

Reducing UK-means to K-means

S. D. Lee Ben Kao

Department of Computer Science

The University of Hong Kong

`sdlee,kao@cs.hku.hk`

Reynold Cheng

Department of Computing

Hong Kong Polytechnic University

`csckcheng@comp.polyu.edu.hk`

The 1st Workshop on Data Mining of Uncertain Data
2007-10-28

Clustering of Uncertain Objects

- Traditional clustering algorithms (e.g. k-means) only handles certain data.
- In real world, uncertainty often arises in data:
 - due to random errors in physical measurements;
 - staleness of real-time data;
 - sampling error
- Considering the uncertainty helps improving clustering accuracy [1]
- We need algorithms to handle the uncertain data.
- e.g. UK-means [1] algorithm: extension to k-means to handle uncertain data.

Traditional k-means Algorithm

- Given: each object represented by a single point
- Task: group the objects into k clusters
- Objective: minimise the distance between an object and its cluster representative point
- The k-means algorithm iteratively:
 - Assigns each object to the cluster it is closest to
 - Updates the representative point of each cluster as the centroid of its member objects
- Until the cluster assignments no longer changes

UK-means Algorithm

- An extension of k-means to handle uncertain objects
- Object:
 - position is uncertain
 - no longer represented as a single point
 - represented as an arbitrary pdf (per object)
- Distance: represented as expected distance (ED)—the expected value of the distance, based on the pdf
- Problem: computation of ED is costly, requiring numerical integration for arbitrary pdf
- The ED calculations have been found to dominate the cost of UK-means [1]

Speeding Up UK-means

There are at last two approaches:

1. Pruning [2]

- Not all EDs need to be computed
- Many can be pruned away by using bounds on EDs
- Upper-/lower-bounds estimated by:
 - Minimum Bounding Rectangle (MBR)
 - Triangle Identity

2. Optimising the ED calculation

- We are taking this approach in this paper

ED (Expected Distance)

$$\text{ED}(o_i, \vec{y}) = \int_{\vec{x} \in R^m} d(\vec{x}, \vec{y}) f_i(\vec{x}) d\vec{x}$$

where $d(\vec{x}, \vec{y})$ is a distance function

$f_i(\vec{x})$ is the pdf of object i

- We restrict ourselves to the mean squared Euclidean distance

$$d(\vec{x}, \vec{y}) = \|\vec{x} - \vec{y}\|^2 = (\vec{x} - \vec{y}) \cdot (\vec{x} - \vec{y})$$

- This is called “Mean Squared Error” in [1]

Calculating ED

$$\text{ED}(o_i, \vec{y}) = \int_{\vec{x} \in R^m} \|\vec{x} - \vec{y}\|^2 f_i(\vec{x}) d\vec{x}$$

- Numerical integration required since f_i is arbitrary
- f_i may be represented as sample points in the space, each associated with a probability of occurrence
- Costly to compute from first principles
- But we can learn from mechanics...

Moment of Inertia

- In mechanics, the Moment of Inertia (M. I.) of an object determines its behaviours in rotational motion
- Definition: M. I. of an object O about axis \vec{y} is:

$$I_{\vec{y}} = \int_{\vec{x} \in O} \|\vec{x} - \vec{y}\|^2 \rho(\vec{x}) d\vec{x}$$

where $\rho(\vec{x}) =$ density at point \vec{x}

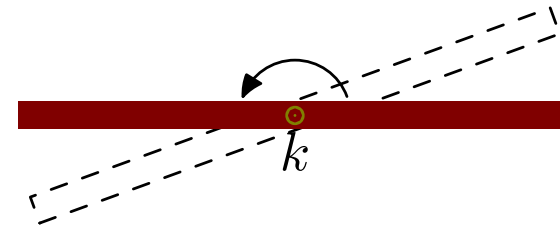
- Compare: $ED(o_i, \vec{y}) = \int_{\vec{x} \in R^m} \|\vec{x} - \vec{y}\|^2 f_i(\vec{x}) d\vec{x}$
- Does the similarity mean theorems on M. I. can be reused for ED?
- How do physicists compute M. I.?

How to calculate Moment of Inertia?

- From first principles:
 - analytical solution can be found for simple shapes about axes of symmetry
 - for arbitrary shapes about arbitrary axes: tedious
 - often impossible to find closed-forms: resort to numerical integration
- Look up from tables
 - For standard shapes (e.g. spheres, cones, cuboids, ...), look up the formula of M. I. from table
- Make use of theorems on M. I.
 - Parallel Axis Theorem
 - Perpendicular Axis Theorem (out of scope)

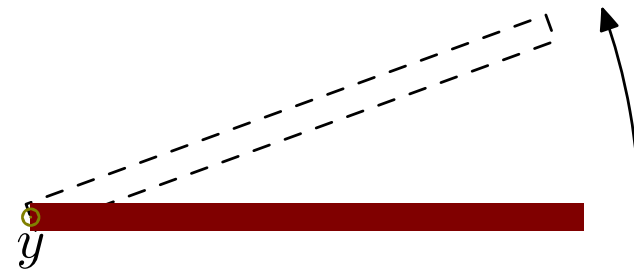
- e.g. consider a thin rod of mass M and length L , and axes perpendicular to its length
 - M. I. about axis through its centre \vec{k}

$$I_{\vec{k}} = \int_{-L/2}^{L/2} \frac{Mr^2}{L} dr = \boxed{\frac{1}{12}ML^2}$$



- M. I. about axis through its end \vec{y}

$$I_{\vec{y}} = \int_0^L \frac{Mr^2}{L} dr = \boxed{\frac{1}{3}ML^2}$$



Parallel Axis Theorem

- Calculating M. I. by integration is often costly
- If $I_{\vec{k}}$ (the M. I. of an object O about an axis X through its centroid \vec{k}) is known, we can calculate $I_{\vec{y}}$ (its M. I. about the axis parallel to X passing through \vec{y}) easily.
- The **Parallel Axis Theorem** states that

$$I_{\vec{y}} = I_{\vec{k}} + M \left\| \vec{y} - \vec{k} \right\|^2$$

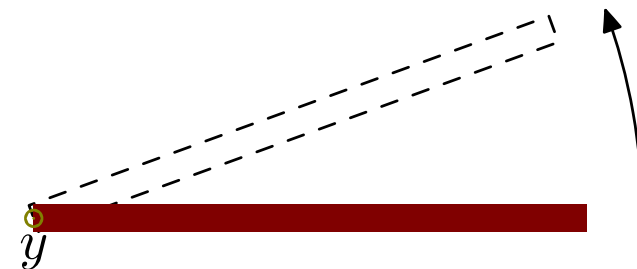
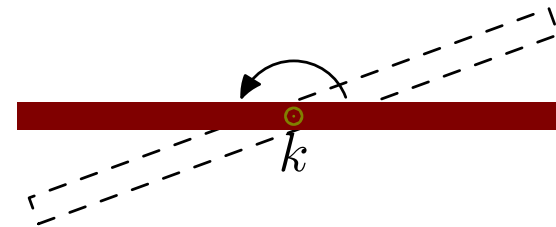
- So, if $I_{\vec{k}}$ is pre-computed (e.g. by looking up from a table of M. I. for common shapes), we can find $I_{\vec{y}}$ efficiently, without doing integration.

- e.g. consider again a thin rod of mass M and length L , and axes perpendicular to its length
 - We know from slide 9 that the M. I. of this rod about the axis through its centroid is

$$I_{\vec{k}} = \boxed{\frac{1}{12}ML^2}$$

- the M. I. of this rod about an axis through its end can be computed using the Parallel Axis Theorem:

$$I_{\vec{y}} = I_{\vec{k}} + M \left\| \vec{y} - \vec{k} \right\|^2 = \frac{1}{12}ML^2 + M \left(\frac{L}{2} \right)^2 = \boxed{\frac{1}{3}ML^2}$$



Analogous Theorem for ED Calculations

- The formulae for ED and M. I. are similar
- Can we find a formula similar to the Parallel Axis Theorem?
- Yes.

$$\text{ED}(o_i, \vec{y}) = \text{ED}(o_i, \vec{k}_i) + \left\| \vec{y} - \vec{k}_i \right\|^2$$

where \vec{k}_i is the centroid of object o_i .

- The derivations are also similar to that of the Parallel Axis Theorem.
- Once $\text{ED}(o_i, \vec{k}_i)$ has been pre-computed, we can find $\text{ED}(o_i, \vec{y})$ any efficiently.
- It follows from this formula that $\text{ED}(o_i, \vec{y})$ is minimised at $\vec{y} = \vec{k}_i$.

Speeding up ED Calculations

$$\text{ED}(o_i, \vec{y}) = \text{ED}(o_i, \vec{k}_i) + \left\| \vec{y} - \vec{k}_i \right\|^2$$

- With this formula, we can compute ED very efficiently:
 1. Compute \vec{k}_i for all uncertain objects o_i
 2. Pre-compute $\text{ED}(o_i, \vec{k}_i)$
 3. Whenever we need to calculate $\text{ED}(o_i, \vec{y})$ for any point \vec{y} , we apply the above formula
- Major overhead: pre-computation of \vec{k}_i and $\text{ED}(o_i, \vec{k}_i)$.
- But this is computed only once.
- Amortised over all iterations of the UK-means algorithm.
- Saves CPU cycles over computing $\text{ED}(o_i, \vec{y})$ from first principles

Eliminating the Pre-Computation Overhead

- Can we avoid the overhead of pre-computing $ED(o_i, \vec{k}_i)$?
- Objective function for the UK-means algorithm:

$$\text{minimise } TED = \left(\sum_{i=1}^n ED(o_i, \vec{k}_i) \right) + \left(\sum_{i=1}^n \left\| \vec{c}_{h(i)} - \vec{k}_i \right\|^2 \right)$$

- But the first term is constant
- It does not vary between the clustering iterations of UK-means
- We would get the same clustering results if we have used:

$$\text{minimise } F = TED - \left(\sum_{i=1}^n ED(o_i, \vec{k}_i) \right) = \left(\sum_{i=1}^n \left\| \vec{c}_{h(i)} - \vec{k}_i \right\|^2 \right)$$

- This eliminates the needs of computing $ED(o_i, \vec{k}_i)$
- We call this the **CK-means** algorithm

Reduction to K-means

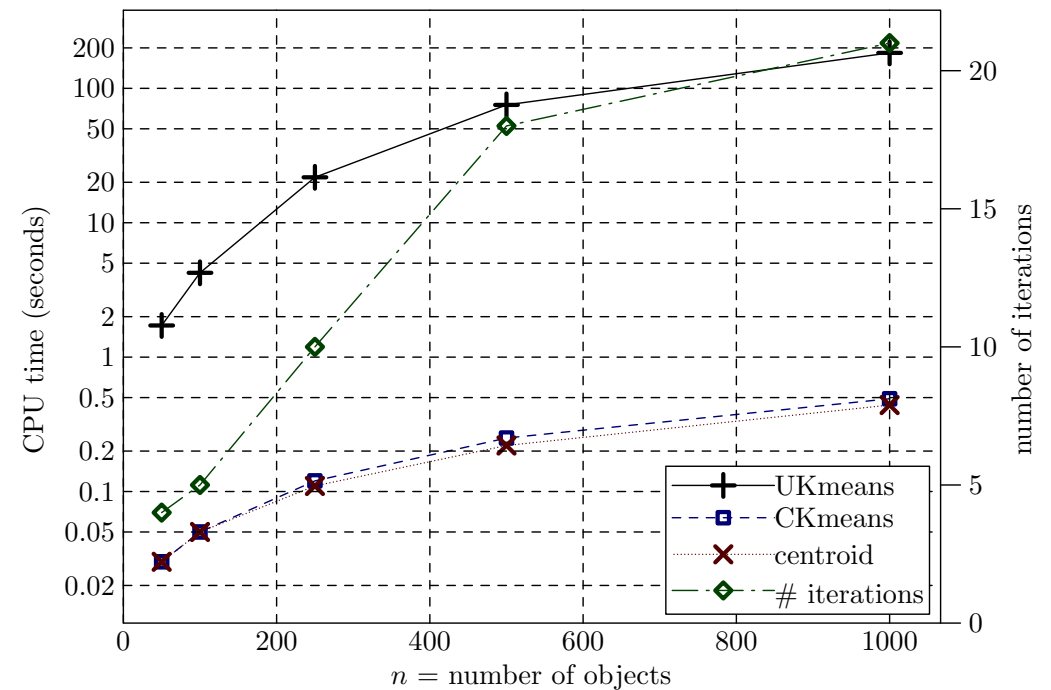
- Let's look at this new object function closely:

$$\text{minimise } F = \left(\sum_{i=1}^n \left\| \vec{c}_{h(i)} - \vec{k}_i \right\|^2 \right)$$

- This is the same objective function,
 1. if we re-run the clustering with traditional K-means
 2. each object is represented by its centroid \vec{k}_i
- This means we have reduced the problem of clustering uncertain data to the traditional problem of clustering certain data
- We can re-use the familiar k-means algorithm
- and enjoy its nice features (e.g. convergence)
- Essentially, CK-means first computes the centroids \vec{k}_i , and then invokes the traditional k-means algorithm.

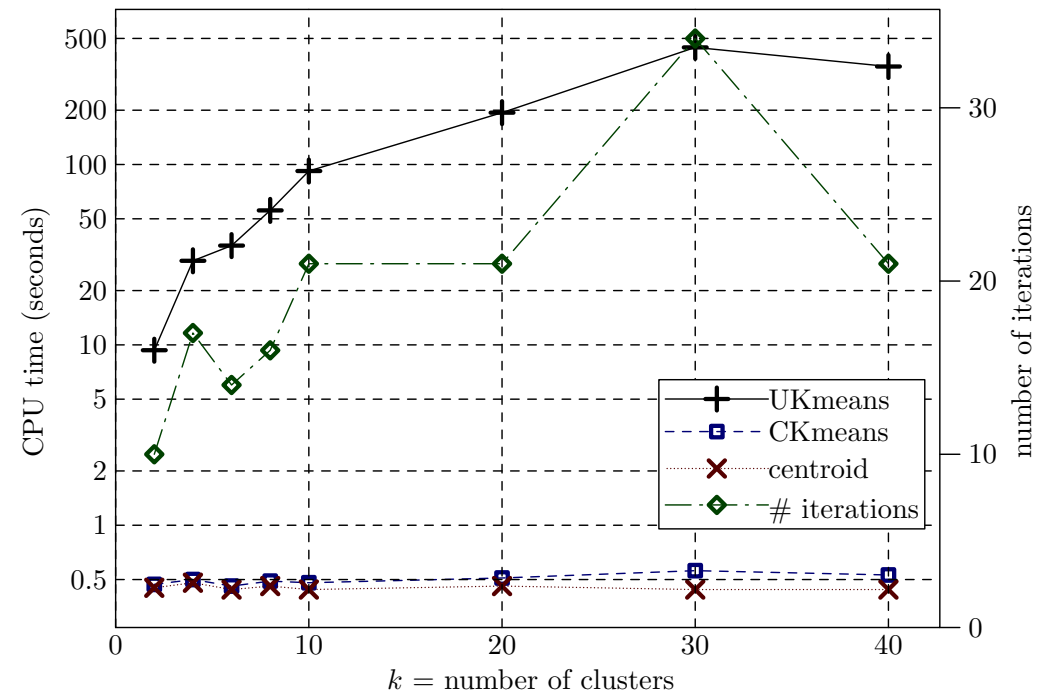
Experiments (1)

- 2D data set was synthesised
- n (50–1000) uncertain objects
- divided into 20 clusters
- Random initial cluster seeds
- 1000 sample points per object
- CK-means vs. UK-means
- CK-means is 57–375 times faster than UK-means
- A time saving of 98.3%–99.76%
- CK-means spent most time on centroid computations
- bigger $n \Rightarrow$ bigger speed up



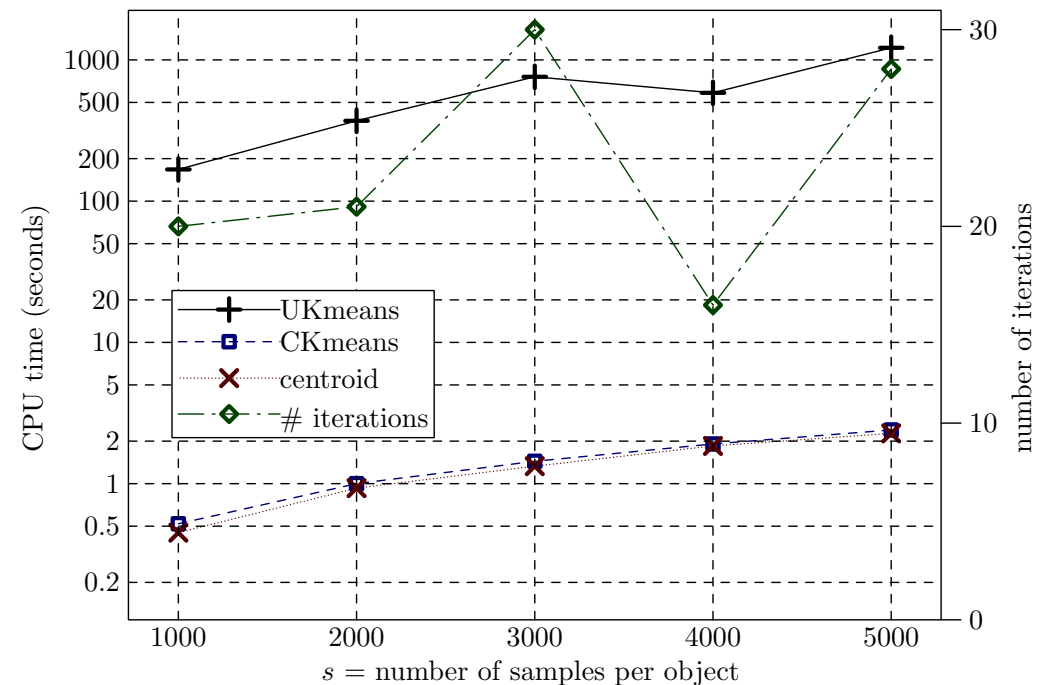
Experiments (2)

- Synthetic data set was used
- 1000 uncertain objects
- divided into k (2–40) clusters
- Random initial cluster seeds
- 1000 sample points per object
- CK-means vs. UK-means
- CK-means is 19.9–796 times faster than UK-means
- CK-means spent most time on centroid computations
- k has little effect on CK-means
- bigger $k \Rightarrow$ UK-means spends more time



Experiments (3)

- Synthetic data set was used
- 1000 uncertain objects
- divided into 25 clusters
- Random initial cluster seeds
- s (1000–5000) sample points per object
- CK-means is 306–527 times faster than UK-means
- CK-means spent most time on centroid computations
- bigger $s \Rightarrow$ both algorithms take more time to finish
- repeating the experiments with 5D data showed similar results



Discussions & Conclusions

- We have derived a formula for computing ED efficiently
- Thereby, accelerating the UK-means algorithm
- An improvement yields CK-means, which further reduces the pre-computation cost
- CK-means turns out to essentially reduce the problem from UK-means to K-means
- But this technique is only valid when ED is mean squared Euclidean distance (L-2 norm).
- Open question: Can we extend this technique to other distance measures? e.g. Manhattan (L-1) distance, maximum-norm (L- ∞) distance

References

- [1] Michael Chau, Reynold Cheng, Ben Kao, and Jackey Ng. Uncertain data mining: An example in clustering location data. In Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2006), volume 3918 of Lecture Notes in Computer Science, pages 199–204, Singapore, 9–12 April 2006. Springer.
- [2] Wang Kay Ngai, Ben Kao, Chun Kit Chui, Reynold Cheng, Michael Chau, and Kevin Y. Yip. Efficient clustering of uncertain data. In Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), pages 436–445, Hong Kong, China, 18–22 December 2006. IEEE Computer Society.