Parallel stability-based `k-means`*

Jason Poulos† Emin Arakelian‡ Fadi Kfoury§

Abstract

Abstract...

1. Introduction

Unsupervised learning is a branch of machine learning that infers patterns from data that has no labels. `k-means` is a popular unsupervised algorithm for finding clusters and cluster centers in data. The goal is to choose k cluster centers to minimize the total squared distance between each data point and its closest center. Given k initial centers chosen uniformly at random from the data points, `k-means` alternates between two steps until convergence: (*assignment step*) each point is assigned to the nearest cluster center and; (*update step*) each center is recomputed as the center of mass of all points assigned to it.

`k-means` is NP-hard, and can be solved in time $O(n^{dk+1} \log n)$, where n is the number of points in d dimensions. While it's hard to parallelize `k-means` itself due to its sequential nature, researchers can find shortcuts in the algorithm, or run numerous `k-means` instances on sub-samples to evaluate the stability of clustering. Our project compares sequential and parallelized versions of the stability-based method.

After briefly reviewing other parallel implementations of `k-means` in Section 2, we describe the stability-based method and its parallel implementation in Section 3. We then describe experiments and results in Section 4. Finally, we draw conclusions in Section 5.

2. Related work

`k-means++` chooses only the first cluster center uniformly at random; subsequent centers are selected from the data points, weighed by a probability proportional to its contribution to the overall error. This initialization algorithm is shown to obtain a set of initial centers that is close to the optimum solution [1].

`k-means||` `k-means++` is ill-suited for massive data because it makes k sequential passes over the data points in order to obtain the initial centers. Bahmani et al. [2] propose a method of parallelizing the initialization that reduces the number of passes, which they call `k-means||`. Instead of sampling a single point in each pass, `k-means||` samples $O(k)$ points and repeats for $O(\log n)$ rounds.

3. Stability-based method

Ben-Hur et al. [3] propose an algorithm that uses stability of clustering with respect to perturbations such as sub-sampling as a means of defining meaningful partitions. Computing the stability measure is the bottleneck, so parallelizing the algorithm will involve enhancements to re-use parts of the computations and avoid storing/writing the full matrix multiplication for computing the measure.

3.1. Parallel implementation

4. Experiments

We compare the algorithms using data from a dialect survey, which includes linguistic binary encoded responses [4]. The linguistic survey data can be used to study dialectometry, or linguistic variations with spatially determined distributions. These data, which feature questions on lexical choice and pronunciation, can be used to identify dialect areas and to determine whether linguistic diversity is isolated or a transitional area between areas of linguistic uniformity.

We implement the serial and parallel stability-based methods in Python and run all implementations on Edison. Performance is evaluated on three dimensions

†`poulos@berkeley.edu`
‡`emin@berkeley.edu`
§`fadi.kfoury@berkeley.edu`

*The code used for this project is available on Github: `https://github.com/plumSemPy/parallel_kmeans`.

– clustering cost, running time, and space complexity – for different values of k.

4.1. Choice of clusters

Figure 2 plots the overlay of the cumulative distributions of the similarity scores for varying values of k. We see that the distribution of similarities for each k are similar and close to 1. A choice of any number of clusters between $2 \leq k \leq 10$ would be reasonable; however, the choice of $k = 2$ appears to be the optimal choice. At $k = 2$, the cumulative distribution converges toward unity faster than the other distributions.

In order to determine the appropriate number of clusters, Figure 3 Figure 3 plots the within–group sum of squares for $k \leq 25$ clusters. We choose $k = 16$ clusters because the within–groups sum of squares (WSS) do not decrease much after $k = 16$. While the location of the "elbow" is ambiguous, it is clear that the greatest difference in reducing WSS happens when k moves from 1 to 2 and that the reduction of WSS diminishes after $k = 2$.

5. Conclusion

References

[1] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[2] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7):622–633, 2012.

[3] A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific symposium on biocomputing*, volume 7, pages 6–17, 2001.

[4] B. Vaux and S. Golder. The harvard dialect survey. *Cambridge, MA: Harvard University Linguistics Department*, 2003.

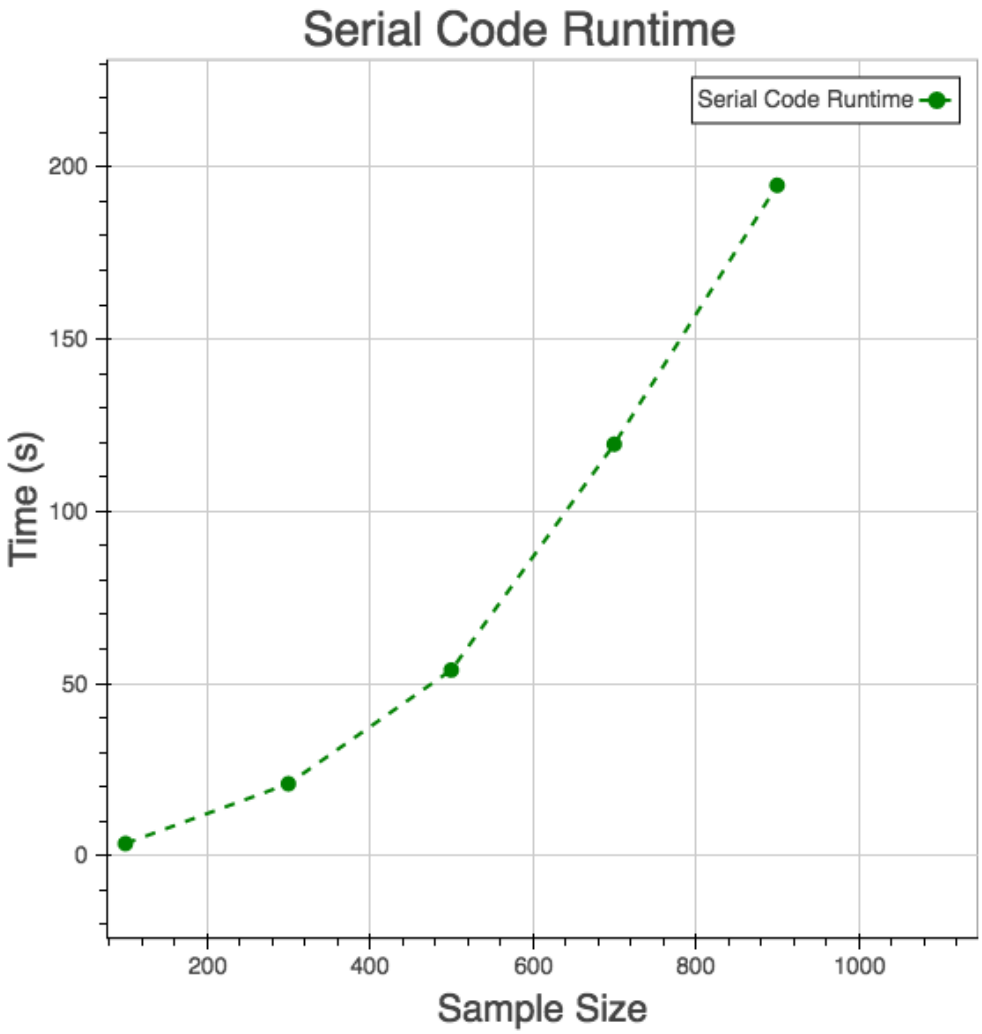

Figure 1. Serial code complexity.

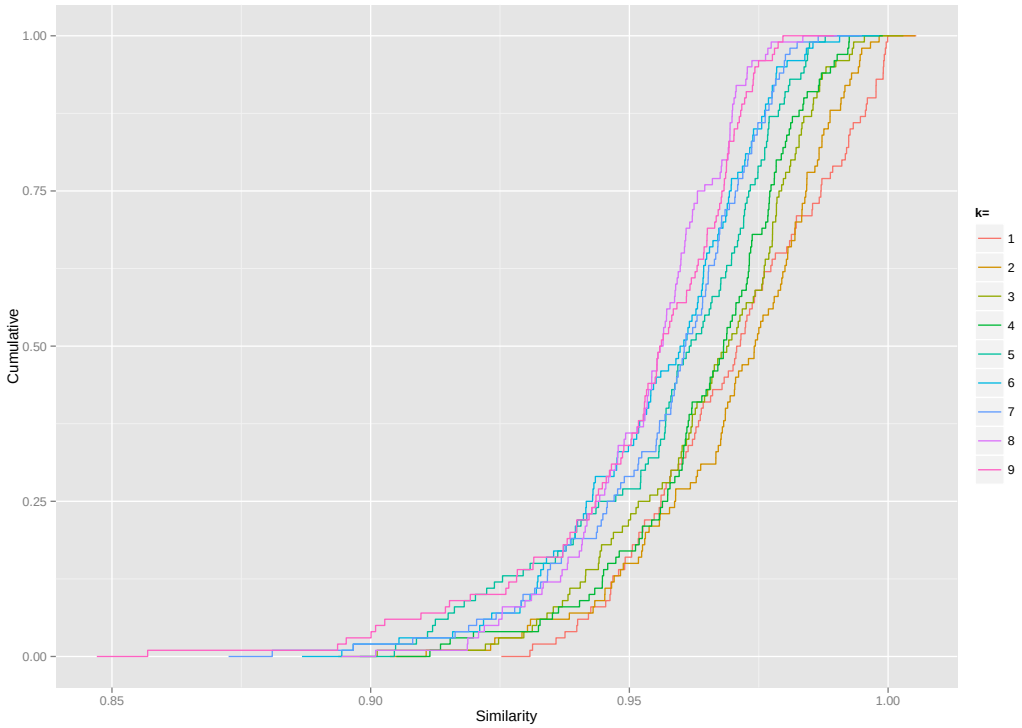

Figure 2. Overlay of the cumulative distributions of the correlation similarity measure for $n = 100$ runs and increasing values of k

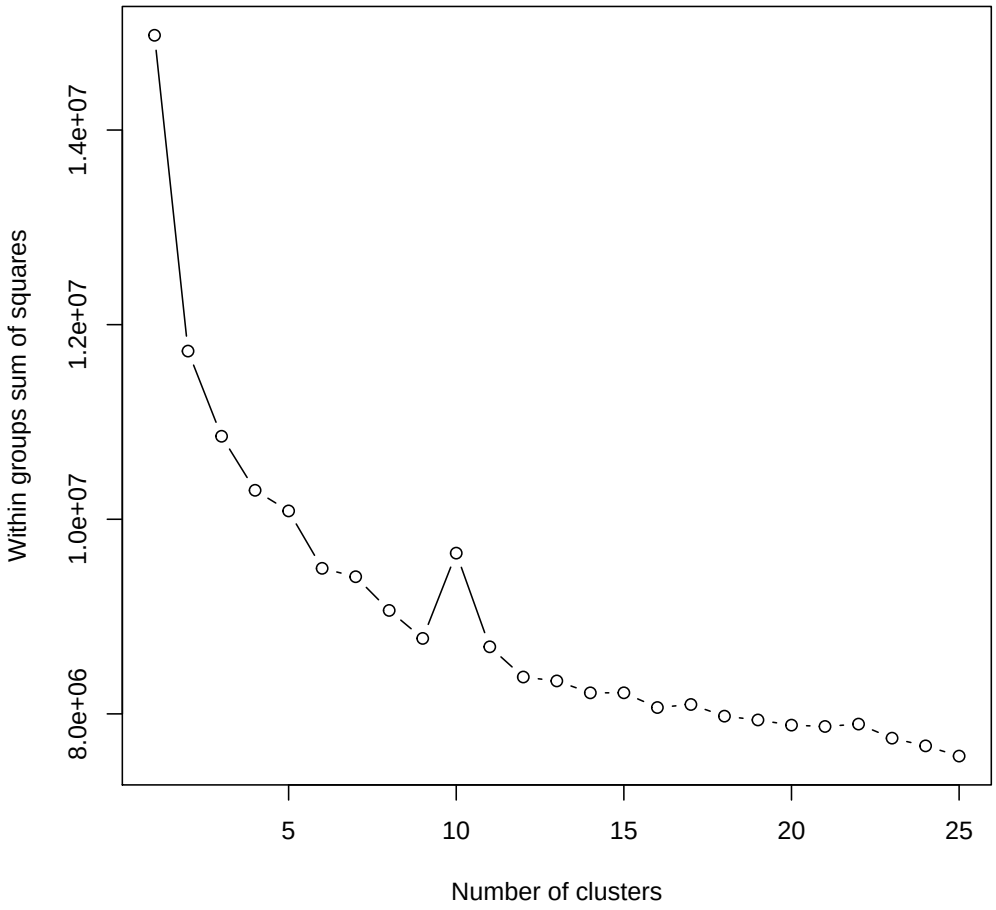

Figure 3. Sum of squared error scree plot of linguistic data. The location of the elbow in the plot suggests a suitable number of clusters for the k–means.