Supplemental Material: Power k-Means Clustering

Jason Xu [1] Kenneth Lange [2]

A. KHM$_p$ as approximate Newton method

We consider the recursion proposed in (Zhang, 2001) for solving the generalized KHM$_p$ objective

$$f(\boldsymbol{\Theta}) = \sum_{i=1}^{N} \frac{k}{\sum_{j=1}^{k} \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^{-p}}.$$

In our notation with $\boldsymbol{w}_i = (w_{i1}, \ldots, w_{ik})^t$, a valid surrogate for each component can be written

$$g(\boldsymbol{\theta}_j) = \sum_i \boldsymbol{w}_i (\|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2)^{p/2}.$$

In this form it is easy to see that g is a strictly convex function, with gradient

$$\nabla g(\boldsymbol{\theta}_j) = p \sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 (\boldsymbol{\theta} - \boldsymbol{x}_i) := p \sum_i \alpha_i (\boldsymbol{\theta} - \boldsymbol{x}_i).$$

When $p > 2$, it is no longer straightforward to solve the stationarity equation $\nabla g(\boldsymbol{\theta}_j) = 0$ explicitly. The authors (Zhang, 2001) instead propose approximately solving by treating the terms $\boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2$ as constants, leading to the recursive update

$$\hat{\boldsymbol{\theta}}_j = \frac{\sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 \boldsymbol{x}_i}{\sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2}.$$

By continuity and strict convexity of g, Brouwer's fixed point theorem indeed guarantees that the sequence $\boldsymbol{\Theta}_m$ will stabilize to some fixed point. However, we show that such an iteration loses monotonicity when $p > 2$, and hence is no longer a descent algorithm. To see this, we examine the second differential

$$\begin{aligned} d^2 g(\boldsymbol{\theta}_j) \quad =& p \sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 \boldsymbol{I} \\ &+ 2p \sum_i \boldsymbol{w}_i (\boldsymbol{\theta}_j - \boldsymbol{x}_i)(\boldsymbol{\theta}_j - \boldsymbol{x}_i)^t. \end{aligned}$$

If we ignore the second term and instead make use of the approximate Hessian

$$H(\boldsymbol{\theta}_j) = p \sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 \boldsymbol{I},$$

then the Newton-type step of the form
$\hat{\boldsymbol{\theta}}_j = \boldsymbol{\theta}_j - H(\boldsymbol{\theta}_j)^{-1} \nabla f(\boldsymbol{\theta}_j)$ recovers the update above: $\hat{\boldsymbol{\theta}}_j$ can be written

$$\begin{aligned} &\boldsymbol{\theta}_j - (p \sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2)^{-1} p \sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 (\boldsymbol{\theta}_j - \boldsymbol{x}_i) \\ = \quad & \frac{\sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2 \boldsymbol{x}_i}{\sum_i \boldsymbol{w}_i \|\boldsymbol{x}_i - \boldsymbol{\theta}_j\|^2}. \end{aligned}$$

Because we have dropped terms toward this approximate Newton step, the update is not guaranteed to be a descent step. Appropriate backtracking may remedy this, though such a maneuver has not been incorporated in KHM$_p$.

Table 1. Adjusted Rand index, k-means++ initialization

	$d=2$	$d=5$	$d=10$	$d=20$	$d=50$	$d=100$	$d=200$
Lloyd	0.760	0.857	0.871	0.878	0.892	0.886	0.901
KHM	0.762	0.843	0.842	0.850	0.870	0.865	0.887
$s_0=-3.0$	**0.775**	**0.876**	**0.939**	**0.963**	0.**988**	**0.982**	**0.987**
$s_0=-9.0$	0.769	0.862	0.892	0.910	0.959	**0.982**	**0.987**
$s_0=-18.0$	0.767	0.858	0.882	0.887	0.923	0.945	0.959

Table 2. Standard deviations, ARI, k-means++ initialization

	$d=2$	$d=5$	$d=10$	$d=20$	$d=50$	$d=100$	$d=200$
Lloyd	0.050	0.034	0.043	0.031	0.034	0.035	0.025
KHM	0.046	0.037	0.040	0.035	0.054	0.035	0.024
$s_0=-3.0$	0.044	0.034	0.024	0.022	0.015	0.016	0.013
$s_0=-9.0$	0.044	0.035	0.040	0.031	0.020	0.016	0.014
$s_0=-18.0$	0.047	0.035	0.043	0.032	0.026	0.027	0.022

B. Additional experimental results

We present additional experimental results and details below. First, we note that though the experiment should coincide to that in (Zhang, 2001) when $d=2$, both Lloyd and KHM perform slightly better in our experiments than reported in their study (although the same trends are conveyed). This is likely due to their choice of running all algorithms for only 40 iterations. We observe it is not uncommon for Lloyd to take more than twice as many steps as they considered to converge, while KHM typically required even more. Recall we add $\pm 10^{-4}$ to all $\theta_{ij}^{(0)}$ when centers are initialized with k-means++, which chooses a subset of the datapoints as the initial guess, to avoid dividing by zero. We run all algorithms until the relative change in objective value falls below $\epsilon = 10^{-6}/\sqrt{d}$, or when any weights fall below machine precision to avoid underflow.

Following (Zhang, 2001; Hamerly & Elkan, 2002), recall we consider the variation of information as well as the root k-means quality ratio defined

$$\sqrt{f_{-\infty}(\hat{\boldsymbol{\theta}})/f_{-\infty}(\hat{\boldsymbol{\theta}}_{\text{optimal}})}$$

where $f_{-\infty}$ denotes the k-means objective. Here $\hat{\boldsymbol{\theta}}_{\text{optimal}}$ is obtained by running Lloyd's algorithm to convergence starting from the true centers $\boldsymbol{\theta}_{\text{true}}$ (depicted as triangles in Figure 2), and is of course itself an approximation to the optimum. Using $\hat{\boldsymbol{\theta}}_{\text{optimal}}$ is more appropriate than judging the quality using $f_{-\infty}(\boldsymbol{\theta}_{\text{true}})$ in the denominator, as the optimum for a given dataset may be quite different than the value used to generate the data.

Here, we additionally report the average and standard deviation of the adjusted Rand index (ARI) in Tables 1 and 2. Again, best performers for each dimension d are boldfaced. Unlike the other measures we have considered, note that a higher ARI value is better. Furthermore, we present results from initializing the center coordinates uniformly rather than seeding with k-means++. This allows us to confirm similar findings from a similar experimental setup that Zhang (2001) pioneered before modern seeding schemes were invented. Though one might argue that there is no obvious reason not to use available seeding schemes, Tables 3 through 8 are illustrative and highlight the differences between methods. In particular, this view makes it more apparent the KHM breaks down in high dimensions, and the improvement gains using power k-means are often more pronounced.

B.1. Further data examples

We consider two classic benchmarks. The BIRCH dataset consists of 100 clusters arranged in a regular grid structure in the plane, with $n = 100,000$ observations. The MNIST database contains $60,000$ examples of handwritten digits ($k=10$ clusters), each represented in a 28×28 field so that $d = 784$. Though our simulation study is modeled after a classic setup that is generous in the sense that data are generated under ideal assumptions for Lloyd's algorithm, these allow us to further compare performance on much larger datasets. On BIRCH, the best(lowest) VI of 20 k-means++ initializations is 0.227 (mean 0.410) using Lloyd's algorithm and 0.164 (mean 0.344) using power k-means. Lloyd's algorithm achieved a lowest root quality ratio of 1.018, with an average of 1.032, compared to 1.012 with an average of 1.024 under power k-means. On

Table 3. Root k-means quality ratio, uniform hypercube initialization

	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$	$d = 100$	$d = 200$
Lloyd	1.175	1.756	2.166	2.519	2.645	2.690	2.689
KHM	**1.074**	1.290	3.545	3.654	3.604	3.607	3.541
$s_0 = -3.0$	1.081	**1.157**	**1.116**	**1.091**	3.082	3.607	3.413
$s_0 = -9.0$	1.114	1.327	1.292	1.216	**1.134**	**1.073**	**1.074**
$s_0 = -18.0$	1.141	1.423	1.440	1.454	1.348	1.360	1.272

Table 4. Variation of information, uniform hypercube initialization

	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$	$d = 100$	$d = 200$
Lloyd	0.756	0.589	0.653	0.767	0.841	0.859	0.872
KHM	**0.720**	0.290	2.926	2.593	2.101	2.064	2.345
$s_0 = -3.0$	0.721	**0.191**	**0.077**	**0.040**	2.204	3.026	2.876
$s_0 = -9.0$	0.767	0.344	0.187	0.089	**0.044**	**0.022**	**0.024**
$s_0 = -18.0$	0.837	0.420	0.270	0.188	0.114	0.110	0.082

Table 5. Adjusted Rand index, uniform hypercube initialization

	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$	$d = 100$	$d = 200$
Lloyd	0.657	0.668	0.626	0.554	0.503	0.487	0.489
KHM	**0.701**	0.870	0.071	0.105	0.179	0.177	0.137
$s_0 = -3.0$	0.695	**0.894**	**0.957**	**0.979**	0.269	0.157	0.115
$s_0 = -9.0$	0.673	0.808	0.894	0.949	**0.974**	**0.988**	**0.988**
$s_0 = -18.0$	0.646	0.766	0.843	0.887	0.930	0.934	0.954

Table 6. Standard Deviations, root k-means quality ratio, uniform hypercube initialization

	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$	$d = 100$	$d = 200$
Lloyd	0.077	0.205	0.266	0.284	0.342	0.310	0.326
KHM	0.036	0.140	0.359	0.340	0.375	0.387	0.345
$s_0 = -3.0$	0.048	0.051	0.067	0.065	1.017	0.387	0.678
$s_0 = -9.0$	0.057	0.088	0.093	0.108	0.101	0.089	0.072
$s_0 = -18.0$	0.086	0.120	0.105	0.143	0.133	0.164	0.122

Table 7. Standard Deviations, variation of information, uniform hypercube initialization

	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$	$d = 100$	$d = 200$
Lloyd	0.113	0.122	0.106	0.158	0.160	0.172	0.165
KHM	0.124	0.141	0.324	0.350	0.351	0.289	0.523
$s_0 = -3.0$	0.112	0.050	0.044	0.031	1.164	0.267	0.821
$s_0 = -9.0$	0.109	0.078	0.058	0.041	0.031	0.025	0.024
$s_0 = -18.0$	0.166	0.090	0.062	0.059	0.046	0.055	0.041

Table 8. Standard Deviations, adjusted Rand index, uniform hypercube initialization

	$d = 2$	$d = 5$	$d = 10$	$d = 20$	$d = 50$	$d = 100$	$d = 200$
Lloyd	0.052	0.066	0.057	0.090	0.083	0.096	0.092
KHM	0.043	0.059	0.043	0.053	0.073	0.068	0.087
$s_0 = -3.0$	0.044	0.026	0.027	0.016	0.365	0.027	0.232
$s_0 = -9.0$	0.049	0.044	0.033	0.024	0.018	0.013	0.013
$s_0 = -18.0$	0.067	0.051	0.033	0.035	0.028	0.033	0.024

MNIST, the differences in performance were negligible: for instance, Lloyd's achieves a minimum VI of 2.123 compared to 2.119 under power k-means, and a minimum objective value of 606.56 vs 606.55 (we cannot evaluate the root quality ratio here for lack of "true centers"). However, our method is preferable in terms of runtime. Lloyd's algorithm requires roughly three times as many iterations to converge. In terms of absolute runtime, our algorithm requires an average of 21.4 seconds on MNIST while Lloyd's algorithm required 68.1 seconds on average. This also shows that our results on computational complexity of the algorithm are backed up empirically in terms of wall-clock time.

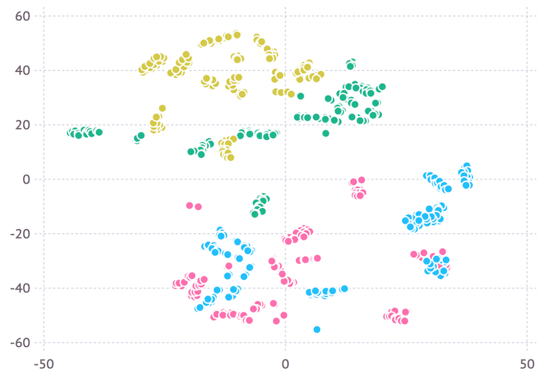
B.2. Mouse protein details

The mouse protein dataset (Higuera et al., 2015) contains expression levels of 77 proteins that produce detectable signals in the nuclear fraction of cortex. These are obtained from 34 trisomic "model" mice (those with Ts65Dn Down syndrome) and 38 wild-type "control" mice, for a total of 72 mice. There are 1080 total measurements obtained, with 510 among the trisomic mice and 570 among controls.

We preprocess the data exactly following Higuera et al. One mouse is missing the majority of protein measurements, and was discarded. Other missing measurements are replaced with the average expression level of the corresponding protein within the same class of mice. All measurements are normalized column-by-column to lie within $[0, 1]$.

Higuera et al. choose a 7×7 self-organizing map (SOM) to cluster control mice, and a 6×6 SOM to cluster the trisomic mice (Higuera et al., 2015). Note that the computational complexity for SOMs is greater by a factor of n compared to power k-means. The number of nodes in the SOM plays the analogous role to the number of centers k, and so we use $k = 49$ and 36 respectively for fair and direct comparison.

Though SOMs underperform under the same measures of quality considered in (Higuera et al., 2015), they provide data visualization. Below, we give an example of a two-dimensional visualization of the mouse protein data using t-distributed stochastic neighbor embedding (t-SNE) (Maaten & Hinton, 2008).


(a) Control

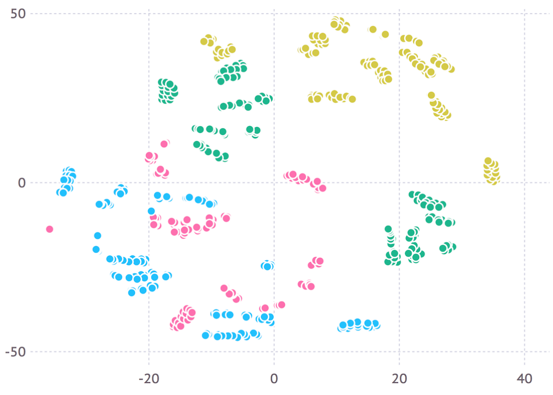

(b) Trisomic

Figure 1. Two-dimensional visualization of mouse protein expression data via t-SNE.

References

Hamerly, G. and Elkan, C. Alternatives to the k-means algorithm that find better clusterings. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, pp. 600–607. ACM, 2002.

Higuera, C., Gardiner, K. J., and Cios, K. J. Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome. *PloS one*, 10(6):e0129126, 2015.

Maaten, L. v. d. and Hinton, G. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

Zhang, B. Generalized k-harmonic means—dynamic weighting of data in unsupervised learning. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pp. 1–13. SIAM, 2001.