The surprising secret identity of the semidefinite relaxation of K-means: manifold learning

Mariano Tepper*  
Flatiron Institute  
Simons Foundation, USA

Anirvan M. Sengupta*  
Dept. of Physics and Astronomy  
Rutgers University, USA

Dmitri Chklovskii*  
Flatiron Institute  
Simons Foundation, USA

Abstract

In recent years, semidefinite programs (SDP) have been the subject of interesting research in the field of clustering. In many cases, these convex programs deliver the same answers as non-convex alternatives and come with a guarantee of optimality. Unexpectedly, we find that a popular semidefinite relaxation of K-means (SDP-KM), learns manifolds present in the data, something not possible with the original K-means formulation. To build an intuitive understanding of its manifold learning capabilities, we develop a theoretical analysis of SDP-KM on idealized datasets. Additionally, we show that SDP-KM even segregates linearly non-separable manifolds. SDP-KM is convex and the globally optimal solution can be found by generic SDP solvers with polynomial time complexity. To overcome poor performance of these solvers on large datasets, we explore efficient algorithms based on the explicit Gramian representation of the problem. These features render SDP-KM a versatile and interesting tool for manifold learning while remaining amenable to theoretical analysis.

1 Introduction

K-means is one of the fundamental and most used clustering techniques [11]. Unfortunately, K-means is NP-hard and thus, in recent years, alternative optimization techniques for this problem have been studied [2, 14, 19]. Among them, semidefinite programming (SDP) has shown great promise. SDP is mainly appealing because of its convexity, coming with a certificate of optimality, and the existence of generic solvers that can find the global solution in polynomial time. A popular and interesting formulation [2, 14] is

$$\max_{Q \in \mathbb{R}^{n \times n}} \text{tr} (DQ) \quad \text{s.t.} \quad Q1 = 1, \quad \text{tr} (Q) = K, \quad Q \succeq 0, \quad Q \geq 0,$$

(SDP-KM)

where $D$ is the Gramian matrix, i.e., $(D)_{ij} = x_i^\top x_j$ where $\{x\}$ are the data points. Its link with K-means is explained in Appendix A. Until now, theoretical efforts have concentrated on showing that SDP-KM is a good surrogate for K-means. Awasthi et al. [2] study its solutions on datasets consisting of linearly separable clusters and demonstrate that they reproduce hard-clustering assignments of K-means. Moreover, the solution to SDP-KM achieves hard clustering even for some datasets on which Lloyd’s algorithm [15] fails (see also [10, 17]). Related problems have been studied in [1, 12, 28].

In this work, we analyze SDP-KM in a different regime than previous studies. Instead of focusing on cases and parameter settings where it approximates the original K-means formulation, we concentrate on alternative settings and discover that SDP-KM is not merely a convex K-means imitator. Fig. 1 shows two examples of this unexpected behavior where SDP-KM dissects the geometry of the data.

*Emails: mtepper@flatironinstitute.org, anirvans@physics.rutgers.edu, dchklovskii@flatironinstitute.org

2 Notation. $(X)_{i,j}, (X)_{j}, (X)_{i}$ denote the $(i,j)$-th entry of matrix $X$, the $j$-th column of $X$, and the $i$-th row of $X$, respectively. For vectors, we employ lowercase and we use a similar notation but with a single index. We write $X \geq 0$ if a matrix $X$ is entry-wise nonnegative and $X \succeq 0$ if that it is positive semidefinite.
Contributions. We first analyze SDP-KM for a simple synthetic example (Sec. 2). Building on this analysis, we suggest that SDP-KM has non-trivial manifold learning capabilities (Sec. 3) and show SDP-KM’s good performance in non-trivial examples, including synthetic and real datasets. Finally, to overcome dataset size limitations of SDP solvers we show that a non-convex Burer-Monteiro-style algorithm [14] shows promise with SDP-KM experimentally and theoretically (Sec. 4). Our software is publicly available at https://github.com/simonsfoundation/sdp_kmeans.

2 Problem analysis

Starting with the appearance of Isomap [22] and locally-linear embedding (LLE) [20], there has been outstanding progress in the area of manifold learning [e.g., 3, 8, 25, 26]. For a data matrix $X = [x_i]_{i=1}^n$ of column-vectors/points $x_i \in \mathbb{R}^d$, the majority of these modern methods have three steps:

1. Determine the neighbors of each point. This can be done in two ways: (1) keep all point within some fixed radius $\rho$ or (2) compute $\kappa$ nearest neighbors.
2. Construct a weighting matrix $W$, where $(W)_{ij} = 0$ if points $i$ and $j$ are not neighbors, and $(W)_{ij}$ is inversely proportional to the distance between points $i$ and $j$ otherwise.
3. Compute an embedding from $W$ that is locally isometric to $X$.

For the third step, many different and powerful approaches have been proposed, from computing shortest paths on a graph [22], to using graph spectral methods [3], to using neural networks [8]. However, the success of these techniques is critically determined by the ability to capture the data structure in the first two steps. Correctly setting either $\rho$ or $\kappa$ is non-trivial task that is left to the user of these techniques. Furthermore, a kernel (most commonly an RBF) is often involved in the second step, adding an additional parameter (the kernel width/scale) to the user to determine. Expectedly, the optimal selection of these parameters plays a critical role in the overall success of the manifold learning process.

SDP-KM departs drastically from this setup as no kernel selection nor nearest neighbor search are involved. Yet, the solution $Q_\ast$ is effectively a kernel which is automatically learned from the data. Because $Q_\ast$ is positive semidefinite it can be factorized as $Q_\ast = Y^\top Y$, defining a feature map from $X$ to $Y$. 

Figure 1: SDP-KM, originally introduced as a convex relaxation of $K$-means clustering, surprisingly performs manifold learning. (a) Learning the manifold of a trefoil knot, which cannot be “untied” in 3D without cutting it. SDP-KM understands that this is a closed manifold, yielding a circulant matrix $Q$, which can be “unfolded” in 2D. (b) Learning multiple manifolds with SDP-KM. Although they are linearly non-separable, SDP-KM correctly finds two submatrices, one for each manifold (for visual clarity, we enhance the contrast of $Q$).
We analyze the case in which the input data to SDP-KM possess rotational symmetry, i.e., data are locally isometric to $X$ where $q \neq 0$. Appendix B, we prove that the solution $Q$ where $W$ is the adjacency matrix of a weighted nearest-neighbors graph. The key to finding a matrix $Y$ that is locally isometric to $X$, while unwrapping the data manifold, is to remove from $W$ the connections between distant points $(X)_{ij}$ and $(X)_{ij}$. This is done with some technique to find nearest neighbors.

SDP-KM also tries to align the output Gramian, $Q$, to the input Gramian, $D$, but discards distant data points differently. As negative entries in $D$ cannot be matched because $Q$ is nonnegative, the best option would be to set the corresponding element of $Q$ to zero. This effectively discards pairs of input data points whose inner product is negative thus enforcing locality in the angular space [6], see Fig. 2. In fact, this argument can be taken further by noting that the constraint $Q1 = 1$ allows us to replace the Gramian $D$ with the negative squared distance matrix,

$$ -\frac{1}{2} \sum_{ij} \|(X)_{ij} - (X)_{ij}\|_2^2(Q)_{ij} = -\sum_i (D)_{ii} \sum_j (Q)_{ij} + tr(DQ) = -tr(D) + tr(DQ). $$

Finally, the constraint $tr(Q) = K$ allows further control of the neighborhood size of SDP-KM (modulating the actual width of its kernel function, see Fig. 2). Next, we develop further intuition about the manifold-learning capabilities of SDP-KM by analyzing theoretically the dataset in Fig. 2.

### 2.1 A formal analysis of a 2D ring dataset

We analyze the case in which the input data to SDP-KM possess rotational symmetry, i.e., data are arranged uniformly on a ring, see Fig. 3. In this case, we can write the SDP as a linear program (LP) in the circular Fourier basis. This new representation allows to visualize that SDP-KM lifts the data into a high-dimensional space, with $K$ controlling its dimensionality.

In the example in Fig. 3, the entries of $D$ can be described by $(D)_{ij} = (x_i^T x_j) = \cos(\alpha_i - \alpha_j)$, where $\alpha_i, \alpha_j$ are the angles of points $x_i, x_j$, respectively (Fig. 2). Since the points are uniformly distributed over the ring, $D$ is a circulant matrix, i.e., $\cos(\alpha_i - \alpha_j) = \cos(\alpha_i + k - \alpha_j + k)$. In Proposition 1 of Appendix B, we prove that the solution $Q$, to SDP-KM is circulant too. Being circulant matrices, $D$ and $Q_*$ are diagonalized by the discrete Fourier transform (DFT), i.e.,

$$ D = F \text{ diag}(d) F^H, \quad Q_* = F \text{ diag}(q) F^H, $$

where $q \geq 0, d \geq 0$ respectively are vectors containing the eigenvalues of $D$ and $Q_*$, and $F \in \mathbb{C}^{n \times n}$ is the unitary DFT matrix, with entries $(F)_{pk} = \frac{1}{\sqrt{n}} \exp(-i2\pi p k/n)$.

Hence, and in accord with the constraint $Q_*1 = 1$, we have that $(F)_{00} = \frac{1}{\sqrt{n}}1$ and $(q)_0 = 1$. 

Figure 3: We regularly sample 100 points on the unit ring. The LP formulation in Problem (8) yields solutions that are practically indistinguishable from the ones given by SDP-KM; the relative error between the matrices $Q_e$ obtained with SDP and LP is $2.41 \cdot 10^{-3}$ and the relative error between their eigenvalues is $2.66 \cdot 10^{-3}$.

2.2 A linear program on the data manifold

We express the objective function and the constraints of SDP-KM in terms of $d$ and $q$, i.e.,

$$\text{tr} (DQ_e) = d^\top q, \quad (5)$$

$$\text{tr} (Q) = 1^\top q = K, \quad (6)$$

$$(Q)_{kk'} = (F)_{k'} \text{diag}(q) (F^\top)_{k'} = \sum_{p=0}^{n-1} \frac{(q)_p \cos \left(2\pi p \frac{k-k'}{n}\right)}{n} \geq 0. \quad (7)$$

This allows us to rewrite SDP-KM as the linear program

$$\max_{q} d^\top q \quad \text{s.t.} \quad (\forall \tau) \ c_{\tau}^\top q \geq 0, \quad 1^\top q = K, \quad q \geq 0, \quad (q)_0 = 1, \quad (8)$$

where $(c_{\tau})_p = \frac{1}{n} \cos \left(2\pi p \frac{\tau}{n}\right)$. As a sanity check, we implemented Problem (8) and, effectively, it provides the same result than SDP-KM, see Fig. 3.

Problem (8) sheds light into the inner workings of SDP-KM. First, the constraint $1^\top q = K$ ensures that $q$ does not grow to infinity and acts as a budget constraint. Let us assume for a moment that we remove the constraint $c_{\tau}^\top q \geq 0$ (the equivalent of $Q \geq 0$). Then, the program will try to set to $K$ the entry of $q$ corresponding to the largest eigenvalue of $d$; this $q$ will violate as $K$ gets bigger the removed constraint (since $(c_{\tau})_p$ is a sinusoid). Then the effect of this constraint is to spread the allocated budget among several eigenvalues (instead of just the largest). The experiment in Fig. 4 confirms this: the number of eigenvalues of $Q_e$ grows with $K$. We can interpret this as increasing the intrinsic dimensionality of the problem in such a way that only local interactions are considered.

2.3 Lifting the ring to a high-dimensional cone

Here, we show that SDP-KM effectively embeds the data manifold into a space where its structure, i.e., rotational symmetry, is preserved. We now make use of the half-wave symmetry in $Q_e$, noting it they can be fully represented with only one half of the Fourier basis. We can then decompose it with the real Fourier basis

$$Q_e = \tilde{F} \text{diag}(\tilde{q}) \tilde{F}^\top, \quad (9)$$

where $\tilde{q} = [(q)_0, (q)_1, (q)_1, \ldots, (q)_{n-1}, (q)_{n-1}]^\top$ and $\tilde{F} \in \mathbb{R}^{n \times n}$ has entries $(p, k = 0, \ldots, n-1)$

$$ (\tilde{F})_{pk} = \begin{cases} \frac{2}{\sqrt{n}} \cos \left(\frac{2\pi p k}{n}\right) & \text{if } k \text{ is even}, \\ \frac{2}{\sqrt{n}} \sin \left(\frac{2\pi p k}{n}\right) & \text{if } k \text{ is odd}. \end{cases} \quad (10)$$

Let $\tilde{Y} = \text{diag}(q)^{1/2} \tilde{F}^\top$. Notice that $\langle \tilde{Y}_{i,1}, \tilde{Y}_{i,1} \|_{F}, \tilde{F}_{i,0} \rangle = (\tilde{F}_{i,1}, \tilde{F}_{i,0}) = \frac{4}{n}$, meaning that the vectors $\tilde{Y}_{i,1}$ are the extreme rays of a right circular cone with the eigenvector $\tilde{F}_{i,0} = \frac{2}{\sqrt{n}} [1, 0, \ldots, 0]^\top$ as its symmetry axis, see Fig. 4(d). Thus, we can interpret the solution to SDP-KM as lifting the 2D ring structure into a cone. As mentioned before, this cone is high-dimensional, with as many directions as needed to preserve the nonnegativity of $Q$.

We identify the rank of the solution $Q$ with the number of active eigenvalues. The bigger the $K$, the higher the rank. The constraint $Q1 = 1$ in SDP-KM leads to a fanning-out effect in the data.
Figure 4: Analysis of the example in Fig. 3. (a) Several solutions \( Q^* \) to SDP-KM. As \( K \) increases, the matrix concentrates more and more toward the diagonal. (b) Number of active eigenvalues in the solution \( Q^* \) to SDP-KM as a function of \( K \). As \( K \) increases, more eigenvalues are active in \( Q^* \). (c) We define the \( h \)-diagonal of \( Q^* \) as the entries \((i, j)\) for which \( i - j = h \). As \( Q^* \) is a circulant matrix, each \( h \)-diagonal contains a single repeated value. We plot these values, assigning a different color to each \( h \). The effect of the scaling constraint \( \text{tr}(Q) = K \) becomes evident: when one \( h \)-diagonal becomes inactive, all remaining \( h \)-diagonals need to be upscaled. (d) The eigenvectors of \( Q^* \) form a high-dimensional cone (we show a cartoon representation with the cone axis in red and the eigenvectors in green).

representation. Intuitively, this fan-out effect is key to the disentanglement of datasets with complex topologies. Spin-model-inspired SDPs for community detection [12] achieve a similar fanning-out by dropping the constraint \( Q_1 = 1 \) and adding the related term \(-\gamma 1^T Q 1\) to the objective function.

With the LP framework and the geometric picture in place, we can begin to understand how the solution evolves as the parameter \( K \) increases from 1 to \( n \). At \( K = 1 \), only the eigenvalue \((q_0)^0\) is active and every vector \((\tilde{Y})_i\) is constant, with entries equal to \(1/n\). With \( K \) slightly above 1, the eigenvalue \((q_1)^1\) becomes active (nonzero), introducing the first nontrivial Fourier component. Geometrically, the vectors \(\{(\tilde{Y})_i\} \) now open up in a narrow cone. As \( K \) increases, the cone widens and, at some point, the angle between two of the vectors reaches \(\pi/2\) (this amounts to hitting the boundary of the nonnegativity constraint in Eq. (7)). Further increase of \( K \) necessitates use of a larger number of Fourier modes. Finally, at \( K = n \) all modes are active and all vectors \(\{(\tilde{Y})_i\} \) become perpendicular to each other. The progression of the number of active modes with \( K \) is depicted in Fig. 4(b).

3 Manifold learning with SDP-KM: Experimental results

In the previous section, we showed that SDP-KM recovers the data manifold in a simple example. This observation is true for more complex datasets for which there is no analytical expression for the transformation that diagonalizes \( Q^* \) (nor \( D \)), see Fig. 1 and examples in the next Section. We visualize the solution \( Q^* \) by embedding it in a low-dimensional space. While our goal is not dimensionality reduction, we learn the data manifold with SDP-KM, and use standard spectral dimensionality reduction to visualize the results.

Recovering multiple manifolds. K-means cannot effectively recover \( K \) distinct manifolds, unless they are linearly separable. Interestingly, SDP-KM does not inherit this property when \( K \) is properly chosen. Of course, if we set \( K \) to the number of manifolds that we want to recover, there is no hope in the general case to obtain a result substantially better than the one obtained with Lloyd’s algorithm [15]. However, if we set \( K \) high enough, SDP-KM is able to use a portion of its budget \( K \) to individually adjust each manifold.

An example with two rings is presented in Fig. 5. We can expect that, as the single ring in Sec. 2 is described by Fourier modes, SDP-KM describes two rings with two sets of Fourier modes with disjoint support; the solution is now arranged as two orthogonal high-dimensional cones, see Fig. 5(b). In a sense, the manifold learning problem is already solved, as there are two circulant submatrices, one for each manifold, with no interactions between them. If the user desires a hard assignment of points to manifolds, we can simply consider \( Q^* \) as the adjacency matrix of a weighted graph and compute its connected components.
Figure 5: Eigenvector structure of the input Gramian and of the solution $Q$ for a dataset of points lying in two rings. (a) The eigenvectors of $D$ ($D$ has rank 2) do not separate the rings. SDP-KM produces two sets of eigenvectors with disjoint support: one set describing the points in each ring (we show all eigenvectors and a detail on the first 3 within each set). (b) The eigenvectors of $Q$ form two orthogonal high-dimensional cones: one cone for each ring (we show a cartoon representation with the cone axis in red and the eigenvectors in green). Notice how these cones become linearly-separable.

Figure 6: Learning multiple manifolds with SDP-KM. The points are arranged in two semicircular manifolds and contaminated with noise. Although they are linearly non-separable, SDP-KM correctly finds two submatrices, one for each manifold (for visual clarity, we enhance the contrast of $Q$).

Discussion of the experimental results. The trefoil knot in Fig. 1(a) is a 1D manifold in 3D; it is the simplest example of a nontrivial knot, meaning that it is not possible to “untie” it in three dimensions without cutting it. However, the manifold learning procedure in Sec. 3 is able to learn the closed 1D manifold. We also present examples using real high-dimensional datasets, recovering in every case structures of interest, see Fig. 7. In figs. 7(a) to 7(c), SDP-KM respectively uncovers the camera rotation, the orientation of the lighting source, and specific handwriting features.

We use several toy datasets that are standard in the literature to demonstrate the multi-manifold learning capabilities of SDP-KM, see figs. 1(b), 5 and 6. In all of these examples, SDP-KM is able to disentangle clusters that are not linearly separable. We also present results for a real dataset in Fig. 8. The dataset is similar to the one in Fig. 7(a) but with two objects; we recover two closed manifolds, each of which corresponds to the viewpoints of one object. The structure of the solution is similar to the one in Fig. 5(b).

4 From convex problems to big data

Standard SDPs involve $O(n^2)$ variables and their resulting time complexity is often $O(n^3)$. Consequently, standard solvers will struggle with large datasets. SDP-KM lends itself to a fast and big-data-friendly implementation [14]. This is done by posing a related problem

$$\max_{Y \in \mathbb{R}^{r \times n}} \text{tr} \left( DY^T Y \right) \quad \text{s.t.} \quad \text{tr} \left( Y^T Y \right) = K, \quad Y^T Y 1 = 1, \quad Y \geq 0.$$ (11)

In this new problem, we have forgone convexity in exchange of reducing the number of unknowns from $O(n^2)$ to $rn$. Kulis et al. [14] set $r = K$. The problematic constraint $Y^T Y \geq 0$, involving $O(n^2)$ terms, has been replaced by the much stronger but easier to enforce $Y \geq 0$. The speed gain is shown in Fig. 9 (in our desktop with 128GB of RAM, cvxpy cannot handle SDP instances bigger than $1200 \times 1200$). See Appendix D for a description of the algorithm.

However, the constraint change is only correct if $Q$ is completely positive. An $n \times n$ matrix $A$ is called completely positive (CP) if there exists $B \geq 0$ such that $A = B^T B$. The least possible number of rows of $B$ is called the cp-rank of $A$. 


Figure 7: Finding two-dimensional embeddings with SDP-KM. (a) 100 images obtained by viewing a teapot from different angles in a plane. The input vectors size is 23028 (76 × 101 pixels, 3 color channels). The manifold uncovers the change in orientation. (b) 256 images from 4 different subjects (each subject is marked with a different color in the figure), obtained by changing the position of the illumination source. The input vectors size is 32256 (192 × 168 pixels). The manifold uncovers the change in illumination (from frontal, to half-illuminated, to dark faces, and back). (c) 500 images handwritten instances of the same digit. The input vectors size is 784 (28 × 28 pixels). On the left and on the right, images of the digits 1 and 2, respectively. The manifold of 1s uncovers their orientation, while the manifold of 2s parameterizes features like size, slant, and line thickness. Details are better perceived by zooming on the plots.

Figure 8: 144 images obtained by viewing a lamp and a horse figurine from different angles in a plane. The input vectors size is 589824 (384 × 512 pixels, 3 color channels). We plot the input data using a 2D spectral embedding (the points corresponding to each object are colored differently). SDP-KM correctly finds two submatrices, one for each manifold (for visual clarity, we enhance the contrast of \( Q \)); furthermore, SDP-KM recovers closed manifolds.

We are thus interested in two questions. First, is the solution \( Q_* \) to SDP-KM completely positive? Answering this question in the affirmative would allow for theoretically sound and fast implementations of SDP-KM. Second, what is the cp-rank of \( Q_* \)? Is it true that \( \text{cp-rank}(Q_*) \leq K \)? This issue is critical, as it determines the number of unknowns in the problem. This topic is by no means trivial. Even if the set of CP matrices forms a convex cone, the problems of determining whether a matrix is inside the set and of projecting a matrix into the set are NP-hard.

It is not hard to prove that, whenever SDP-KM produces a hard-clustering \( Q_* \) (see Awasthi et al. [2] for such conditions), \( Q_* \) is CP. When SDP-KM produces a soft-clustering \( Q_* \), as in all of the examples in this paper, the question becomes more difficult.

A matrix \( A \) such that \( A \geq 0 \) and \( A \succeq 0 \) is called doubly nonnegative (DN). Obviously, \( Q_* \) is DN. From the definitions, every CP matrix is DN but the reverse is not necessarily the case. For very small examples \((n \leq 4)\), we know that the sets of CP and of DN matrices are equivalent [16].

Let us now go back to the example in Sec. 2 (points arranged regularly on a ring). For this example, we can establish a simple sufficient condition on \( K \), for \( Q_* \) to be CP. Recall that if \( D \) is circulant, \( Q_* \) is circulant (see Proposition 1 in Appendix B). In Proposition 2 of Appendix B, we prove that if the solution \( Q_* \) to SDP-KM is a circulant matrix, then it is CP for every \( K \leq 3/2 \) or \( K \geq \frac{n}{2} \). Naturally, more theory is needed to shed light into this problem in general scenarios.

Complementarily, we have studied the questions raised in this section from an experimental viewpoint. We use the symmetric nonnegative matrix factorization (SNMF) of \( Q_* \), see Appendix C, as a proxy.
The non-convex solver is much faster than the convex one (despite comparing our non-optimized python code of the former, versus a highly optimized C++ code of the latter). The speedup also comes with a lesser memory usage.

Interestingly, setting $r = K$ produces “softer” solutions. This suggests that the cp-rank of $Q_*$ is (much) greater than $K$.

for checking whether $Q_*$ is CP. The rationale is that if the approximation with SNMF is very tight, it is highly likely that $Q_*$ is CP. These experiments are presented in Fig. 10. We found that, with a properly chosen rank $r$, SNMF can indeed accurately approximate $Q_*$. However, setting $r = K$ is in general not enough and leads to a poor reconstruction. These two facts support the idea that $Q_*$ is CP, but has a cp-rank much higher than $K$.

Our experiments with the non-convex algorithm in Appendix D lead to similar conclusions as those with SNMF, see Fig. 11. Setting $r = K$, leads to a poor approximation of $Q_*$ and, as observed by Kulis et al. [14], to hard-clustering. Setting $r \gg K$ leads to much improved reconstructions.

5 Conclusions

In this work, we showed that SDP-KM can learn multiple low-dimensional data manifolds in high-dimensional spaces. An SDP instance, it is convex and can be solved in polynomial-time. Unlike most manifold learning algorithms, the user does not need to select/use a kernel and no nearest neighbors searches are involved. Theoretical and empirical evidence suggests that the solutions to SDP-KM are completely positive for many standard datasets: we can thus safely use a Burer-Monteiro-style non-convex algorithm that scales better with the dataset size than traditional SDP solvers, allowing to handle much larger datasets.
A Relationship with $K$-means

$K$-means seeks to cluster a dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ by computing

$$\min_{\mathbf{C}_K} \sum_{k=1}^K \sum_{\mathbf{x}_i \in \mathbf{C}_k} \|\mathbf{x}_i - \frac{1}{|\mathbf{C}_k|} \sum_{\mathbf{x}_j \in \mathbf{C}_k} \mathbf{x}_j\|_2^2, \quad (K\text{-means})$$

where $\mathbf{C}_K = \{\mathbf{C}_k\}_{k=1}^K$ is a partition of $\mathbf{X}$, i.e., $\mathbf{C}_k \cap \mathbf{C}_k' = \emptyset$ and $\bigcup_k \mathbf{C}_k = \mathbf{X}$. Albeit its popularity, it is known to be NP-Hard and, in practice, users employ an heuristic [15, originally developed in 1957] to find a solution.

The objective function of $K$-means, henceforth denoted $J_K$, can be rewritten as

$$J_K = -\sum_{k=1}^K \frac{1}{|\mathbf{C}_k|} \sum_{i,j} \mathbf{x}_i^\top \mathbf{x}_j \cdot (\mathbf{z}_k^\top \mathbf{z}_k)_{ij} = -\sum_{i,j} (\mathbf{X}^\top \mathbf{X})_{ij} (\mathbf{Y}^\top \mathbf{Y})_{ij} = -\text{tr} \left( \mathbf{X}^\top \mathbf{X} \mathbf{Y}^\top \mathbf{Y} \right).$$

By construction, the matrix $\mathbf{Q} = \mathbf{Y}^\top \mathbf{Y}$ exhibits the following properties

$$\mathbf{Q}1 = 1, \quad \text{tr} (\mathbf{Q}) = K. \quad (15)$$

Let $\mathbf{D}$ be the Gramian matrix, i.e., $\mathbf{D} = \mathbf{X}^\top \mathbf{X}$. We can then re-cast $K$-means as the optimization problem

$$\max_{\mathbf{Y} \in \mathcal{V}_Y} \begin{cases} \mathbf{Q}1 = 1, & \text{tr} (\mathbf{D}) \mathbf{Q} \quad \text{s.t.} \quad \text{tr} (\mathbf{Q}) = K, \quad \text{rank} (\mathbf{Q}) = K, \quad \mathbf{Q} \geq 0, \mathbf{Q} \geq 0. \end{cases} \quad (16)$$

where $\mathcal{V}_Y = \{0\} \cup \{|\mathbf{C}_k|^{-1/2}\}_{k=1}^K$. Seeking to apply the desirable properties of SDP to $K$-means, we can pose [14, 19]

$$\max_{\mathbf{Q} \in \mathbb{R}^{n \times n}} \begin{cases} \mathbf{Q}1 = 1, & \text{tr} (\mathbf{Q}) = K, \quad \text{rank} (\mathbf{Q}) = K, \quad \mathbf{Q} \geq 0, \mathbf{Q} \geq 0. \end{cases} \quad (17)$$

where mixed-integer program is relaxed into the real-valued nonnegative program, directly optimizing over $\mathbf{Q}$. SDP-KM is as a relaxation of this problem, simply obtained by removing the rank constraint.

B Proofs

Let $\mathbf{P}$ be the cyclic permutation matrix, i.e., $\mathbf{P} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. For any circulant matrix $\mathbf{D}$, we have that $\mathbf{P}^\top \mathbf{D} \mathbf{P} = \mathbf{D}$ (notice that this implies that $\mathbf{P}^s \mathbf{D} \mathbf{P}^s = \mathbf{D}$ for any $s \in \mathbb{N}$).

**Proposition 1.** Let $\mathbf{Q}_* \mathbf{P}$ be the solution to SDP-KM. If $\mathbf{P}^\top \mathbf{D} \mathbf{P} = \mathbf{D}$, then $\mathbf{P}^\top \mathbf{Q}_* \mathbf{P}$ is also the solution to SDP-KM. Since the problem is convex, $\mathbf{P}^\top \mathbf{Q}_* \mathbf{P} = \mathbf{Q}_*$, and $\mathbf{Q}_*$ must be a circulant matrix.
Proof. We have that for the objective function
\[
\text{tr} (DP^TQ, P) = \text{tr} (PDP^TQ, P) = \text{tr} (DQ, P)
\] (18)
and for the constraints, using \(PP^T = I\),
\[
\text{tr} (P^TQ, P) = \text{tr} (DQ, P) = K,
\] (19)
\[
P^TQ, P1 = P^TQ, I = P^T I = 1.
\] (20)
Since \(z^TQ, z \geq 0\), then \(z^TP^TQ, Pz \geq 0\). Finally, \(Q, s \geq 0\) implies that \(P^TQ, P \geq 0\).

Proposition 2. If the solution \(Q, s\) to SDP-KM is a circulant matrix, then it is CP for every \(K \leq 3/2\) or \(K \geq \frac{n}{2}\).

Proof. For \(K \leq 3/2\), plugging the constraint \(Q, 1 = 1\) into Corollary 2.6 in [21, p. 7] gives the desired result.

Let us address \(K \geq \frac{n}{2}\). Kaykobad [13] proved that every diagonally dominant matrix \(A\), i.e., \(|(A)_{ii}| \geq \sum_{j \neq i} |(A)_{ij}|\) for all \(i\), is a CP matrix. We have to prove then that \(Q, s \geq 0\) is diagonally dominant. We have \(\text{tr} (Q, s) = K\) and, since \(Q, s\) is circulant, all \((Q, s)_{ij}\) have the same value. Then, \((Q, s)_{ii} = K/n\). From \(Q, 1 = 1\), \(\sum_{j \neq i} (Q, s)_{ij} = 1 - (Q, s)_{ii} = 1 - K/n\). Hence, \(Q, s\) is diagonally dominant for \(K \geq \frac{n}{2}\).

C Symmetric NMF

In this section, we present the algorithm used to compute the symmetric NMF of a matrix \(A \in \mathbb{R}^{n \times n}\), defined as
\[
\min_{Y \in \mathbb{R}^{n \times r}} \|A - YY^T\|_F^2 \quad \text{s.t.} \quad Y \geq 0.
\] (SNMF)

We use the alternating direction method of multipliers (ADMM) to solve it. In short, ADMM solves convex optimization problems by breaking them into smaller subproblems, which are individually easier to handle. It has also been extended to handle non-convex problems, e.g., to solve several flavors of NMF [7, 27, 23, 24].

Problem (SNMF) can be equivalently re-formulated as
\[
\min_{Y \in \mathbb{R}^{n \times r}} \|A - YX^T\|_F^2 \quad \text{s.t.} \quad Y = X, \ Y \geq 0, \ X \geq 0,
\] (21)
and we consider its augmented Lagrangian,
\[
\mathcal{L} (X, Y, \Gamma) = \frac{1}{2} \|A - YX^T\|_F^2 + \frac{\sigma}{2} \|Y - X\|_F^2 - \text{tr} (\Gamma^T (Y - X))
\] (22)
where \(\Gamma\) is a Lagrange multiplier, \(\sigma\) is a penalty parameter.

The ADMM algorithm works in a coordinate descent fashion, successively minimizing \(\mathcal{L}\) with respect to \(X, Y\), one at a time while fixing the other at its most recent value and then updating the multiplier \(\Gamma\). For the problem at hand, these steps are
\[
X^{(t+1)} = \arg\min_{X \geq 0} \mathcal{L} (X, X^{(t)}, \Gamma^{(t)}), \quad (23a)
\]
\[
Y^{(t+1)} = \arg\min_{Y \geq 0} \mathcal{L} (X^{(t+1)}, Y, \Gamma^{(t)}), \quad (23b)
\]
\[
\Gamma^{(t+1)} = \Gamma^{(t)} - \eta \sigma (X^{(t+1)} - Y^{(t+1)}), \quad (23c)
\]
In our experiments, we fix \(\eta\) and \(\sigma\) to 1. We initialize the algorithm with a random matrix.
D Non-convex SDP solver

We follow the algorithm proposed by [14, 9] to solve Problem (11). Our approach has a small but fundamental difference: instead of setting \( r = K \), we allow for \( r \geq K \). We define the augmented Lagrangian of Problem (11) as

\[
\mathcal{L}(Y, \mu, \lambda) = -\text{tr}(DY^TY) + \frac{\sigma}{2} \|Y^TY - I\|^2_2 - \mu^T(Y^TY - I)
\]

where \( \mu, \lambda \) are Lagrange multipliers, \( \sigma, \varphi \) are penalty parameters. We obtain \( Y \) by running the steps

\[
Y^{(t+1)} = \arg\min_{Y \geq 0} \mathcal{L}(Y, \mu^{(t)}, \lambda^{(t)}),
\]

\[
\mu^{(t+1)} = \mu^{(t)} - \eta \sigma (Y^TY - I),
\]

\[
\lambda^{(t+1)} = \lambda^{(t)} - \eta \varphi (\text{tr}(Y^TY) - K).
\]

This is a non-standard approach since the minimization over \( Y \) (the gradient \( \partial \mathcal{L}/\partial Y \) is given by [9]) is a non-convex problem. Although there are no guarantees about the convergence of the procedure, theoretical assurances for related problems have been presented by [4]. To perform the minimization with respect to \( Y \), we use the L-BFGS-B algorithm [5] with bound constraints \((Y)_{ij} \in [0,1])].

Finally, the initialization to the overall iterative algorithm is done with symmetric nonnegative matrix factorization, see Appendix C. In our experiments, we fix \( \eta, \varphi, \) and \( \sigma \) to 1 and prenormalize \( D \) (dividing by its Frobenius norm).

References

[1] Amini, A. and Levina, E. On semidefinite relaxations for the block model. Technical report, arXiv:1406.5647, 2014. URL http://arxiv.org/abs/1406.5647.

[2] Awasthi, P., Bandeira, A., Charikar, M., Krishnaswamy, R., Villar, S., and Ward, R. Relax, no need to round: Integrality of clustering formulations. In ITCS, New York, NY, USA, 2015. ISBN 978-1-4503-3333-7. doi: 10.1145/2688073.2688116. URL http://doi.acm.org/10.1145/2688073.2688116.

[3] Belkin, M. and Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation, 15(6):1373–1396, 2003. ISSN 0899-7667. doi: 10.1162/089976603321780317. URL http://www.mitpressjournals.org/doi/10.1162/089976603321780317.

[4] Boumal, N., Voroninski, V., and Bandeira, A. The non-convex Burer-Monteiro approach works on smooth semidefinite programs. In NIPS, 2016.

[5] Byrd, R., Lu, P., Nocedal, J., and Zhu, C. A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific Computing, 16(5):1190–1208, 1995. ISSN 1064-8275. doi: 10.1137/0916069. URL http://epubs.siam.org/doi/10.1137/0916069http://epubs.siam.org/doi/abs/10.1137/0916069.

[6] Cho, Y. and Saul, L. Kernel methods for deep learning. NIPS, pp. 342–350. 2009. URL http://papers.nips.cc/paper/3628-kernel-methods-for-deep-learning%5Cnfile:///Files/30/30FC697D-BC63-4E21-94BA-D6C76FA225B2.pdf.

[7] Févotte, C. and Idier, J. Algorithms for nonnegative matrix factorization with the \( \beta \)-divergence. Neural Computation, 23(9):2421–2456, 2011. ISSN 0899-7667. doi: 10.1162/NECO_a_00168. URL http://www.mitpressjournals.org/doi/abs/10.1162/NECO_a_00168.

[8] Hadsell, R., Chopra, S., and LeCun, Y. Dimensionality reduction by learning an invariant mapping. In CVPR, 2006. ISBN 0-7695-2597-0. doi: 10.1109/CVPR.2006.100. URL http://ieeexplore.ieee.org/document/1640964/.

[9] Hou, Y., Whang, J., Gleich, D., and Dhillon, I. Non-exhaustive, overlapping clustering via low-rank semidefinite programming categories and subject descriptors. In KDD, New York, New York, USA, 2015. ISBN 9781450336642. doi: 10.1145/2783258.2783398. URL http://dl.acm.org/citation.cfm?doid=2783258.2783398.
[10] Iguchi, T., Mixon, D., Peterson, J., and Villar, S. On the tightness of an SDP relaxation of k-means. Technical report, arXiv:1505.04778, 2015. URL http://arxiv.org/abs/1505.04778.

[11] Jain, A. Data clustering: 50 years beyond K-means. Pattern Recognition Letters, 31(8):651–666, 2010. ISSN 01678655. doi: 10.1016/j.patrec.2009.09.011.

[12] Javanmard, A., Montanari, A., and Ricci-Tersenghi, F. Phase transitions in semidefinite relaxations. Technical report, arXiv:1511.08769, 2015.

[13] Kaykobad, M. On nonnegative factorization of matrices. Linear Algebra and its Applications, 96:27–33, 1987. ISSN 00243795. doi: 10.1016/0024-3795(87)90334-X. URL http://linkinghub.elsevier.com/retrieve/pii/002437958790334X.

[14] Kulis, B., Surendran, A., and Platt, J. Fast low-rank semidefinite programming for embedding and clustering. In AISTATS, 2007.

[15] Lloyd, S. Least squares quantization in PCM. IEEE Transactions on Information Theory, 28(2):129–137, 1982. ISSN 0018-9448. doi: 10.1109/TIT.1982.1056489. URL http://ieeexplore.ieee.org/document/1056489/.

[16] Maxfield, J. and Minc, H. On the matrix equation $X'X=A$. Proceedings of the Edinburgh Mathematical Society (Series 2), 13(02):125–129, 1962.

[17] Mixon, D., Villar, S., and Ward, R. Clustering subgaussian mixtures by semidefinite programming. Technical report, arXiv:1602.06612, 2016. URL http://arxiv.org/abs/1602.06612.

[18] O’Donoghue, B., Chu, E., Parikh, N., and Boyd, S. Conic optimization via operator splitting and homogeneous self-dual embedding. Journal of Optimization Theory and Applications, 169(3):1042–1068, 2016. ISSN 0022-3239. doi: 10.1007/s10957-016-0892-3. URL http://link.springer.com/10.1007/s10957-016-0892-3.

[19] Peng, J. and Wei, Y. Approximating K-means-type Clustering via Semidefinite Programming. SIAM Journal on Optimization, 18(1):186–205, jan 2007. ISSN 1052-6234. doi: 10.1137/050641983. URL http://epubs.siam.org/doi/10.1137/050641983.

[20] Roweis, S. and Saul, L. Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science, 290(5500), 2000.

[21] So, W. and Xu, C. When does cp-rank equal rank? Technical report, arXiv:1308.3193, 2013. URL http://arxiv.org/abs/1308.3193.

[22] Tenenbaum, J., Silva, V., and Langford, J. A global geometric framework for nonlinear dimensionality reduction. Science, 290(5500), 2000.

[23] Tepper, M. and Sapiro, G. A bi-clustering framework for consensus problems. SIAM Journal on Imaging Sciences, 7(4):2488–2525, 2014.

[24] Tepper, M. and Sapiro, G. Compressed nonnegative matrix factorization is fast and accurate. IEEE Transactions on Signal Processing, 64(9):2269–2283, 2016.

[25] Weinberger, K. and Saul, L. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2, pp. 1683–1686, 2006.

[26] Weiss, Y., Torralba, A., and Fergus, R. Spectral hashing. NIPS, 2008. doi: 10.1.1.141.7577.

[27] Xu, Y., Yin, W., Wen, Z., and Zhang, Y. An alternating direction algorithm for matrix completion with nonnegative factors. Front. Math. China, 7(2):365–384, 2012.

[28] Yu, Y., Neufeld, J., Kiros, R., Zhang, X., and Schuurmans, D. Regularizers versus losses for nonlinear dimensionality reduction. In ICML, 2012. ISBN 9781450312851.