# AM-SURE 2023 Final Report:
# Clustering with General Costs

Daniel Wang
dtw@brown.edu
daniel_wang1@alumni.brown.edu

July 28, 2023

## Preface

The following is a report based on work completed as part of the 10-week Applied Mathematics Summer Undergraduate Research Experience (AM-SURE) program, hosted by the Courant Institute of Mathematical Sciences at New York University during the summer of 2023. All mistakes are my own.

## 1 Factor discovery through optimal transport

### 1.1 The factor discovery problem

Consider a dataset of observations $\{x^i\}$ realized from some unknown distribution $\rho(x|z)\gamma(z)$, where $x \in \mathbb{R}^d$ is our variable of interest.

The goal of the factor discovery problem is to construct factors $z$, which are in general vector-valued, that explain the behavior of $x$. Having such an explanation lends itself to several uses. One such use is the ability to make predictions of $x$. Another is the construction of a simpler representation of $x$. Some well-known existing procedures for accomplishing the latter are clustering, which identifies a discrete-valued $z$ that represents the class of each observation, and the method of principal components, which identifies a continuous-valued $z$ that reduces the dimensionality of the observations while preserving meaningful variability.

The—as of this writing—unpublished draft "Factor discovery through optimal transport" [Tabak, 2023] provides a far more detailed treatment of the factor discovery problem, which has been simplified and condensed here.

### 1.2 The barycenter problem with known factors

Suppose, for a moment, that known factors $z_k$ are provided with the dataset $\{x^i, z_k^i\}$. To extract the relationship between the factors $z$ and the observations

$x$, we seek a transformation $y = T(x, z)$ that removes all variability among the observations that is attributable to $z_k$. In other words, the transformed data $y$ should be independent of $z_k$:

$$y \perp z_k. \tag{1}$$

However, we do not wish to explain away more variability than just that attributable to $z_k$. Otherwise, we could simply transform all of the observations to a single point, leaving them absolutely indistinguishable! Therefore, we will seek the transformation that simultaneously minimizes an integral of the transport cost $c(x, y)$ required to deform the source distribution $\rho(x)$ into the target distribution $\mu(y)$. This is a formulation of the barycenter problem in optimal transport:

$$\underset{y=T(x,z)}{\operatorname{argmin}} \iint c(x,y)\rho(x|z)\gamma(z)\,\mathrm{d}x\,\mathrm{d}z \quad \text{such that } y \perp z_k. \tag{2}$$

## 1.3 The complete statement of the factor discovery problem with hidden factors

Now, in the reality of the factor discovery problem, we will not have access to these known factors $z_k$. Even if we do, our objective still lies in discovering new, hidden factors $z_h$. Recall that, at a high level, we seek

$$\begin{aligned} &\min \mathbf{Variability}(T_\sharp(\rho(x), \gamma(z_h)) \\ &= \min \mathbf{Variability}(\mu(y)) \end{aligned} \tag{3}$$

for some definition of **Variability**(), all while minimizing the deformation caused by the transport map $T(x, z)$. In the case of the canonical, quadratic cost $c(x, y) = \|x - y\|^2$, we define

$$\begin{aligned} \mathbf{Variability}(\mu(y)) &= \min_{\hat{y}} \int \|y - \hat{y}\|^2 \mu(y)\,\mathrm{d}y \\ &= \int \|y - \mathbb{E}[y]\|^2 \mu(y)\,\mathrm{d}y \\ &= \sigma^2(\mu(y)), \end{aligned} \tag{4}$$

which is simply the variance. Omitting some details, it can then be proven, still adhering to the quadratic cost, that

$$\sigma^2(\mu(y)) = \sigma^2(\mu(y)) + [\min_{y=T(x,z)} \iint \|x - y\|^2 \rho(x|z)\gamma(z)\,\mathrm{d}x\,\mathrm{d}z \text{ s.t. } y \perp z_h]. \tag{5}$$

In order words, the variance of the original distribution can be decomposed into a sum: the variance of the transformed distribution plus the reduction in variance imputed to transport. Although Equation 5 shows a relation specific to the quadratic cost, it is natural to state, more generally, that

$$\begin{aligned} &\mathbf{Variability}(\rho(x)) \\ &= \mathbf{Variability}(\mu(y)) + [\min_{y=T(x,z_h)} \iint c(x,y)\rho(x|z)\gamma(z)\,\mathrm{d}x\,\mathrm{d}z \text{ s.t. } y \perp z_h], \end{aligned} \tag{6}$$

which can be made rigorous by a careful choice of **Variability**().

Since **Variability**$(\rho(x))$ is fixed in our problem, we can—rather than minimize **Variability**$(\mu(y))$—instead maximize the transport cost over $z_h$. Thus, we can augment the minimization problem in Expression 2 to arrive at our complete problem statement of factor discovery through optimal transport:

$$\max_{z_h} \left[ \min_{y=T(x,z)} \iint c(x,y)\rho(x|z)\gamma(z)\,\mathrm{d}x\,\mathrm{d}z \quad \text{such that } y \perp z_h \right]. \tag{7}$$

# 2 Clustering as a form of factor discovery

## 2.1 A relaxation to *k*-means

If the sought factor is known to be discrete, meaning that

$$z^i \in \{z_1, z_2, \dots, z_p, \dots, z_k\} \;\forall\, i, \tag{8}$$

the factor discovery problem becomes clustering; we seek to partition our dataset $\{x^i\}$ into $k$ classes. In the case of such a discrete, categorical factor, a natural relaxation of the full independence condition is to demand that the conditional mean $\overline{y}(z)$ of the transformed data be independent of $z$. Figure 1 provides a rough illustration of such a transformation. Note that, in general, the conditional distributions (labeled as clusters 1, 2, and 3 in this data-driven case) need not and will not be isomorphic.

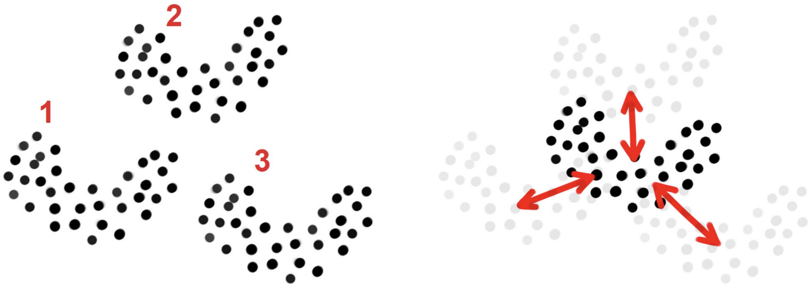


Figure 1: Transforming three clusters of data to render their means identical [Yang and Tabak, 2020].

A second premise of our relaxation is to adopt the quadratic transport cost. With these two premises,

1. $\overline{y} = \overline{y}(z_a) = \overline{y}(z_b) \;\forall\; z_a, z_b \in \{z_1, \dots, z_k\}$
2. $c(x,y) = \|x-y\|^2$,

it can be shown that the factor discovery problem in Expression 7 becomes

$$\max_{I_p} \sum_{p=1}^{k} [I_p]\, \|\overline{y} - \overline{x}(z_p)\|^2 \tag{9}$$

where $I_p$ is the set of the identities of all observations assigned to class $p$ and $[I_p]$ is its cardinality and where $\overline{x}(z_p)$ is the mean of all observations assigned to class $p$. In other words, we seek to assign the observations so as to maximize a weighted sum of the costs of transporting the conditional means. Via a relation similar to that in Equation 6, this maximization problem is equivalent to the minimization problem

$$\min_{I_p} \sum_{p=1}^{k} \sum_{i \in I_p} \|x^i - \overline{x}(z_p)\|^2. \tag{10}$$

Further, it can be shown that in the limit of numerous observations belonging to each class, Expression 10 is equivalent to the global solution sought by the well-known $k$-means algorithm [Tabak, 2023].

## 2.2 Standard *k*-means clustering

The standard $k$-means algorithm is as follows:

**Algorithm 1** $k$-Means Clustering Algorithm

```
Require: Dataset $\{x^i\}$, Number of clusters $k$
Ensure: Centroids $\{\omega_p\}_{p=1}^k$
  Initialize $\{\omega_p\}$ at $k$ random data points from $\{x^i\}$
  while Centroids change do
      for each point $x^i$ in $\{x^i\}$ do
          $z^i \leftarrow \operatorname{argmin}_{p \in \{1:k\}} ||x^i - \omega_p||$ (i.e. assign $x^i$ to the nearest centroid)
      end for
      for each centroid $\omega_p$ in $\{\omega_p\}$ do
          Update $\omega_p$ to be the centroid of all points currently assigned to class $p$
      end for
  end while
```

The $k$-**means** algorithm in particular arises from our relaxation because of our second premise: adopting the quadratic cost. While best known by its explicit formula, the arithmetic mean $\overline{x}$ of a set of real numbers $\{x^i \in \mathbb{R}\}_{i=1}^N$

$$\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x^i \tag{11}$$

happens also to be the solution to the optimization problem

$$\operatorname*{argmin}_{\hat{x}} \sum_{i=1}^{N} |x^i - \hat{x}|^2, \tag{12}$$

which was implied earlier by our substitution in Equation 4. For a vector-valued set, the mean is simply the vector of component-wise means.

### 2.3 An extension to $k$-medians

What if we had chosen not the quadratic cost $c(x, y) = \|x - y\|^2$ but some other cost? One alternative that immediately comes to mind is the linear cost $c(x, y) = \|x - y\|$ (i.e. the euclidean distance). It seems natural then, given a set of real numbers $\{x^i \in \mathbb{R}\}_{i=0}^N$, to think of

$$\operatorname*{argmin}_{\hat{x}} \sum_{i=0}^{N} |x^i - \hat{x}| \tag{13}$$

as some sort of "modified mean" of the set. In fact, this quantity already goes by a familiar name:

$$\begin{aligned} &\operatorname*{argmin}_{\hat{x}} \sum_{i=0}^{N} |x^i - \hat{x}| \\ &= x^{\frac{N}{2}} \\ &= \textbf{Median}\left(\{x^i \in \mathbb{R}\}_{i=0}^N\right). \end{aligned} \tag{14}$$

It is the median, the middle value, of the set. In that spirit, we might consider an algorithm $k$-**medians** that seeks $k$ points, each of which minimizes the sum of the distances to its attributed points. Naïvely, we might try to do this component-wise on a set of vector-valued data. For the 2D dataset of 11 points shown in Figure 2, the component-wise medians yield the red point. What we intuitively want is the more natural "geometric median", drawn approximately in the right-side diagram.

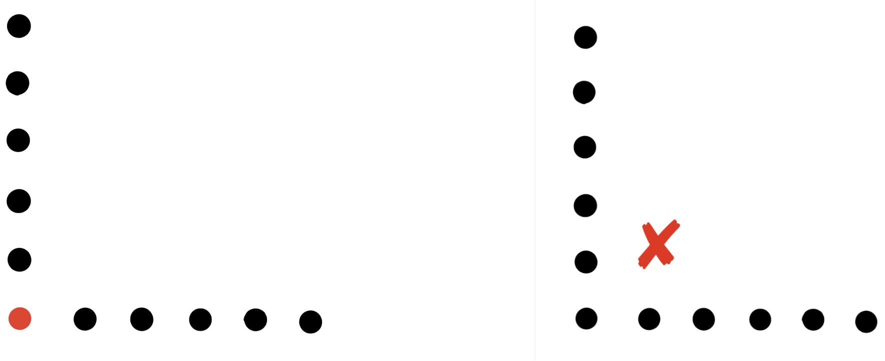


Figure 2: Comparison of component-wise (left) and geometric (right) medians of a 2D dataset containing 11 points.

The aforementioned naïve approach of finding component-wise medians is actually the block separable formulation of another problem, which is the minimization of the sum of manhattan distances (i.e. $L^1$ norms):

$$\underset{\hat{x}}{\operatorname{argmin}} \sum_{i=1}^{N} \|x^i - \hat{x}\|_1. \tag{15}$$

The geometric median of a set of vectors is simply the minimizer of the sum of euclidean distances (i.e. $L^2$ norms),

$$\underset{\hat{x}}{\operatorname{argmin}} \sum_{i=1}^{N} \|x^i - \hat{x}\|_2, \tag{16}$$

which has no component-wise formulation. In fact, it has no explicit solution outside a few special cases [Drezner et al., 2002].

## 2.4 An extension to general costs

Having seen $k$-medians, we might wish to extend the standard $k$-means algorithm to further costs yet, arriving at an algorithm for what one might call $k$-“general centers”:

**Algorithm 2** $k$-“General Centers” Clustering Algorithm

```
Require: Dataset $\{x^i\}$, Number of clusters $k$
Ensure: Centers $\{\omega_p\}$
  Initialize $\{\omega_p\}$ at $k$ random data points from $\{x^i\}$
  while Centers change do
      for each point $x^i$ in $\{x^i\}$ do
          $z^i \leftarrow \underset{p \in \{1:k\}}{\operatorname{argmin}} c(x^i, \omega_p)$ (i.e. assign $x^i$ to the nearest center)
      end for
      for each center $\omega_p$ in $\{\omega_p\}$ do
          Update $\omega_p$ to be $\underset{\hat{\omega}}{\operatorname{argmin}} \sum_{i \in I_p} c(x^i, \hat{\omega})$
      end for
  end while
```

# 3 *k*-GenCenters

## 3.1 The module (available on GitHub)

$k$-GenCenters (short for $k$-"General Centers") is a new module that can perform Algorithm 2 for a variety of transport costs $c(x, y)$. As of this writing, these include the $L^p$ norm for various values of $p$,

$$c(x,y) = \left( \sum_{d=1}^{D} |x_d - y_d|^p \right)^{\frac{1}{p}}, \quad x, y \in \mathbb{R}^D, \quad p \in \mathbb{R}, \quad p \geq 1, \tag{17}$$

and some powers of the euclidean distance,

$$c(x,y) = \left( \sqrt{\sum_{d=1}^{D} (x_d - y_d)^2} \right)^{n}, \quad x, y \in \mathbb{R}^D, \quad n \in \mathbb{N}. \tag{18}$$

The $k$-GenCenters module is written in Python and styled after the well-known `sklearn.cluster.KMeans` class from the `scikit-learn` library but provides some functionalities that do not exist in its `sklearn` counterpart. These include a method that evaluates the accuracy of the clustering against the true labels (if provided) and a method to generate the Voronoi diagram of the clustering. Future users are encouraged to contribute custom costs and additional functionalities. In theory, the transport cost used in $k$-GenCenters could be quite exotic; it need not be translation invariant or even obey the metric space axioms! Figure 3 shows a demonstration of the $k$-GenCenters module for a few costs.

The $k$-GenCenters algorithm has a time complexity of $\mathcal{O}(ndki)$, where $n$ is the number of data points, $d$ is the dimensionality of the data, $k$ is how many centers are sought, and $i$ is how many iterations are undertaken during each update of the centers. The value of $i$ can be tuned by the user and does not apply to standard $k$-means; $i$ applies to $k$-medians, which is implemented using an iterative algorithm [Weiszfeld, 1937], and to most other costs offered by $k$-GenCenters, which are implemented using gradient descent as of this writing.

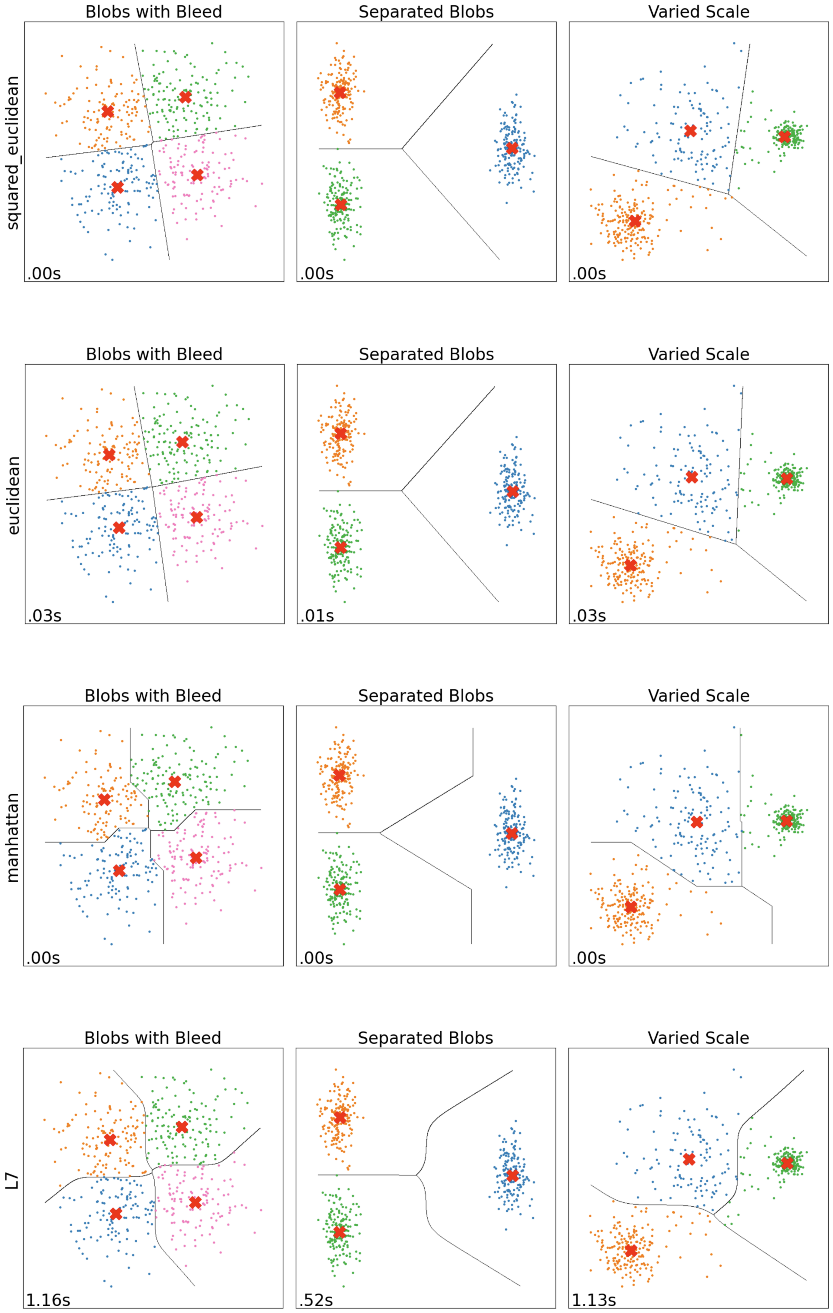


Figure 3: Demonstrating the $k$-GenCenters module on three toy datasets. Using `squared_euclidean` and `euclidean` costs, the first two rows show $k$-means and $k$-medians, respectively. A red `x` shows the final position of each center, coloration of the data shows the final assignments, Voronoi boundaries are drawn in black, and runtime is printed in the southwest corner of each plot.

## 3.2 A comparison of $k$-means and $k$-medians

Of the costs that $k$-GenCenters offers, `squared_euclidean` and `euclidean` run fastest. They are also closely related, providing for an apt comparison of $k$-means and $k$-medians. In the interest of brevity, some results have been omitted from this and subsequent sections; the motivated reader might consult the accompanying presentation slides, which contain additional pertinent graphics.

### 3.2.1 Robustness to outliers

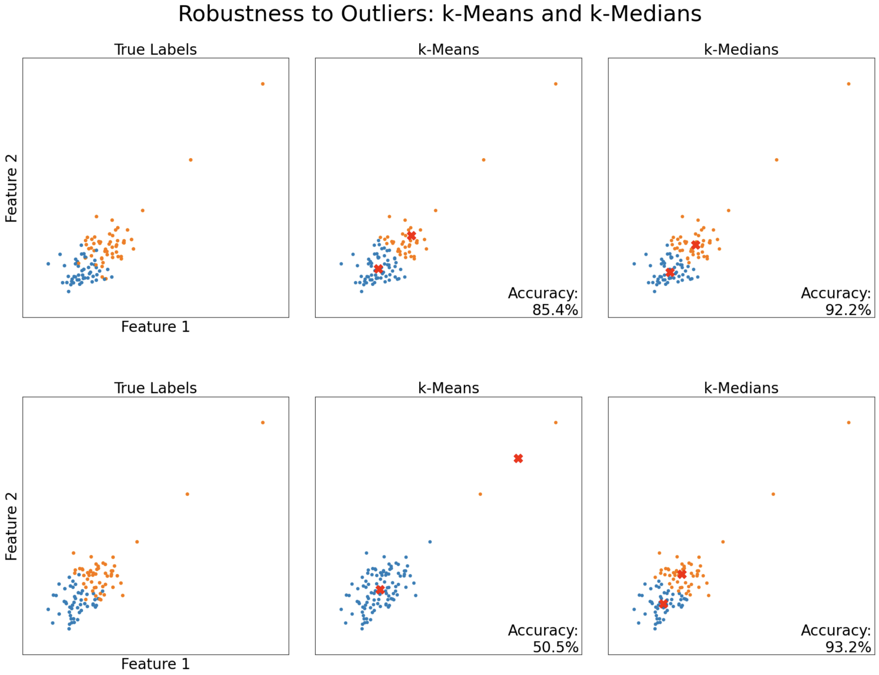


Figure 4: Differential failure of $k$-means relative to $k$-medians on a toy dataset with outliers, despite identical initialization. The first row shows a relatively inaccurate clustering found by $k$-means due to the costly pull of outliers. The second row of plots shows $k$-means catastrophically trapped in a local optimum. The leftmost column simply shows the dataset colored by its ground-truth labels.

One would expect $k$-medians to be more robust to outliers than $k$-means since the `squared_euclidean` cost explodes for faraway points. As result, $k$-means tends to find an optimum that has lower absolute accuracy than $k$-medians on outlier-ridden datasets. Figure 4 illustrates this weakness of $k$-means on an admittedly antagonistic toy dataset.

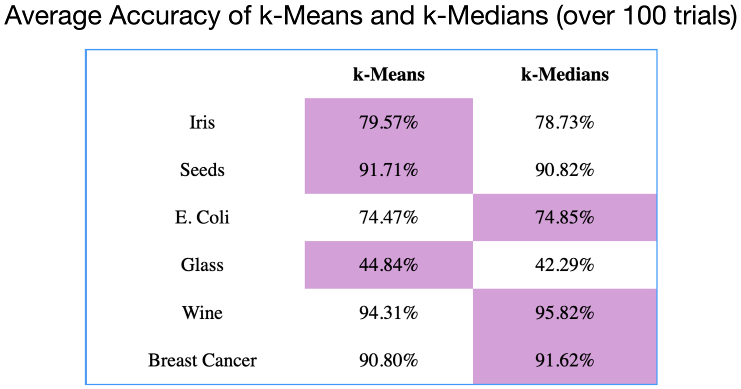
### 3.2.2 Performance on real-world datasets

For a more practical test of the relative merit of $k$-means and $k$-medians, we apply both algorithms to a gamut of real-world datasets. Table 1 shows the datasets that were used, along with some of their properties.

| Dataset | Size | Dimensions | Classes |
|---|---|---|---|
| Iris | 150 | 4 | 3 |
| Seeds | 210 | 7 | 3 |
| E. coli | 336 | 7 | 8 |
| Glass | 214 | 9 | 6 |
| Wine | 178 | 13 | 3 |
| Breast Cancer | 569 | 30 | 2 |

Table 1: Specifications of the six real-world datasets used to compare $k$-means and $k$-medians.

Table 2 shows the average accuracy over 100 trials of $k$-means and $k$-medians, with a purple fill indicating the winning algorithm on each dataset. Based on the number of wins, $k$-means and $k$-medians seem tied, each winning four out of the eight datasets. There appears to be a trend that favors $k$-medians on higher-dimensional datasets, but this trend is inconclusive. Moreover, given that pairwise distances between points tend to become more uniform in keeping with the so-called "curse of dimensionality" [Steinbach et al., 2004], one would even expect the `squared_euclidean` cost to *help* produce meaningful differentiation of the data. For these reasons, such a trend seems spurious.



Average Accuracy of k-Means and k-Medians (over 100 trials)

| | **k-Means** | **k-Medians** |
|---|---|---|
| Iris | 79.57% | 78.73% |
| Seeds | 91.71% | 90.82% |
| E. Coli | 74.47% | 74.85% |
| Glass | 44.84% | 42.29% |
| Wine | 94.31% | 95.82% |
| Breast Cancer | 90.80% | 91.62% |

Table 2: Average accuracy over 100 trials of $k$-means and $k$-medians on six real-world datasets. A purple fill indicates the higher value in each row (i.e. the highlight shows which algorithm was more accurate).

Another important consideration in comparing $k$-means and $k$-medians, not thoroughly explored in this investigation, is the stability of each algorithm. As clustering is inherently an unsupervised learning task, the true labels are not usually available. One might be willing to make a significant sacrifice in average accuracy to gain some stability, to avoid catastrophic outcomes, to achieve a worst-case guarantee, etc.

#### 3.2.3 A verdict

Historically, an advantage of k-means over k-medians has been its speed; there is an explicit formula for the centroid, so the update step is fast. However, many datasets are practically modest in their size, dimensionality, and class count, including those heretofore used in this comparison. Even the time required to run $k$-medians on one of these datasets is trivial—a fraction of a second on a modern laptop computer. Therefore, the ostensible advantage of $k$-means over $k$-medians on such datasets is greatly diminished. Especially if one anticipates an outlier-ridden dataset, $k$-medians may be the better option.

### 3.3 Improving initialization

In the spirit of clustering with general costs, something else that we might attempt is to improve the initialization of the centers.

#### 3.3.1 The Forgy initialization

The Forgy initialization is the canonical initialization of the $k$-means algorithm. Each center is initialized randomly, with uniform probability over the data points [Peña et al., 1999]:

$$\mathbb{P}(\omega_p = x^i) = \frac{1}{N} \; \forall \, p, i. \tag{19}$$

#### 3.3.2 The random partition initialization

The random partition initialization does not initialize the centers. Instead, it initializes the *assignments* with uniform probability over the classes [Peña et al., 1999], effectively partitioning the dataset and accomplishing the first step of the main **while** loop in Algorithm 1:

$$\mathbb{P}(z^i = p) = \frac{1}{k} \; \forall \, i, p. \tag{20}$$

The random partition initialization tends to produce centers that start close together, all near the heart of the dataset, and then migrate apart as the algorithm proceeds.

#### 3.3.3 The $k$-means++ initialization

The $k$-means++ algorithm is a relatively modern technique that proposes an improvement over traditional initializations. Again, as in Forgy, we initialize the centers. However, we do not do this with uniform probability over the data points. Instead, we initialize the centers successively with a probability over the data points that grows with distance from the pre-existing centers [Arthur and Vassilvitskii, 2007]:

$$\mathbb{P}(\omega_p = x^i) = \frac{\min\limits_{\omega_{l<p}} \|x^i - \omega_l\|_2^2}{\sum_{j=1}^N \min\limits_{\omega_{l<p}} \|x^j - \omega_l\|_2^2} \;\forall\; i, p. \tag{21}$$

Thus, we incentivize the centers to spread out. Such spread helps to mitigate undesirable local optima and other issues.

### 3.3.4 The $k$-GenCenters++ initialization

In line with this flavor of argument, we might propose a bespoke initialization called $k$-GenCenters++, in which the aforementioned probability of placing a center at a data point grows not with the square of the euclidean distance but with an arbitrary transport cost:

$$\mathbb{P}(\omega_p = x^i) = \frac{\min\limits_{\omega_{l<p}} c(x^i, \omega_l)}{\sum_{j=1}^N \min\limits_{\omega_{l<p}} c(x^j, \omega_l)} \;\forall\; i, p. \tag{22}$$

For instance, one might consider replacing the squared euclidean distance with the *cubed* euclidean distance, which would provide an even more extreme incentive to place new centers far from pre-existing centers. Some preliminary results appear to show that a $k$-GenCenters++ initialization with the cubed euclidean distance $c(x, y) = \|x - y\|_2^3$ provides greater average accuracy on *certain* real-world datasets. It is worth mentioning that the effect size amounts to only a very small improvement over $k$-means++, and even then, this behavior seems to hold true only on particular datasets. Further study would be needed for a conclusive finding.

## 3.4 Ongoing work

The study of clustering with general costs may merit future work along several avenues, including a more thorough treatment of “$k$-means **vs** $k$-medians” and of improving initialization, which were earlier discussed. The following are areas of recent or ongoing work but by no means represent an exhaustive list of the directions in which we could take our inquiry.

### 3.4.1 Must-link constraints

One current area of interest as of this writing is clustering with must-link constraints. Suppose that we know that some of our observations share a factor (e.g. they originated from the same person). This provides some useful information, and the task of clustering becomes a semi-supervised learning task. We say that these points sharing a factor are “must-link” [Huang et al., 2008], and they must be assigned to the same cluster by the time our algorithm terminates.

The perspective of optimal transport with general costs offers a delightfully natural approach to this problem, namely, to assign all points in each must-link set $I_m$ to the center that minimizes the sum of the transport costs during every assignment step of Algorithm 2:

$$z^i = \underset{p}{\operatorname{argmin}} \sum_{i \in I_m} c(x^i, \omega_p) \;\forall\, m. \tag{23}$$

We can think of this as the "cheapest" center for the must-link set.

Under this technique, the centers can be expected to converge much faster, providing a speedup to the algorithm. Currently, the implementation of the $k$-GenCenters module available on GitHub is equipped with this must-link functionality, and the motivated reader is encouraged to try it for herself.

### 3.4.2 Weighted surrogates

Given large must-link sets, we may have an opportunity to achieve another speedup yet. We could reduce the number of operations required during every assignment step by replacing the must-link set with a weighted surrogate and then assigning all members of the must-link set based on some weighted cost of transporting that surrogate. Make the assignments

$$z^i = \underset{p}{\operatorname{argmin}}\, W(c(S(\{x^i\}_{i \in I_m}), \omega_p)) \;\forall\, m \tag{24}$$

where $W()$ is some weighting function chosen carefully based on $c()$ (this could be as simple as a scaling factor like the cardinality of the must-link set $[I_m]$) and where $S(\{x^i\}_{i \in I_m})$ is some surrogate point chosen carefully based on $c()$ that aggregates the points in the must-link set (this could be as simple as the centroid). Employing such a weighted surrogate might allow us to avoid entirely the business of summing the individual transport costs of all points belonging to a large must-link set. This substitution may be especially practical in the early iterations of the $k$-GenCenters algorithm, when the centers need only migrate quickly and roughly in the right direction. Toward the later iterations, we may wish for our algorithm to adaptively reintroduce information about the individual points of each must-link set so that the final, converged assignments are as accurate as possible.

## 4 Acknowledgments

The author would like to thank Ryan Shìjié Du, Andrew Lipnick, Nina Mortensen, and Esteban G. Tabak for their generous mentorship on this project. The author remains grateful also to Aleksandar Donev, Ryan Shìjié Du, Fortino Garcia, and Olivia Pomerenk for their work in organizing the AM-SURE 2023 program.

# References

David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 9780898716245.

Zvi Drezner, Kathrin Klamroth, Anita Schöbel, and George Wesolowsky. The weber problem. *Facility Location: Applications and Theory*, 01 2002. doi: 10.1007/978-3-642-56082-8_1.

Haichao Huang, Yong Cheng, and Ruilian Zhao. A semi-supervised clustering algorithm based on must-link set. In Changjie Tang, Charles X. Ling, Xiaofang Zhou, Nick J. Cercone, and Xue Li, editors, *Advanced Data Mining and Applications*, pages 492–499, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-88192-6.

J.M Peña, J.A Lozano, and P Larrañaga. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters*, 20(10):1027–1040, 1999. ISSN 0167-8655. doi: 10.1016/S0167-8655(99)00069-0.

Michael Steinbach, Levent Ertöz, and Vipin Kumar. *The Challenges of Clustering High Dimensional Data*, pages 273–309. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-662-08968-2. doi: 10.1007/978-3-662-08968-2_16.

Esteban G. Tabak. Factor discovery through optimal transport. Unpublished manuscript, June 2023.

Endre Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386, 1937.

Hongkang Yang and Esteban G Tabak. Clustering, factor discovery and optimal transport. *Information and Inference: A Journal of the IMA*, 10(4):1353–1387, 12 2020. ISSN 2049-8772. doi: 10.1093/imaiai/iaaa040.