Research Article

# Fixed-Sized Clusters k-Means

**Mikko I. Malinen[1], Pasi Fränti[1]**

1. School of Computing, University of Eastern Finland, Joensuu, Finland

**We present a $k$-means-based clustering algorithm, which optimizes the mean square error, for given cluster sizes. A straightforward application is balanced clustering, where the sizes of each cluster are equal. In the $k$-means assignment phase, the algorithm solves an assignment problem using the Hungarian algorithm. This makes the assignment phase time complexity $O(n^3)$. This enables clustering of datasets of size more than 5000 points.**

## 1. Introduction

Euclidean sum-of-squares clustering is an NP-hard problem[1], which groups $n$ data points into $k$ clusters so that intra-cluster distances are low and inter-cluster distances are high. Each group is represented by a center point (centroid). The most common criterion to optimize is the mean square error (MSE):

$$\mathrm{MSE} = \sum_{j=1}^{k} \sum_{X_i \in C_j} \frac{\|X_i - C_j\|^2}{n}, \tag{1}$$

where $X_i$ denotes data point locations and $C_j$ denotes centroid locations. $k$-Means[2] is the most commonly used clustering algorithm, which provides a local minimum of MSE given the number of clusters as input. $k$-Means algorithm consists of two repeatedly executed steps:

**Assignment step:** Assign the data points to clusters specified by the nearest centroid:

$$P_j^{(t)} = \{X_i : \|X_i - C_j^{(t)}\| \leq \|X_i - C_{j^*}^{(t)}\| \\ \forall\, j^* = 1, \ldots, k\}$$

**Update step:** Calculate the mean of each cluster:

$$C_j^{(t+1)} = \frac{1}{|P_j^{(t)}|} \sum_{X_i \in P_j^{(t)}} X_i$$

 doi.org/10.32388/V36UBJ 

These steps are repeated until the centroid locations do not change anymore. $k$-Means assignment step and the update step are optimal with respect to MSE: The partitioning step minimizes MSE for a given set of centroids; the update step minimizes MSE for a given partitioning. The solution therefore converges to a local optimum but without guarantee of global optimality. To get better results than in the $k$-means, slower agglomerative algorithms[3][4][5] or more complex $k$-means variants[6][7][8][9] are sometimes used.

In *balanced clustering*[10][11], there are equal or more equal number of points in each cluster than in traditional clustering. Balanced clustering is desirable, for example, in divide-and-conquer methods, where the divide step is done by clustering.

Balanced clustering, in general, is a 2-objective optimization problem in which two goals contradict each other: to minimize MSE and to balance cluster sizes. Traditional clustering aims to minimize MSE without considering cluster size balance. Balancing, on the other hand, would be trivial if we did not care about MSE; simply by dividing points to equal size clusters randomly.

We next review some articles that have size constraits on clusters. Constrained $k$-means[12] allows putting lower bound on cluster sizes. Data clustering with size constraints[13] transforms the problem into a binary integer linear programming problem. The biggest dataset in their experiments is of size 625 points, indicating that the algorithm is not suitable for bigger datasets. Data Clustering with Cluster Size Constraits[14] allows putting upper bounds on cluster sizes. Their biggest dataset tested is 2000 points that also indicates that bigger datasets take too much time. Our proposed algorithm allows clustering up to circa 5000 points.

## 2. Fixed-sized clusters $k$-means

To describe Fixed-sized clusters $k$-means, we need to define what is an assignment problem. The formal definition of assignment problem (or linear assignment problem) is as follows. Given two sets ( $A$ and $S$), of equal size and with a weight function $W : A \times S \to \mathbb{R}$. The goal is to find a bijection $f : A \to S$ so that the cost function is minimized:

$$\text{Cost} = \sum_{a \in A} W(a, f(a)).$$

In the context of the proposed algorithm, sets $A$ and $S$ correspond respectively to cluster slots and to data points, see Figure 1.

In Fixed-sized clusters $k$-means, we proceed as in $k$-means, but the assignment phase is different: Instead of selecting the nearest centroids we have $n$ pre-allocated slots, and datapoints can be assigned only to these slots; see Figure 1.

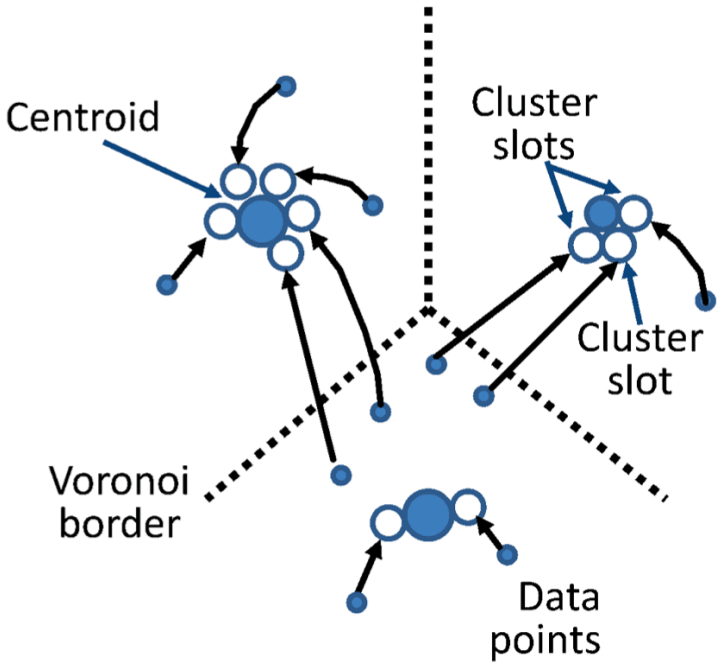


**Figure 1.** Assigning points to centroids via cluster slots.


doi.org/10.32388/V36UBJ

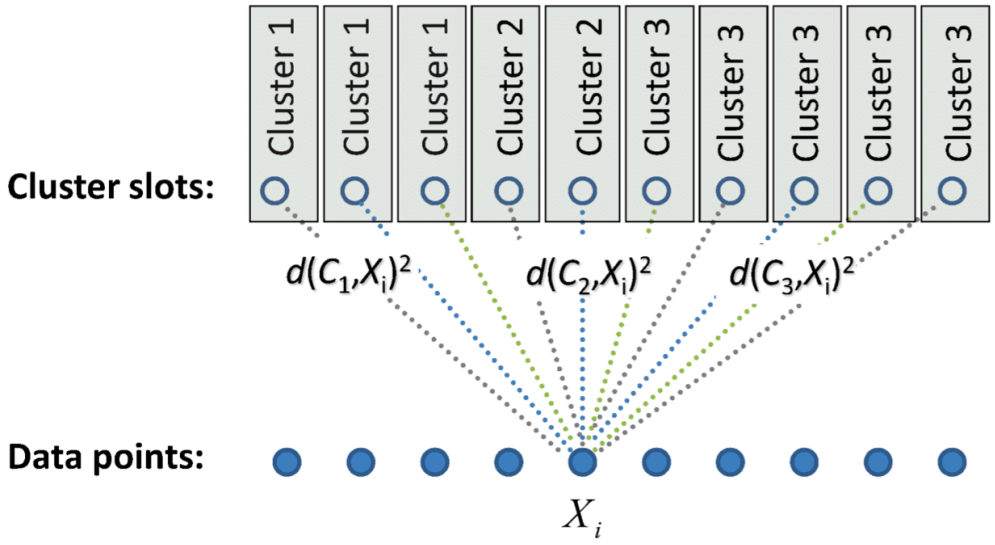


**Figure 2.** Minimum MSE calculation with fixed-sized clusters. Modeling with bipartite graph.

To find an assignment that minimizes MSE, we solve an assignment problem using the Hungarian algorithm[15]. First we construct a bipartite graph consisting $n$ datapoints and $n$ cluster slots, see Figure 2. We then partition the cluster slots in clusters in fixed sizes.

We give centroid locations to partitioned cluster slots, one centroid to each cluster. The initial centroid locations can be drawn randomly from all data points. The edge weight is the squared distance from the point to the cluster centroid it is assigned to. Contrary to standard assignment problem with fixed weights, here the weights dynamically change after each $k$-Means iteration according to the newly calculated centroids. After this, we perform the Hungarian algorithm to get the minimal weight pairing. The squared distances are stored in a $n \times n$ matrix, for the sake of the Hungarian algorithm. The update step is similar to that of $k$-means, where the new centroids are calculated as the means of the data points assigned to each cluster:

$$C_i^{(t+1)} = \frac{1}{n_i} \cdot \sum_{X_j \in C_i^{(t)}} X_j. \tag{2}$$

The weights of the edges are updated immediately after the update step. The pseudocode of the algorithm is in Algorithm 1. In calculation of edge weights, the cumulative sum of cluster sizes is

$$c(j) = \sum_{l=[1..j]} n_l \qquad \forall j \in [1..k], \tag{3}$$


doi.org/10.32388/V36UBJ

where $n_l$:s are cluster sizes and the number of cluster slot is denoted by $a$ and

$$\text{argmin}_j \, c(j) \geq a \tag{4}$$

is used in calculation of cluster where a cluster slot belongs to. So the edge weights are calculated by

$$W(a,i) = dist(X_i, C^t_{\text{argmin}_j c(j) \geq a})^2 \quad \forall a \in [1..n] \quad \forall i \in [1..n]. \tag{5}$$

After convergence of the algorithm the partition of points $X_i$, $i \in [1..n]$, is

$$X_{f(a)} \in P_{\text{argmin}_j c(j) \geq a}. \tag{6}$$

**Algorithm 1** Fixed-sised clusters $k$-Means
Input: dataset $X$, cluster sizes $n_l$, number of clusters $k$
Output: partitioning of dataset.

Initialize centroid locations $C^0$.
$t \leftarrow 0$
**repeat**
  Assignment step:
    Calculate edge weights. Eq. 5
    Solve an Assignment problem.
  Update step:
    Calculate new centroid locations $C^{t+1}$. Eq. 2
  $t \leftarrow t + 1$
**until** centroid locations do not change.
Output partitioning.

There is a convergence result in [12] (Proposition 2.3) for Constrained $k$-means. The result says that the algorithm terminates in a finite number of iterations at a partitioning that is locally optimal. At each iteration, the cluster assignment step cannot increase the objective function of Constrained $k$-means (3) in [12]. The cluster update step will either strictly decrease the value of the objective function or the algorithm will terminate. Since there are a finite number of ways to assign $m$ points to $k$ clusters so that cluster $h$ has at least $\tau_h$ points, since Constrained $k$-means algorithm does not permit repeated assignments, and since the objective of Constrained $k$-means (3) in [12] is strictly nonincreasing and bounded below by zero, the algorithm must terminate at some cluster assignment that is locally optimal. The same convergence result applies to Fixed-sized clusters $k$-means as well. The assignment step is optimal with respect to MSE because of pairing and the update step is optimal, because MSE is clusterwise minimized as is in $k$-means.

## 3. Time Complexity

Time complexity of the assignment step in $k$-means is $O(k \cdot n)$. The assignment step of the proposed Fixed-sized clusters $k$-means algorithm can be solved in $O(n^3)$ time with the Hungarian algorithm.

## 4. Application: Seating plan

As an application we present calculating a seating plan, where compatibility of persons within tables is optimized. First we need a compatibility matrix, where compatibility distance is given for every pair of persons. This has to be done manually.

$$D = \begin{pmatrix} 0 & d_{12} & . & . & . \\ d_{21} & 0 & & & \\ . & & . & & \\ . & & & . & \\ . & & & & . \end{pmatrix} \tag{7}$$

Then we need to do multidimensional scaling[16] giving $D$ as argument and the result is the data $X$ in higher dimensional space, but distances preserved. Then we do Fixed-sized k-Means giving data $X$, sizes of tables $n_l$ and number of tables $k$ as arguments. The output is the seating plan.

### *4.1. Experiments*

We tested the algorithm by creating a seating plan for Mikko I. Malinen's doctoral dissertation evening party in 2015. There were 22 persons invited. In compatibility distance matrix there were $22 \cdot 22 = 484$ distances. Sizes of tables were $[4\ 4\ 5\ 6\ 3]$ and $k$ was 5. Data $X$ became 10-dimensional. We repeated the algorithm 1000 times. This took only few seconds. People were happy with the seating plan. The software for both Fixed-sized clusters $k$-means and Seating plan are available from http://cs.uef.fi/~mmali/software/.

## 5. Conclusions

We presented an algorithm for clustering giving cluster sizes as constraints. The algorithm is practical up to 5000 points data. As an application we presented creating a seating plan for f.eg. parties.

## References


1. ^ *Aloise D, Deshpande A, Hansen P, Popat P (2009). "NP-hardness of Euclidean sum-of-squares clustering". Mach. Learn.. 75: 245--248.*
2. ^ *MacQueen J. Some methods of classification and analysis of multivariate observations. Proc. 5th Berkeley Symp. Mathemat. Statist. Probability. 1: 281–296 (1967).*
3. ^ *Equitz WH (1989). "A New Vector Quantization Clustering Algorithm". IEEE Trans. Acoust., Speech, Signal Processing. 37: 1568–1575.*
4. ^ *Fr\uooe4nti P, Virmajoki O, Hautam\uooe4ki V (2006). "Fast agglomerative clustering using a k-nearest neighbor graph". IEEE Trans. on Pattern Analysis and Machine Intelligence. 28 (11): 1875--1881.*
5. ^ *Fr\uooe4nti P, Virmajoki O (2006). "Iterative shrinking method for clustering problems". Pattern Recognition. 39 (5): 761--765.*
6. ^ *Arthur D, Vassilvitskii S. k-means++: the advantages of careful seeding. In: SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 2007. p. 1027-1035.*
7. ^ *Fr\"anti P, Kivij\"arvi J (2000). "Randomized local search algorithm for the clustering problem". Pattern Anal. Appl.. 3 (4): 358--369.*
8. ^ *Pelleg D, Moore A. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In: Proceedings of the Seventeenth International Conference on Machine Learning. San Francisco: Morgan Kaufmann; 2000. p. 727-734.*
9. ^ *Likas A, Vlassis N, Verbeek JJ (2003). "The global k-means clustering algorithm". Pattern Recognition. 36: 451--461.*
10. ^ *Malinen MI, Fr\"anti P. Balanced K-means for clustering. In: Joint Int. Workshop on Structural, Syntactic, and Statistical Pattern Recognition (S+SSPR 2014), LNCS 8621. Joensuu, Finland; 2014.*
11. ^ *Malinen MI, Fr\"anti P (2023). "All-pairwise squared distances lead to more balanced clustering". Applied Computing and Intelligence. 3(1): 93--115.*
12. a, b, c, d *Bradley PS, Bennett KP, Demiriz A. Constrained K-Means Clustering. Tech. rep., MSR-TR-2000-65, Microsoft Research; 2000.*
13. ^ *Zhu S, Wang D, Li T (2010). "Data clustering with size constraints". Knowledge-Based Systems. 23 (8): 883--889.*

14. ^ *Ganganath N, Cheng CT, Tse CK. "Data Clustering with Cluster Size Constraints Using a Modified k-means Algorithm, The Pre-Published Version".*

15. ^ *Burkhard R, Dell'Amico M, Martello S. Assignment Problems (Revised reprint). Philadelphia: SIAM; 2012.*

16. ^ *Cox TF, Cox MAA. Multidimensional scaling. London: Chapman & Hall; 1994.*

## Declarations

**Funding:** No specific funding was received for this work.

**Potential competing interests:** No potential competing interests to declare.