearest other cluster. This method is closely related to k-medoids clustering and the PAM algorithm [17; 34], which cluster the data around k representative objects (called medoids), minimizing the distances to the medoids. In contrast to k-means (which minimizes squared errors), this method can also be used to optimize Euclidean or Manhattan distance; but it is mostly of interest for distances where the mean is not useful.[2] As Silhouette is fairly expensive to compute, it can be simplified by using the distance from the cluster center or medoid instead of the average distance. But it turns out that we can try to directly optimize this measure using PAM-like algorithms [41; 21].

3.6 Information-theoretic criteria

A different idea to choose the optimum number of clusters is based on the principle of minimum description length. Here, a k-means solution is considered better, if the data can be encoded more compactly. Increasing the number of centers means that the input data is approximated more closely (and hence needs less to encode the deviations), but at the same time, we also need to store more cluster centers. This intuition nicely fits the idea of approximating data and data quantization. X-means [27] integrates this with k-means in an algorithm that dynamically increases the number of clusters as long as a cluster quality criterion improves. They proposed to use the Bayesian Information Criterion (BIC) of Schwarz [36], who also proposed the Akaike Information Criterion (AIC). The original X-means version appears to have an error, the fixed equation of Foglia and Hancock [13] appears to work better. G-means [15] uses Anderson-Darling tests instead to decide when to accept a new cluster, and when to stop increasing k.

[2]While the arithmetic means in k-means do *not* minimize Euclidean or Manhattan distance, it is often good enough to be useful for many applications.

3.7 Simulation-based criteria

Tibshirani et al. [40] propose the gap statistic, which estimates a baseline SSE'_k obtained by clustering uniform random data sets. They then choose a k using

$$\text{Gap}_k := E[\log \text{SSE}'_k] - \log \text{SSE}_k ,$$

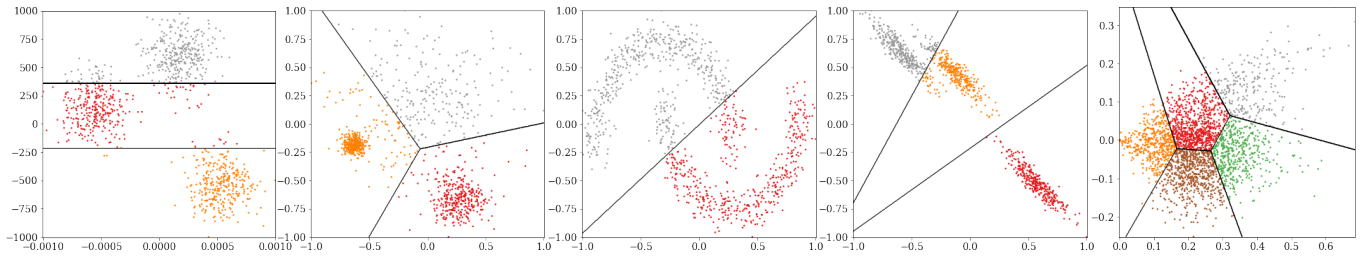
and picking the smallest k such that $\text{Gap}_k \geq \text{Gap}_{k-1} - s_{k+1}$ where s_{k+1} is the standard deviation of the estimates. This works decently well for synthetic data, but most interestingly it failed to recognize the uniform data as unclustered. For the more challenging data sets, the estimated number of clusters was unstable with the default sample sizes.

As we are not convinced that the "novel" approach we constructed above is clearly superior to VRC, BIC, or the Gap statistic, we suggest that you simply use one of these approaches to choose k, and rather pay attention to the way you preprocess your data for k-means. Because "garbage in, garbage out" – if your data is not prepared well, none of the clustering results will be good.

4. THE TRUE CHALLENGES OF K-MEANS

While the difficulty of choosing k is easily noticed by the user, as he has to specify this parameter, it nevertheless remains much more difficult to obtain meaningful and useful results from k-means than commonly anticipated. If we study the foundations of k-means, and the relationship to Gaussian mixture modeling, we can observe that k-means assumes errors to be invariant across the entire data space, whereas in full Gaussian mixture modeling, the deviation from a cluster in certain directions weight less than in other, and depend on the individual clusters. We can consider k-means as a limit case of Gaussian mixture modeling, where we perform (i) all clusters have an identical, spherical shape, and (ii) we make hard cluster assignments, for example by making the cluster standard deviations tend to zero. We will not go into (ii) here in detail (see, e.g., Bishop [3]). The observation of interest is that in k-means we somewhat assume that all clusters have the same spherical shape. This is simply a consequence of the sum of squared errors (Eq. 1) not including any weights depending on the cluster or axis. This is a reasonable simplification if we assume that our data set was generated from k pure signals (corresponding to the cluster centers) plus i.i.d. Gaussian noise.

This also leads to many situations where k-means will not work well: for example (i) if the axes have very different scales, and clusters are separated on the scales of low variance as in Figure 4a, (ii) if the cluster diameters are very different, yet the clusters are close, as in Figure 4b and (iii) when the clusters are not generated by Gaussian errors around an origin but have a non-convex shape as in Figure 4c and common in geodata, (iv) there are correlations in the data and some directions are more important than others, as in Figure 4d, (v) the input data is not continuous, or (vi) the similarity of objects is not well captured by Euclidean distance. When dealing with complex data, such as text data, it is fairly common that we first have to "vectorize" it, for example using TF-IDF, and/or applying some dimensionality reduction technique such as principal components analysis (PCA). Figure 4e shows such a data set, containing 5 groups from the well-known 20newsgroups data set, reduced to two dimensions with TF-IDF and PCA. It exhibits a typical "conical" shape with a tip at the zero, then



(a) Poorly scaled (b) Varying diameter (c) Non-convex (d) Correlated (e) Poorly transformed

Figure 4: Examples of data sets where k-means cannot be expected to work well.

extending into the first component, often seen with PCA on sparse input data, with the first component often capturing an overall vector length. In this case, all signal suitable for clustering was destroyed by this naïve preprocessing – yet this is a common combination of preprocessing techniques recommended in various blogs. The first two cases and the fourth case can be solved much better by using Gaussian mixture modeling [8], in the third case – common for example in data that follows geographic features – DBSCAN [11; 35] and other density-based clustering algorithms often are a better choice. For (v), the choice of a suitable clustering algorithm tends to become difficult, although modifications of k-means to categorical variables exist, for example, the k-modes algorithm [16]; There exist many clustering algorithms that allow using other distance functions, such as classic hierarchical clustering, k-medoids clustering [18; 33], spherical k-means [9; 32] and DBSCAN [11; 35]. When looking at clustering results reported using k-means, in many cases the results suffer from at least one of these additional problems as well.

5. CONCLUSION

Given the prevalence of the elbow method in education, online media (such as Wikipedia[3]), and even clustering research (as evidenced by the many proposals to automatically identify an elbow), it appears to be due to warn of using this method and to emphasize that much better alternatives such as the variance-ratio criterion (VRC) of Calinski and Harabasz [6], the Bayesian Information Criterion (BIC), or the Gap statistics should always be preferred instead. While the problems of the elbow approach have been discussed several times in the literature (e.g., [24; 19]), this knowledge of clustering basics appears to have been largely forgotten in today's machine learning community and hence needs to be communicated again. Educators should omit the method or at least explain better alternatives. Data scientists must be made wary of drawing conclusions from clustering results because of such problems, and must not rely on evaluation measures telling them what is "best". Reviewers of scientific literature should probably even reject conclusions drawn from choosing the "optimal" k using such an unreliable method. In the long run, we must accept that there is no "optimal" solution in cluster analysis, but it is an explorative approach that may yield multiple interesting solutions, and interestingness necessarily is a subjective decision of the user.

[3]E.g, `https://en.wikipedia.org/w/index.php?title=Elbow_method_(clustering)&oldid=1099441401` as of 2022

6. REFERENCES

[1] Aloise, D., Deshpande, A., Hansen, P., and Popat, P. NP-hardness of euclidean sum-of-squares clustering. *Mach. Learn. 75*, 2 (2009), 245–248.

[2] Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., and Perona, I. An extensive comparative study of cluster validity indices. *Pattern Recognit. 46*, 1 (2013), 243–256.

[3] Bishop, C. M. *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer, 2007.

[4] Bock, H.-H. *Clustering Methods: A History of k-Means Algorithms*. Springer, 2007, pp. 161–172.

[5] Bonner, R. E. On some clustering techniques. *IBM J. Res. Dev. 8*, 1 (1964), 22–32.

[6] Caliński, T., and Harabasz, J. A dendrite method for cluster analysis. *Communications in Statistics 3*, 1 (1974), 1–27.

[7] Davies, D. L., and Bouldin, D. W. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-1*, 2 (1979), 224–227.

[8] Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological) 39*, 1 (1977), 1–22.

[9] Dhillon, I. S., and Modha, D. S. Concept decompositions for large sparse text data using clustering. *Mach. Learn. 42*, 1/2 (2001), 143–175.

[10] Dunn, J. C. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics 3*, 3 (1973), 32–57.

[11] Ester, M., Kriegel, H., Sander, J., and Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery and Data Mining, KDD* (1996), pp. 226–231.

[12] Estivill-Castro, V. Why so many clustering algorithms: a position paper. *SIGKDD Explor. 4*, 1 (2002), 65–75.

[13] Foglia, A., and Hancock, B. Notes on bayesian information criterion calculation for x-means clustering, 2012.

[14] Friedman, H. P., and Rubin, J. On some invariant criteria for grouping data. *Journal of the American Statistical Association 62*, 320 (1967), 1159–1178.

[15] Hamerly, G., and Elkan, C. Learning the k in k-means. In *Neural Information Processing Systems, NIPS* (2003), pp. 281–288.

[16] Huang, Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov. 2*, 3 (1998), 283–304.

[17] Kaufman, L., and Rousseeuw, P. J. Clustering by means of medoids. In *Statistical Data Analysis Based on the L_1 Norm and Related Methods*, Y. Dodge, Ed. North-Holland, 1987, pp. 405–416.

[18] Kaufman, L., and Rousseeuw, P. J. Partitioning around medoids (program PAM). In *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Ltd, 1990, ch. 2, pp. 68–125.

[19] Ketchen, D. J., and Shook, C. L. The application of cluster analysis in strategic management research: An analysis and critique. *Strategic Management Journal 17*, 6 (1996), 441–458.

[20] Krzanowski, W. J., and Lai, Y. T. A criterion for determining the number of groups in a data set using sum-of-squares clustering. *Biometrics 44*, 1 (1988), 23–34.

[21] Lenssen, L., and Schubert, E. Clustering by direct optimization of the medoid silhouette. In *Similarity Search and Applications* (2022), pp. 190–204.

[22] Mahajan, M., Nimbhorkar, P., and Varadarajan, K. R. The planar k-means problem is NP-hard. In *WALCOM: Algorithms and Computation* (2009), pp. 274–285.

[23] Marriott, F. H. C. Practical problems in a method of cluster analysis. *Biometrics 27*, 3 (1971), 501–514.

[24] Milligan, G. W., and Cooper, M. C. An examination of procedures for determining the number of clusters in a data set. *Psychometrika 50*, 2 (June 1985), 159–179.

[25] Novikov, A. PyClustering: Data mining library. *Journal of Open Source Software 4*, 36 (2019), 1230.

[26] Onumanyi, A. J., Molokomme, D. N., Isaac, S. J., and Abu-Mahfouz, A. M. Autoelbow: An automatic elbow detection method for estimating the number of clusters in a dataset. *Applied Sciences 12*, 15 (2022).

[27] Pelleg, D., and Moore, A. W. X-means: Extending k-means with efficient estimation of the number of clusters. In *Int. Conf. Machine Learning (ICML)* (2000), pp. 727–734.

[28] Pham, D. T., Dimov, S. S., and Nguyen, C. D. Selection of k in k-means clustering. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 219*, 1 (2005), 103–119.

[29] Rousseeuw, P. J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math. 20* (1987), 53–65.

[30] Salvador, S., and Chan, P. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *Tools with Artificial Intelligence (ICTAI)* (2004), pp. 576–584.

[31] Satopää, V., Albrecht, J. R., Irwin, D. E., and Raghavan, B. Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In *Distributed Computing Systems (ICDCS) Workshops* (2011), pp. 166–171.

[32] Schubert, E., Lang, A., and Feher, G. Accelerating spherical k-means. In *Similarity Search and Applications* (2021), pp. 217–231.

[33] Schubert, E., and Rousseeuw, P. J. Faster k-medoids clustering: Improving the PAM, CLARA, and CLARANS algorithms. In *Similarity Search and Applications, SISAP* (2019), pp. 171–187.

[34] Schubert, E., and Rousseeuw, P. J. Fast and eager k-medoids clustering: O(k) runtime improvement of the PAM, CLARA, and CLARANS algorithms. *Inf. Syst. 101* (2021), 101804.

[35] Schubert, E., Sander, J., Ester, M., Kriegel, H., and Xu, X. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst. 42*, 3 (2017), 19:1–19:21.

[36] Schwarz, G. Estimating the Dimension of a Model. *The Annals of Statistics 6*, 2 (1978), 461 – 464.

[37] Shi, C., Wei, B., Wei, S., Wang, W., Liu, H., and Liu, J. A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm. *EURASIP J. Wirel. Commun. Netw. 2021*, 1 (2021), 31.

[38] Sugar, C. A., and James, G. M. Finding the number of clusters in a dataset: An information-theoretic approach. *Journal of the American Statistical Association 98*, 463 (2003), 750–763.

[39] Thorndike, R. L. Who belongs in the family? *Psychometrika 18*, 4 (1953), 267–276.

[40] Tibshirani, R., Walther, G., and Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63*, 2 (2001), 411–423.

[41] Van der Laan, M., Pollard, K., and Bryan, J. A new partitioning around medoids algorithm. *Journal of Statistical Computation and Simulation 73*, 8 (2003), 575–584.

[42] Zhang, Y., Mandziuk, J., Quek, H. C., and Goh, W. Curvature-based method for determining the number of clusters. *Inf. Sci. 415* (2017), 414–428.