Analyzing Iterated Linear Optimization as a Rounding Step for Semidefinite Programming

Teressa Chambers

Brown University

September 17, 2022

Overview

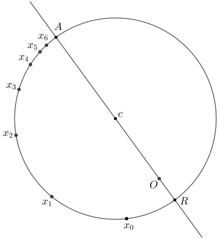
1. Laying the Groundwork
2. Current Investigations
3. Future Directions

Iterated Linear Optimization

Let $\triangle \subset \mathbb{R}^n$ be a compact, convex set. Define $T : \mathbb{R}^n \to \triangle$ as follows:

$$T(x) = \operatorname*{argmax}_{y \in \triangle} x \cdot y$$

Fixed point iteration is the construction of a sequence $x_{i+1} = T(x_i)$ starting with x_0 and ending when $x_{i+1} = x_i$.



Felzenszwalb, Klivans, and Paul. "Iterated Linear Optimization" and "Clustering with Semidefinite Programming and Fixed Point Iteration." 2021, 2022. arXiv:2012.02213, arXiv:2012.09202

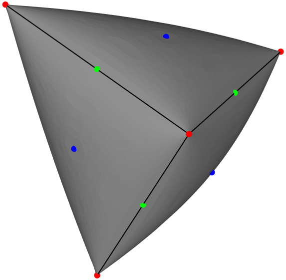
Elliptopes, Summarized

The elliptope $\mathcal{L}_n$ is defined as follows:

$$\mathcal{L}_n = \{X \in \mathcal{S}(n) | X \succeq 0, X_{ii} = 1\}$$

The vertices of the elliptope are symmetric matrices of rank 1, whose entries are all in {-1,1}.

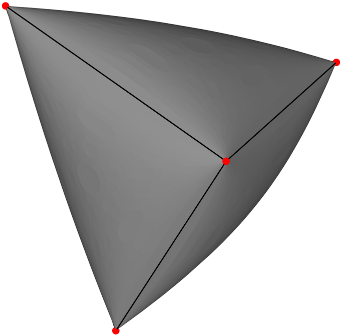
The elliptope can be represented geometrically in $\mathbb{R}^{\frac{n(n-1)}{2}}$:

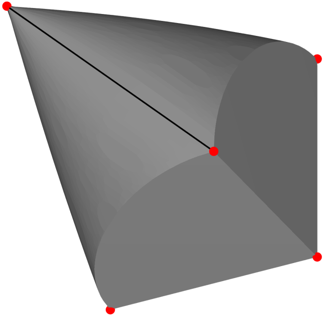


The k-way Difference

The k-way elliptope $\mathcal{L}_{n,k}$:

$$\mathcal{L}_{n,k} = \{X \in \mathcal{L}_n | X_{ij} \geq -\frac{1}{k-1}\}$$





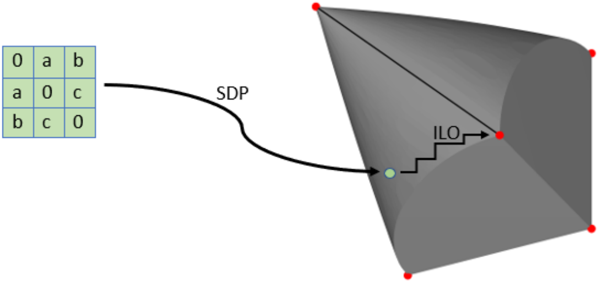
Using the Elliptope for Clustering

Consider a set of n data points in $\mathbb{R}^d$.

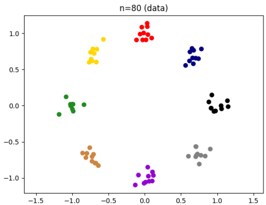
Step 1: Construct an n-by-n symmetric matrix M of the Euclidean distances between data points

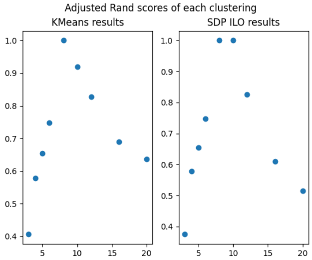
Step 2: Solve a semidefinite programming (SDP) problem to get a matrix Z on the k-way elliptope $\mathcal{L}_{n,k}$

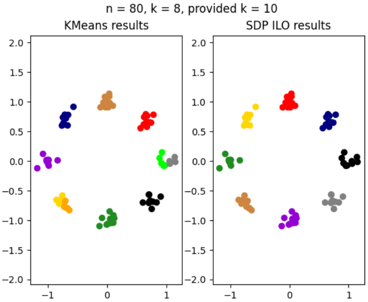
Step 3: Use iterated linear optimization (ILO) to round Z to a vertex, representing a partition of the data

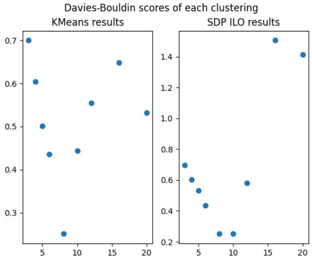


Clustering Performance Analysis

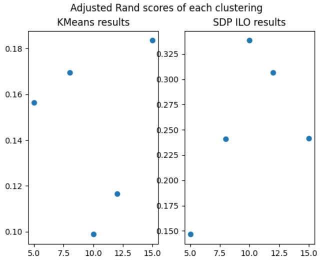


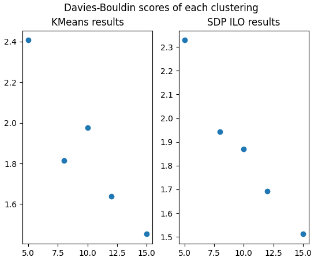


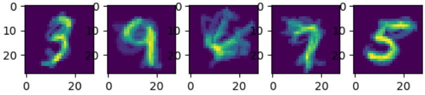


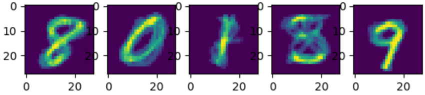


Clustering on MNIST









Future Directions

A few immediate investigations:

- "Niceness" conjecture: If $k = m$ and the maximum distance between any two points in a true cluster is smaller than the distance between any two points in different true clusters, the algorithm will return the correct clusters.
- Partition preference: How and why does the algorithm avoid "extreme" partitions even when these are the true partitions of data?
- Comparison testing: Similar experiments will be performed to compare the algorithm to another SDP-based method (Mixon et al., 2016).

Does this iterative algorithm **almost always** converge to a vertex of the k-way elliptope?