k is the Magic Number — Inferring the Number of Clusters Through Nonparametric Concentration Inequalities

Sibylle Hess (✉) and Wouter Duivesteijn

Data Mining Group, Technische Universiteit Eindhoven, Eindhoven, the Netherlands
{s.c.hess,w.duivesteijn}@tue.nl

Abstract. Most convex and nonconvex clustering algorithms come with one crucial parameter: the k in k-means. To this day, there is not one generally accepted way to accurately determine this parameter. Popular methods are simple yet theoretically unfounded, such as searching for an elbow in the curve of a given cost measure. In contrast, statistically founded methods often make strict assumptions over the data distribution or come with their own optimization scheme for the clustering objective. This limits either the set of applicable datasets or clustering algorithms. In this paper, we strive to determine the number of clusters by answering a simple question: given two clusters, is it likely that they jointly stem from a single distribution? To this end, we propose a bound on the probability that two clusters originate from the distribution of the unified cluster, specified only by the sample mean and variance. Our method is applicable as a simple wrapper to the result of any clustering method minimizing the objective of k-means, which includes Gaussian mixtures and Spectral Clustering. We focus in our experimental evaluation on an application for nonconvex clustering and demonstrate the suitability of our theoretical results. Our SpecialK clustering algorithm automatically determines the appropriate value for k, without requiring any data transformation or projection, and without assumptions on the data distribution. Additionally, it is capable to decide that the data consists of only a single cluster, which many existing algorithms cannot.

Keywords: k-means · concentration inequalities · spectral clustering · model selection · one-cluster clustering · nonparametric statistics.

1 Introduction

When creating a solution to the task of clustering —finding a natural partitioning of the records of a dataset into k groups— the holy grail is to automatically determine k. The current state of the art in clustering research has not yet achieved this holy grail to a satisfactory degree. Since many papers on parameter-free clustering exist, this statement might sound unnecessarily polemic without further elaboration. Hence, we must describe what constitutes "satisfactory", which we will do by describing unsatisfactory aspects of otherwise perfectly fine (and often actually quite interesting) clustering solutions. Some solutions can only handle convex cluster shapes [17], while most real-life phenomena are not necessarily convex. Some solutions manage to avoid determining k, at the cost of having to specify another parameter that effectively controls k [1]. Some solutions define a cluster criterion and an algorithm to iteratively mine the data: the best single cluster is found, after which the algorithm is run again on the data minus that cluster, etcetera [11]; this runs the risk of finding a local optimum. We, by contrast, introduce a clustering algorithm that can handle nonconvex shapes, is devoid of parameters that demand the user to directly or indirectly set k, and finds the global optimum for its optimization criterion.

We propose a probability bound on the operator norm of centered, symmetric decompositions based on the matrix Bernstein concentration inequality. We apply this bound to assess whether two given clusters are likely to stem from the distribution of the unified cluster. Our bound provides a statistically founded decision criterion over the minimum similarity within one cluster and the maximum similarity between two clusters: this entails judgment on whether two clusters should be separate or unified. Our method is easy to implement and statistically nonparametric. Applied on spectral clustering methods, to the best of the authors' knowledge, providing a statistically founded way to automatically determine k is entirely new.

We incorporate our bound in an algorithm called SpecialK, since it provides a method for SPEctral Clustering to Infer the Appropriate Level k. On synthetic datasets, SpecialK outperforms some competitors, while performing roughly on par with another competitor. However, when unleashed on a synthetic dataset consisting of just random noise, all competitors detect all kinds of clusters, while only SpecialK correctly determines the value of k to be one. If you need an algorithm that can correctly identify a sea of noise, SpecialK is the only choice. On four real-life datasets with class labels associated to the data points, we illustrate how all available algorithms, including SpecialK, often cannot correctly determine the value of k corresponding to the number of classes available in the dataset. We argue that this is an artefact of a methodological mismatch: the class labels indicate one specific natural grouping of the points in the dataset, but the task of clustering is to retrieve *any* natural grouping of the points in the dataset, not necessarily the one encoded in the class label. Hence, such an evaluation is fundamentally unfit to assess the performance of clustering methods.

2 Three Sides of a Coin: *k*-means, Gaussian Mixtures and Spectral Clustering

To embed our work in already existing related work, and to describe the preliminary building blocks required as foundation on which to build our work, we must first introduce some notation. We write $\mathbf{1}$ for a constant vector of ones, whose dimension can be derived from context unless otherwise specified. We denote with $\mathbb{1}^{m\times k}$ the set of all binary matrices which indicate a partition of m elements into k sets. Such a partition is computed by k-means clustering;

every element belongs to exactly one cluster. Let $D \in \mathbb{R}^{m\times n}$ be a data matrix, collecting m points $D_{j\cdot}$, which we identify with their index j. The objective of k-means is equivalent to solving the following matrix factorization problem:

$$\min_{Y,X} \left\|D - YX^\top\right\|^2 \quad \text{s.t.} \quad Y \in \mathbb{1}^{m\times k}, X \in \mathbb{R}^{n\times k}. \tag{1}$$

The matrix Y indicates the cluster assignments; point j is in cluster c if $Y_{jc} = 1$. The matrix X represents the cluster centers, which are given in matrix notation as $X = D^\top Y \left(Y^\top Y\right)^{-1}$. The well-known optimization scheme of k-means, *Lloyd's algorithm* [17], employs the convexity of the k-means problem if one of the matrices X or Y is fixed. The algorithm performs an alternating minimization, updating Y to assign each point to the cluster with the nearest center, and updating $X_{\cdot c}$ as the mean of all points assigned to cluster c.

2.1 Gaussian Mixtures

The updates of Lloyd's algorithm correspond to the expectation and maximization steps of the EM-algorithm [3, 2]. In this probabilistic view, we assume that every data point $D_{j\cdot}$ is independently sampled by first selecting cluster c with probability π_c, and then sampling point $D_{j\cdot}$ from the Gaussian distribution:

$$p(\xi|c) = \frac{1}{\sqrt{2\pi\epsilon}} \exp\left(-\frac{1}{2\epsilon}\left\|\xi - X_{\cdot c}\right\|^2\right) \sim \mathcal{N}(\xi|X_{\cdot c}, \epsilon I).$$

This assumes that the covariance matrix of the Gaussian distribution is equal for all clusters: $\Sigma_c = \epsilon I$. From this sampling procedure, we compute the log-likelihood for the data and cluster assignments:

$$\begin{aligned}\log p(D, Y|X, \epsilon I, \pi) &= \log\left(\prod_{j=1}^{m}\prod_{c=1}^{k}\left(\frac{\pi_c}{\sqrt{2\pi\epsilon}}\exp\left(-\frac{1}{2\epsilon}\left\|D_{j\cdot} - X_{\cdot c}^\top\right\|^2\right)\right)^{Y_{jc}}\right)\\ &= \sum_{j=1}^{m}\sum_{c=1}^{k} Y_{jc}\left(\ln\left(\frac{\pi_c}{\sqrt{2\pi\epsilon}}\right) - \frac{1}{2\epsilon}\left\|D_{j\cdot} - X_{\cdot c}^\top\right\|^2\right)\\ &= -\frac{1}{2\epsilon}\left\|D - YX^\top\right\|^2 - \frac{m}{2}\ln(2\pi\epsilon) + \sum_{c=1}^{k}|Y_{\cdot c}|\ln\left(\pi_c\right).\end{aligned}$$

Hence, if $\epsilon > 0$ is small enough, maximizing the log-likelihood of the Gaussian mixture model is equivalent to solving the k-means problem.

2.2 Maximum Similarity versus Minimum Cut

There are multiple alternative formulations of the k-means problem. Altering the Frobenius norm in Equation (1) with the identity $\|A\|^2 = \mathrm{tr}(AA^\top)$ begets:

$$\begin{aligned}\left\|D - YX^\top\right\|^2 &= \|D\|^2 - 2\,\mathrm{tr}\left(D^\top YX^\top\right) + \mathrm{tr}\left(XY^\top YX^\top\right)\\ &= \|D\|^2 - \mathrm{tr}\left(Y^\top DD^\top Y\left(Y^\top Y\right)^{-1}\right),\end{aligned}$$

where the last equality derives from inserting the optimal X, given Y. Thus, we transform the k-means objective, defined in terms of distances to cluster centers, to an objective defined solely on similarity of data points. The matrix $DD^\top$ represents similarity between points, measured via the inner product $sim(j,l) = D_{j\cdot}D_{l\cdot}^\top$. The k-means objective in Equation (1) is thus equivalent to the *maximum similarity problem* for a similarity matrix $W = DD^\top$:

$$\max_{Y\in\mathbb{1}^{m\times k}} Sim(W,Y) = \sum_c R(W, Y_{\cdot c}), \quad R(W,y) = \frac{y^\top W y}{|y|}. \tag{2}$$

Here, we introduce the function $R(W,y)$, which is known as the *Rayleigh coefficient* [10], returning the ratio similarity of points within cluster y.

An alternative to maximizing the *similarity within* a cluster, is to minimize the *similarity between* clusters. This is known as the *ratio cut* problem, stated for a symmetric similarity matrix W as:

$$\min_{Y\in\mathbb{1}^{m\times k}} Cut(W,Y) = \sum_s C(W,Y), \quad C(W,y) = \frac{y^\top W \bar{y}}{|y|}.$$

The function $C(W,y)$ sums the similarities between points indicated by cluster y and the remaining points indicated by $\bar{y} = \mathbf{1} - y$. Imagining the similarity matrix W as a weighted adjacency matrix of a graph, the function $C(W,y)$ sums the weights of the edges which would be cut if we *cut out* the cluster y from the graph. Defining the matrix $L = \operatorname{diag}(W\mathbf{1}) - W$, also known as the *difference graph Laplacian* [5], we have $C(W,y) = R(L,y)$. As a result, the maximum similarity problem with respect to the similarity matrix $-L$ is equivalent to the minimum cut problem with similarity matrix W.

2.3 Spectral Clustering

If similarities are defined via the inner product, then the similarity in Equation (2) is maximized when *every* point in a cluster is similar to *every other* point in that cluster. As a result, the obtained clusters by k-means have convex shapes. If we expect nonconvex cluster shapes, then our similarities should only locally be compared. This is possible, e.g., by defining the similarity matrix as the adjacency matrix to the kNN graph or the ϵ-neighborhood graph. Clustering methods employing such similarities are known as *spectral clustering* [20]. It is related to minimizing the cut for the graph Laplacian of a given similarity matrix. Spectral clustering computes a truncated eigendecomposition of $W = -L \approx V^{(k+1)}\Lambda^{(k+1)}{V^{(k+1)}}^\top$, where $\Lambda^{(k+1)}$ is a diagonal matrix having the $(k+1)$ largest eigenvalues on its diagonal $\Lambda_{11} \geq \ldots \geq \Lambda_{k+1k+1}$, and $V^{(k+1)}$ represents the corresponding eigenvectors. Graph Laplacian theory says that the eigenvectors to the largest eigenvalue indicate the connected components of the graph, while in practical clustering application the entire graph is assumed to be connected. To honor this assumption, the first eigenvector is omitted from the matrix $V^{(k+1)}$, which is subsequently discretized by k-means clustering [18].

Considering the relation between the minimum cut objective and k-means clustering, the objective to minimize the $Cut(L,Y)$ is actually equivalent to solving k-means clustering for a data matrix D such that $W = DD^\top$. This relation was recently examined [9], with the disillusioning result that k-means clustering on the decomposition matrix D usually returns a local optimum, whose objective value is close to the global minimum but whose clustering is unfavorable. Consequently, the authors propose the algorithm SpectACl, approximating the similarity matrix by a matrix product of projected eigenvectors, such that:

$$W \approx DD^\top, \quad D_{ji} = \left|V_{ji}^{(n)}\right| \left|\Lambda_{ii}^{(n)}\right|^{-(1/2)} \tag{3}$$

for a large enough dimensionality $n > k$. Although this increases the rank of the factorization from k in traditional spectral clustering to $n > k$, the search space, which is spanned by the vectors $D_{\cdot i}$, is reduced in SpectACl. The projection of the orthogonal eigenvectors $V_{\cdot i}$ to the positive orthant introduces linear dependencies among the projections $D_{\cdot i}$.

2.4 Estimating *k*

Depending on the view on k-means clustering —as a matrix factorization, a Gaussian mixture model, or a graph-cut algorithm— we might define various strategies to derive the correct k. The *elbow* strategy is arguably the most general approach. Plotting the value of the objective function for every model when increasing k, a kink in the curve is supposed to indicate the correct number of clusters. With regard to spectral clustering, the elbow method is usually deployed on the largest eigenvalues of the Laplacian, called *eigengap heuristic* [18]. Depending on the application, the elbow may not be easy to spot, and the selection of k boils down to a subjective trade-off between data approximation and model complexity.

To manage this trade-off in a less subjective manner, one can define a cost measure beforehand. Popular approaches employ Minimum Description Length (MDL) [4, 14] or the Bayesian Information Criterion (BIC) [21]. The nonconvex clustering method Self-Tuning Spectral Clustering (STSC) [24] defines such a cost measure on the basis of spectral properties of the graph Laplacian. The k-means discretization step is replaced by the minimization of this cost measure, resulting in a rotated eigenvector matrix which approximates the form of partitioning matrices, having only one nonzero entry in every row. The definition of the cost measure derives from the observation that a suitable rotation of the eigenvectors also defines a transformation of the graph Laplacian into a block-diagonal form. In this form, the connected components in the graph represent the clustering structure. STSC then chooses the largest number k obtaining minimal costs, from a set of considered numbers of clusters.

The definition of a cost measure may also rely on statistical properties of the dataset. Tibshirani et al. deliver the statistical foundations for the elbow method with the gap statistic [22]. Given a reference distribution, the gap statistic chooses the value of k for which the gap between the approximation error and its expected value is the largest. The expected value is estimated by sampling the data of the reference distribution, and computing a clustering for every sampled dataset and every setting of k.

The score-based methods cannot deliver a guarantee over the quality of the gained model. This is where statistical methods come into play, whose decisions over the number of clusters are based on statistical tests. GMeans [7] performs statistical tests for the hypothesis that the points in one cluster are Gaussian distributed. PGMeans [6] improves over GMeans, by applying the Kolmogorov-Smirnov test for the goodness of fit between one-dimensional random projections of the data and the Gaussian mixture model. They empirically show that this approach is also suitable for non-Gaussian data. An alternative to the Normality assumption is to assume that every cluster follows a unimodal distribution in a suitable space, which can be validated by the dip test. DipMeans provides a wrapper for k-means-related algorithms, testing for individual data points whether the distances to other points follow a unimodal distribution [12]. Maurus and Plant argue that this approach is sensitive to noise and propose the algorithm SkinnyDip, focusing on scenarios with high background noise [19]. Here, the authors assume that a basis transformation exists such that the clusters form a unimodal shape in all coordinate directions. All these approaches require a data transformation or projection, in order to apply the one-dimensional tests.

3 A Nonparametric Bound

We propose a bound on the probability that a specific pair of clusters is generated by a single cluster distribution. Our bound relies on concentration inequalities, which have as input the mean and variance of the unified cluster distribution, which are easy to estimate. No assumptions on the distribution shape (e.g., Gaussian) must be made, and no projection is required. The core concentration inequality which we employ is the matrix Bernstein inequality.

Theorem 1 (Matrix Bernstein [23, Theorem 1.4]). *Consider a sequence of independent, random, symmetric matrices $A_i \in \mathbb{R}^{m\times m}$ for $1 \leq i \leq n$. Assume that each random matrix satisfies:*

$$\mathbb{E}[A_i] = \mathbf{0} \qquad \text{and} \qquad \|A_i\|_{op} \leq \nu \text{ almost surely,}$$

and set $\sigma^2 = \left\|\sum_i \mathbb{E}[A_i^2]\right\|_{op}$. Then, for all $t \geq 0$:

$$\mathbb{P}\left(\left\|\sum_i A_i\right\|_{op} \geq t\right) \leq m \exp\left(-\frac{1}{2}\frac{t^2}{\sigma^2 + \nu t/3}\right).$$

The matrix Bernstein bound employs the operator norm. For real-valued, symmetric matrices this equals the maximum eigenvalue in magnitude:

$$\|A\|_{op} = \sup_{\|x\|=1} \|Ax\| = \max_{1\leq j\leq m} |\lambda_j(A)| = \max_{x\in\mathbb{R}^m} R(A,x). \tag{4}$$

The relationship to the Rayleigh coefficient is important. This relationship is easy to derive by substituting A with its eigendecomposition. We derive the following central result for the product matrix of centered random matrices.

Theorem 2 (ZZ Top Bound). *Let $Z_{\cdot i} \in \mathbb{R}^m$ be independent samples of a random vector with mean zero, such that $\|Z_{\cdot i}\| \leq 1$ for $1 \leq i \leq n$. Further, assume that $\mathbb{E}[Z_{ji}Z_{li}] = 0$ for $j \neq l$ and $\mathbb{E}[Z_{ji}^2] = \sigma^2$ for $0 < \sigma^2 < 1$ and $1 \leq j \leq m$. Then, for $t > 0$:*

$$\mathbb{P}\left(\left\|ZZ^\top - n\sigma^2 I\right\|_{op} \geq t\right) \leq m \exp\left(-\frac{1}{2} \cdot \frac{t^2}{n\sigma^2 + t/3}\right).$$

Proof. We apply the matrix Bernstein inequality (Theorem 1) to the sum of random matrices:

$$ZZ^\top - n\sigma^2 I = \sum_i \left(Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2 I\right).$$

Assuming that the expected values satisfy $\mathbb{E}[Z_{ji}Z_{li}] = 0$ for $j \neq l$ and $\mathbb{E}[Z_{ji}Z_{ji}] = \sigma^2$, yields that the random matrix $A_i = Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2$ has mean zero:

$$\mathbb{E}\left[A_i\right] = \mathbb{E}\left[Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2 I\right] = \left(\mathbb{E}\left[Z_{ji}Z_{li}\right]\right)_{jl} - \sigma^2 I = 0.$$

The operator norm of the random matrices A_i is bounded by one:

$$\begin{aligned}\|A_i\|_{op} &= \left\|Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2 I\right\|_{op} = \sup_{\|x\|=1} x^\top \left(Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2 I\right) x = \sup_{\|x\|=1} \left(Z_{\cdot i}^\top x\right)^2 - \sigma^2 \\ &\leq 1 - \sigma^2 \leq 1.\end{aligned}$$

The expected value of A_i^2 is:

$$\mathbb{E}\left[\left(Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2 I\right)\left(Z_{\cdot i}Z_{\cdot i}^\top - \sigma^2 I\right)\right] \leq \mathbb{E}\left[Z_{\cdot i}Z_{\cdot i}^\top - 2\sigma^2 Z_{\cdot i}Z_{\cdot i}^\top + \sigma^4 I\right] = \sigma^2 I.$$

Thus, the norm of the total variance is:

$$\left\|\sum_i \mathbb{E}\left[A_i A_i\right]\right\|_{op} = \left\|n\sigma^2 I\right\|_{op} = n\sigma^2.$$

The matrix Bernstein inequality then yields the result. □

Applying Equation (4) to Theorem 2 yields the following corollary.

Corollary 1. *Let $Z_{\cdot i} \in \mathbb{R}^m$ be independent samples of a random vector with mean zero, such that $\|Z_{\cdot i}\| \leq 1$ for $1 \leq i \leq n$. Further assume that $\mathbb{E}[Z_{\cdot i}Z_{\cdot i}^\top] = \sigma^2 I$ for $0 < \sigma^2 < 1$ and $1 \leq i \leq n$. Let $y \in \{0,1\}^m$ be an indicator vector of a cluster candidate, and denote:*

$$t = R\left(ZZ^\top, y\right) - n\sigma^2. \tag{5}$$

Then, the probability that an indicator vector $y^* \in \{0,1\}^m$ exists with a Rayleigh coefficient such that $R(ZZ^\top, y) \leq R(ZZ^\top, y^*)$, is bounded as follows:

$$\mathbb{P}\left(\max_{y^* \in \{0,1\}^m} R\left(ZZ^\top, y^*\right) - n\sigma^2 \geq t\right) \leq m \exp\left(-\frac{1}{2} \cdot \frac{t^2}{n\sigma^2 + t/3}\right).$$

In practice, we must estimate the mean and variance of a candidate cluster. In this case, the relationship between the Rayleigh coefficient for the centered random matrix Z and the original data matrix is specified as follows (assuming the data matrix D is reduced to the observations belonging to a single cluster).

Remark 1. Assume we want to bound the probability that two clusters indicated by $y, \bar{y} \in \{0,1\}^m$ are parts of one unified cluster represented by $D \in \mathbb{R}^{m \times n}$. We denote with $\mu = \frac{1}{m} D^\top \mathbf{1}$ the vector of sample means over the columns of D. The Rayleigh coefficient of y with respect to the columnwise centered matrix $Z = D - \mathbf{1}\mu^\top$ is equal to (see supplementary material [8] for full derivation):

$$R\left(ZZ^\top, y\right) = \frac{|\bar{y}|}{m}\left(\frac{|\bar{y}|}{m} R\left(DD^\top, y\right) - \frac{|y|}{m} Cut\left(DD^\top, [y \ \bar{y}]\right) + \frac{|y|}{m} R\left(DD^\top, \bar{y}\right)\right)$$

The higher $R\left(ZZ^\top, y\right)$ is, the higher t is in Equation (5), and the lower the probability is that y indicates a subset of the cluster represented by D. Remark 1 shows that the probability of y indicating a subset of the larger cluster D, is determined by three things: the similarity within each of the candidate clusters ($R(DD^\top, y)$ and $R(DD^\top, \bar{y})$), the rational cut between these clusters ($Cut\left(DD^\top, [y \ \bar{y}]\right)$), and the ratio of points belonging to the one versus the other cluster ($|y|$ versus $|\bar{y}|$). As a result, the ZZ Top Bound provides a natural balance of the within- and between-cluster similarity governing acceptance or rejection of a given clustering.

3.1 A Strategy to Find a Suitable Number of Clusters

Remark 1 and Corollary 1 provide a method to bound the probability that two clusters are generated by the same distribution. Let us go through an example to discuss how we can employ the proposed bounds in a practical setting. Imagine two clusterings, the one employing a larger cluster covering the records denoted by the index set $\mathcal{J} \subset \{1, \ldots, m\}$, having center $\mu \in \mathbb{R}^n$; the other containing a subset of $\mathcal{J}$, indicated by y. Assume the following.

Assumption 1. *If the indices $\mathcal{J}$ form a* true *cluster, then the columns $D_{\mathcal{J}i}$ are independent samples of a random vector with mean $\mu_i \mathbf{1} \in \mathbb{R}^{|\mathcal{J}|}$, for $\mu_i \in \mathbb{R}$.*

We then define the $|\mathcal{J}|$-dimensional scaled and centered sample vectors:

$$Z_{\cdot i} = \frac{1}{\|D_{\mathcal{J}i}\|}(D_{\mathcal{J}i} - \mu_i \mathbf{1}) \quad \text{for all } 1 \leq i \leq n.$$

Now, if one were to assume that $y \in \{0,1\}^m$ satisfies:

$$R(ZZ^\top, y) \geq \sqrt{2n\sigma^2 \ln\left(\frac{m}{\alpha}\right) + \frac{1}{9}\ln\left(\frac{m}{\alpha}\right)^2} + n\sigma^2 + \frac{1}{3}\ln\left(\frac{m}{\alpha}\right), \qquad (6)$$

then Corollary 1 implies that the probability of $\mathcal{J}$ being a true cluster and Equation (6) holding is at most α. Hence, if α is small enough, then we conclude that

Algorithm 1 SpecialK(W, n, α)

```
W ≈ $V^{(n)}\Lambda^{(n)}{V^{(n)}}^\top$                                  ▷ Compute truncated eigendecomposition
$D_{ji} = \left|V^{(n)}_{ji}\right| |\Lambda_{ii}|^{1/2}$                  ▷ For all $1 \le j \le m$, $1 \le i \le n$
for $k = 1, \dots$ do
    $Y^{(k)} \leftarrow$K-MEANS$(D, k)$
    for $c_1, c_2 \in \{1, \dots, k\}, c_1 > c_2$ do
        $\mathcal{J} \leftarrow \left\{j \middle| Y^{(k)}_{jc_1} + Y^{(k)}_{jc_2} > 0\right\}$
        $Z_{\cdot i} \leftarrow \frac{1}{\|D_{\mathcal{J}i}\|}\left(D_{\mathcal{J}i} - \frac{|D_{\mathcal{J}i}|}{|\mathcal{J}|}\mathbf{1}\right)$        ▷ For all $j \in \mathcal{J}$
        $\sigma^2 \leftarrow \frac{1}{n|\mathcal{J}|}\sum_{j,i} Z_{ji}^2$                          ▷ Sample variance
        $t \leftarrow \max\left\{R(ZZ^\top, Y_{\mathcal{J}c}) - n\sigma^2 \middle| c \in \{c_1, c_2\}\right\}$
        if $|\mathcal{J}| \exp\left(-\frac{1}{2} \cdot \frac{t^2}{n\sigma^2 + t/3}\right) > \alpha$ then
            return $Y^{(k-1)}$
```

$\mathcal{J}$ is not a true cluster; this conclusion is wrong only with the small probability α (which functions as a user-set significance level of a hypothesis test).

Assumption 1 may not hold for all datasets. In particular, the assumption that the column vectors of the data matrix (comprising points from only one cluster) are sampled with the same variance parameter and a mean vector which is reducible to a scaled constant one vector, is not generally valid. Especially if the features of the dataset come from varying domains, the cluster assumptions may not hold. In this paper, we evaluate the ZZ Top Bound in the scope of spectral clustering, where the feature domains are comparable; every feature corresponds to one eigenvector. In particular, we consider a decomposition of the similarity matrix as shown in Equation (3). The rank of this decomposition is independent from the expected number of k, unlike in traditional spectral clustering algorithms. Hence, a factor matrix D as computed in Equation (3) can be treated like ordinary k-means input data.

We propose Algorithm 1, called SpecialK, since it provides a method for SPEctral Clustering to Infer the Appropriate Level k. Its input is a similarity matrix W, the feature dimensionality of the computed embedding, and the significance level $\alpha > 0$. In the first two steps, the symmetric decomposition $W \approx DD^\top$ is computed. For an increasing number of clusters, we compute a k-means clustering. For every pair of clusters, we compute the probability that both clusters are actually subsets of the unified cluster. If this probability is larger than the significance level α, then we conclude that the current clustering splits an actual cluster into two and we return the previous model.

4 Experiments

In comparative experiments, our state-of-the-art competitors are Self-Tuning Spectral Clustering (STSC) [24] and PGMeans [6], whose implementations

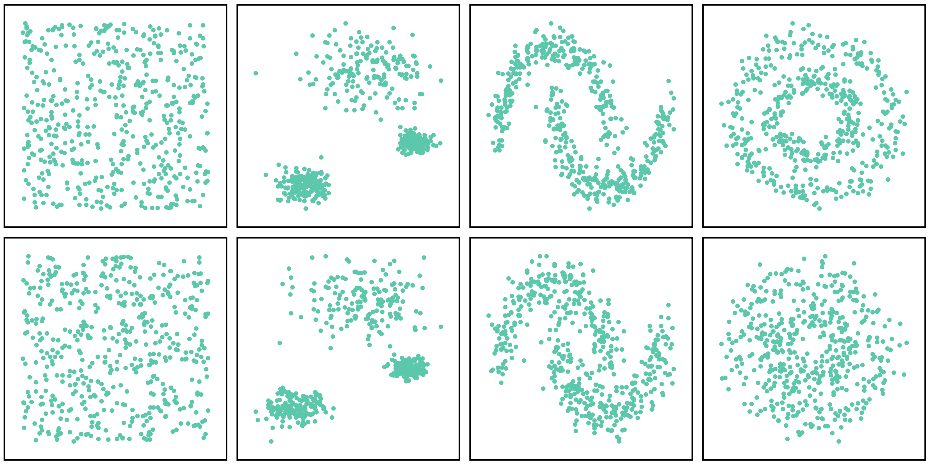

Fig. 1. Visualization of the datasets (from left to right: random, three blobs, two moons, and two circles) with noise parameters equal to 0.1 (top row) and 0.15 (bottom row).

have been kindly provided by the authors. Since we strive for applicability on nonconvex cluster shapes, we apply PGMeans to the projected eigenvectors (as computed in Line 2 in Algorithm 1) of a given similarity matrix. We set for PGMeans the significance level for every test of 12 random projections to $\alpha = 0.01/12$. We also included SkinnyDip [19] in our evaluation. However, applying this algorithm on the decomposition matrix results in a vast number of returned clusters ($\hat{k} \approx 50$ while the actual value is $k \leq 3$) where most of the data points are attributed to noise. Since this result is clearly wrong, we eschew further comparison with this algorithm.

We consider two variants of similarity matrices: W_R and W_C. The former employs the ϵ-neighborhood adjacency matrix, where ϵ is set such that 99% of the data points have at least ten neighbors; the latter employs the symmetrically normalized adjacency matrix of the kNN graph. STSC computes its own similarity matrix for a given set of points. To do so, it requires a number of considered neighbors, which we set to the default value of 15. Note that the result of STSC comes with its own quality measurement for the computed models; higher is better. We provide a Python implementation of SpecialK[1]. In this implementation, we do not assess for all possible pairs of clusters if they emerge from one distribution, but only for the ten cluster pairs having the highest cut.

4.1 Synthetic Experiments

We generate benchmark datasets, using the scikit library for Python. Figure 1 provides examples of the generated datasets, which come in four types of seeded cluster shape (random, blobs, moons, and circles), and a noise parameter (set to

[1] `https://github.com/Sibylse/SpecialK`

0.1 and 0.15, respectively, in the figure). For each shape and noise specification, we generate $m = 1500$ data points. The noise is Gaussian, as provided by the scikit noise parameter (cf. `http://scikit-learn.org`). This parameter takes a numeric value, for which we investigate ten settings: we traverse the range $[0, 0.225]$ by increments of size 0.025. For every shape and noise setting, we generate five datasets. Unless otherwise specified, we employ a dimensionality of $n = 200$ as parameter for SpecialK. However, we use a different rank of $n = 50$ for PGMeans, whose results benefit from this setting. We set SpecialK's significance level to $\alpha = 0.01$. For all algorithms, we consider only values of $k \in \{1, \ldots, 5\}$; Figure 1 illustrates that higher values are clearly nonsense.

In Figure 2, we plot every method's estimated number of clusters for the four datasets (rows) and two similarity matrices (columns) W_R and W_C (since STSC employs its own similarity matrix, the plot with respect to STSC does not vary for these settings: STSC behaves exactly the same in left-column and right-column subplots on the same dataset). In every subplot, the x-axis denotes the setting for the noise parameter. On the y-axis we aggregate the number of clusters detected by each of the three competitors; the correct number of clusters (3 for blobs, 2 for moons and circles, and 1 for random) for each subplot is highlighted by a pink band. Every column in every subplot corresponds to a single setting of shape and noise; recall that we generated five version of the dataset for such a setting. The column now gathers for each of the three competitors (marked in various shapes; see the legend of the figure) which setting of k it determines to be true how often out of the five times. This frequency is represented by mark size: for instance, if PGMeans determines five distinct values of k for the five datasets, we get five small squares in the column, but if it determines the same value of k for all five datasets, we get one big square in the column. An algorithm performs well if many of its big marks fall in the highlighted band.

Figure 2 illustrates that PGMeans is all over the place. For the moons and circles datasets, it correctly identifies the number of clusters at low noise levels, but from a certain noise level onwards, it substantially overestimates k. On the moons dataset under W_R this behavior is subtle; under W_C and on the circles dataset the jump from 2 to 5 clusters is jarring. On the blobs dataset, which really isn't that difficult a task, PGMeans systematically underestimates k. STSC, on the other hand, does quite well. It doesn't make a single mistake on the blobs dataset. STSC generally has the right idea on the circles and moons datasets: at low noise levels, it correctly determines k, and at higher noise levels, it alternates between the correct number of clusters and an overestimation. Conversely, SpecialK has a tendency to err on the other side, if at all. On the circles dataset, it correctly identifies the number of clusters at low noise levels, and packs all observations into a single cluster at high noise levels, which, visually, looks about right (cf. Figure 1, lower right). On the blobs dataset, SpecialK generally finds the right level of k, only making incidental mistakes. Performance seems more erratic on the two moons dataset, especially with the W_C similarity matrix. Similarity matrices to this dataset exhibit unusual effects, which are due to the symmetry of the two clusters [9]. To counter these effects,

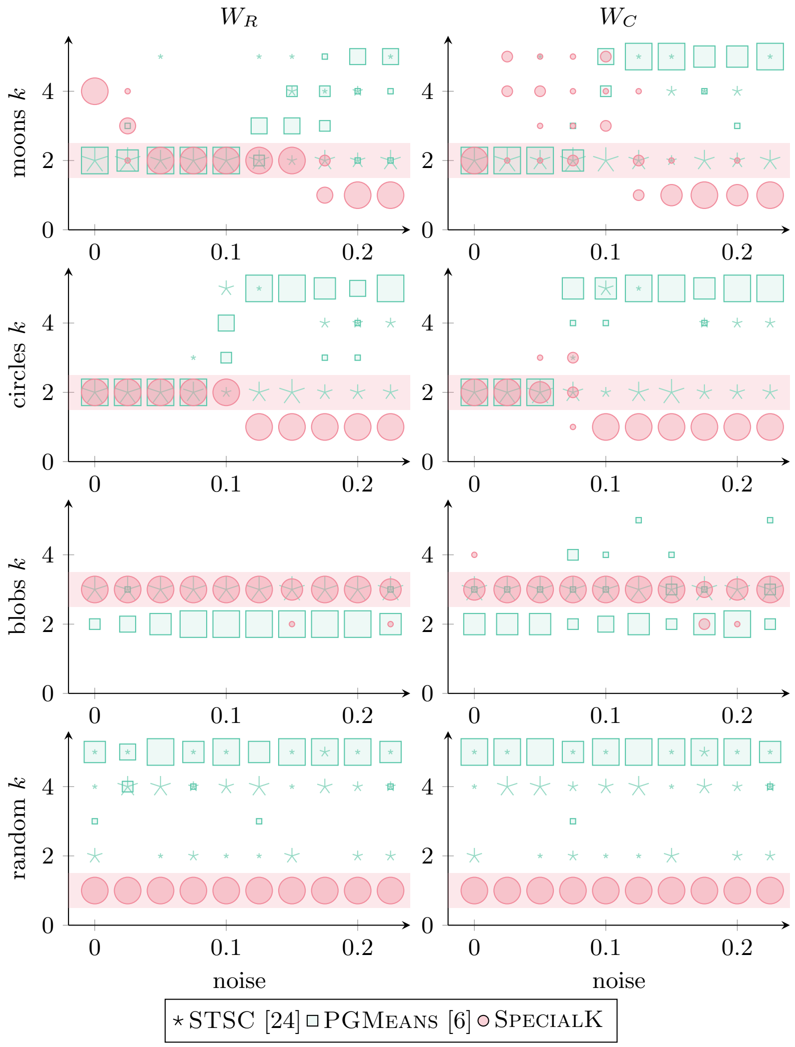

Fig. 2. Variation of noise, comparison of the derived number of clusters for the two moons, two circles, three blobs, and random datasets.

we discard all eigenvectors which are extremely correlated, which we define as having an absolute Spearman rank correlation $|\rho| > 0.95$. Subsequently, the rank is correctly estimated until the noise makes the two clusters appear as one.

The bottom row of Figure 2 is quite revealing. It illustrates how both STSC and PGMeans are prone to overfitting. On data that is pure noise, both these methods tend to find several clusters anyway, despite there being no natural grouping in the dataset: it really rather is one monolithic whole. SpecialK is the only algorithm capable of identifying this, and it does so without a single

Table 1. Experimental results on real-world datasets. The left half contains metadata on the employed datasets: names, numbers of rows (m) and columns (d), and the real number of classes in the data (Actual k). The right half contains the results of the experiments: the number of classes k determined by the algorithms STSC and SpecialK, the latter parameterized with similarity matrices W_R and W_C.

Dataset	m	d	Actual k	Determined k		
				STSC	SpecialK	
					W_R	W_C
Pulsar	17 898	9	2	2	2	2
Sloan	10 000	16	3	4	4	2
MNIST	60 000	784	10	2	2	3
HMNIST	5 000	4 096	8	3	4	4

mistake. Oddly, STSC seems to favor an even number of clusters in random data. PGMeans tends to favor defining as many clusters as possible on random data.

Empirical results on the sensitivity to the input parameter n (the employed number of eigenvectors) are given in the supplementary material [8].

4.2 Real-World Data Experiments

Experiments on synthetic datasets provide ample evidence that PGMeans cannot compete with STSC and SpecialK. Hence, we conduct experiments with only those latter two algorithms on selected real-world datasets, whose characteristics are summarized in the left half of Table 1. The Pulsar dataset[2] contains samples of Pulsar candidates, where the positive class of real Pulsar examples poses a minority against noise effects. The Sloan dataset[3] comprises measurements of the Sloan Digital Sky Survey, where every observation belongs either to a star, a galaxy, or a quasar. The MNIST dataset [16] is a well-known collection of handwritten numbers: the ten classes are the ten digits from zero to nine. The HMNIST dataset [13] comprises histology tiles from patients with colorectal cancer. The classes correspond to eight types of tissue. For these real-world datasets, we increase SpecialK's parameter to $n = 1000$, since the real-world datasets have at least three times as many examples as the synthetic datasets.

Results of the procedure when we let k simply increase up to eleven are given in Table 2. By p we denote the maximum of the probability bounds SpecialK computes, as outlined in line 10 of Algorithm 1 and mirrored at the end of Corollary 1. For STSC, we output the quality values on which the algorithm bases its decisions (higher is better). Additionally, we give the Normalized Mutual Information (NMI) scores between the constructed clustering and the actual class labels, matched via the Hungarian algorithm [15]; typically, higher is better.

STSC returns the k for which the quality column contains the highest value. By Algorithm 1, SpecialK returns the lowest k for which its p-value is below

[2] `https://www.kaggle.com/pavanraj159/predicting-pulsar-star-in-the-universe`

[3] `https://www.kaggle.com/lucidlenn/sloan-digital-sky-survey`

Table 2. NMI scores, probability bounds (p) and costs for the MNIST dataset. The selected rank and the corresponding NMI score is highlighted for every method.

	SpecialK				STSC [24]	
	W_C		W_R			
k	NMI	p	NMI	p	NMI	quality
2	0.317	10^{-6}	**0.195**	$\mathbf{10^{-23}}$	**0.306**	**0.987**
3	**0.518**	$\mathbf{10^{-4}}$	0.207	1.000	0.290	0.978
4	0.668	0.019	0.244	1.000	0.282	0.969
5	0.687	0.011	0.281	1.000	0.274	0.970
6	0.759	0.004	0.294	1.000	0.271	0.970
7	0.760	1.000	0.311	1.000	0.287	0.948
8	0.759	1.000	0.333	1.000	0.279	0.954
9	0.757	1.000	0.347	1.000	0.277	0.956
10	0.756	1.000	0.350	1.000	0.297	0.942
11	0.747	1.000	0.348	1.000	0.362	0.957

α, while the p-value for $k+1$ is above α. The selected values are highlighted in Table 2, and the determined values for k are entered in the right half of Table 1.

Across all datasets, the right half of Table 1 gives the determined values for k. All methods reconstruct the actual k well on the Pulsar dataset, but none of the determined values for k are equal to the actual value for k on the other three datasets. On Sloan, both methods are in the correct ballpark. On HMNIST, SpecialK is closer to the right answer than STSC. On MNIST, the true value of k is 10, but both methods determine a substantially lower number of clusters as appropriate. One can get more information on the behavior of the algorithms by taking a closer look at Table 2; similar tables for the other datasets can be found in the supplementary material [8].

For STSC, the highest NMI value is actually obtained for $k = 11$, which is a too high number of clusters, but quite close to the actual k. However, the computed quality does not mirror this finding. Also, the NMI value for the actual $k = 10$ is substantially lower than the NMI for $k = 11$, and the NMI for $k = 10$ is lower than the NMI value for the selected $k = 2$. Hence, NMI cannot just replace the quality in STSC. For SpecialK, p-value behavior is unambiguous under W_R. Notice that NMI peaks at the right spot.

Under W_C, things get more interesting. While the p-value for $k = 4$ indeed surpasses the threshold α, a slightly less restrictive setting (in many corners of science, $\alpha = 5\%$ is considered acceptable) would have changed the outcome to $k = 6$. At that stage, the p-value suddenly unambiguously shoots to 1; more than 6 clusters is definitely not acceptable. We see this behavior persist through the other datasets as well: there is always a specific value of k, such that the p-values for all $k' > k$ are drastically higher than the p-values for all $k'' \leq k$. While Algorithm 1 makes an automated decision (and this is a desirable property of an algorithm; no post-hoc subjective decision is necessary), if an end-user wants to invest the time to look at the table and select the correct number of clusters themselves, the p-values give a clear and unambiguous direction to that decision.

The entire last paragraph glosses over the fact that the actual k for MNIST is not 6, but 10. In fact, at first sight, the right half of Table 1 paints an unpleasant picture when compared to the Actual k column in the left half. However, the correct conclusion is that that column label is misleading. We have some real-world datasets with a given class label, providing a natural grouping of the data into k clusters. The task of clustering is also to find a natural grouping in the dataset. Clustering, however, is not necessarily built to *reconstruct any given* natural grouping: this task is unsupervised! Hence, if an algorithm finds a natural group on the MNIST dataset of relatively bulbous digits, this is a rousing success in terms of the clustering task. However, this group encompasses the digits 6, 8, and 9 (and perhaps others), which reduces the cardinality of the resulting clustering when compared to the clustering that partitions all digits. Therefore, no hard conclusions can be drawn from the determined k not matching the actual k. This is a cautionary tale (also pointed out in [9]), warning against a form of evaluation that is pervasive in clustering, but does not measure the level of success it ought to measure in a solution to the clustering task.

5 Conclusions

We propose a probability bound, that enables to make a hard, statistically founded decision on the question whether two clusters should be fused together or kept apart. Given a significance level α (with the usual connotation and canonical value settings), this results in an algorithm for spectral clustering, automatically determining the appropriate number of clusters k. Since it provides a method for SPEctral Clustering to Infer the Appropriate Level k, the algorithm is dubbed SpecialK. Automatically determining k in a statistically nonparametric manner for clusters with nonconvex shapes is, to the best of our knowledge, a novel contribution to data mining research. Also, unlike existing algorithms, SpecialK can decide that the data encompasses only one cluster.

SpecialK is built to automatically make a decision on which k to select, which it does by comparing subsequent p-values provided by the probability bound, and checking whether they undercut the significance level α. As a consequence, the user can elect to simply be satisfied with whatever k SpecialK provides. In the experiments on the MNIST dataset, we have seen that the perfect setting of k is in the eye of the beholder: several people can have several contrasting opinions of what constitutes a natural grouping of the data in a real-world setting. In such a case, one can extract more meaningful information out of the results SpecialK provides, by looking into the table of NMI scores and p-values for a range of settings of k. Eliciting meaning from this table is a subjective task. However, in all such tables for all datasets we have seen so far, there is a clear watershed moment where k gets too big, and relatively low p-values are followed by dramatically high p-values for any higher k. Turning this soft observation into a hard procedure would be interesting future work.

References

1. Alamgir, M., von Luxburg, U.: Multi-agent random walks for local clustering on graphs. In: Proc. ICDM, pp. 18–27 (2010)
2. Bauckhage, C., Drachen, A., Sifa, R.: Clustering game behavior data. IEEE Transactions on Computational Intelligence and AI in Games **7**(3), 266–278 (2015)
3. Bishop, C.M.: Pattern recognition and machine learning. Springer (2006)
4. Böhm, C., Faloutsos, C., Pan, J.Y., Plant, C.: RIC: Parameter-free noise-robust clustering. Transactions on Knowledge Discovery from Data **1**(3), 10 (2007)
5. Chung, F.R.K.: Spectral Graph Theory. American Mathematical Society (1997)
6. Feng, Y., Hamerly, G.: PG-means: learning the number of clusters in data. In: Advances in neural information processing systems, pp. 393–400 (2007)
7. Hamerly, G., Elkan, C.: Learning the k in k-means. In: Advances in neural information processing systems, pp. 281–288 (2004)
8. Hess, S., Duivesteijn, W.: k is the magic number — supplementary material. In: arXiv, to appear (2019)
9. Hess, S., Duivesteijn, W., Honysz, P., Morik, K.: The SpectACl of nonconvex clustering: A spectral approach to density-based clustering. In: Proc. AAAI (2019)
10. Horn, R.A., Johnson, C.A.: Matrix Analysis. Cambridge University Press (1985)
11. Hou, J., Sha, C., Chi, L., Xia, Q., Qi, N.: Merging dominant sets and DBSCAN for robust clustering and image segmentation. In: Proc. ICIP, pp. 4422–4426 (2014)
12. Kalogeratos, A., Likas, A.: Dip-means: an incremental clustering method for estimating the number of clusters. In: Advances in neural information processing systems, pp. 2393–2401 (2012)
13. Kather, J.N., Weis, C.A., Bianconi, F., Melchers, S.M., Schad, L.R., Gaiser, T., Marx, A., Zöllner, F.G.: Multi-class texture analysis in colorectal cancer histology. Scientific reports **6** (2016)
14. Kontkanen, P., Myllymäki, P., Buntine, W., Rissanen, J., Tirri, H.: An MDL framework for data clustering. In: Neural information processing series. Advances in minimum description length theory and applications, pp. 323–353 (2005)
15. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval research logistics quarterly **2**(1-2), 83–97 (1955)
16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
17. Lloyd, S.P.: Least square quantization in PCM. IEEE Transactions on Information Theory **28**(2), 129–137 (1982)
18. von Luxburg, U.: A tutorial on spectral clustering. Statistics and computing **17**(4), 395–416 (2007)
19. Maurus, S., Plant, C.: Skinny-dip: clustering in a sea of noise. In: Proc. KDD, pp. 1055–1064 (2016)
20. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Proc. NIPS, pp. 849–856 (2001)
21. Pelleg, D., Moore, A.W.: X-means: Extending k-means with efficient estimation of the number of clusters. In: Proc. ICML, pp. 727–734 (2000)
22. Tibshirani, R., Walther, G., 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), 411–423 (2001)
23. Tropp, J.A.: User-friendly tail bounds for sums of random matrices. Foundations of computational mathematics **12**(4), 389–434 (2012)
24. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: Proc. NIPS, pp. 1601–1608 (2005)