Estimating Number of Clusters in a data set via the Gap Statistics

- Tibshirani , Walter, Hastie

Presented by: Md Anees Parwez

https://fractionshub.com/

Problem Statement

The Challenge

Finding the **appropriate number of clusters** in cluster analysis

Standard clustering algorithms like K-means require prior information in order to specify the number of clusters.

Proposed Solution

Method of **Gap Statistics**

This uses outputs of ordinary clustering algorithm and automatically determines number of clusters by standardizing Within Cluster Sum of Squares.

Outline

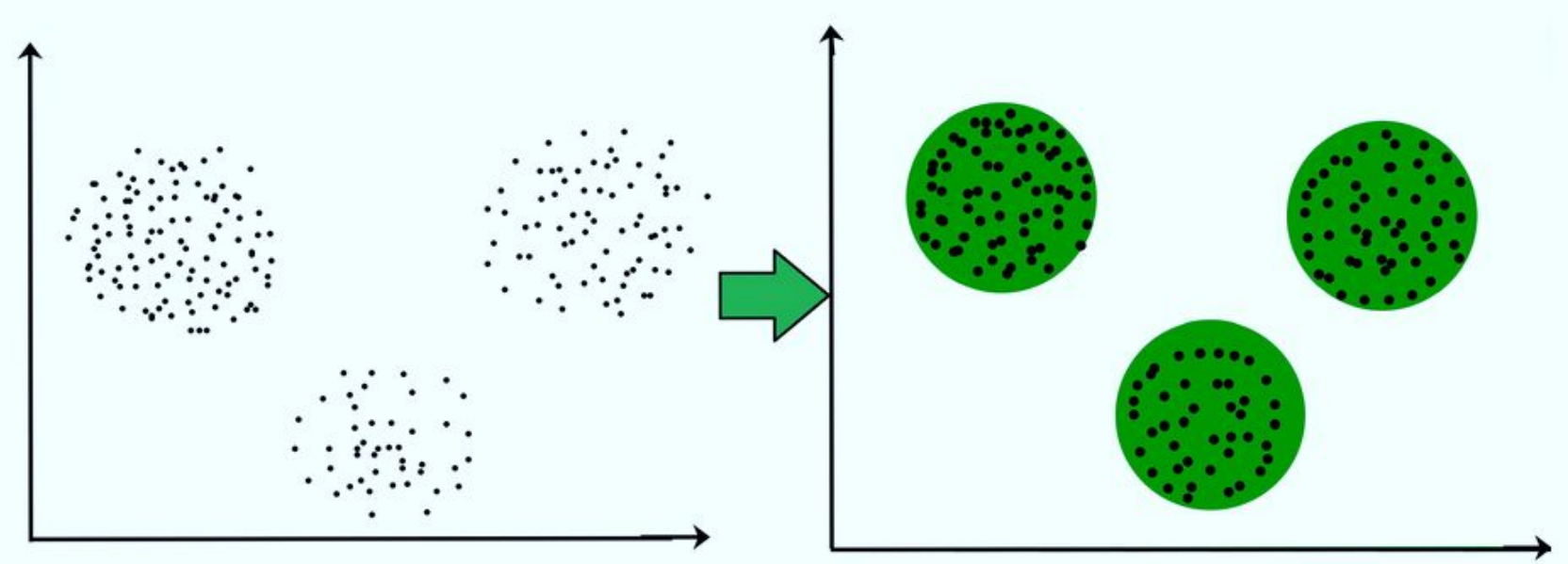
- Clustering
- K-means

Gap Statistics

- Motivation
- Null Distribution
- Implementation
- Simulation Results

Clustering

Grouping objects in sets: Objects within a cluster are as **homogeneous** as possible, whereas objects from different clusters are as **heterogeneous** as possible.



Clustering is an unsupervised learning method,
so input data sets are without labelled responses

Image Source: Geeks for Geeks

Clustering

- Grouping objects in sets: Objects within a cluster are as **homogeneous** as possible, whereas objects from different clusters are as **heterogeneous** as possible.

  Balance between *individuality* and *indistinguishability*

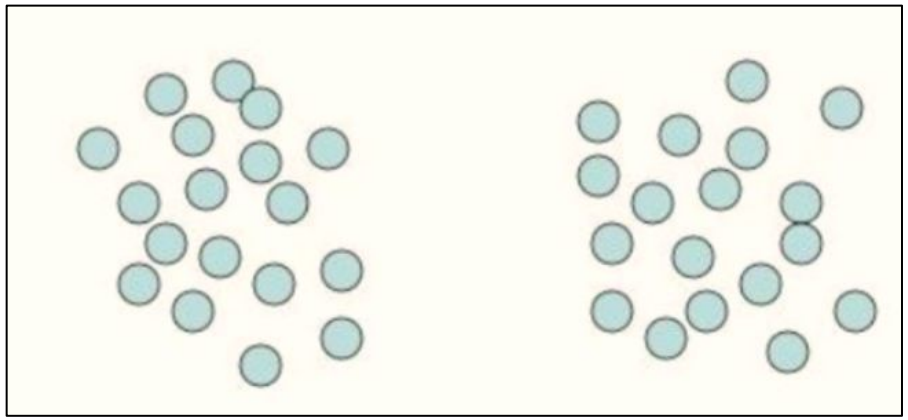
- It is an unsupervised learning method, so input data sets are without labelled responses

- Who can benefit?
  - Companies use it for Customer segmentation and provide different offers to different customer groups
  - Advertisement firm could spend money more effectively
  - Police can utilize it to identify crime prone localities
  - Document Classification based on tags, topics and content

Clusters of Clustering

Hard to present a rigorous definition of a "*cluster*".

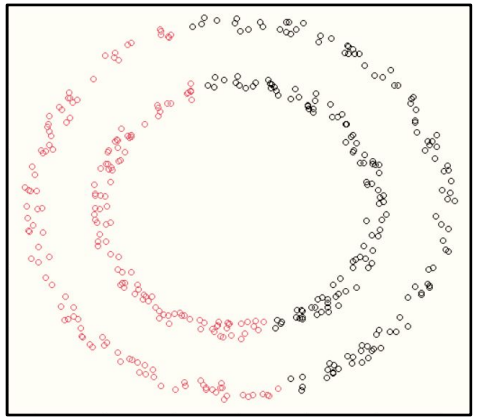
Problem 1 : Scale dependency



Two Clusters?

But, what if each point represents a thousand objects?

Problem 2 : Distance as Similarity measure



Clusters as two circles of different radii?

But, distance based clustering algorithm would yield depicted clusters.

Mathematical Framework

K-Means

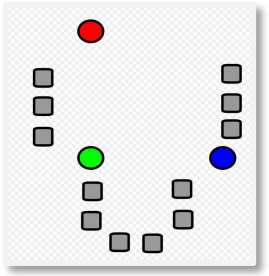
- K-means is one of the simplest , fundamental clustering algorithm.
- Aims to partition n observations into K clusters (**fixed apriori**).
- The cluster center (means) represent the cluster.
- Each observation belongs to the cluster with the nearest mean.
- It seeks to find cluster centers $\mu_1, \ldots \mu_K$ such that

$$J_k = \sum_{k=1}^{K} \sum_{i \in C_k} ||x_i - \mu_k||^2 \text{ is minimized}$$

K-Means: Lyod's Algorithm

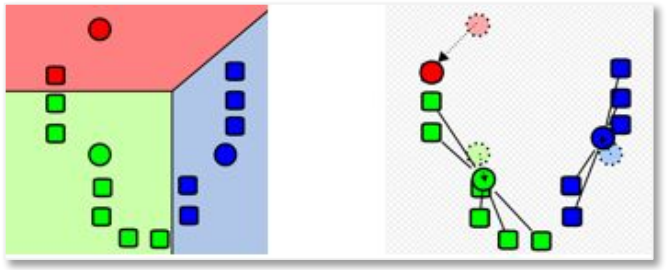
Initialize

Randomly select k centers.



Iterate

1. Assign data point to cluster whose distance from center is minimum of all the clusters.
2. Update the cluster means using all points in that cluster.



Terminate

Stop when no more new assignment are made.

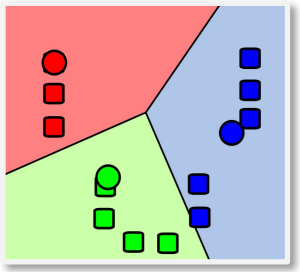
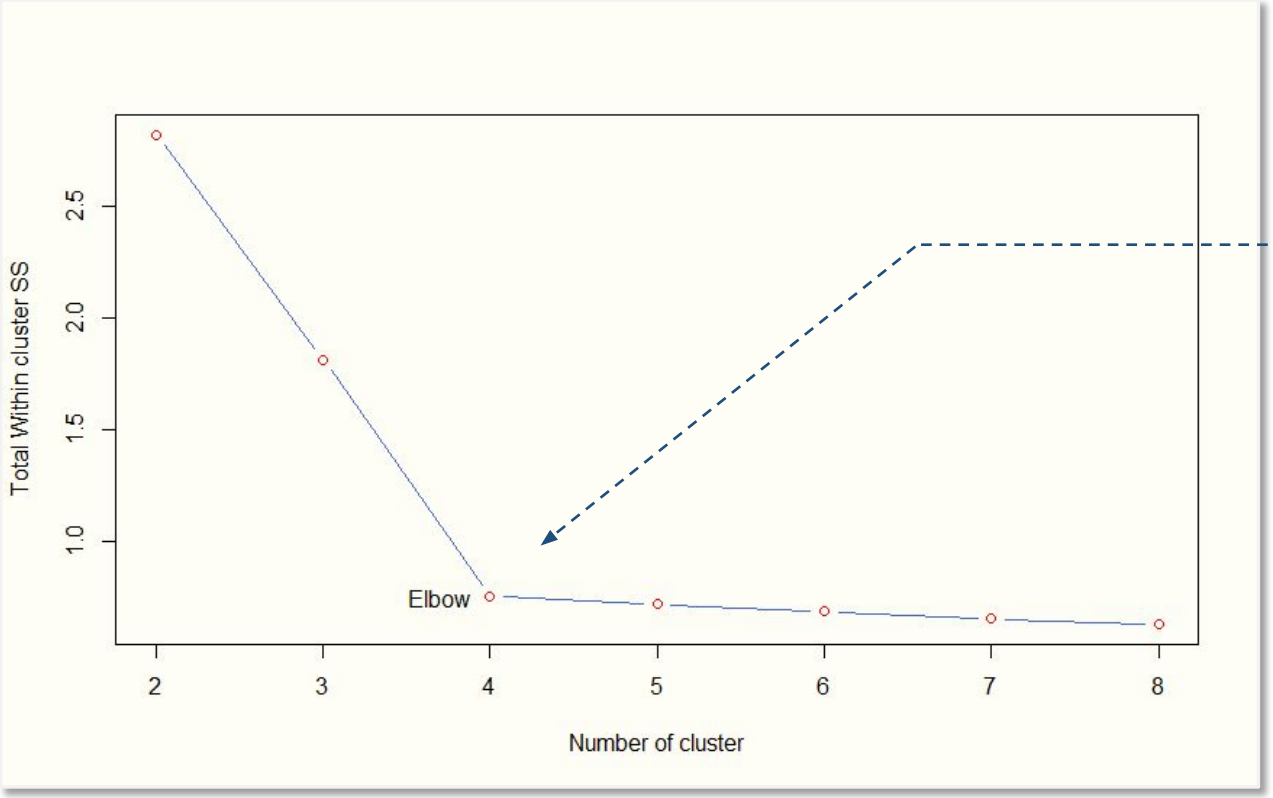


Image Source: Wikipedia

K-Means: Elbow Phenomenon

How do we choose k?

K-means aims to minimize J_k. Hence, given an option, we can consider multiple k and check which k minimizes it.



Mathematics and heuristics show that in case of well separated clusters W_k decreases rapidly with increase in k, until an "**elbow**" is reached. After elbow, rate of decrease of W_k is significantly slowed.

Therefore, choose that k after which curve flattens significantly (i.e., elbow is reached)

However, in real world data, the clusters are not well segregated. So the elbow is not easily identifiable.

Notations

- Let $\{x_{ij}\}\ i = 1, \ldots n;\ j = 1, \ldots p$ are n many p-dimensional vectors which constitute data
- Set $d_{ij} = \left|\left|x_i - x_j\right|\right|^2 = \sum_k \left|\left|x_{ik} - x_{jk}\right|\right|^2$ is Euclidean distance between two points
- Let there be k clusters $C_1, \ldots C_k$ with r^{th} cluster size n_r
- Set $D_r = \sum_{i,i' \in C_r} d_{ii'}$
- $W_k = \sum_{r=1}^{k} \frac{1}{2n_r} \times D_r$

Can also be written as:

$$W_k = \sum_r \sum_{x_i \in C_r} \left|\left|x_i - \overline{x_r}\right|\right|^2$$

W_k is nothing but pooled within cluster sum of squares

Moving Towards Gap Statistics

Objective

Optimum W_k $\Rightarrow$ Optimum $\log(W_k)$

Procedure

Standardize the graph of $\log(W_k)$ by comparing it with its expectation, $\mathrm{E}^*[\log(W_k)]$ (under an appropriate null assumption of data distribution)

Selection

Our estimate will be k for which $\log(W_k)$ falls farthest from this reference curve of $\mathrm{E}^*[\log(W_k)]$, i.e., that k for which $\mathrm{Gap(k)} = \mathrm{E}^*[\log(W_k)] - \log(W_k)$ is largest

Suppose we standardize (W_k) by dividing it with its expected value, $E^*(W_k)$. We would choose k such that:

$$\frac{W_k}{E^*(W_k)} \; is\ minimum \quad \Rightarrow \quad \log\left(\frac{W_k}{E^*(W_k)}\right) is\ minimum$$

$$\Rightarrow \log\left(E^*(W_k)\right) - \log(W_k) \quad is\ maximum$$

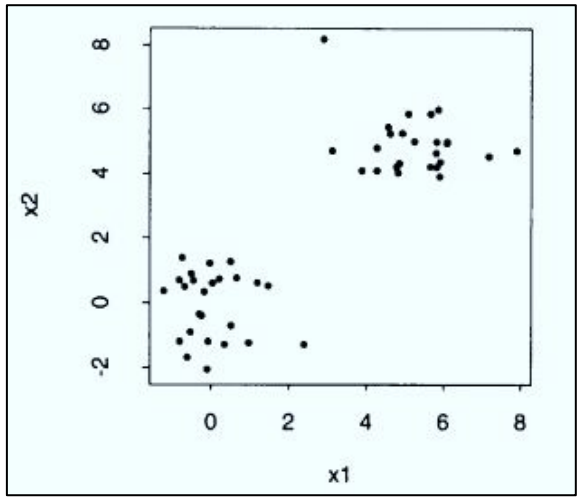
Alternate motivation

Interpreting $\mathrm{E}^*[log(W_k)]$

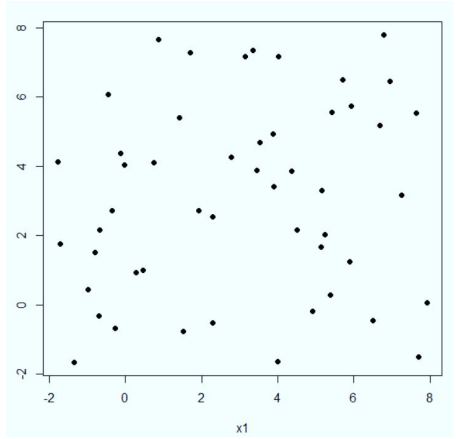
Assume null distribution ($\mathcal{N}$) , is uniform in p dimension.
a) n Samples are drawn from $\mathcal{N}$
b) Clustering is performed on sample, resulting in k clusters.
c) What is expected value of $log(W_k)$?

Assuming the centers of k clusters formed align in equally spaced the expectation under $\mathcal{N}$, $\mathrm{E}^*[log(W_k)]$ is:

$$\log(pn/12) - (2/p)logk + constant$$



Actual Distribution

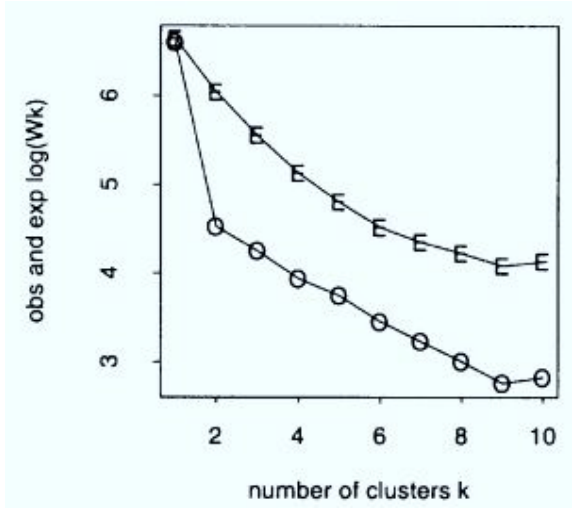


Null distribution

Now if actual given data has K well separated cluster:

- For $k < K$, $log(W_k)$ decreases faster *(with k)* than its expected rate $(2/p)log(k)$ under $\mathcal{N}$ (because $\mathcal{N}$ *has no cluster*).
- $k > K$, is adding an unnecessary cluster in center of approximately uniform cloud. So $log(W_k)$ will decrease more slowly than its expected rate.

So for k = K, gap would be maximum



$\log(W_k)$: O & $E^*[\log(W_k)]$: E

Null Distribution

The null model is taken as a *single component* model with a *log concave* density

A multi-component assumption sometimes result in erroneously rejecting 1-component model[1].

Hence we assume a single-component in our null hypothesis itself and reject it in favor of a k-component model if strongest evidence for any such k>1 warrants.

In a multivariate distribution, it is impossible to set confidence interval for the number or modes[2].

Since we need strong unimodality we use log concave density, $f(x) = e^{\phi(x)}$ where $\phi(x)$ is a concave function.

Let S_p be the set of all single component log concave densities on $\mathbb{R}_p$

[1]: Roeder 1994 [2]: Walter, 2001

Null Distribution: Insights from K-Means Clustering

Consider k-means clustering:

- $MSE_X(k) = E\left[\min_{\mu \in A_k} ||X - \mu||^2\right]$ with a $k - point$ set $A_k \in \mathbb{R}^p$ chosen to minimize this quantity. This is simply population version of W_k.

- Population version of gap statistic for k means clustering:

$$g(k) = \log\left(\frac{MSE_{X^*}(k)}{MSE_{X^*}(1)}\right) - \log\left(\frac{MSE_X(k)}{MSE_X(1)}\right)$$

  We subtracted logarithm of variances to make $g(1) = 0$.

- We're looking for least favorable single component null distribution X^* such that $g(k) \leq 0 \ \ \forall X \in S_p, \forall k \geq 1$.

Null Distribution: Theory

Theorem 1

For p = 1 , $\forall$ k $\geq$ 1,

$$\inf_{X \in S_p} \log\left(\frac{MSE_X(k)}{MSE_X(1)}\right) = \log\left(\frac{MSE_{U[0,1]}(k)}{MSE_{U[0,1]}(1)}\right)$$

Among all unimodal distribution the uniform produces most spurious clusters.

Theorem 2

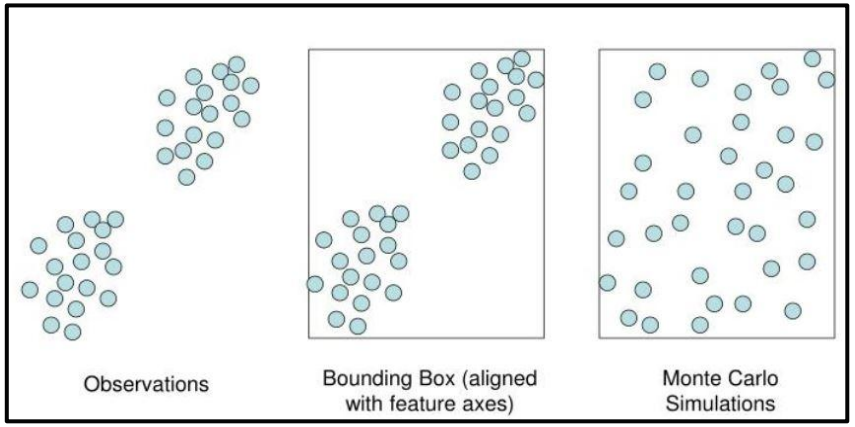
In higher dimension, no log concave distribution solves

$$\inf_{X \in S_p} \log\left(\frac{MSE_X(k)}{MSE_X(1)}\right)$$

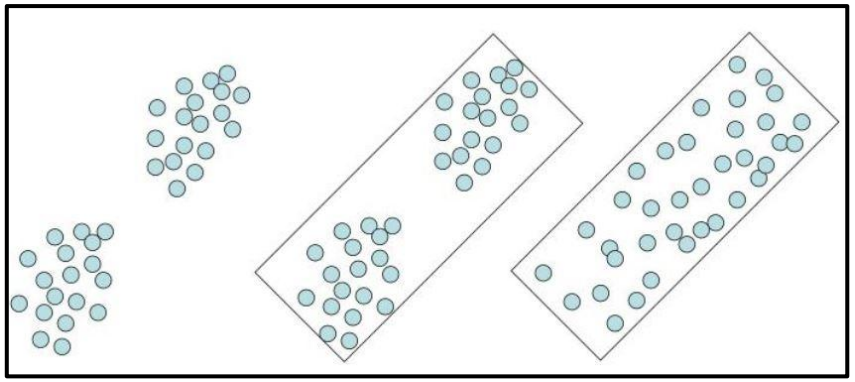
Authors suggest to mimic the 1-D case of uniform, along each component in higher dimension

Null Distribution: Suggestions for Higher Dimension

Suggestion 1: Generate each reference feature uniformly over range of observed value for that feature.



Suggestion 2: Generate reference features uniformly over a box aligned along Principal Components of data



Gap Statistics: Implementation

- We estimate $E^*[\log(W_k)]$ by an average B copies of $\log(W_k^*)$ each of which is computed from Monte Carlo samples $X_1^*, X_2^*, \ldots, X_n^*$, drawn from our reference distribution.
- $sd(k)$: Standard deviation of B Monte Carlo estimates $\log(W_k^*)$.
- Accounting simulation error results in quantity $s_k = \sqrt{1 + \frac{1}{B}}\ sd(k)$
- Choose the cluster size $\hat{k}$ to be the smallest k such that:

$$gap(k) \geq gap(k+1) - s_{k+1}$$

Error tolerance: This one standard deviation rule is known to work well empirically

Algorithm

Step 1: Clustering the data

Cluster the observed data for each value of k and compute W_k for each k.

Step 2: Estimating $E^*[\log(W_k)]$

i. Generate B reference distribution using either suggestion.
ii. Perform clustering on each set to compute $W_{kb}^* \; for \; b = 1,..B$ for each
iii. For each k, $E^*[\log(W_k)] = \frac{1}{B}\sum_b(\log(W_{kb}^*)$

Note that this is estimated value

Step 3: Other statistics

i. Compute : $gap(k) = E^*[\log(W_k)] - \log(W_k)$

ii. Compute: $sd(k) = \sqrt{\frac{1}{B} \times \sum_b(\log(W_{kb}^*) - E^*[\log(W_k)])}$

iii. Use $sd(k)$ to compute s_k.

Step 4: Choosing $\hat{k}$

Optimal $\hat{k}$: the smallest k $\quad s.t. \quad$ $gap(k) \geq gap(k+1) - s_{k+1}$

Other Methods for Choosing K

- **Calinski and Harabasz (1974)**

  Maximize w.r.t k the quantity: $CH(k) = \frac{B(k)/(k-1)}{W(k)/(n-k)}$

  B(K) is between cluster SS, where W(K) is within cluster SS.

- **Krzanowski and Lai (1985)**

  Maximize w.r.t k the quantity: $KL(k) = \frac{DIFF(k)}{DIFF(k+1)}$

  Here, $DIFF(k) = (k-1)^{2/p} W_{k-1} - k^{2/p} W_k$

- **Hartigan (1975)**

  $$H(k) = \left[\frac{W_k}{W_{k+1}} - 1\right] / (n-k-1).$$

  The estimated number of clusters is taken as the smallest k such that $H(k) \leq 10$

Simulations

Using simulated data, we compare the gap statistics with other existing methods *(50 realizations were generated in each setting).*

(a): Null (single cluster) data in 10 D

200 data points uniformly distributed over unit square in 10 dimension

Method	*Estimates of the following numbers of clusters $\hat{k}$:*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Null model in 10 dimensions										
CH	0‡	50	0	0	0	0	0	0	0	0
KL	0‡	29	5	3	3	2	2	0	0	0
Hartigan	0‡	0	1	20	21	6	0	0	0	0
Silhouette	0‡	49	1	0	0	0	0	0	0	0
Gap/unif	49‡	1	0	0	0	0	0	0	0	0
Gap/pc	50‡	0	0	0	0	0	0	0	0	0

‡ Refers to correct number of clusters

(b): 3 clusters in 2 D

3 clusters of standard normal data:
centered at (0,0) ,(0,5) ,(5,-3)
with 25, 25, 50 observations respectively.

Method	*Estimates of the following numbers of clusters $\hat{k}$:*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
3-cluster model										
CH	0	0	50‡	0	0	0	0	0	0	0
KL	0	0	39‡	0	5	1	1	2	0	0
Hartigan	0	0	1‡	8	19	13	3	3	2	1
Silhouette	0	0	50‡	0	0	0	0	0	0	0
Gap/unif	1	0	49‡	0	0	0	0	0	0	0
Gap/pc	2	0	48‡	0	0	0	0	0	0	0

Gap/unif : uses uniform reference distribution

Gap/pc: using principal component reference distribution

Simulations

(c): 4 cluster in 3 D

Each was randomly chosen to have
25 or 50 points in 3 dimension
with standard normal distributions and centers
randomly chosen from N(0,5I).

Method	*Estimates of the following numbers of clusters $\hat{k}$:*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Random 4-cluster model in 3 dimensions										
CH	0	0	0	42‡	8	0	0	0	0	0
KL	0	0	0	35‡	5	3	3	3	0	0
Hartigan	0	1	7	3‡	9	12	8	2	3	5
Silhouette	0	20	15	15‡	0	0	0	0	0	0
Gap/unif	0	1	2	47‡	0	0	0	0	0	0
Gap/pc	2	2	4	42‡	0	0	0	0	0	0

(d): 2 elongated clusters in 3 D

Cluster 1: Set $x_1 = x_2 = x_3 = t$
t takes 100 values from -0.5 to 0.5
and gaussian noise added.
Cluster 2: same as *Cluster 1* but 10 is added to each feature.

Method	*Estimates of the following numbers of clusters $\hat{k}$:*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
2 elongated clusters										
CH	0	0‡	0	0	0	0	0	7	16	27
KL	0	50‡	0	0	0	0	0	0	0	0
Hartigan	0	0‡	0	1	0	2	1	5	6	35
Gap/unif	0	0‡	17	16	2	14	1	0	0	0
Gap/pc	0	50‡	0	0	0	0	0	0	0	0

Conclusion

- Other methods do well, except in non null setting where only gap statistics shows a reasonable performance

- Gap/unif does well except in elongated clustering simulation, where oblique shape adversely affects its performance.

The **Gap/pc method** is a clear winner overall

THANK YOU