

Weighted k -Centers & Lloyd's Algorithm

Jonathan Friedman & Taylor Killian

December 9, 2015

Abstract

When determining the optimal location of distribution centers (e.g. warehouses, regional retail, or neighborhood commercial property), consideration must be made to maximize customer utility. It is assumed that individuals prefer the distance from them to any distribution center be minimized. To select the optimal location of distribution centers that minimize the average person's distance from any center, Lloyd's algorithm was altered to solve a variant of weighted k -centers. This augmented clustering algorithm was used to simulate a variety of distribution paradigms (differentiated by the weighting function used) using the 2010 U.S. Census population data for Massachusetts which provides a dense decision surface to optimize over. Results reaffirm the intuition that dense areas of high population, subject to weighting function, are optimal locations to place centers. These solutions are highly dependent on the granularity of the data and how the population at each data point is weighted.

1 Introduction

Serving the maximum number of people at minimum expense is an important, canonical problem in the retail and commercial industries. A company must balance placing their distribution centers near as many people as possible with the cost associated with opening and maintaining a large number of centers. A general formulation of this problem known as the metric k -center problem asks for a placement of n distribution centers that minimizes the maximum distance from any point to a center [1]. In this paper, we examine the variant in which each point has a weight that affects its distance from the center point; for this report the weight is considered to be a function of the population at each point. We also seek to minimize the sum of the maximum distances rather than the maximum individual distance. Efficient algorithms to solve this problem exist in one dimension [7]. For the two-dimensional case we adapt Lloyd's algorithm, an algorithm commonly used to find evenly spaced points and subsets within a given set of data, to work on the 2010 US Census data from the state of Massachusetts by modifying the clustering step.

If no attempt is made to avoid local extrema, then this problem is easy to solve for one center. In section 3, we give a solution for one center using compass

search. However, compass search is not viable for a large number of points because the number of required directions to search grows quickly. In section 4, we give a solution method for an arbitrary number of centers using the variant on Lloyd’s algorithm. In section 5, we use our algorithm to optimize placement of distribution centers in the population data of Massachusetts, subject to different constraints. The two we considered were separate weighting functions, used when adjusting the cluster centers, and the minimum population we allowed to influence the adjustment of these centers.

2 Background & Data

2.1 Background

Let \mathcal{C} be the set of distribution centers, let \mathcal{X} be the set of points which the centers serve, and let d be a distance metric. In the general metric k -center problem, the goal is to minimize

$$\max_{p_i \in \mathcal{X}} \min_{c \in \mathcal{C}} d(c, p_i)$$

Now let ξ_i be the population at point $p_i \in \mathcal{X}$, and w be a weight function (we consider only weight functions where $\xi_i > \xi_j \implies w(\xi_i) \geq w(\xi_j)$, more exotic functions could be useful in other contexts). In the version of the weighted k -center problem that we consider, the goal is to minimize our cost function, the “weighted mean distance” function

$$\frac{\sum_i w(\xi_i) \min_{c \in \mathcal{C}} d(c, p_i)}{|\mathcal{X}|}$$

Note that while the minimum achieved in the standard metric k -center problem corresponds to distance, the minimum achieved in the weighted problem does not have a consistent physical interpretation across all weighting functions. When w is the identity function, it corresponds to minimizing the distance from the average person, rather than the average point, to its nearest distribution center. In general, it can be intuitively thought of as a weighted analogue to this mean distance, but its precise interpretation depends on the domain and context of the problem.

The choice of w is also context and domain dependent, as shown below in Section 5. A superlinear w could be used in a situation in which it is not as useful to place distribution centers near areas of low population density; for example, an expensive specialty store might place great importance on serving urban areas. A sublinear w could be used in a situation in which optimally serving areas of high population density is not important, as in the case of a big-box retailer that customers generally visit infrequently.

Unless otherwise specified, d is the standard Euclidean distance. Of course, this distance metric does not capture the true surface distance between points on the Earth. Since we test our algorithm on geographic data, we should,

strictly speaking, use a different distance function such as the haversine formula. Euclidean distance provides a good approximation on sufficiently small regions such as the state of Massachusetts, where the curvature of the Earth is effectively negligible when calculating the distance between two points.

2.2 Census Data

Throughout the paper, we test our algorithms using census block population data for Massachusetts from the 2010 U.S. Census. A census block is the smallest geographic unit for which population data is collected for every resident. Population data on a block level is available at [3], and can be parsed using geographic information software such as the Python Shapefile Library [6]. To generate a representative point for each block, we enclose the block in a bounding rectangle, then take the center of that rectangle to be the location of the block.

Although the 2010 Census divided Massachusetts into 157,508 total census blocks, 61,174 of those blocks have a population of zero (since every point in the United States is within a census block, many of which are often delimited by geophysical features, several blocks are located in lakes, national parks, shopping malls, and so forth) [8]. We ignore these "empty" blocks in our analysis because they are of little interest in the context of placing distribution centers to serve potential customers.

3 One Distribution Center

When $k = 1$, the weighted k -centers problem is easily solved via compass search. Compass search is a simple method of minimizing a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ that does not require computation of the gradient. The general structure of the algorithm is as follows: Starting from a starting location and with an initial step size and a set of directions that form a positive spanning set of \mathbb{R}^n , find a direction for which moving by the step size decreases the function and execute that step. If there is no such direction, decrease the step size and search anew for a direction that decreases the function. The algorithm terminates when the step size falls below a given tolerance. A more detailed description is given in Algorithm 1 (see [4] for further details and proof of convergence). Compass search is susceptible to convergence within local extrema, which is usually avoided by running the search several times with randomly chosen starting points to more fully explore the space of possible solutions.

As above, we choose f to be

$$\frac{\sum_i w(\xi_i)d(c, p_i)}{|\mathcal{X}|}.$$

Figure 1 shows distribution center placement for various w . As expected, the distribution center is consistently placed near Boston. Under $w(x) = \sqrt{x}$, the center moves farther away from Boston as the influence of high population to the

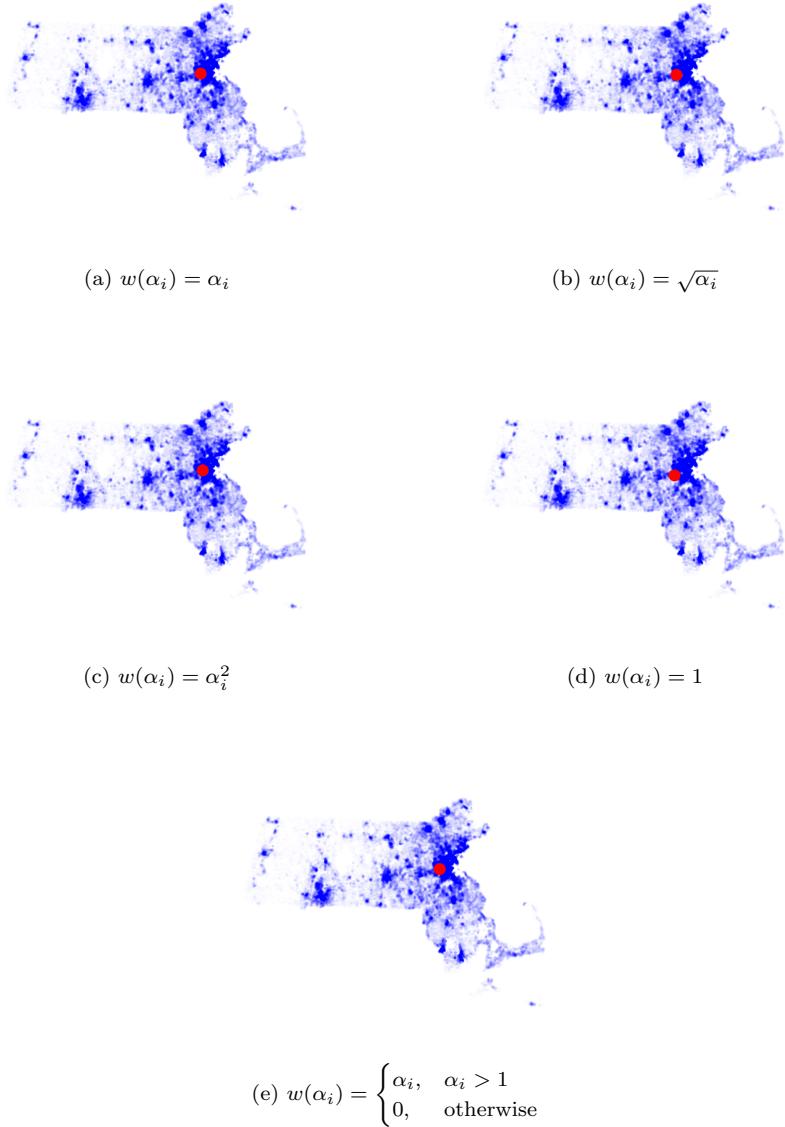


Figure 1: Distribution center placement under various weighting functions. The centers are shown in red.

objective function is decreased, and under $w(x) = x^2$, the center moves closer to the downtown neighborhoods of the city. However, the choice of weighting

function appears to have little effect on placement of the center. Even when population is not taken into account, the center remains fairly close to Boston by virtue of the large number of census blocks located there.

This method works well for one distribution center, but the number of directions to search grows quickly with the number of centers. Consider the case of n clusters $c_1, \dots, c_n \in \mathbb{R}^2$. Our function to minimize is now the full weighted mean distance function

$$\frac{\sum_i w(\xi_i) \min_{c \in \mathcal{C}} d(p_i, c)}{|\mathcal{X}|}$$

A positive spanning set for \mathbb{R}^n must contain at least $n + 1$ vectors [5], so we must consider up to three possible directions at each step for each p_i . Determining the exact number of directions that must be searched for n points in order to guarantee convergence to some maximum is beyond the scope of this report. Suffice it to say that better, more scaleable methods exist. One such algorithm is Lloyd's algorithm, a variant of which is described in the next section.

Algorithm 1 Compass Sort

Let $\lambda_1, \dots, \lambda_m$ be a positive spanning set of \mathbb{R}^n . Choose a starting guess p_0 , a step size s_0 , a scaling constant $\alpha < 1$, and a tolerance t .

```

while  $s_k > t$  do
    if  $f(p_k + s_k \lambda_i) < f(p_k)$  for some  $\lambda_i$  then
         $p_{k+1} = p_k + s_k \lambda_i$ 
         $s_{k+1} = s_k$ 
    else
         $p_{k+1} = p_k$ 
         $s_{k+1} = \alpha s_k$ 
    end if
end while
return  $p_{k+1}$ 
```

4 Multiple Distribution Centers

Lloyd's algorithm provides a more scaleable solution for multiple distribution centers. This algorithm was originally applied to problems in information theory [2], but has been adapted to other fields such as clustering. A detailed description is given in Algorithm 2 (see [1] for further details). The general algorithm works as follows: first, assign, via some method, starting locations for the centers c_i . Then, assign each center c_j a cluster Γ of points that are closer to c_i than to any other c_k , $j \neq k$. Next, adjust the location of the c_i to the centers of their respective clusters. Repeat until the c_i do not move between iterations.

In the unweighted problem, the centers of the clusters can be computed by taking the mean of each component of the points in the cluster. In the weighted

problem, this method does not work, since the center of the cluster has become a weighted mean of the points. Instead, in the center adjusting step, we find the new location for a center c_j in cluster Γ by solving the optimization problem

$$\operatorname{argmin}_{c_j} \sum_{\{i:p_i \in \Gamma\}} w(\xi_i) d(p_i, c_j)$$

This is equivalent to finding a root of the gradient

$$\left[\sum_{\{i:p_i \in \Gamma\}} w(\xi_i) \left(\frac{p_{ix} - c_{j,x}}{d(p_i, c_j)} \right), \sum_{\{i:p_i \in \Gamma\}} w(\xi_i) \left(\frac{p_{iy} - c_{j,y}}{d(p_i, c_j)} \right) \right]^T$$

We randomly assign starting centers, though other initialization methods may be used. This suffices to find a solution for an arbitrary number of distribution centers.

To find a solution such that the cost function is below a given threshold and the number of centers is minimized, run the algorithm for 1, 2, ... centers until the cost function falls below the threshold. Lloyd's algorithm is not guaranteed to produce a global minimum, so to more fully explore the space, we run the algorithm several times, stochastically varying the initialization points, for each choice of k .

Algorithm 2 Lloyd's Algorithm with Weighting

Randomly choose n initial centers $\{c_1^1, \dots, c_n^1\} = \mathcal{C}$, and let $\{\gamma_1, \dots, \gamma_n\} = \Gamma$ be their corresponding clusters.

```

while  $\exists c_i : c_i^k \neq c_i^{k-1}$  do
    for  $\gamma_i \in \Gamma$  do
         $\gamma_i = \{x \in \mathcal{X} : d(x, c_i^k) < d(x, c_j^k) \forall j \neq i\}$ 
    end for
    for  $c_i \in \mathcal{C}$  do
         $c_i^k = \operatorname{argmin}_{\chi} \sum_{\{i:p_i \in \gamma\}} w(\xi_i) d(p_i, \chi)$ 
    end for
end while
```

5 Experiments

As in the case of one distribution center, we test our algorithm on U.S. census data from Massachusetts. In order to compare results with the same tolerance and different w , we develop a simple rescaling such that under each function w , all calculated weights satisfy $\{w(\xi_i) \in [0, 1] \mid \sum_i w(\xi_i) = 1\}$.

WILL REWORK THIS PARAGRAPH... Initially, we solve the problem of placing the minimum number of centers such that the weighted mean distance with $w(\xi) = \xi$ is below a given initial tolerance level. Then, for every other weighting function w , we solve the problem for under that same tolerance.

We hoped to be able to compare the different clusterings produced by each weighting function. We were unable to determine any feasible way of doing this across separate clusterings without assuming there to be a unique underlying truth set that we anticipate our algorithm will find, an assumption we were unwilling to make. We hesitate in making such an assumption due to the stochastic nature with which we initialized our clustering and by applying some conservatism in our optimization approach. Each time a clustering is finalized, there is no guarantee that the points in a single cluster will remain in that same cluster after the next iteration. This is particularly true when comparing the final product of two clusterings produced by different weighting functions, when the structure of the clustering and the centers themselves are significantly different. The solution we used to avoid incongruous comparisons from cluster to cluster was to randomly partition our dataset into a test and validation set, with an 85/15 splitting. The concept with having a dedicated validation set would be that the clusters will have been defined and optimized based on the test set. The validation set could be then used to serve as a means to check whether our newly determined clustering also minimizes the distance of an independent set of points in the space to the cluster centers. We hoped that having a uniform set of points, for all clusterings, would sufficiently serve to provide a common measure of the performance any particular clustering has in minimizing the distances from the validation set to the cluster centers.

5.1 Rescaling

THIS WILL GO INTO THE DISCUSSION AROUND PART 3 (Check comments) Without rescaling, it is difficult to compare results with different choices of w . For example, the weighted mean distance with $w(\xi) = \xi^2$ will necessarily be much greater than the weighted mean distance with $w(\xi) = \sqrt{\xi}$. Therefore, we divide both the tolerance and the cost function by

$$\sum_i w(\xi_i)$$

This roughly corresponds to a rescaling to the same scale of the case when $w(\xi) = \xi$. Rescaling the tolerance is equivalent to choosing a smaller tolerance to start with. However, rescaling is a systematic way of converting a tolerance with a physical interpretation to one suited to the problem.

5.2 Effect of Number of Centers on Cost

Figure 2 shows the cluster arrangements for unscaled linear weighting for $k = 1, \dots, 10$ clusters. Also shown in figure 3 is the validation cost as a function of number of clusters.

As expected, centers are consistently located near urban areas. Also as expected, cost decreases, albeit erratically, with the number of centers. The five center arrangement, for example, results in higher cost than the four center arrangement, this is likely due to the random initialization of the centers. Had

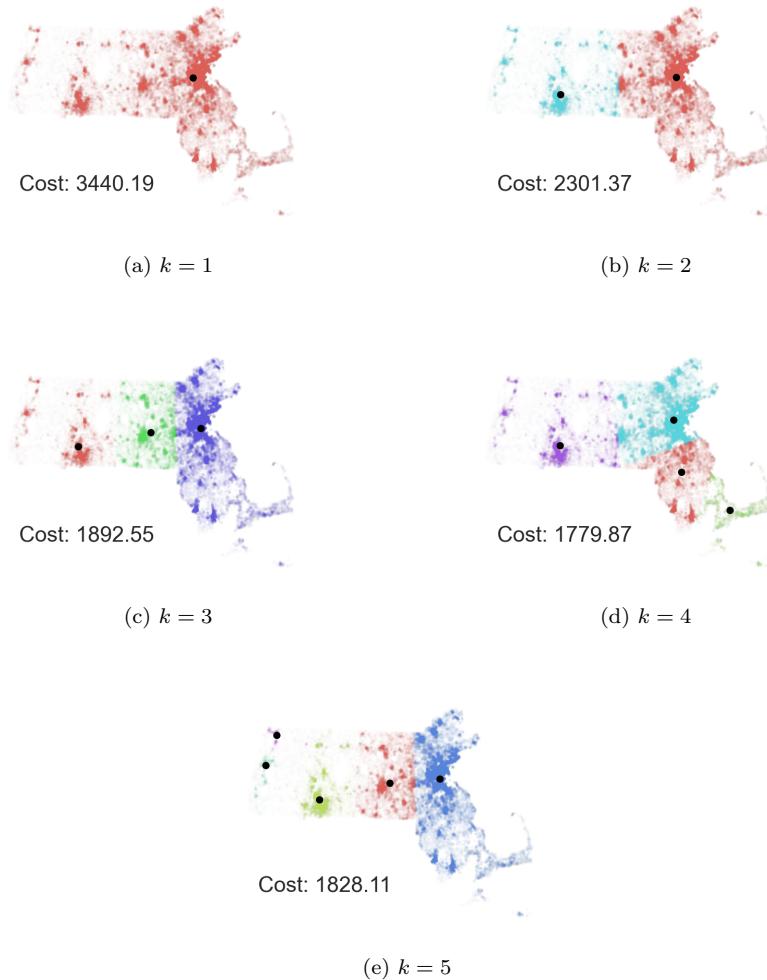


Figure 2: Distribution center placement and validation cost under linear weighting for $k = 1, \dots, 5$. The value of the cost function is displayed, and centers are shown in black.

time permitted, performing multiple runs and then averaging their results would have produced a smoother curve with fewer unexpected results. Nonetheless, the results are consistent with expectations and exhibits behavior from the established clustering theory. That is, with an increase in the number of clusters, the cost decays asymptotically.

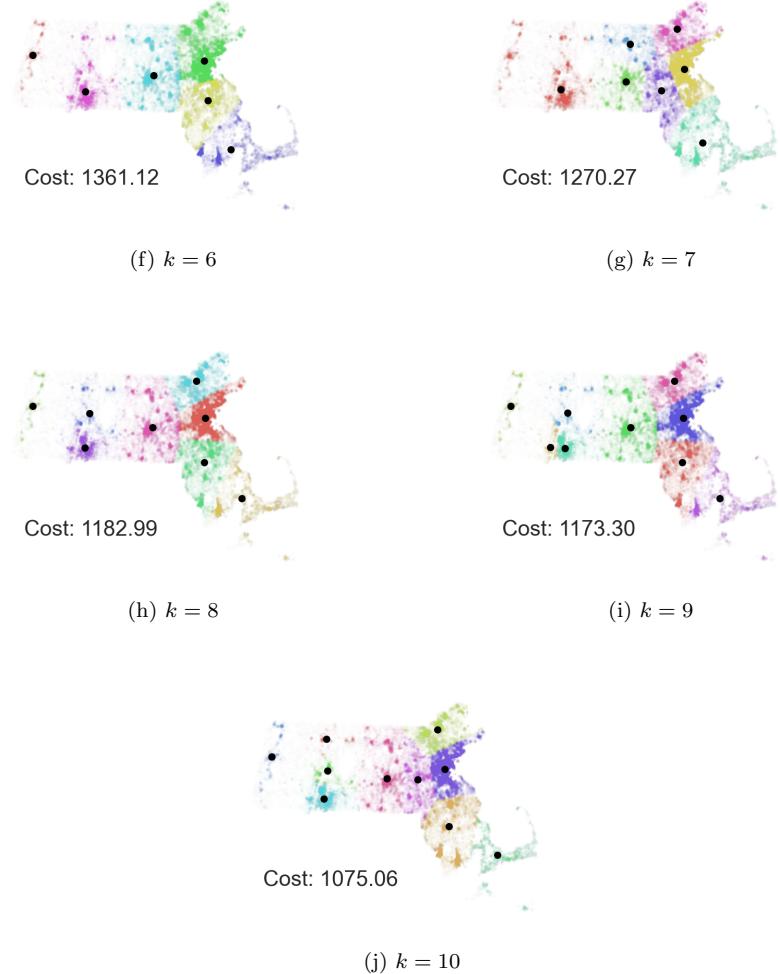


Figure 2: Distribution center placement and validation cost under linear weighting for $k = 6, \dots, 10$. The value of the cost function is displayed, and centers are shown in black.

5.3 Effect of Population Thresholding on Cost

Figure 4 shows the effects of population thresholding on center placement for selected population thresholds. Recall that a population threshold of n means that points, and corresponding weights, with population less than n are neglected when adjusting the cluster centers c_i . All thresholds were tested with five clusters.

The results are not surprising. Increasing the threshold does not have a

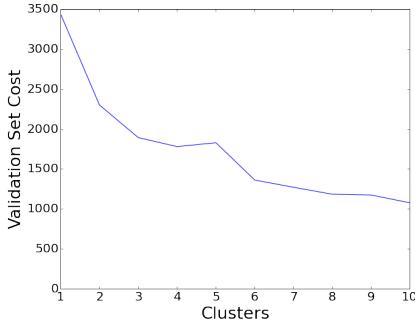


Figure 3: Validation cost vs. number of clusters

significant effect on placement of centers on a micro level. For instance, the centers near Boston and Springfield are always in roughly the same places. A sufficiently high threshold does serve to all but ignore less populated areas. At a population threshold of 100, the Boston center serves the entirety of south-eastern Massachusetts and the cape.

5.4 Comparison of Weighting Functions

Figure 5 shows the results of generating 10 clusters under four different weighting functions other than the default linear function that was demonstrated above. As in the case of one distribution center, choice of weighting function has little effect on center location. Log weighting, which curbs the influence of points with high population, does not show substantially different center placement from square weighting, which increases the influence of high population. It is possible that this behavior is a result of some non-general aspect of our problem, such as the population distribution and density of Massachusetts, or the discrete and often highly spread out distribution of points, but in our case, the weighting function has little effect on center placement.

6 Conclusions & Future Work

The choice of weighting function is a potential area of further, concentrated study. We assume that the weighting function is chosen based on *a priori* domain knowledge, which suggests an investigation into data-driven choices of weighting functions for known business problems. The focus of this project was the development of an augmentation to Lloyd’s algorithm that scaled with the number of clusters. We were successful to that end and are pleased with our clustering results.

Another interesting future variation that lends itself to study is how the clustering changes with the granularity of the population data. We utilized US Census[3] block data that, while sufficiently detailed for a macro analysis

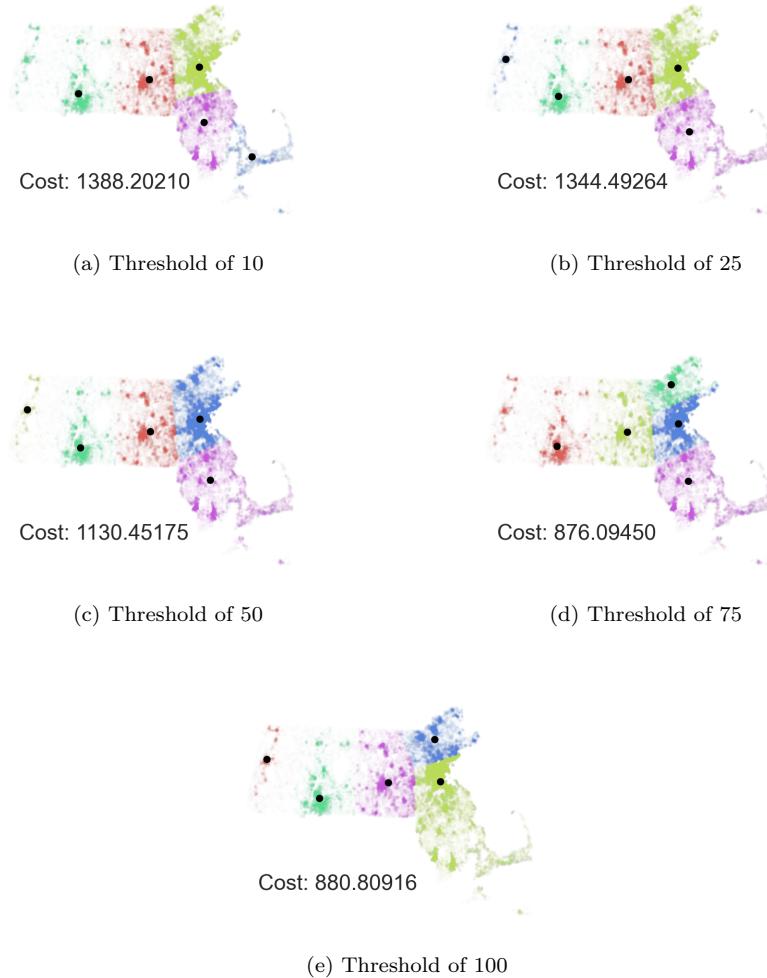


Figure 4: Distribution center placement under linear weighting for various population thresholds. The value of the cost function is displayed, and centers are shown in black.

of Massachusetts, is not detailed enough to provide adequate insight for the optimal location of distribution and/or retail centers within a densely populated city. If we had access to parcel data (storefronts and other properties at the street level) it is likely that our clustering solution would be fundamentally different from those presented in this paper.

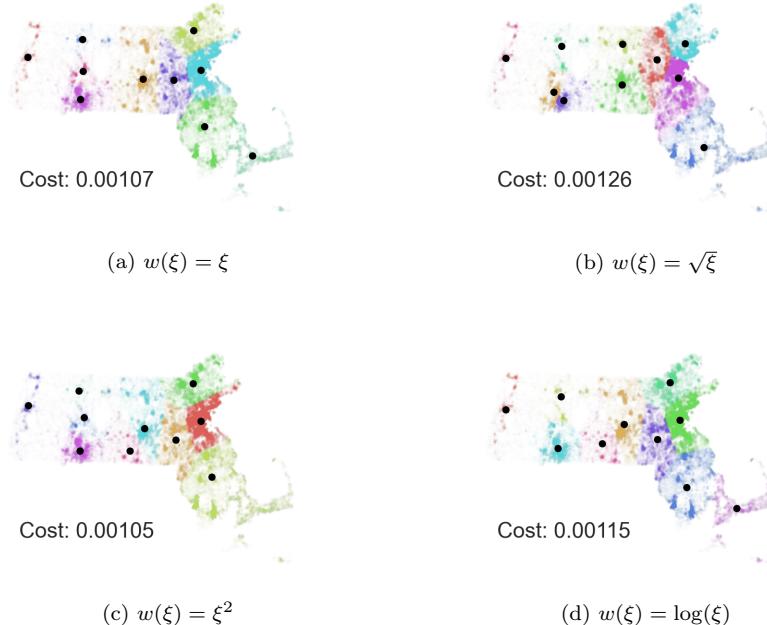


Figure 5: Distribution center placement under various weighting functions. The value of the cost function is displayed, and centers are shown in black.

References

- [1] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C., Silverman, R., & Wu, A. Y (2002). An Efficient k -Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24(7), pp. 881-892.
- [2] Lloyd, S. P (1982). Least Squares Quantization in PCM. *IEEE Transactions on Information Theory* 28(2), pp. 129-137.
- [3] U.S. Census Bureau. *TIGER/Line with Selected Demographic and Economic Data: Population & Housing Unit Counts*. Retrieved from <https://www.census.gov/geo/maps-data/data/tiger-data.html>
- [4] Kolda, T. G., Lewis, R. M., & Torczon, V (2003). Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods. *SIAM Review*, 45(3), pp. 385-482.
- [5] Davis, C. (1954). Theory of Positive Linear Dependence. *Journal of American Mathematics*, 76(4), pp. 733-746.

- [6] *Python Shapefile Library*. Github repository. Retrieved from <https://github.com/GeospatialPython/pyshp>
- [7] Chen, D. Z., Li, J., & Wang, H. Efficient Algorithms for the One-Dimensional k -Center Problem (2015). *Theoretical Computer Science* 592, pp. 135-142.
- [8] Rossiter, K. (2011, July 20). What are census blocks? [Web log post]. Retrieved from <http://blogs.census.gov/2011/07/20/what-are-census-blocks/>