Fast indefinite multi-point (IMP) clustering

Working notes

BY S. CHANDRASEKARAN

May 7, 2014

July 31, 2014

1 Introduction

Let the M columns of the matrix $X \in \mathbb{R}^{N \times M}$ denote points in $\mathbb{R}^N$ that we would like to cluster, using the K-means algorithm for example. The word cluster does not seem to have a unique mathematical meaning in the literature, but is used in a variety of situations for different purposes. Loosely speaking one would like to partition the M columns X_j of X into mutually exclusive subsets such that columns in the same subset are close to each other while columns in different subsets are far apart from each other. Suppose we divide the columns into K subsets $\mathcal{S}_k$ for $0 \leqslant k < K$. Then a possible mathematical problem that captures the intent is to pick subsets $\mathcal{S}_k$ such that the sum of the squares of the intra-cluster separations,

$$\sum_{0 \leqslant k < K} \sum_{X_i, X_j \in \mathcal{S}_k} \|X_i - X_j\|^2,$$

is minimized, where $\|\cdot\|$ denotes the Euclidean norm.

(One reason for squaring the distance is to simplify the optimization algorithm when the Eucliden norm is used.) However, for reasons of practicality and efficiency, the K-means algorithm uses a different (but equivalent in the case of squared Euclidean distance with weights) formulation.

The $\mathcal{S}_k$ are restricted to Voronoi cells. To each $\mathcal{S}_k$ there is assigned a column Y_k such that $x \in \mathcal{S}_k$ if k is the smallest integer for which

$$\|x - Y_k\| = \min_{0 \leqslant l < K} \|x - Y_l\|.$$

While this can be used with arbitrary norms, that is of no interest in this paper, and we will continue assuming that the norm is the standard Euclidean norm. In this formulation it is usually conventional to measure the goodness of a clustering via the expression,

$$\sum_{0 \leqslant k < K} \sum_{X_j \in \mathcal{S}_k} \|X_j - Y_k\|^2.$$

This considerably decreases the flop count of algorithms that try minimize the above expression, as there are many fewer terms involved if $K \ll M$.

There are many algorithms that directly or indirectly try minimize the above expression over the K columns Y_j. However it is difficult to the find the global minimum and the quality of the local minimum may not be good, though there does not necessarily seem to be agreement over this in the literature, as the precise local minima at which the algorithm stops depends on the starting point.

The aim of this paper is to consider a larger class of objective functions for choosing the partitioning of the columns X_j, in order to provide more flexibility in practice, while at the same time retaining the guaranteed descent feature of the standard K-means algorithm. In fact the standard K-means algorithm will be a special case. However, we do not claim that the clusters our algorithm computes will be better in practice; this must be determined by the data and intended use of the clusters. (We provide data on some synthetic data sets in the experimental section.) Nor do we provide statistical justifications for our choice of objective functions, even though these would clearly be of great interest, as this would be non-trivial and would deviate from the main aim of the paper.

We propose two changes to the standard K-means algorithm. The first is that we try to bring in some flavor of hierarchical clustering, and the second is that we bring in an explicit penalty term for inter-cluster distance.

To mimic one level of hierarchical clustering we represent the subset $\mathcal{S}_k$ as a union of Voronoi cells $\mathcal{S}_{k;l}$:

$$\mathcal{S}_k = \cup_l \mathcal{S}_{k;l}.$$

To penalize gaps between clusters we bring in terms of the form

$$-\|Y_{k;l} - Y_{p;i}\|^2.$$

Note the negative sign which encourages these distances to become larger during the optimization process.

The rest of the paper works through the details of a K-means style algorithm with guaranteed descent.

2 Notation

Let $\mathbb{R}$ denote the set of reals, $\mathbb{N}$ the set of non-negative integers, and $\mathbb{N}_+$ the set of positive integers. Let $\mathbb{N}_p = \{0, 1, ..., p-1\}$, for $p \in \mathbb{N}$, with $\mathbb{N}_0 = \{\}$.

Let $\|\cdot\|$ denote the standard Euclidean 2–norm. Let $\|\cdot\|_F$ denote the Frobenius norm.

Let e denote the vector of all ones; the dimension will be apparent from the context.

Breaking from custom, we will place row indices on the left. For example, ${}_iA_j$ will denote the (i, j)-th entry of the matrix A. We will also use A_j to denote the j-th column of A, while ${}_iA$ will denote the i-th row of A. We will use a double index notation for block matrices. So ${}_{p;}A_{k;}$ will denote the (p, k)-th block sub-matrix of A, and ${}_{p;i}A_{k;j}$ will denote the (i, j)-th entry of the block ${}_{p;}A_{k;}$. Frequently our row and column indices will start with 0 rather than 1.

Let $N, L, K \in \mathbb{N}_+$. Let $Y \in \mathbb{R}^{N \times L}$. Block partition the columns of Y into K blocks:

$$Y = (\ Y_{0;}\ \ Y_{1;}\ \ \cdots\ \ Y_{K-1;}\ ).$$

Let $\lambda \in \mathbb{N}_+^K$ for $K \in \mathbb{N}_+$, and let $Y_{k;} \in \mathbb{R}^{N \times \lambda_k}$ for $k \in \mathbb{N}_K$; that is λ_k denotes the number of columns in $Y_{k;}$ and

$$\sum_{k \in \mathbb{N}_K} \lambda_k = L.$$

Using Y and λ, partition $\mathbb{R}^N$ into K mutually disjoint subsets $\mathcal{S}_k$ according to the following membership rule: $x \in \mathbb{R}^N$ is assigned to $\mathcal{S}_k$ if k is the smallest integer for which

$$\min_{l \in \mathbb{N}_{\lambda_k}} \|x - Y_{k;l}\| = \min_{p \in \mathbb{N}_K} \min_{j \in \mathbb{N}_{\lambda_p}} \|x - Y_{p;j}\|.$$

Let $\mathcal{S}_{k;l}$, for $l \in \mathbb{N}_{\lambda_k}$, denote λ_k mutually disjoint subsets of $\mathcal{S}_k$. The membership rule for $\mathcal{S}_{k;l}$ is as follows: $x \in \mathcal{S}_k$ is assigned to $\mathcal{S}_{k;l}$ if l is the smallest integer for which

$$\|x - Y_{k;l}\| = \min_{n \in \mathbb{N}_{\lambda_k}} \|x - Y_{k;n}\|.$$

We will call $\mathcal{S}_{k;l}$ as a sub-cluster and $\mathcal{S}_k$ as a cluster.

Let $X_{k;l}$ denote the sub-matrix of X that contains all the columns of X that lie in $\mathcal{S}_{k;l}$.

3 Problem

Let $M, N \in \mathbb{N}_+$. Let $1 < L_1 \in \mathbb{N}_+$. Let $X \in \mathbb{R}^{N \times M}$ and $\Omega \in \mathbb{R}^N$ be given. Let $\alpha, \beta \geqslant 0$ and $\varsigma, \gamma > 0$ be given. Let $Y \in \mathbb{R}^{N \times L}$ for some $0 < L \leqslant L_1$. Let $K \in \mathbb{N}_+$ and $\lambda \in \mathbb{N}_+^K$ such that $\sum_{k \in \mathbb{N}_K} \lambda_k = L$. Let

$$\begin{aligned} F(Y, \lambda) &= \sum_{k \in \mathbb{N}_K} \sum_{l \in \mathbb{N}_{\lambda_k}} \|X_{k;l} - Y_{k;l} e^T\|_F^2 \\ &+ \alpha \sum_{k \in \mathbb{N}_K} \sum_{l < n \in \mathbb{N}_{\lambda_k}} \|Y_{k;l} - Y_{k;n}\|^2 \\ &+ \frac{\beta}{\gamma} \sum_{k < p \in \mathbb{N}_K} \sum_{l \in \mathbb{N}_{\lambda_k}, j \in \mathbb{N}_{\lambda_p}} (1 - \gamma \|Y_{k;l} - Y_{p;j}\|^2) \\ &+ \varsigma \|Y - \Omega e^T\|_F^2. \end{aligned} \tag{1}$$

Given X, Ω, α, β, ς, γ, L_1, find L, Y, K and λ, which solves the minimization problem

$$\min_{Y,\lambda} F(Y,\lambda),$$

when

$$\beta < \frac{\varsigma}{2(L_1-1)}.$$

The global optimum is hard to find, so we settle for a "local" minima, though the word "local" is dubious in a discrete setting. The bound on β is needed to ensure that F is bounded from below. We recommend choosing γ to be reasonably small so as to discourage the formation of empty sub-clusters. (See Proposition 5.) A good default choice for Ω is the global mean

$$\Omega = \frac{Xe}{M}.$$

The role of α is to encourage sub-clusters belonging to a single cluster to be close together, while the role of β and γ is to encourage clusters to be well-separated. The role of ς is purely technical at this point; it keeps F bounded from below when some sub-clusters become empty.

There are several components to this problem and it is difficult to find a linear presentation. Assuming that the reader is familiar with the K-means algorithm we begin with a rough outline of the algorithm and then present the details. Our goal is a guaranteed descent algorithm to a local minimum.

4 The Algorithm

The algorithm is a form of block coordinate descent, complicated by the presence of a combinatorial part. The algorithm proceeds in multiple stages. In each stage we guarantee that F is non-increasing.

1. Initialize Y (essentially randomly from columns of X) with $K = L_1$. (Section 5.)
2. Assign columns of X by the nearest center rule. (Section 6.)
3. **Repeat**:
   - A) Compute C, T and R. (Section 7.)
   - B) **For each** column $Y_{k;l}$:
     - a) **If** sub-cluster $\mathcal{S}_{k;l}$ is empty delete if descent is possible. (Section 7.2.)
     - b) **Else** among the following choices, **pick** the **one** with maximum descent:
       - i. Split off sub-cluster into its own cluster if descent is possible. (Section 7.3.)
       - ii. Transfer sub-cluster to another cluster if descent is possible. (Section 7.4.)
       - iii. Swap with another sub-cluster if descent is possible. (Section 7.5.)
     - c) Update λ, T, R and other variables as needed. (Section 12.7.)
   - C) Freeze all partitions $\mathcal{S}_{k;l}$ and move Y to the nearest critical point. (Section 8.)
   - D) **For each** column X_j:
     - a) **If** $L < L_1$ and if $X_j = Y_{K;0}$ would lead to descent take this path. (Section 7.1.)
     - b) **Else** assign to nearest $Y_{k;l}$ (guarantee descent). (Section 6.)
     - c) **If** X_j changed membership, freeze all the partitions $\mathcal{S}_{k;l}$ and modify Y to reach nearest local minimum (guarantee descent). (Section 8.)
4. **Until** no (significant) descent

We point out a key difference with what most people call Lloyd's [15] or Forgy's [12] version of the K-means algorithm: we update the centers every time X_j is re-assigned. This second version of K-means is known to be more efficient in the Euclidean case [13, 14]. Furthermore, it guarantees (modulo floating-point errors) that no empty clusters will be produced by K-means, which is a frequent problem in Forgy's version.

We now present the details of the various steps. We also give some details about the local minima where the algorithm can stop. Some of the detailed algebraic calculations are relegated to the appendix.

Proposition 1. *The cost of one loop, steps B, C and D, is $O(NML + L^2)$ flops. Each step of the loop is guaranteed not to increase the objective function.*

Proof. Established in the propositions below. □

5 Initializing Y

We use standard random techniques. Let $Q \in \mathbb{N}$ be a positive integer.

1. Choose Q columns of X randomly. Find one with the largest separation and keep as the first column of Y.
2. $L_1 - 1$ times do the following:
   a) Choose Q columns randomly from X. Keep the one furthest from the current set of Y columns as the next column of Y. If the largest distance is zero, this step must be repeated until the largest distance becomes non-zero.

Choose $K = L_1$, so every column of Y corresponds to a cluster, and there are L_1 initial clusters. As long as there are enough distinct columns in X this process will terminate in a finite number of iterations and cost $O(Q^2 N + QNL_1^2)$ flops.

This is the costliest stage and to balance the cost we must pick Q such that $Q \ll \min(\sqrt{MIL_1}, MI/L_1)$ where I is the number of iterations that the algorithm will run, which is of course not available a priori.

6 Assigning X_j

Column X_j is assigned to $\mathcal{S}_{k;l}$ following the standard membership rule described in Section 2. The cost of assigning one X_j is $O(NL)$ flops. The objective function is guaranteed not to increase, and strict decrease is assured if X_j changes its membership.

The first time we are guaranteed that each $\mathcal{S}_{k;l} = \mathcal{S}_k$ will be non-empty. The remaining times this guarantee is not available.

The total cost of assigning all columns of X is $O(NML)$ flops.

7 Re-arranging sub-clusters

The symmetric two-dimensional array C will be used to hold the distances between the columns of Y. Let

$$\begin{aligned}
{}_{k;l}C_{k;l} &= \|Y_{k;l} - \Omega\|^2, && k \in \mathbb{N}_K, l \in \mathbb{N}_{\lambda_k},\\
{}_{k;l}C_{k;n} &= \|Y_{k;l} - Y_{k;n}\|^2, && l \neq n \in \mathbb{N}_{\lambda_k},\\
{}_{k;l}C_{p;i} &= \|Y_{k;l} - Y_{p;i}\|^2, && k \neq p \in \mathbb{N}_K, l \in \mathbb{N}_{\lambda_k}, i \in \mathbb{N}_{\lambda_p}.
\end{aligned}$$

The two-dimensional array T will be used to hold the distances between columns of Y and sub-clusters, while the one-dimensional array R will hold the distances from columns of Y to all sub-clusters that it is not a member of. Let

$$\begin{aligned}
{}_{k;l}T_k &= \sum_{l \neq n \in \mathbb{N}_{\lambda_k}} {}_{k;l}C_{k;n}, && k \in \mathbb{N}_K, l \in \mathbb{N}_{\lambda_k},\\
{}_{k;l}T_p &= \sum_{i \in \mathbb{N}_{\lambda_p}} {}_{k;l}C_{p;i}, && k \neq p \in \mathbb{N}_K, l \in \mathbb{N}_{\lambda_k},\\
R_{k;l} &= \sum_{k \neq p \in \mathbb{N}_K} {}_{k;l}T_p, && k \in \mathbb{N}_K, l \in \mathbb{N}_{\lambda_k}.
\end{aligned}$$

Note that $C \in \mathbb{R}^{L\times L}$, $T \in \mathbb{R}^{L\times K}$ and $R \in \mathbb{R}^{L}$.

Proposition 2. *C, T and R can be computed in $O(NL^2)$ flops.*

Proof. It is easy to see that C can be computed in $O(NL^2)$ flops.

${}_{k;l}T_k$ can be computed in $O(\lambda_k)$ flops. With some care ${}_{k;}T_k$ can be computed in $O(\lambda_k)$ flops too. ${}_{k;l}T_p$ can be computed in $O(\lambda_p)$ flops, and ${}_{k;}T_p$ can be computed in $O(\lambda_p \lambda_k)$ flops. Hence T can be computed in $O(L^2)$ flops.

$R_{k;l}$ can be computed in $O(K)$ flops, and $R_{k;}$ can be computed in $O(\lambda_k K)$ flops. Therefore R can be computed in $O(KL)$ flops, or, more generously, in $O(L^2)$ flops. □

Based on C, T and R, we re-arrange the sub-clusters, changing the membership of at most two columns of Y at a time, while still ensuring descent of F. In this section alone we will let F_1 denote the value of F before the intended operation, and let F_2 denote the value of F after the intended operation. The operation will cause a strict decrease in the value of F, if $F_1 - F_2 > 0$.

There are four possible operations we consider for each column $Y_{k;l}$, and there is a fifth one for introducing a new $Y_{k;l}$.

Note that we do not entertain operations that look at three columns of Y at the same time, since the step will become L times slower. However, in some situations it might be worthwhile to do so, especially if $L \sim N$.

7.1 Assigning X_j its own cluster

If $L < L_1$ there is room to create new clusters.

Proposition 3. *If $L < L_1$ and $X_j \in \mathcal{S}_{k;l}$, then introducing $Y_{K;0} = X_j$ will result in*

$$F_1 - F_2 = \|X_j - Y_{k;l}\|^2 + \beta\left(\|X_j e^T - Y\|_F^2 - \frac{L}{\gamma}\right) - \varsigma\,\|X_j - \Omega\|^2.$$

This also gives one way to interpret γ, and one of the effects of ς—it discourages the creation of clusters away from Ω. It also shows one of the significant differences with K-means, where new clusters can be easily introduced, and the number of clusters must be controlled explicitly. In our algorithm the constants indirectly influence the number of clusters and should produce a smoother way to tune the algorithm in practice.

Note that $Y_{k;l}$ occurs twice in the above expression.

Proof. From equation (1) we can compute

$$F_1 - F_2 \;=\; \|X_j - Y_{k;l}\|^2 - \varsigma\,\|X_j - \Omega\|^2 - \frac{\beta}{\gamma}(L - \gamma\|X_j e^T - Y\|_F^2).$$

□

7.2 Deleting an empty sub-cluster

It possible that after the columns of X have been re-assigned to the columns of Y, some subset $\mathcal{S}_{k;l}$ associated with $Y_{k;l}$ might be empty. In this case we give priority to deleting this sub-cluster if possible and decrement L by 1 if we succeed.

Proposition 4. *If $\mathcal{S}_{k;l} = \{\}$ then deleting $Y_{k;l}$ will result in*

$$F_1 - F_2 = \alpha\,{}_{k;l}T_k + \varsigma_{\,k;l}C_{k;l} + \frac{\beta}{\gamma}(L - \lambda_k) - \beta R_{k;l}. \tag{2}$$

Once C, T and R, are available, this can be computed in $O(1)$ flops.

Proof. From equation (6) we can compute

$$F_1 - F_2 = \alpha_{\,k;l} T_k + \frac{\beta}{\gamma}(L - \lambda_k - \gamma R_{k;l}) + \varsigma_{\,k;l} C_{k;l}.$$

□

Proposition 5. *If*

$$\gamma \leqslant \frac{\varsigma - 2\,\beta\,(L_1 - 1)}{2\left(\|X\|_2 \sqrt{M} + \varsigma\,\|\Omega\|_2 \sqrt{L_1}\right)}, \tag{3}$$

then expression (2) is always non-negative.

Proof. Follows from Proposition 15. □

Thus with sufficiently small choice of γ the algorithm will always delete empty sub-clusters. We do not recommend setting γ this small as the upper bound is extremely loose and would not enforce large gaps between clusters.

If $Y_{k;l}$ is deleted then we need to update C, T and R efficiently. See Proposition 23.

7.3 Splitting off a sub-cluster

We now develop a criteria to check if $Y_{k;l}$ should be split off into its own cluster in case $\lambda_k > 1$, and if K should be increased by one.

Proposition 6. *If $Y_{k;l}$ is split off into its own cluster*

$$F_1 - F_2 = (\alpha + \beta)_{\,k;l} T_k - \frac{\beta}{\gamma}(\lambda_k - 1).$$

This can be computed in $O(1)$ flops once T and λ are available.

Proof. We can compute from equation (6)

$$\begin{aligned} F_1 - F_2 &= \alpha_{\,k;l} T_k + \frac{\beta}{\gamma}(L - \lambda_k - \gamma R_{k;l}) + \varsigma_{\,k;l} C_{k;l} \\ &- \frac{\beta}{\gamma}(L - \lambda_k - \gamma R_{k;l}) - \varsigma_{\,k;l} C_{k;l} \\ &- \frac{\beta}{\gamma}(\lambda_k - 1 - \gamma_{\,k;l} T_k) \\ &= (\alpha + \beta)_{\,k;l} T_k - \frac{\beta}{\gamma}(\lambda_k - 1). \end{aligned}$$

□

We note that large values for γ will encourage sub-clusters to split off. This is another reason to keep γ reasonably small. This also gives a good thumb rule for tuning γ on synthetic data sets.

If $Y_{k;l}$ is split off then C, T and R have to be updated efficiently. See Proposition 24.

7.4 Transferring a sub-cluster to another cluster

Proposition 7. *If $Y_{k;l}$ is transferred from cluster k to cluster p then*

$$F_1 - F_2 = (\alpha + \beta)\,({}_{k;l} T_k - {}_{k;l} T_p) - \frac{\beta}{\gamma}(\lambda_k - \lambda_p - 1),$$

and this can be computed in $O(1)$ flops once T and λ are available.

Proof. We compute from equation (6)

$$\begin{aligned} F_1 - F_2 &= \alpha_{\,k;l} T_k + \varsigma_{\,k;l} C_{k;l} \\ &+ \frac{\beta}{\gamma}(\lambda_p - \gamma_{\,k;l} T_p) + \frac{\beta}{\gamma}(L - \lambda_p - \lambda_k - \gamma\,(R_{k;l} - {}_{k;l}T_p)) \\ &- \alpha_{\,k;l} T_p - \varsigma_{\,k;l} C_{k;l} \\ &- \frac{\beta}{\gamma}(\lambda_k - 1 - \gamma_{\,k;l} T_k) - \frac{\beta}{\gamma}(L - \lambda_p - \lambda_k - \gamma\,(R_{k;l} - {}_{k;l}T_p)) \\ &= (\alpha+\beta)_{\,k;l} T_k - (\alpha+\beta)_{\,k;l} T_p + \frac{\beta}{\gamma}(\lambda_p - \lambda_k + 1). \end{aligned}$$

□

If $Y_{k;l}$ is tranferred to cluster p, then T and R have to be updated efficiently. See Proposition 25. Note that if γ is not sufficiently large, huge clusters will gobble up small clusters.

7.5 Swapping two sub-clusters

Proposition 8. *If $Y_{k;l}$ is swapped with $Y_{p;i}$ then*

$$F_1 - F_2 = (\alpha+\beta)\,({}_{k;l}T_k + {}_{p;i}T_p - {}_{k;l}T_p - {}_{p;i}T_k + 2\,{}_{k;l}C_{p;i}).$$

Proof. From equation (6) we can compute

$$\begin{aligned} F_1 - F_2 &= \alpha_{\,k;l} T_k + \frac{\beta}{\gamma}(\lambda_p - \gamma_{\,k;l} T_p) \\ &+ \alpha_{\,p;i} T_p + \frac{\beta}{\gamma}(\lambda_k - \gamma_{\,p;i} T_k) \\ &- \alpha\,({}_{p;i}T_k - {}_{p;i}C_{k;l}) - \alpha\,({}_{k;l}T_p - {}_{k;l}C_{p;i}) \\ &- \frac{\beta}{\gamma}(\lambda_p - \gamma_{\,p;i} T_p - \gamma_{\,p;i} C_{k;l}) - \frac{\beta}{\gamma}(\lambda_k - \gamma_{\,k;l} T_k - \gamma_{\,k;l} C_{p;i}) \\ &= (\alpha+\beta)_{k;l} T_k + (\alpha+\beta)_{p;i} T_p \\ &- (\alpha+\beta)_{k;l} T_p - (\alpha+\beta)_{p;i} T_k \\ &+ {}_{p;i}C_{k;l}(\alpha+\beta) + {}_{k;l}C_{p;i}(\alpha+\beta). \end{aligned}$$

□

If the swap is carried out T and R must be updated. See Proposition 26.

8 Descending Y

Suppose we freeze the membership of the columns of X in the subsets $\mathcal{S}_{k;l}$. What is the optimal choice for Y? In the K-means case the optimal choice is clearly the mean of each cluster. In our case the answer is only a little more complicated.

Define the matrix $A \in \mathbb{R}^{L \times L}$ as follows:

$$\begin{aligned} {}_{k;l}A_{k;l} &= |\mathcal{S}_{k;l}| + \alpha\,(\lambda_k - 1) + \varsigma - \beta\,(L - \lambda_k), && (4) \\ {}_{k;l}A_{k;n} &= -\alpha, && l \neq n \in \mathbb{N}_{\lambda_k}, \\ {}_{k;l}A_{p;i} &= +\beta, && k \neq p \in \mathbb{N}_K, i \in \mathbb{N}_{\lambda_p}, \end{aligned}$$

where $|\mathcal{S}|$ denotes the cardinality of the set $\mathcal{S}$. Define the matrix $W \in \mathbb{R}^{N \times L}$ as follows:

$$W_{k;l} = X_{k;l}\, e.$$

We assume that $W_{k;l} = 0$ if $\mathcal{S}_{k;l} = \{\}$.

Proposition 9. *For a fixed set of $\mathcal{S}_{k;l}$ and a fixed λ, there is exactly one minimum point:*

$$Y = (W + \varsigma\,\Omega e^T)\,A^{-1}.$$

Proof. See proof of Proposition 16 and the discussion leading up to it. □

Proposition 10. *For a fixed set of $\mathcal{S}_{k;l}$ and a fixed λ, the unique critical point Y can be computed in $O(NL)$ flops.*

Proof. See Proposition 22 and the argument leading to it. □

Note that this cost is comparable to the cost of assigning one column X_j to its optimal sub-cluster. So re-computing Y for every such assignment only affects the constant in the flop count. However, the reason for doing so, is similar to that of the non-standard K-means algorithm: it reduces the chance of producing empty sub-clusters.

9 Numerical experiments

Tests were carried out on a bunch of random synthetic data that were generated as follows. Let $M_1 \in \mathbb{N}_+$. Let $T_{k;} \in \mathbb{R}^{N \times M_1}$ be a random matrix with entries chosen uniformly from $[-1, 1]$. Let $S \in \mathbb{R}^{N \times K}$ be a random matrix with entries chosen uniformly from $[-1, 1]$. Let $D \in \mathbb{R}^{K \times K}$ be a diagonal matrix with entries chosen uniformly from $[-\nu, \nu]$ for $\nu \in \mathbb{R}$ and fixed. Let $X_{k;} = T_{k;} + {}_kD_k\, S_k\, e^T$.

Let $C_{k;}$ represent a clustering of the columns of X into K_1 clusters. Let ${}_k\mathfrak{C}_l$ denote the number of columns of $X_{l;}$ that are in $C_{k;}$. Let

$$\mathfrak{K}_l = \text{argmax}_{k \in \mathbb{N}_{K_1}}\, {}_k\mathfrak{C}_l,$$

and

$${}_k\mathfrak{L} = \text{argmax}_{l \in \mathbb{N}_K}\, {}_k\mathfrak{C}_l.$$

Let

$$\mathbb{S}(C) = \sum_{l \in \mathbb{N}_K} \sum_{k \in \mathbb{N}_{K_1}, k \neq \mathfrak{K}_l} {}_k\mathfrak{C}_l + \sum_{k \in \mathbb{N}_{K_1}} \sum_{l \in \mathbb{N}_K, l \neq \mathfrak{L}_k} {}_k\mathfrak{C}_l.$$

($\mathbb{S}(C)$ is the sum of false positives and negatives when those concepts make sense.) We can simplify this

$$\begin{aligned} \mathbb{S}(C) &= \sum_{l \in \mathbb{N}_K} (M_1 - {}_{\mathfrak{K}(l)}\mathfrak{C}_l) + \sum_{k \in \mathbb{N}_{K_1}} (|C_{k;}| - {}_k\mathfrak{C}_{\mathfrak{L}(k)}) \\ &= M - \sum_{l \in \mathbb{N}_K} {}_{\mathfrak{K}(l)}\mathfrak{C}_l + M - \sum_{k \in \mathbb{N}_{K_1}} {}_k\mathfrak{C}_{\mathfrak{L}(k)} \\ &= 2M - \sum_{l \in \mathbb{N}_K} {}_{\mathfrak{K}(l)}\mathfrak{C}_l - \sum_{k \in \mathbb{N}_{K_1}} {}_k\mathfrak{C}_{\mathfrak{L}(k)}. \end{aligned}$$

Using the nearest center rule we can partition the columns of X into K clusters $\Xi_{k;}$ using the K columns of SD. Note that these may not be the same as $X_{k;}$. Let Y and Λ be computed by the new method, and let the resulting clustering of the columns of X into K_1 clusters be denoted by $Z_{k;}$. We will measure the goodness of Y and Λ by the number

$$\text{score IMP} = \mathbb{S}(\Xi) - \mathbb{S}(Z).$$

This number lies in the range $[-2M, 2M]$ and bigger numbers are taken to indicate that the clustering Z was good in some sense. Note that one advantage of this measure is that it does not benefit lumping all of X into a single cluster, or, breaking it all up into M clusters, since every clustering is compared to the putative right clustering $X_{k;}$. One disadvantage is that sometimes $X_{k;}$ is not the right clustering and our method can be penalized for finding it.

Note that a score of 0 should be considered as excellent for this data set.

We will also compare against a K-means algorithm. Our implementation uses the same initialization routine as `IMP` except with a much larger value of Q. It also updates the cluster centers every time a column of X is re-assigned. We did this so as to avoid empty clusters. The K-means algorithm was provided with the correct value of K and ran it until it reached a local minimum. In fact we did this for both algorithms. `IMP` was provided with a starting value of $L_1 = 2K$ and left to work out the true number of clusters. Both algorithms were run until they reached their local minimum and no attempt was made at early termination.

We chose some of the parameters as follows:

$$\begin{aligned}
L_1 &= 2K, \\
M &= \begin{cases} KM_1, & \text{no outliers,} \\ (K+1)M_1, & \text{with } M_1 \text{ outliers,} \end{cases} \\
\varsigma &= \frac{M_1}{2000}, \\
\beta &= \frac{\varsigma}{2.00001\,(L_1-1)}, \\
\alpha &= 2\,(L_1-1)\,\beta, \\
Q \text{ (for } K\text{-means)} &= K, \\
Q \text{ (for } \texttt{fIMP}) &= \begin{cases} 30, & \text{if } K=100, \\ 40, & \text{if } K=200, \end{cases} \\
\nu &= 10\,K^{1/N}\log_2(K), \\
\text{Number of trials} &= \begin{cases} 100, & \text{if } K=100, \\ 50, & \text{if } K=200. \end{cases}
\end{aligned}$$

The experimental results for $K=100$ are summarized in Table 1 and for $K=200$ in Table 2.

N	M_1	$\gamma \times 10^4$	Losses$_1$	fIMP score	Losses$_2$	K-means score	Time fIMP	Time K-means
7	30	4	42	−6	8	−31	0.60	0.04
14	30	4	53	−6	14	−28	1.08	0.07
28	60	4	47	−8	11	−58	4.03	0.26
56	120	3	26	−15	1	−129	17.32	1.16
112	240	3	23	−14	6	−237	80.81	5.35

Table 1. Experimental results for $K=100$. The column Losses$_1$ reports the number of times the fIMP score was strictly negative out of 100 trials. The column "fIMP score" reports the average score for fIMP across 100 trials. The columns Losses$_2$ reports the number of times the score for fIMP was strictly worse than the score for K-means out of 100 trials. The column "K-means score" reports the average score for K-means. The column "Time fIMP" reports the average running time for fIMP in seconds. The column "Time K-means" reports the average running time for K-means in seconds.

N	M_1	$\gamma \times 10^4$	Losses$_1$	fIMP score	Losses$_2$	K-means score	Time fIMP	Time K-means
7	30	2.5	36	−8	9	−39	4.60	0.23
14	30	2	40	−14	9	−48	8.12	0.39
28	60	2	33	−16	4	−83	29.60	1.23
56	120	2	33	−37	8	−158	138.91	5.60
112	240	0.8	16	−73	3	−296	665.94	23.98

Table 2. Experimental results for $K=200$. The column Losses$_1$ reports the number of times the fIMP score was strictly negative out of 50 trials. The column "fIMP score" reports the average score for fIMP across 50 trials. The columns Losses$_2$ reports the number of times the score for fIMP was strictly worse than the score for K-means out of 50 trials. The column "K-means score" reports the average score for K-means. The column "Time fIMP" reports the average running time for fIMP in seconds. The column "Time K-means" reports the average running time for K-means in seconds. Note that compared to Table 1 *both* M and K are doubled in corresponding rows.

For this specific synthetic data set and initialization strategy we conjecture that fIMP takes $O(NMK^2)$ flops to find a local minima while K-means takes $O(NMK)$ flops. We conjecture that on average fIMP comes close to the global minimum while K-means is off by about one cluster. It is crucial to note that this synthetic data set has no outliers, the clusters tend to collide more frequently near the origin, and the clusters are convex (cubical) in shape. So these conjectures are only in this very restrictive setting.

The next set of experiments was essentially a repeat of the previous set with outliers thrown in. In particular for every run we added M_1 points distributed randomly in $[-\nu, \nu]^N$. The scoring however was restricted to the non-outliers and there was no penalty for empty clusters. To enable K-means to do well in this situation we seeded it with $K+M_1$ centers. For fIMP we chose $L_1=2K$ as usual. The results for $K=100$ are shown in Table 3 and for $K=200$ shown in Table 4.

N	M_1	$\gamma \times 10^4$	Losses$_1$	fIMP score	Losses$_2$	K-means score	Time fIMP	Time K-means
7	30	4	33	−6	12	−24	0.45	0.06
14	30	4	54	−9	12	−27	0.80	0.10
28	60	4	32	−8	0	−86	2.33	0.57
56	120	3	21	−15	0	−400	9.71	4.00
112	240	3	14	−26	0	−1428	46.36	32.22

Table 3. Experimental results for $K = 100$ with M_1 outliers. The column Losses$_1$ reports the number of times the fIMP score was strictly negative out of 100 trials. The column "fIMP score" reports the average score for fIMP across 100 trials. The columns Losses$_2$ reports the number of times the score for fIMP was strictly worse than the score for K-means out of 100 trials. The column "K-means score" reports the average score for K-means. The column "Time fIMP" reports the average running time for fIMP in seconds. The column "Time K-means" reports the average running time for K-means in seconds.

N	M_1	$\gamma \times 10^4$	Losses$_1$	fIMP score	Losses$_2$	K-means score	Time fIMP	Time K-means
7	30	2	31	−14	4	−44	3.98	0.29
14	30	2	39	−10	11	−33	6.99	0.45
28	60	2	34	−21	2	−90	22.13	2.00
56	120	2	25	−27	0	−337	96.93	11.89
112	240	1	19	−71	0	−1602	500.21	84.31

Table 4. Experimental results for $K = 200$ with M_1 outliers. The column Losses$_1$ reports the number of times the fIMP score was strictly negative out of 50 trials. The column "fIMP score" reports the average score for fIMP across 50 trials. The columns Losses$_2$ reports the number of times the score for fIMP was strictly worse than the score for K-means out of 50 trials. The column "K-means score" reports the average score for K-means. The column "Time fIMP" reports the average running time for fIMP in seconds. The column "Time K-means" reports the average running time for K-means in seconds. Note that compared to Table 1 *both* M and K are doubled in corresponding rows.

10 Existing work

For a summary of early literature see [6, 8].

It is known that when there really are K clusters and enough effort is expended then, in some cases, K-means will converge quickly to the right solution [1, 11]. On the other hand it is known that for ill-fated configurations K-means can take a long time to converge [2].

Work has also been done on heuristic starting methods, for example [3], which also estimate K.

The work of Lindsten, Ohlsson and Lung [4, 5], replaces the K-means objective function with a convex objective function with one continuous regularization parameter that replaces the discrete parameter K. This objective function bears some resemblance to ours. However, we note that their objective function has sums of norms, and not their squares, and so is more expensive to optimize and requires traditional interior point methods. Furthermore, there are no negative terms in their objective function and hence no explicit penalty for inter-cluster distances. Finally every cluster is represented by a single center as in the classical K-means method and the algorithm starts with M (versus L_1) cluster centers.

Another modified K-means objective function is that of [17]. However, this does not include negative terms either.

Hierarchical clustering algorithms are asymptotically slower than K-means but there has been work on making them faster, for example [7, 9].

Our algorithm leans heavily on the standard Euclidean norm, but other measures of similarity can be important in practice [10].

Multi-point representations of clusters is implicit in the work of Rose *et. al.* on deterministic annealing [16].

Multi-point representations are also used in learning vector quantization [18].

1. Rafail Ostrovsky, Yuval Rabani, Leonard J.Schulman, Chaitanya Swamy, "*The Effectiveness of Lloyd-Type Methods for the k-Means Problem*," 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), 2006.

2. Andrea Vattani, "*K-means requires exponentially many iterations even in the plane*," DCG, 2011.

3. A.McCallum, K.Nigam and L.H.Ungar, "Efficient Clustering of High-Dimensional Data Set with Application to Reference Matching," Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 169-178, 2000.
4. Henrik Ohlsson and Lennart Ljung, "A Convex Approach to Subspace Clustering," 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC) Orlando, FL, USA, December 12-15, 2011.
5. Fredrik Lindsten, Henrik Ohlsson and Lennart Ljung, "Just Relax and Come Clustering! A Convexification of k-Means Clustering," Technical report from Automatic Control at Linköpings universitet, Report no.: LiTH-ISY-R-2992, 2011.
6. Rui Xu and Donald Wunsch II, "Survey of Clustering Algorithms," IEEE Transactions on Neural Networks, vol. 16, no. 3, May 2005.
7. David Eppstein, "Fast hierarchical clustering and other applications of dynamic closest pairs," Journal of Experimental Algorithmics, volume 5, article no. 1, 2000.
8. P.Berkhin, "A Survey of Clustering Data Mining Techniques," in Jacob Kogan, Charles Nicholas and Marc Teboulle, editors, "*Grouping Multidimensional Data*," Springer, 2006.
9. J.A.Aslam, E.Pelekhov and D.Rus, "The Star Clustering Algorithm for Information Organization," in Jacob Kogan, Charles Nicholas and Marc Teboulle, editors, "*Grouping Multidimensional Data*," Springer, 2006.
10. M.Teboulle, P.Berkhin, I.Dhillon, Y.Guan and J.Kogan, "Clustering with Entropy-Like k-Means Algorithms," in Jacob Kogan, Charles Nicholas and Marc Teboulle, editors, "*Grouping Multidimensional Data*," Springer, 2006.
11. Z.Volkovich, J.Kogan and C.Nicholas. "Sampling Methods for Building Initial Partitions," in Jacob Kogan, Charles Nicholas and Marc Teboulle, editors, "*Grouping Multidimensional Data*," Springer, 2006.
12. E.Forgy, "Cluster analysis of multivariate data: efficiency vs. interpretability of classifications," Biometrics, 21(3):768, 1965.
13. B.Larsen and C.Aone, "Fast and effective text mining using linear-time document clustering," in Proceedings of the 5th ACM SIGKDD, pages 16–22, San Diego, CA, USA, 1999.
14. M.Steinbach, G.Karypis, and V.Kumar, "A comparison of document clustering techniques," in Proceedings of the 6th ACM SIGKDD, World Text Mining Conference, Boston, MA, USA, 2000.
15. Stuart P.Lloyd, "Least squares quantization in PCM," IEEE Transactions on Information Theory, 28(2):129–137, 1982.
16. K.Rose, E.Gurewitz and G.C.Fox, "A deterministic annealing approach to clustering," Pattern Recognition Letters, 11:589–594, 1990.
17. J.L.Marroquin and F.Girosi, "Some extensions of the k-means algorithm for image segmentation and pattern classification," Technical Report A.I. Memo 1390, MIT Press, Cambridge, MA, USA, 1993.
18. Teuvo Kohonen, "Self-Organization and Associative Memory," Springer series in Information Sciences, 1989, Springer.

11 Future work

We have presented a single family of new objective functions for clustering. The advantage of this family is that it retains the efficient time complexity of K-means while allowing a different set of local minima that can be tuned via a few parameters.

However, there is a much larger family of objective functions that can be explored. For example, we could consider other power laws on the distance, and we can allow many more constants.

Our objective function can be viewed as an average linkage 2-level hierarchical clustering with gap penalties scheme. The linkage can be viewed as a complete graph with cluster centers as vertices. A very useful model would be to replace this graph with a spanning tree that is chosen dynamically to allow non-spherical clusters and decrease the objective function (single linkage with gap penalties). However the details for the corresponding fast solver becomes more complicated, so this will be addressed in a separate paper.

Our objective function corresponds to a spring–mass model, where some of the springs can be viewed either as having a negative spring constant, or, as being wrapped around through the point at ∞. Based on this physical model, one can see that we can consistently develop other objective functions, for example using spring–mass–charge models (the exponents will no longer just be +2). What these model will loose is the unique local minima for a fixed partition, but fast gradient descent will still be possible. The details will be presented elsewhere.

It also would be nice to develop simple statistical models that can guide the user in the choice of the objective functions.

Some of the more complicated objective functions require more sophisticated fast solvers to locate the best Y. So this consideration also influences the choice of the objective function.

Our algorithm requires $O(L^2)$ working space memory. One can implement an algorithm that requires less working space memory but more flops.

Our algorithm has an $O(L^2)$ combinatorial part in each iteration. This makes it non-scalable with respect to the number of clusters. We can make it $O(L)$ by only considering $O(1)$ random cluster centers during the combinatorial phase. The details will be presented elsewhere.

Applying deterministic annealing to these new objective functions will also be interesting.

12 Appendix

12.1 Additional notation

Let

$$A \otimes B = \begin{pmatrix} {}_1A_1\,B & {}_1A_2\,B & \cdots \\ {}_2A_1\,B & {}_2A_2\,B & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix},$$

where A and B are two matrices. Let

$$\mathrm{vec}(A) = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \end{pmatrix}.$$

Let

$$\mathrm{tr}(A) = \sum_{i=1}^{n} {}_iA_i,$$

where $A \in \mathbb{R}^{n \times n}$.

Let $Y_{-k;l}$ denote the sub-matrix of $Y_{k;}$ obtained by dropping $Y_{k;l}$ from $Y_{k;}$. Let $Y_{-k;}$ denote the sub-matrix of Y obtained by dropping the sub-matrix $Y_{k;}$ from Y.

12.2 Sums of square distances

Note that

$$\begin{aligned} \|A - ye^T\|_F^2 &= \mathrm{tr}((A - ye^T)^T(A - ye^T)) \\ &= \mathrm{tr}((A^T - ey^T)(A - ye^T)) \\ &= \mathrm{tr}(A^TA - A^Tye^T - ey^TA + y^Tyee^T) \\ &= \mathrm{tr}(A^TA) - 2\,y^TAe + y^Tye^Te \\ &= \|A\|_F^2 - 2\,y^TAe + \|y\|^2\,n, \end{aligned} \tag{5}$$

where n is the number of columns in A.

12.3 Re-writing F

All the terms in F that depend on the single column $Y_{k;l}$:

$$\begin{aligned} F(Y,\lambda) &= \|Y_{k;l}\,e^T - X_{k;l}\|_F^2 + \alpha\,\|Y_{k;l}\,e^T - Y_{-k;l}\|_F^2 \\ &+ \frac{\beta}{\gamma}\,(L - \lambda_k - \gamma\,\|Y_{k;l}\,e^T - Y_{-k}\|_F^2) \\ &+ \varsigma\,\|Y_{k;l} - \Omega\|^2 + \text{terms independent of } Y_{k;l}. \end{aligned} \tag{6}$$

We can use identity (5) and expand it as:

$$\begin{aligned} F(Y,\lambda) &= \|X_{k;l}\|_F^2 - 2Y_{k;l}^T\,X_{k;l}\,e + \|Y_{k;l}\|^2\,|\mathcal{S}_{k;l}| \\ &+ \alpha\,\|Y_{-k;l}\|_F^2 - 2\alpha\,Y_{k;l}^T\,Y_{-k;l}\,e + \alpha\,\|Y_{k;l}\|_2^2\,(\lambda_k - 1) \\ &+ \frac{\beta}{\gamma}\,(L - \lambda_k - \gamma\,\|Y_{-k}\|_F^2 + 2\,\gamma\,Y_{k;l}^T\,Y_{-k}\,e - \gamma\,\|Y_{k;l}\|^2\,(L - \lambda_k)) \\ &+ \varsigma\,\|\Omega\|^2 - 2\varsigma\,Y_{k;l}^T\,\Omega + \varsigma\,\|Y_{k;l}\|^2 \\ &+ \text{terms independent of } Y_{k;l}. \end{aligned}$$

Gathering terms we get

$$\begin{aligned} F(Y,\lambda) &= \|X_{k;l}\|_F^2 + \alpha\,\|Y_{-k;l}\|_F^2 + \frac{\beta}{\gamma}\,(L - \lambda_k - \gamma\,\|Y_{-k}\|_F^2) + \varsigma\,\|\Omega\|^2 \\ &- 2Y_{k;l}^T\,(X_{k;l}\,e + \alpha\,Y_{-k;l}\,e + \varsigma\,\Omega - \beta\,Y_{-k}\,e) \\ &+ \|Y_{k;l}\|^2\,(|\mathcal{S}_{k;l}| + \alpha\,(\lambda_k - 1) + \varsigma - \beta\,(L - \lambda_k)) \\ &+ \text{terms independent of } Y_{k;l}. \end{aligned} \tag{7}$$

12.4 Gradient of F

From equation (7) we can compute the gradient:

$$\begin{aligned} \frac{1}{2}\,\frac{\partial F(Y,\lambda)}{\partial Y_{k;l}} &= Y_{k;l}\,(|\mathcal{S}_{k;l}| + \alpha\,(\lambda_k - 1) + \varsigma - \beta\,(L - \lambda_k)) \\ &- (X_{k;l}\,e + \alpha\,Y_{-k;l}\,e + \varsigma\,\Omega - \beta\,Y_{-k}\,e). \end{aligned} \tag{8}$$

Note that this is the $(k;l)$-th block component of the gradient viewed as a column vector.

It is useful to re-write this in a simpler form. First we observe that A is symmetric.

Proposition 11. *$A = A^T$.*

Proof. Obvious. □

Now we can write the gradient of F as

$$\frac{1}{2}\,\frac{\partial F(Y,\lambda)}{\partial Y} = YA - (W + \varsigma\,\Omega\,e^T), \tag{9}$$

where the gradient is now written as a matrix for convenience. In standard column form we can write

$$\frac{1}{2}\,\frac{\partial F(Y,\lambda)}{\partial Y} = (A \otimes I)\,\mathrm{vec}(Y) - \mathrm{vec}(W) - \varsigma e \otimes \Omega, \tag{10}$$

where we used the fact that A is symmetric.

Proposition 12. *A is strictly diagonally dominant and positive definite and*

$$\|A^{-1}\|_2 < \frac{1}{\varsigma - 2\,\beta\,(L-1)},$$

when $L>1$.

Proof. We claim the diagonal entries are positive:

$$|\mathcal{S}_{k;l}|+\alpha\,(\lambda_k{-}1)+\varsigma-\beta\,(L-\lambda_k)>0,$$

since

$$\begin{aligned}|\mathcal{S}_{k;l}|+\alpha\,(\lambda_k{-}1)+\varsigma-\beta\,(L-\lambda_k) &\geqslant\\ \varsigma+\lambda_k(\alpha+\beta)-\alpha-L\,\beta &\geqslant\\ \varsigma+\alpha+\beta-\alpha-L\,\beta &\geqslant\\ \varsigma-(L-1)\,\beta &> 0,\end{aligned}$$

since by assumption

$$\lambda_k{\geqslant}1, \qquad\qquad \beta<\frac{\varsigma}{2(L_1-1)}<\frac{\varsigma}{L-1}.$$

The sum of the absolute values of the entries in row $(k;l)$ is given by

$$\alpha\,(\lambda_k{-}1)+\beta\,(L-\lambda_k).$$

We claim that this sum is strictly smaller than the corresponding diagonal term since

$$\begin{aligned}|\mathcal{S}_{k;l}|+\alpha\,(\lambda_k{-}1)+\varsigma-\beta\,(L-\lambda_k)-\alpha\,(\lambda_k{-}1)-\beta\,(L-\lambda_k) &=\\ |\mathcal{S}_{k;l}|+\varsigma-2\,\beta\,(L-\lambda_k) &\geqslant\\ \varsigma-2\,\beta\,(L-1) &> 0,\end{aligned}$$

since by assumption

$$\beta<\frac{\varsigma}{2(L_1-1)}\leqslant\frac{\varsigma}{2(L-1)}.$$

Applying Gerschgorin's theorem we also obtain

$$\lambda_{\min}(A)\geqslant\varsigma-2\,\beta\,(L-1)>0,$$

from which we obtain the desired upper bound on the 2-norm of A^{-1}. □

Proposition 13.

$$\|W\|_2\leqslant\|X\|_2\sqrt{M}.$$

Proof. From

$$W_{k;l}=X_{k;l}\,e$$

we obtain

$$\|W_{k;l}\|_2\leqslant\|X\|_2\sqrt{|\mathcal{S}_{k;l}|}.$$

Therefore

$$\|W\|_2\leqslant\|W\|_F\leqslant\|X\|_2\sqrt{M}.$$ □

Proposition 14. *All critical points of* F *are uniformly bounded.*

Proof. $L>1$ is the non-trivial case. From Proposition 12, it follows that the critical points that are solutions of the equation

$$\frac{\partial F(Y,\lambda)}{\partial Y}=YA-(W+\varsigma\Omega e^T)=0,$$

satisfy the bound

$$\|Y\|=\|(W+\varsigma\Omega e^T)\,A^{-1}\|\leqslant\|W+\varsigma\Omega e^T\|\,\|A^{-1}\|\leqslant\frac{\|W\|+\varsigma\,\|\Omega\|\,\|e\|}{\varsigma-2\,\beta\,(L-1)}<\infty,$$

for any sub-multiplicative norm.

Since F is differentiable and bounded from below, there are no other critical points to consider.

Using Proposition 13 we also have the explicit uniform upper bound

$$\|Y\|_2 \leqslant \frac{\|X\|_2 \sqrt{M} + \varsigma \|\Omega\|_2 \sqrt{L}}{\varsigma - 2\beta(L-1)} \leqslant \frac{\|X\|_2 \sqrt{M} + \varsigma \|\Omega\|_2 \sqrt{L_1}}{\varsigma - 2\beta(L_1-1)}. \qquad \square$$

Proposition 15.

$$R_{k;l} \leqslant \frac{2L_1\left(\|X\|_2 \sqrt{M} + \varsigma \|\Omega\|_2 \sqrt{L_1}\right)}{\varsigma - 2\beta(L_1-1)}.$$

Proof. Follows from the previous proposition and the easily established upper bound

$$R_{k;l} \leqslant 2L_1 \|Y\|_2. \qquad \square$$

12.5 Hessian of F

From equation (8) we can compute the Hessian of F, denoted as $2H$:

$$\begin{aligned}
{}_{k;l}H_{k;l} = \frac{1}{2}\frac{\partial^2 F(Y,\lambda)}{\partial^2 Y_{k;l}} &= (|\mathcal{S}_{k;l}| + \alpha(\lambda_k - 1) + \varsigma - \beta(L - \lambda_k))\, I, \\
{}_{k;l}H_{k;n} = \frac{1}{2}\frac{\partial^2 F(Y,\lambda)}{\partial Y_{k;l}\partial Y_{k;n}} &= -\alpha I, \\
{}_{k;l}H_{p;i} = \frac{1}{2}\frac{\partial^2 F(Y,\lambda)}{\partial Y_{k;l}\partial Y_{p;i}} &= \beta I.
\end{aligned}$$

We can represent the Hessian in matrix form as

$$H = A \otimes I,$$

which also follows from equation (10).

Proposition 16. *All critical points of F are of the form $Y = (W + \varsigma\Omega e^T)\, A^{-1}$ and correspond to local minima of F.*

Proof. Follows from the positive-definiteness of the Hessian H. $\square$

Note that the formula for the critical point is a bit deceptive in appearance. For example, there is more than one critical point, since the choice of λ and $\mathcal{S}_{k;l}$ determine W and A.

12.6 Rapid application of A^{-1}

We will depend on the Sherman–Morrison–Woodbury (SMW) formula

$$(I + UV^T)^{-1} = (I - U(I + V^T U)^{-1} V^T). \tag{11}$$

Let D denote the diagonal matrix

$${}_{k;l}D_{k;l} = |\mathcal{S}_{k;l}| + \alpha\lambda_k + \varsigma - \beta(L - \lambda_k).$$

All ${}_{k;l}D_{k;l}$ can be computed in $O(L)$ flops once $|\mathcal{S}_{k;l}|$ is known.

Note that

$${}_{k;l}D_{k;l} = {}_{k;l}A_{k;l} + \alpha \geqslant 0$$

since we have assumed that

$$\alpha \geqslant 0.$$

Let B denote the block-diagonal matrix

$$ {}_{k;}B_{k;} = (\alpha+\beta)\, e e^T, \qquad\qquad k \in \mathbb{N}_K, $$

where the size of each block is chosen such that

$$ A = D - B + \beta e e^T. $$

We first compute $(D-B)^{-1}$ noting that we just have to invert the K diagonal blocks

$$ {}_{k;}D_{k;} - {}_{k;}B_{k;} = {}_{k;}D_{k;} - (\alpha+\beta)\, e e^T = {}_{k;}D_{k;}^{1/2}\Big(I - (\alpha+\beta)\, {}_{k;}D_{k;}^{-1/2}\, e e^T\, {}_{k;}D_{k;}^{-1/2}\Big) {}_{k;}D_{k;}^{1/2}. $$

Let

$$ \tau_k = e^T\, {}_{k;}D_{k;}^{-1}\, e = \sum_{l \in \mathbb{N}_{\lambda_k}} \frac{1}{{}_{k;l}D_{k;l}}, $$

where $\tau \in \mathbb{R}^K$ can be computed in $O(L)$ flops if ${}_{k;l}D_{k;l}$ is available.

Proposition 17. $$ ({}_{k;}D_{k;} - {}_{k;}B_{k;})^{-1} = {}_{k;}D_{k;}^{-1} + \frac{\alpha+\beta}{1-(\alpha+\beta)\,\tau_k}\, {}_{k;}D_{k;}^{-1}\, e e^T\, {}_{k;}D_{k;}^{-1}. $$

Proof. By equation (11). □

Proposition 18. $$ \sigma = e^T (D-B)^{-1} e = \sum_{k \in \mathbb{N}_K} \frac{\tau_k}{1-(\alpha+\beta)\,\tau_k}. $$

Proof. We calculate:

$$ \begin{aligned} \sigma = e^T(D-B)^{-1}e &= \sum_{k\in\mathbb{N}_K} e^T({}_{k;}D_{k;} - {}_{k;}B_{k;})^{-1} e \\ &= \sum_{k\in\mathbb{N}_K} \left(e^T\, {}_{k;}D_{k;}^{-1} e + \frac{\alpha+\beta}{1-(\alpha+\beta)\,\tau_k}\, e^T\, {}_{k;}D_{k;}^{-1}\, e e^T\, {}_{k;}D_{k;}^{-1}\, e \right) \\ &= \sum_{k\in\mathbb{N}_K} \left(\tau_k + \frac{\alpha+\beta}{1-(\alpha+\beta)\,\tau_k}\, \tau_k^2 \right) \\ &= \sum_{k\in\mathbb{N}_K} \left(\frac{\tau_k(1-(\alpha+\beta)\,\tau_k) + (\alpha+\beta)\,\tau_k^2}{1-(\alpha+\beta)\,\tau_k} \right) \\ &= \sum_{k\in\mathbb{N}_K} \left(\frac{\tau_k - (\alpha+\beta)\,\tau_k^2 + (\alpha+\beta)\,\tau_k^2}{1-(\alpha+\beta)\,\tau_k} \right) \\ &= \sum_{k\in\mathbb{N}_K} \frac{\tau_k}{1-(\alpha+\beta)\,\tau_k}. \end{aligned} $$

□

Proposition 19. $$ A^{-1} = (D-B)^{-1} - \frac{\beta}{1+\beta\sigma}(D-B)^{-1} e e^T (D-B)^{-1}. $$

Proof. By equation (11)

$$ \begin{aligned} A^{-1} &= (D - B + \beta e e^T)^{-1} \\ &= (D-B)^{-1} - \frac{\beta}{1+\beta\sigma}(D-B)^{-1} e e^T (D-B)^{-1}. \end{aligned} $$

□

Proposition 20. $$ \epsilon_{k;l} = ((D-B)^{-1} e)_{k;l} = \frac{1}{(1-(\alpha+\beta)\,\tau_k)\, {}_{k;l}D_{k;l}}. $$

The computation costs $O(L)$ flops once ${}_{k;l}D_{k;l}$ and τ_k have been computed.

Proof. Direct computation with formulas we have already found:

$$\begin{aligned}((D-B)^{-1}e)_{k;} &= ({}_{k;}D_{k;} - {}_{k;}B_{k;})^{-1}e \\ &= {}_{k;}D_{k;}^{-1}e + \frac{\alpha+\beta}{1-(\alpha+\beta)\,\tau_k}\,{}_{k;}D_{k;}^{-1}\,e\,e^T\,{}_{k;}D_{k;}^{-1}e \\ &= {}_{k;}D_{k;}^{-1}e\left(1+\frac{\alpha+\beta}{1-(\alpha+\beta)\,\tau_k}\,e^T\,{}_{k;}D_{k;}^{-1}e\right) \\ &= {}_{k;}D_{k;}^{-1}e\left(1+\frac{(\alpha+\beta)\,\tau_k}{1-(\alpha+\beta)\,\tau_k}\right) \\ &= {}_{k;}D_{k;}^{-1}e\left(\frac{1-(\alpha+\beta)\,\tau_k+(\alpha+\beta)\,\tau_k}{1-(\alpha+\beta)\,\tau_k}\right) \\ &= \frac{{}_{k;}D_{k;}^{-1}e}{1-(\alpha+\beta)\,\tau_k}.\end{aligned}$$

□

Proposition 21. *Let*

$$\psi_{k;l} = \frac{z_{k;l}}{{}_{k;l}D_{k;l}} \qquad\qquad \mu_k = \sum_{l\in\mathbb{N}_{\lambda_k}} \psi_{k;l}.$$

Then

$$w_{k;l} = ((D-B)^{-1}z)_{k;l} = \psi_{k;l} + \frac{(\alpha+\beta)\,\mu_k}{(1-(\alpha+\beta)\,\tau_k)\,{}_{k;l}D_{k;l}}.$$

The cost of computing $w=(D-B)^{-1}z$ is $O(L)$ flops once ${}_{k;l}D_{k;l}$ is available.

Proof. Direct computation with formulas we have already found:

$$\begin{aligned}((D-B)^{-1}z)_{k;} &= ({}_{k;}D_{k;} - {}_{k;}B_{k;})^{-1}\,z_{k;} \\ &= {}_{k;}D_{k;}^{-1}z_{k;} + \frac{\alpha+\beta}{1-(\alpha+\beta)\,\tau_k}\,{}_{k;}D_{k;}^{-1}\,e\,e^T\,{}_{k;}D_{k;}^{-1}\,z_{k;} \\ &= \psi_{k;} + \frac{(\alpha+\beta)\left(e^T\,{}_{k;}D_{k;}^{-1}\,z_{k;}\right)}{1-(\alpha+\beta)\,\tau_k}\,{}_{k;}D_{k;}^{-1}\,e \\ &= \psi_{k;} + \frac{(\alpha+\beta)\,\mu_k}{(1-(\alpha+\beta)\,\tau_k)}\,{}_{k;}D_{k;}^{-1}\,e.\end{aligned}$$

□

Proposition 22.

$$A^{-1}z = w - \frac{\beta\,(\epsilon^T z)}{1+\beta\sigma}\,\epsilon.$$

The cost is $O(L)$ flops once σ, w and ϵ have been computed.

Proof. Direct computation with formulas we have already found:

$$\begin{aligned}A^{-1}z &= (D-B)^{-1}z - \frac{\beta}{1+\beta\sigma}(D-B)^{-1}e\,e^T\,(D-B)^{-1}z \\ &= w - \frac{\beta\,(e^T\,(D-B)^{-1}z)}{1+\beta\sigma}(D-B)^{-1}e \\ &= w - \frac{\beta\,(\epsilon^T z)}{1+\beta\sigma}\,\epsilon.\end{aligned}$$

□

12.7 Updating C, T and R

When $Y_{k;l}$'s membership in cluster k is changed we have to update C, T and R efficiently. The following propositions aid in this task, but are no means sufficient for the programmer, since the indices of the various quantities must be updated too.

Proposition 23. *If $\mathcal{S}_{k;l} = \{\}$ and $Y_{k;l}$ is deleted, then the new T and R can be computed from the following formulas:*

$$\begin{aligned}
{}_{k;n}T_k &\leftarrow {}_{k;n}T_k - {}_{k;n}C_{k;l}, && l \neq n \in \mathbb{N}_{\lambda_k},\\
{}_{p;i}T_k &\leftarrow {}_{p;i}T_k - {}_{p;i}C_{k;l}, && k \neq p \in \mathbb{N}_K, i \in \mathbb{N}_{\lambda_p},\\
R_{p;i} &\leftarrow R_{p;i} - {}_{p;i}C_{k;l}, && k \neq p \in \mathbb{N}_K, i \in \mathbb{N}_{\lambda_p}.
\end{aligned}$$

This costs $O(L)$ flops.

Proposition 24. *If $Y_{k;l}$ is split off from cluster k, then let $Y_{K;0} = Y_{k;l}$ and*

$$\begin{aligned}
{}_{K;0}C_{p;i} &= {}_{k;l}C_{p;i}, && p \in \mathbb{N}_K, i \in \mathbb{N}_{\lambda_p},\\
{}_{k;n}T_k &\leftarrow {}_{k;n}T_k - {}_{k;n}C_{k;l}, && l \neq n \in \mathbb{N}_{\lambda_k},\\
{}_{p;i}T_k &\leftarrow {}_{p;i}T_k - {}_{p;i}C_{k;l}, && k \neq p \in \mathbb{N}_K, i \in \mathbb{N}_{\lambda_p},\\
{}_{p;i}T_K &= {}_{p;i}C_{k;l}, && k \neq p \in \mathbb{N}_K, i \in \mathbb{N}_{\lambda_p},\\
{}_{K;0}T_p &= {}_{k;l}T_p, && p \in \mathbb{N}_K,\\
R_{K;0} &= R_{k;l} + {}_{k;l}T_k.
\end{aligned}$$

The cost of this update is $O(L)$ flops.

Proposition 25. *If $Y_{k;l}$ is transferred to cluster p, then*

$$\begin{aligned}
{}_{q;j}T_k &\leftarrow {}_{q;j}T_k - {}_{p;i}C_{k;l},\\
{}_{q;j}T_p &\leftarrow {}_{q;j}T_p + {}_{p;i}C_{k;l},\\
R_{k;l} &\leftarrow R_{k;l} - {}_{k;l}T_p + {}_{k;l}T_k,\\
R_{p;i} &\leftarrow R_{p;i} - {}_{p;i}C_{k;l}.
\end{aligned}$$

Proposition 26. *If $Y_{k;l}$ is swapped with $Y_{p;i}$, then*

$$\begin{aligned}
{}_{k;n}T_k &\leftarrow {}_{k;n}T_k - {}_{k;n}C_{k;l} + {}_{k;n}C_{p;i}\\
{}_{p;j}T_p &\leftarrow {}_{p;j}T_p + {}_{p;j}C_{k;l} - {}_{p;j}C_{p;i}\\
{}_{k;n}T_p &\leftarrow {}_{k;n}T_p + {}_{k;n}C_{k;l} - {}_{k;n}C_{p;i}\\
{}_{p;j}T_k &\leftarrow {}_{p;j}T_k + {}_{p;j}C_{p;i} - {}_{p;j}C_{k;l}\\
R_{k;n} &\leftarrow R_{k;n} - {}_{k;n}C_{p;i} + {}_{k;n}C_{k;l}\\
R_{p;j} &\leftarrow R_{p;j} - {}_{p;j}C_{k;l} + {}_{p;j}C_{p;i}.
\end{aligned}$$

The cost of the update is $O(L)$ flops.