LUND UNIVERSITY

Non-negative Spherical Relaxations for Universe-Free Multi-matching and Clustering

Thunberg, Johan; Bernard, Florian

[Link to publication](#)

Non-Negative Spherical Relaxations for Universe-Free Multi-Matching and Clustering

Johan Thunberg[1] and Florian Bernard[2]

[1] Halmstad University, Sweden
[2] University of Bonn, Germany

Abstract. We propose a novel non-negative spherical relaxation for optimization problems over binary matrices with injectivity constraints, which in particular has applications in multi-matching and clustering. We relax respective binary matrix constraints to the (high-dimensional) non-negative sphere. To optimize our relaxed problem, we use a conditional power iteration method to iteratively improve the objective function, while at same time sweeping over a continuous scalar parameter that is (indirectly) related to the universe size (or number of clusters). Opposed to existing procedures that require to fix the integer universe size before optimization, our method automatically adjusts the analogous continuous parameter. Furthermore, while our approach shares similarities with spectral multi-matching and spectral clustering, our formulation has the strong advantage that we do not rely on additional post-processing procedures to obtain binary results. Our method shows compelling results in various multi-matching and clustering settings, even when compared to methods that use the ground truth universe size (or number of clusters).

Keywords: Multi-matching · Clustering · Spectral methods · Spectral clustering · Permutation synchronization.

1 Introduction

In this work we propose a novel method that generalizes the power iteration algorithm to handle non-negative matrices with rows on the unit sphere (i.e. the intersection of the non-negative orthant and the unit sphere), referred to as "non-negative sphere". This is highly relevant for optimization problems that aim to find assignments between elements, such as they occur in multi-matching (e.g. for finding correspondences between keypoints in a collection of images [58]), or in clustering problems, where points are to be grouped according to some similarity criterion [43]. While there is a range of approaches based on spectral matrix decompositions, such as spectral permutation synchronization [58, 60, 54, 8] or spectral clustering [69], existing procedures have the downside that they require that the universe size (or number of clusters) k is known, and additionally require post-processing to find binary solutions representing discrete assignments.

Our method circumvents these shortcomings by using a similarity matrix to group vectors on the $(k-1)$-dimensional non-negative sphere, where k is an

(arbitrarily loose) upper bound for the universe size (i.e. the total number of unique elements across all objects in multi-matching problems, or the number of clusters in clustering problems). Our procedure shares resemblance with power iterations and orthogonal iterations for eigenspace computation [35], but optimizes over non-negative matrix *rows* with unit length (instead of orthogonal *columns*). In addition, we automatically determine a single real-valued parameter that is (indirectly) related to the (unknown) integer number k, which we address by utilizing specific properties that directly arise from our novel formulation. We summarize our main contributions as follows:

- Our formulation does not require the knowledge of k. Instead, we simply require to select an (arbitrarily loose) upper bound of k, which corresponds to the number of columns of our matrix optimization variable. We emphasize that it is not possible to choose a too large upper bound that would impair the result quality.
- Opposed to related existing methods, our method does not require any additional post-processing in order to find discrete assignments.
- We show that our procedure leads to state-of-the-art results in diverse multi-matching and clustering problems.

2   Related Work

In the following we provide an overview of the relevance of spectral methods for multi-matching and clustering.

Multi-matching and permutation synchronization. Multi-matching refers to the problem of finding correspondences across a collection of objects (e.g. correspondences between keypoints in images, or between vertices of meshes, etc.). Many approaches are based on permutation synchronization [58, 60, 54, 41, 14, 10, 39, 13, 8], which is a procedure that establishes cycle consistency in the set of pairwise matchings. The main idea is to represent pairwise matchings in a large block matrix, and then use low-rank matrix factorization (e.g. via spectral decompositions, convex or non-convex optimization) to extract cycle-consistent matchings. Similarly, analogous strategies have been used for other types of pairwise transformations, such as functional maps [40, 33], or spatial transformations [9, 4]. While permutation synchronization can be interpreted as a multi-object version of the linear assignment problem [56], respective approaches generally do not consider pairwise terms and thereby are unable to take into account geometric consistency. Instead, to consider geometric consistency, a multi-object version of the quadratic assignment problem [48, 49] can be considered, which, however, leads to an objective function that is a fourth-order polynomial. The latter was successfully addressed based on a higher-order projected power iteration method [11]. All of the mentioned approaches require that the universe size is known (or accurately estimated). In contrast, in this work we propose an approach that does not rely on this information.

Spectral clustering and related work. Clustering is well studied and thousands of algorithms have appeared [43] since the k-means algorithm was in-

troduced over sixty years ago [64]. Spectral clustering was first developed in the context of graph partitioning [30], and a large number of related approaches have been proposed [61, 26, 46, 57, 25, 5, 76, 53]. Essentially, in spectral clustering the intractable NP-hard clustering problem [28] is "relaxed" to a tractable eigenvector problem, whose solution is then eventually rounded to approximate a solution of the original problem. Typically, pairwise distances between data points are computed first, which are then represented as a distance matrix, i.e. an object that reflects dissimilarities between data points. Subsequently, a kernel function is applied to the distance matrix [74, 75, 70, 71] in order to convert it to a similarity matrix. Kernel functions range from fixed Gaussians [57] to adaptive approaches [71, 44]. Once a suitable similarity matrix is obtained, eigenvectors are computed as a low-dimensional embedding of the similarity matrix, which is then used for clustering. A common way to compute spectral clustering solutions is via power iterations or orthogonal iterations [35]. Spectral formulations furthermore appear as relaxations to normalized cut and min-cut formulations for clustering [16, 65]. There is also a connection to kernel-based k-means [23]. Recently there have been several works that incorporate learning of balance parameters in the spectral clustering formulation [17, 18, 59].

Spherical clustering refers to the clustering of points distributed over the (unit) sphere. Two common methods for spherical clustering are spherical k-means [24] and online spherical k-means [77]. The k-means clustering method [64, 29] is a special case of a Gaussian mixture model in which all priors are equal for the components, and all parameters are equal for the concentrations. The spherical k-means methods share analogous properties with respect to mixture models based on the von Mises-Fisher distribution on the unit sphere [7, 6], which uses cosine-functions of angular differences. This transitions into generative model-based (or parametric models) for spherical clustering, for which many approaches have been suggested [78, 27, 47, 12, 63]. In this context the Expectation Maximization algorithm is commonly used [6, 36]. While spherical clustering approaches have the objective to cluster points distributed over the sphere, in our approach we consider the clustering of points in any space. To this end, we consider a conditional power iteration method [31, 42] that iteratively finds points on the non-negative sphere in order to separate them into clusters.

Generalizations of power iterations. Due to its simplicity and its high practical value, there have been a diverse range of generalizations of the power iteration method. Among them are the higher-order power method [22], the incorporation of additional penalties [45], a variant for nonlinear eigenproblems [37], a max-pooling-based variant [21], tensor power iterations [62], coordinate-wise power iterations [52], projected power iterations [19], higher-order projected power iterations [11], or a sparsity-inducing formulation [8]. Moreover, the power iteration algorithm can also be interpreted as a special case of the Frank-Wolfe algorithm [31, 42] applied to Rayleigh quotient optimization.

3   Preliminaries

We define the set of binary row-stochastic matrices with m rows and k columns as $\mathcal{P}_{m,k}$. Another description of $\mathcal{P}_{m,k}$ is that it comprises m-by-k matrices with rows equal to the canonical basis vectors in $\mathbb{R}^k$. Suppose a data matrix $X \in \mathbb{R}^{m \times k}$ is given, where the rows represent a total of m data points in k dimensions. A similarity matrix $W \in \mathbb{R}^{m \times m}$ for X is a non-negative symmetric matrix, where each element W_{ij} captures the similarity (in some appropriate sense) between the row i (i.e. data point i) and row j (i.e. data point j) of X. Larger values indicate higher similarity, and smaller values indicate lower similarity.

Consider the nonlinear least-squares optimization problem on the form

$$\underset{U \in \mathcal{P} \subset \mathcal{P}_{m,k}}{\arg\min}\ f(\beta, W, U), \tag{1}$$

where $\mathcal{P}$ is a subset of $\mathcal{P}_{m,k}$ that depends on the type of problem we want to solve, and

$$f(\beta, W, U) = \|W - \beta U U^T\|_{\mathrm{F}}^2 = \mathrm{tr}(W^2) - 2\beta\,\mathrm{tr}(U^T W U) + \beta^2\,\mathrm{tr}(U U^T U U^T), \tag{2}$$

where $\beta > 0$ is some parameter of choice. Note that this formulation only involves W (and not a data matrix X), which could have been obtained without using a data matrix X in applications such as graph clustering or permutation synchronization.

As mentioned, the choice of $\mathcal{P}$ in (1) depends on the type of problem considered. In clustering, $\mathcal{P}$ is equal to $\mathcal{P}_{m,k}$. In multi-matching, $\mathcal{P}$ has additional structure. In such cases $\mathcal{P}$ is the set of matrices on the form $U = [U_1^T, U_2^T, \ldots, U_q^T]^T$, where $q < m$ and each U_i is a partial permutation matrix, i.e. $U_i \in \mathbb{P}_{m_i q}$ with

$$\mathbb{P}_{m_i q} := \{X \in \{0,1\}^{m_i \times k} : X\mathbf{1}_k \leq \mathbf{1}_{m_i},\ \mathbf{1}_{m_i}^T X \leq \mathbf{1}_k^T\} \quad \text{and} \quad \sum_{i=1}^{q} m_i = m. \tag{3}$$

3.1   Non-Negative Spherical Relaxation

We revisit equation (2) and note that for $U \in \mathcal{P}_{m,k}$ it holds that

$$\mathrm{tr}(U U^T U U^T) = \mathrm{tr}((U^T U)(U^T U)) = \mathrm{tr}(U^T \mathbf{1}_{m,m} U), \tag{4}$$

where $\mathbf{1}_{m,m}$ is the matrix in $\mathbb{R}^{m \times m}$ whose elements are all 1. This means that minimization of f in (2) over $\mathcal{P}_{m,k}$ may equivalently be expressed as maximization of

$$2\beta\,\mathrm{tr}(U^T W U) - \beta^2\,\mathrm{tr}(U^T \mathbf{1}_{m,m} U) \tag{5}$$

over $\mathcal{P}_{m,k}$, since the constant term in (2) can be ignored. We introduce the quadratic form formulation as

$$\underset{U \in \mathcal{P}_{m,k}}{\arg\max}\, g(\alpha, W, U), \tag{6}$$

where
$$g(\alpha, W, U) = (1-\alpha)\operatorname{tr}(U^T W U) - \alpha \operatorname{tr}(U^T \mathbf{1}_{m,m} U). \tag{7}$$

We see that (6) is equivalent to maximization of (5) by setting $\alpha = \beta^2/(\beta^2 + 2\beta) > 0$. The maximization of (5) is in turn equivalent to minimization of (2), as stated above. Now we define its **non-negative spherical relaxation** as

$$\underset{U \in \operatorname{Sp}^+_{m,k}}{\arg\max}\, g(\alpha, W, U), \tag{8}$$

where $\operatorname{Sp}^+_{m,k} = \{[u_1^T, u_2^T, \ldots, u_m^T]^T \in \mathbb{R}^{m \times k} : u_i \geq 0 \text{ and } u_i u_i^T = 1 \,\forall\, i\}$, i.e. each row u_i of a matrix U in $\operatorname{Sp}^+_{m,k}$ has non-negative elements and unit length (a point on the non-negative part of the unit sphere). An illustration of this relaxation is provided in Fig. 1 for $m = 3$.
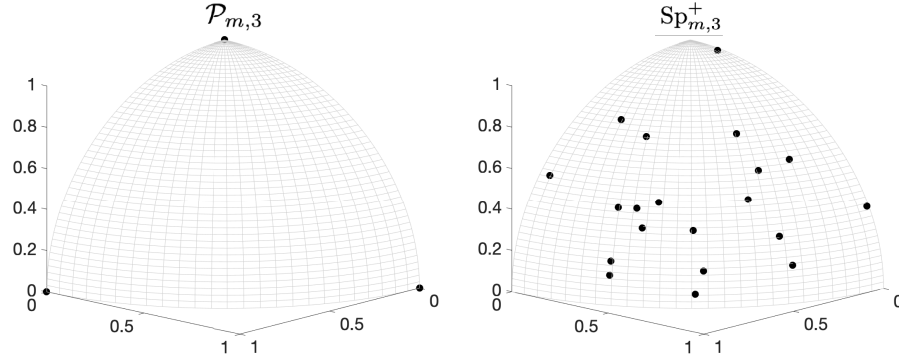
Fig. 1. Left: points in the set $\mathcal{P}_{m,3}$ are corner points on the non-negative sphere. **Right:** points in the relaxed set $\operatorname{Sp}^+_{m,3}$ lie on the non-negative sphere.

3.2 Conditional Power Iteration

We define two operations for re-balancing the similarity matrix W before used in the optimization problem, the **normalization**

$$W \leftarrow DWD, \text{ where } D = \sqrt{m}\,\operatorname{diag}(W\mathbf{1}_m)^{-\frac{1}{2}}; \tag{9}$$

and the κ-**shift**

$$W \leftarrow W + \kappa\mathbf{1}_{m,m}, \tag{10}$$

for a $\kappa > 0$. There is an efficient (local) optimization method for (8) (with the re-balanced W) referred to as *conditional power iterations*, see Algorithm 1. In the algorithm we use the matrix

$$V(\alpha) = \delta(\alpha)\mathbf{I}_m + (1-\alpha)W - \alpha\mathbf{1}_{m,m}, \tag{11}$$

where $\delta(\alpha) \geq 0$ is some function of α such that $V(\alpha)$ is positive definite. We will throughout assume δ is given by minus the smallest eigenvalue of $(1 - \alpha)W - \alpha \mathbf{1}_{m,m}$ plus some additional positive epsilon.

Input: $U_0 \in \mathcal{S}$
Output: U
Initialise: $t \leftarrow 0$
repeat
$\quad U_{t+1} \leftarrow \underset{\tilde{U} \in \mathcal{S}}{\arg\max} \ \mathrm{tr}(U_t^T V(\alpha)\tilde{U})$
$\quad t \leftarrow t+1$
until *convergence*
$U \leftarrow U_t$

Algorithm 1: Conditional power iterations

The algorithm resembles the conditional gradient method (also known as the Frank-Wolfe method [42]), for objectives of quadratic form, where we allow for $\mathcal{S}$ to be any (not necessarily convex) compact subset of $\mathbb{R}^{m \times k}$. In the following, we summarize convergence properties of Algorithm (1).

Proposition 1 *The following holds for the conditional power iterations in Algorithm* (1).
1. $tr(U_{t+1}^T V(\alpha)U_{t+1}) \geq tr(U_t^T V(\alpha)U_t)$.
2. There is $\delta > 0$, s.t. $tr(U_t^T V(\alpha)U_t) \to \delta$ as $t \to \infty$, i.e. the objective converges.
3. For any $\epsilon > 0$ there is a $T \in \mathbb{N}$ (as a function of ϵ) s.t. $\sum_{t=T}^{\infty} \|U_{t+1} - U_t\|_F^2 \leq \epsilon$.

A proof of Proposition 1 is provided in the appendix. For manifolds one can, in general, further strengthen condition 3: the solution converges to a critical point, and converges almost everywhere to a local maximum [51].

Using Algorithm 1: when $\mathcal{S} = \mathrm{Sp}_{m,k}^+$, we immediately note that the maximizers of (8) are equal to those of $\mathrm{tr}(U^T V(\alpha)U)$. This can be observed by noting that $\mathrm{tr}(U^T(\delta(\alpha)\mathbf{I}_m)U) = \delta(\alpha)\mathrm{tr}(UU^T) = \delta(\alpha)\mathrm{tr}(U^T U) = \delta(\alpha)m$ is constant for $U \in \mathrm{Sp}_{m,k}^+$.

The main update in Algorithm 1 reads

$$U_{t+1} \leftarrow \underset{\tilde{U} \in \mathrm{Sp}_{m,k}^+}{\arg\max} \mathrm{tr}(U_t^T V(\alpha)\tilde{U}). \qquad (12)$$

Since the rows of matrices in the set $\mathrm{Sp}_{m,k}^+$ are independent, the maximization in (12) can be solved by performing maximizing for each column of $U_t^T V$ individually. An efficient solution to this per-column maximization problem is presented in Algorithm 2.

In the following we assume that $\mathcal{S} = \mathrm{Sp}_{m,k}^+$ and that the maximization problem in (12) is solved using Algorithm 2. Given the output U to Algorithm 1, we subsequently perform a rounding to obtain a matrix $\tilde{U} \in \mathrm{Sp}_{m,k}^+ \cap \mathcal{P}_{m,k}$, from $U \in \mathrm{Sp}_{m,k}^+$.

Input: $x \in \mathbb{R}^k$
Output: $y \in \mathrm{Sp}_{1,k}^+$
if $\max_i x_i \leq 0$ **then**
    $y \leftarrow \mathbf{0}_k$
    $i = \arg\max_i x_i$ // `select any solution if not unique`
    $y_i \leftarrow 1$
else
    $y \leftarrow \max(x, 0)$ (element-wise maximization)
    $y \leftarrow \frac{y}{\|y\|_2}$
end

Algorithm 2: Projection onto non-negative sphere

4 Universe-Size-Free Procedure

In Sec. 3.2 we provided a strategy for local optimization of (8) for a given α. The problem is that this parameter α is not given. It does not make sense to, for example, choose $\alpha = 1$, since then the optimal solution U^* does not depend on W. The main approach in this section is to select a "good" α by using the convergence rate of Algorithm 1 for different choices of α. Regarding the choice of k, our only requirement in the proposed procedure is that it is sufficiently large.

4.1 Description of the method

First the matrix W is normalized, see (9), and consecutively κ-shifted, see (10), (with some small $\kappa > 0$). The procedure is then to run two consecutive algorithms: α-sweep, see Algorithm 3 below, and α-pick, see Algorithm 4 below. The method is illustrated on a high level for a multi-matching problem in Fig. 2. The main intuition behind the algorithms is illustrated for a clustering problem in Fig. 3.

α**-sweep:** Initially we choose $\alpha = 1$ and k to some sufficiently large positive integer. We choose U_0 to the unique matrix $K \in \mathbb{R}^{m,k}$ for which there is a matrix R and integer $\bar{m} > m$ such that $[K^T, R^T]^T = [I_k, I_k, \ldots, I_k]^T \in \mathbb{R}^{\bar{m} \times k}$. In other words, U_0 comprises the upper part (the first m rows) of a tall matrix with repeated identity matrix blocks. Then we decrease α (from 1) to 0 with a specified step-size ϵ_α. For each α considered on the path from 1 to 0, we run the conditional power iteration Algorithm 1 (outlined in Section 3.2) for a few fixed number of steps to update U. Thus, when we update α with the step size ϵ_α, we use the final U obtained from for the previous α as initialization. For each α considered, we compute the difference between the objective function with U from the last iteration and the objective function with U from the second last iteration. We call this difference $\eta(\alpha)$ and it is a measure of how fast the objective converges for each considered α.

W

1	0	0	0	0	1	1	0	0
0	1	1	0	1	0	0	1	0
0	1	1	0	1	0	0	0	1
0	0	0	1	0	1	1	0	0
0	1	1	0	1	0	1	0	0
1	0	0	1	0	1	0	0	1
1	0	0	1	1	0	1	0	0
0	1	0	0	0	0	0	1	0
0	0	1	0	0	1	0	0	1

α-sweep + α-pick

U

0.7307	0	0	0.5081	0	0.4560
0	0.6663	0	0	0.7457	0
0	0.0334	0.4911	0	0.8705	0
0.7307	0	0	0.5081	0	0.4560
0	0.0383	0	0	0.9993	0
0.6527	0	0.4495	0.4539	0	0.4074
0.7307	0	0	0.5081	0	0.4560
0	0.9987	0	0	0.0517	0
0	0	1.0000	0	0	0

U_0^T

1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0	0

Solve LAP for the four blocks

$\tilde{U}\tilde{U}^T$

1	0	0	1	0	1	1	0	0
0	1	1	0	1	0	0	0	0
0	1	1	0	1	0	0	0	0
1	0	0	1	0	1	1	0	0
0	1	1	0	1	0	0	0	0
1	0	0	1	0	1	1	0	0
1	0	0	1	0	1	1	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1

$\tilde{U}$

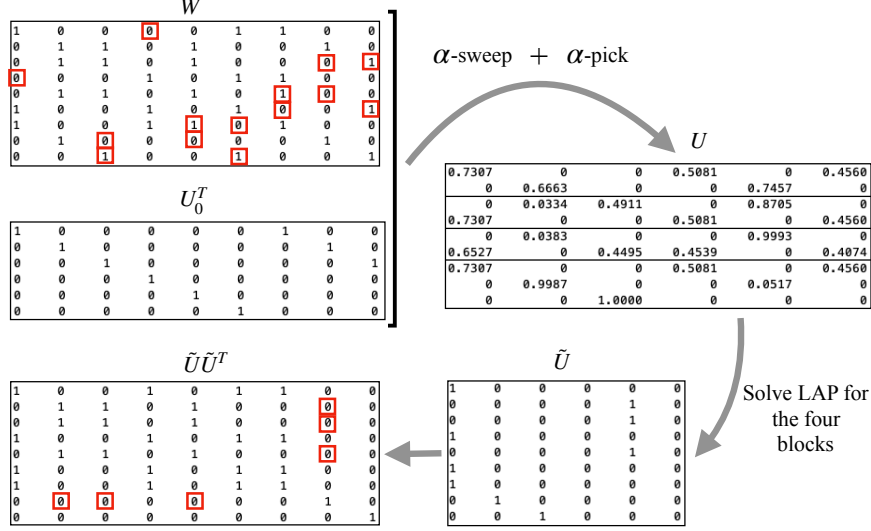| 1 | 0 | 0 | 0 | 0 | 0 |
0	0	0	0	1	0
0	0	0	0	1	0
1	0	0	0	0	0
0	0	0	0	1	0
1	0	0	0	0	0
1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0

Fig. 2. Illustration of the method for a small multi-matching problem. Universe size (number of unique points) is $k^* = 3$, number of images is 4, each containing either 2 or 3 point points in correspondence with points in other images. The sum of all points (in all objects) is $m = 9$. We choose $k = 6$ for the method. There are two inputs to the method: the noisy similarity matrix W, where erroneous entries are marked by red; and the matrix U_0. The main procedure α-sweep (Algorithm 3) and α-pick (Algorithm 4) produces a matrix $U \in \mathrm{Sp}_{9,6}^+$. The linear assignment problem (LAP) is solved for each of the four blocks of U to produce the matrix $\tilde{U}$, which defines the correspondences.

α**-pick:** This proceedure is illustrated in the right sub-figure of Fig. 3. We find the first local maximum of η from the right, referred to as "Right max" in the figure. Then we find the first local maximum of η from the left, referred to as "Left max" in the figure. Then we pick α^* as the corresponding α for the minimum η between "Left max" and "Right max", shown with a green dot in the figure. In the figure we also show the F-score (blue line), which is the harmonic mean of precision and recall. The black dashed line illustrates the corresponding F-score for the picked α^*. The procedure for finding the index of the value α^* in the α-vector is formally described in Algorithm 4.

4.2 Brief Motivation

We begin by addressing initialization of U_0 and why α is decreased from 1 to 0 in the α-sweep algorithm. For $\alpha = 1$, one can show that K is an optimal solution to (6). Optimal solutions for $\alpha = 0$ comprise matrices with all rows equal. But, the optimal solutions for $\alpha = 0$ comprise a critical set for all $\alpha \in [0, 1]$. Hence the solution will not change when α is increased from 0 to 1, as we would always get the same trivial solutions when varying α from 0 to 1. The pink curve in Fig. 3

Input: W, k, ϵ_α, ϵ_η N k
Output: α_{tot}, η_{tot}, U_{tot}
Initialise: $\alpha \leftarrow 1$, $\alpha_{tot} \leftarrow [\,]$, $h_{tot} \leftarrow [\,]$, $U_0 \leftarrow K$, $U_{tot} \leftarrow U_0$
// outer-loop sweeps discretely over α
repeat
    $\alpha \leftarrow \alpha - \epsilon_\alpha$
    $\lambda_{\min} \leftarrow \min(\mathrm{eig}((1-\alpha)W - \alpha\mathbf{1}_{m,m}))$ // smallest eigenvalue
    $V \leftarrow (\lambda_{\min} + \epsilon_\eta)\mathbf{I}_m + (1-\alpha)W - \alpha\mathbf{1}_{m,m}$
    $i \leftarrow 0$,
    // inner-loop uses Algorithm 1
    repeat
        $U_{t+1} \leftarrow \underset{\tilde{U}\in\mathrm{Sp}^+_{m,k}}{\arg\max}\ \mathrm{tr}(U_t^T V(\alpha)\tilde{U})$
        $i \leftarrow i + 1$
        $t \leftarrow t + 1$
    until $i = N$
    $\alpha_{\mathrm{tot}} \leftarrow [\alpha_{\mathrm{tot}}, \alpha]$
    $U_{tot} \leftarrow [U_{tot}|U_t]$
    $\eta_{\mathrm{tot}} \leftarrow [\eta_{\mathrm{tot}}, |g(\alpha, W, U_t) - g(\alpha, W, U_{t-1})|]$
until $\alpha - \epsilon_\alpha < 0$

Algorithm 3: α-sweep

Input: α_{tot} // from Algorithm 4
Output: i*
Initialise: $i_r \leftarrow 0$, $i_l \leftarrow \mathrm{length}(\eta_{\mathrm{tot}})$ // length of the vector
repeat
    $i_r \leftarrow i_r + 1$
until $\eta_{tot}(i_r + 1) < \eta_{tot}(i_r)$ // value at subsequent index in vector is
    smaller

repeat
    $i_l \leftarrow i_l - 1$
until $\eta_{tot}(i_l - 1) < \eta_{tot}(i_l)$ // value at previous index in vector is
    smaller

i* $\leftarrow i_r + \mathrm{minIndex}(\eta_{\mathrm{tot}}(i_r : i_l))$ // index of minimum value between the
    two maxima

Algorithm 4: α-pick

would be replaced by a straight line. Now, with a κ-shift, the optimal solutions for $\alpha > 0$ small enough are the matrices with all rows equal. The objective will be the same when varying α in that region. Thus the η-values will be close to zero in that region, see Fig. 3 when α is smaller than 0.43. If the similarity matrix is "free of noise", there is a region of α-values for which the optimal solution is retrieved and η on that interval should be close to 0 (convergence of objective is assured by Proposition 1). The objective value, however, is different from that when $\alpha = 0$ or $\alpha = 1$. By varying α between 1 and 0, one should first
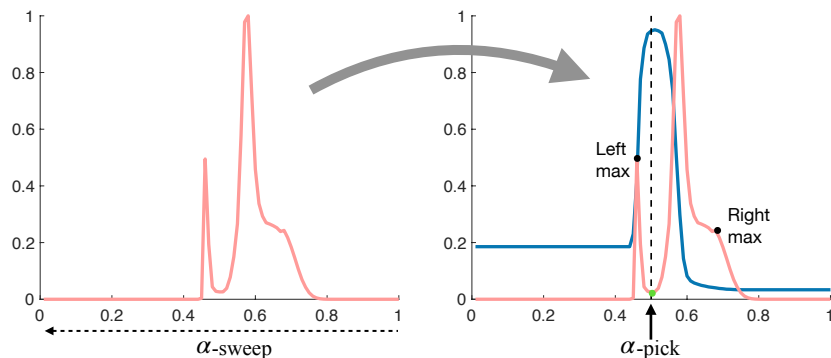
Fig. 3. The method is used for a sample clustering problem for illustration. **Left**: the computed measure of convergence rate η (pink) as a function of α for Algorithm 3 (i.e. α-sweep). **Right**: an α-value is picked for the (pink) η-curve using Algorithm 4 (i.e. α-pick). For illustration, the F-score for the solution for each α is shown in blue and the dashed line shows the F-score value of the solution corresponding to the picked α.

experience an increase in η, then a local minimum, then an increase, and then a local minimum again.

5   Experiments

5.1   Multi-Image Matching

Multi-image matching refers to the problem of establishing correspondences between given keypoints in a collection of images [58, 79, 73, 68, 66, 11, 14, 13]. Our procedure for these type of problems is illustrated with an example in Fig. 2. As rounding, we solve q (partial) linear assignment problems for the q blocks in the output matrix U.

CMU house image sequence. We begin by considering the CMU house image sequence [1], from which we generate a range of multi-image matching problems following the procedure of [58]. We compare our method with **MatchEig** [54], **Spectral** [58], **SparseStiefelOpt** [8], **MatchALS** [79], **NmfSync** [10]. In this experiment for **Our** we use $k = 100$, $\epsilon_\alpha = 0.01$, $N = 5$, $\epsilon_\eta = 0$, $\kappa = 0.01$. We present results in Fig. 4 in terms of F-score, i.e. harmonic mean of precision and recall.

WILLOW-ObjectClass data set. Next we follow the procedure in [2] to perform multi-image matching for the WILLOW-ObjectClass data set [3]. Sample images are shown in Fig. 5. We compare **Our** method in terms of F-score with **PG** [72], **Spectral** [58], **MatchLift** [20], **MatchALS** [79], **NmfSync** [10], and **SparseStiefelOpt** [8]. Except **Our**, all methods use the known ground-truth universe size for the multi-matching. For **Our** we use $k = 100$, $\epsilon_\alpha = 0.01$,
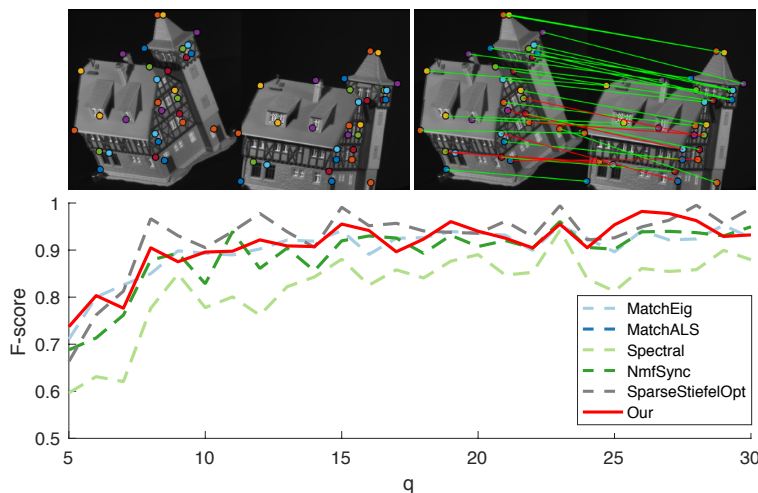
Fig. 4. Top left: two images from the CMU house image sequence. **Top right**: example of correct vs. incorrect pairwise matchings. Green lines show correct matchings, whereas red lines show incorrect matchings. **Bottom**: quantitative comparison of state-of-the-art multi-image matching methods in terms of the F-Score. All methods with dashed lines *use the ground-truth universe size for matching*, whereas **Our** (solid) does not.

$N = 20$, $\epsilon_\eta = 0$, $\kappa = 0$. Results are shown in Table 1, where it can be seen that our method is competitive without knowing the universe size.
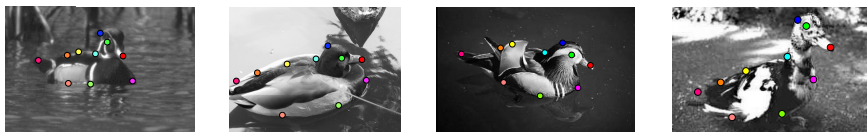

Fig. 5. Sample images with corresponding ground truth feature correspondences from the WILLOW-ObjectClass data set.

Multi-matching with partial observations. We continue with experiments on simulated (synthetic) problems where only a subset of features are observed for each of the q objects (corresponding to images). This means that the number of points m_i is not necessarily equal to m_j for $i \neq j$. We conduct a series of experiments where we generate problems using the following parameters. q: number of objects; d: ground truth universe size; ρ: probability that a feature is observed (for $\rho = 0$ it holds $m_i = 0$ for all i, and for $\rho = 1$ it holds $m_i = d$ for all i); σ: this parameter specifies the degree of error in the similarity matrix W. It is the portion of randomly shuffled ground truth matchings for pairs of objects.

	Car	Duck	Face	Motorbike	Winebottle	universe size
Our	0.985	0.935	1	0.983	1	estimated
PG	0.990	0.925	1	1	1	ground truth
Spectral	0.990	0.929	1	0.975	1	ground truth
Matchlift	0.990	0.925	1	0.976	1	ground truth
MatchALS	0.980	0.921	1	0.960	1	ground truth
NmfSync	0.990	0.925	1	0.975	1	ground truth
SparseStiefelOpt	0.980	0.861	1	0.975	1	ground truth

Table 1. WILLOW dataset F-score comparison for different methods. Horizontally: image data sets. Vertically: methods.

An evaluation for different parameter settings is shown in Fig. 6, where average F-scores over 50 problems are presented for each parameter setting.
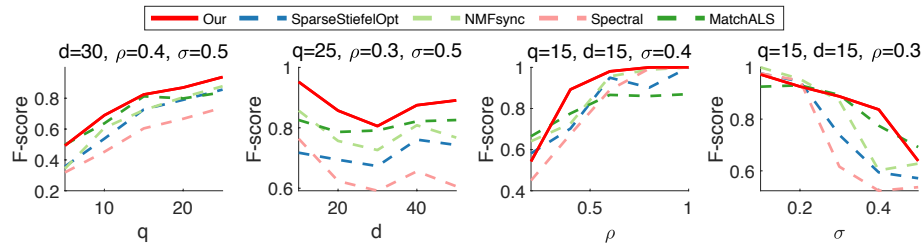

Fig. 6. F-score ($\uparrow$) comparison of **Our** (Red) and four other methods for synchronization of partial permutation matrices. Methods with dashed lines use ground-truth universe size (i.e. d) in the matching, whereas the method with solid line (**Our**) does not. In each setting three of the parameters (q, d, ρ, σ) are constant, whereas the fourth is varying along the x-axis. The y-axis represents the average F-score over 50 problem instances for each choice of the varying parameter. All methods except **Our** use the universe size for the synchronization.

5.2   Clustering

Random binary similarity matrices. We begin the clustering evaluation by considering binary similarity matrices as input, which can be seen as the adjacency matrix of an unweighted graph, and hence the setting is in line with graph clustering. The similarity matrix represents matchings between m points randomly assigned to one of the k^* clusters. In this context, two parameters capture the noise level of the similaity matrices considered: $0 \leq \rho \leq 1$ and $0 \leq \nu \leq 1$, where the former captures the degree of randomness in the elements, and the latter captures the degree of missing data (i.e. the degree of connectivity of the graph). Higher values are more challenging.
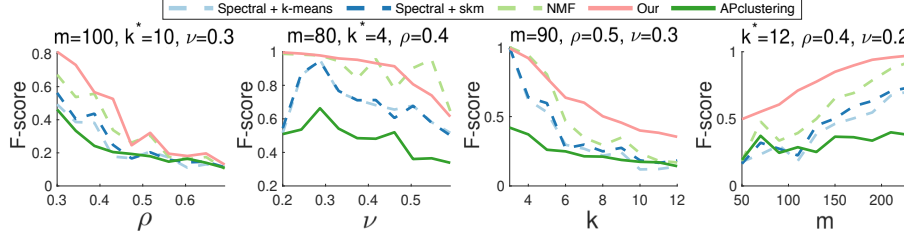
Fig. 7. F-score ($\uparrow$) comparison of **Our** (pink) and five other clustering methods. In each setting three of the parameters (m, k^*, ρ, ν) are constant, whereas the fourth is varying along the x-axis. The y-axis represents the average F-score over 50 problem instances for each choice of the varying parameter. Methods with solid lines assume no knowledge of k^*, whereas methods with dashed lines use k^* as the number of clusters to determine.

We compare our method to existing clustering methods in this setting. Results are provided for four different settings in Fig. 7. In each of the four settings one parameter varies, whereas the others are fixed. The average F-scores are shown for 50 simulations for each choice of the varying parameter. The algorithms we compare are the following ones. **Our**: Algorithm 3 and Algorithm 4; **APclustering:** Affinity propagation clustering [15, 32]; **NMF**: Non-negative matrix factorization of W [34] with multiplicative updates [50] and simple rounding; **Spectral + kmeans**: Spectral clustering with k-means clustering [55] using rounding as in [57]; **Spectral + skm**: same as **Spectral + kmeans** except rounding with spherical k-means clustering [38]. Of the compared methods, only **Our** and **APclustering** (solid lines) do not use k^* explicitly in the clustering. Regarding **Our**, we have chosen the following parameters in all experiments in Fig. 7: $k = 100$, $\epsilon_\alpha = 0.01$, $N = 20$ $\epsilon_\eta = 0$, $\kappa = 0$.

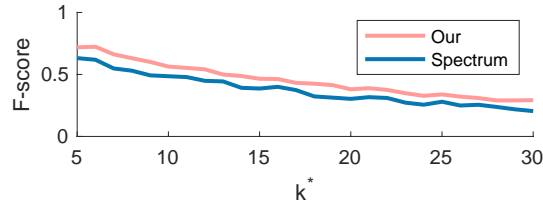**Gaussian mixture models.** Here we consider data from 2D Gaussian mixture

Fig. 8. Quantitative comparison in terms of the mean F-Score ($\uparrow$) between our method and **Spectrum** for varying number of mixture components k^*.

models (GMM), where the number of mixture components is k^* and the m rows of a data matrix X correspond to m observations. The generated problems are

challenging with highly overlapping clusters and different scales. We used the adaptive kernel proposed by [44] to construct a similarity matrix W and used their method **Spectrum** provided in [55, 67] as comparison method, which is a self-tuning spectral clustering method using the adaptive kernel, followed by a procedure for selecting the appropriate number of clusters, as well as a final step of Gaussian mixture model fitting. Results are shown in Fig. 8, where we vary the number of k^* ground truth clusters. For each choice of k^*, we randomly generate 30 GMMs. In this generation of models, the mean distance between cluster centers is 1 and mean variance within clusters is 0.7. For each GMM we generate 200 sample points to create a problem for evaluation.

5.3   Discussion and Limitations

Once we know the index i^* from Algorithm 4, we can run Algorithm 3 again for i^* many steps in the outer loop and then terminate. An alternative is to save the cluster assignments after each iteration in a vector. For clustering problems, the rounding procedure does not comprise a computational bottleneck (its computational complexity is $\mathcal{O}(km)$). Thus it can be added in each iteration of the outer loop to observe how the number clusters vary with varying α. From a computational perspective the per-iteration complexity of Algorithm 1 is equivalent to that of the orthogonal iteration algorithm for computing eigenvectors. The computational time is reasonable for moderate problem sizes. For example, the largest computational time observed for all problems used for Fig. 7 in Section 5, was 1.34 seconds on a 2,7 GHz Intel Core i7 MacBook Pro. However, the proposed method suffers from some shortcomings from a computational perspective. Increasing N and decreasing ϵ_α results in more iterations (the total number being N times the number of discretization steps of α). The optimal initialization with $U_0 = K$ helps to reduce the N needed to get a good solution. However, the eigenvalue computation in each iteration could provide a computational bottleneck. This can be alleviated by using more efficient approximation methods and additionally use a previously computed eigenvector as initialization for the next iteration in the outer loop.

6   Conclusion

In this work have presented a novel non-negative spherical relaxation for multi-matching and clustering without the knowledge of universe size. An efficient and easy-to-implement conditional power iterations method is used for optimization, where a continuous parameter is selected by comparing convergence rates for values thereof. The solution for this parameter-choice is subsequently rounded to determine the assignments. The method is competitive against state-of-the art methods that assume knowledge of the universe size.

References

1. CMU/VASC image database. http://www.cs.cmu.edu/afs/cs/project/vision/vasc/idb/www/html/motion/
2. MultiObject-Matching. https://github.com/qianqianwang68/multiway
3. WILLOW-objectclass. https://www.di.ens.fr/willow/research/graphlearning/
4. Arrigoni, F., Rossi, B., Fusiello, A.: Spectral synchronization of multiple views in se (3). SIAM Journal on Imaging Sciences **9**(4), 1963–1990 (2016)
5. Bach, F.R., Jordan, M.I.: Learning spectral clustering, with application to speech separation. The Journal of Machine Learning Research **7**, 1963–2001 (2006)
6. Banerjee, A., Dhillon, I., Ghosh, J., Sra, S.: Clustering on hyperspheres using expectation maximization. Department of Computer Sciences, University of Texas, Tech. Rep. TR-03-07 (2003)
7. Banerjee, A., Ghosh, J.: Frequency sensitive competitive learning for clustering on high-dimensional hyperspheres. In: Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290). vol. 2, pp. 1590–1595. IEEE (2002)
8. Bernard, F., Cremers, D., Thunberg, J.: Sparse quadratic optimisation over the stiefel manifold with application to permutation synchronisation. In: NeurIPS (2021)
9. Bernard, F., Thunberg, J., Gemmar, P., Hertel, F., Husch, A., Goncalves, J.: A solution for multi-alignment by transformation synchronisation. In: CVPR (2015)
10. Bernard, F., Thunberg, J., Goncalves, J., Theobalt, C.: Synchronisation of partial multi-matchings via non-negative factorisations. Pattern Recognition **92**, 146–155 (2019)
11. Bernard, F., Thunberg, J., Swoboda, P., Theobalt, C.: Hippi: Higher-order projected power iterations for scalable multi-matching. In: CVPR (2019)
12. Bijral, A.S., Breitenbach, M., Grudic, G.: Mixture of watson distributions: a generative model for hyperspherical embeddings. In: Artificial Intelligence and Statistics. pp. 35–42. PMLR (2007)
13. Birdal, T., Golyanik, V., Theobalt, C., Guibas, L.J.: Quantum permutation synchronization. In: CVPR (2021)
14. Birdal, T., Simsekli, U.: Probabilistic permutation synchronization using the riemannian structure of the birkhoff polytope. In: CVPR (2019)
15. Bodenhofer, U., Kothmeier, A., Hochreiter, S.: Apcluster: an r package for affinity propagation clustering. Bioinformatics **27**(17), 2463–2464 (2011)
16. Chan, P.K., Schlag, F.: Spectral k-way ratio-cut partitioning and clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **13**(9) (1994)
17. Chen, X., Hong, W., Nie, F., Huang, J.Z., Shen, L.: Enhanced balanced min cut. International Journal of Computer Vision **128**(7) (2020)
18. Chen, X., Zhexue Haung, J., Nie, F., Chen, R., Wu, Q.: A self-balanced min-cut algorithm for image clustering. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2061–2069 (2017)
19. Chen, Y., Candès, E.J.: The projected power method: An efficient algorithm for joint alignment from pairwise differences. Communications on Pure and Applied Mathematics **71**(8), 1648–1714 (2018)
20. Chen, Y., Guibas, L.J., Huang, Q.X.: Near-Optimal Joint Object Matching via Convex Relaxation. In: ICML (2014)

21. Cho, M., Sun, J., Duchenne, O., Ponce, J.: Finding matches in a haystack: A max-pooling strategy for graph matching in the presence of outliers. In: CVPR (2014)
22. De Lathauwer, L., Comon, P., De Moor, B., Vandewalle, J.: Higher-order power method. Nonlinear Theory and its Applications, NOLTA95 **1** (1995)
23. Dhillon, I.S., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 551–556 (2004)
24. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Machine learning **42**(1), 143–175 (2001)
25. Ding, C., He, X., Simon, H.D.: On the equivalence of nonnegative matrix factorization and spectral clustering. In: Proceedings of the 2005 SIAM international conference on data mining. pp. 606–610. SIAM (2005)
26. Donath, W.E., Hoffman, A.J.: Lower bounds for the partitioning of graphs. In: Selected Papers Of Alan J Hoffman: With Commentary, pp. 437–442. World Scientific (2003)
27. Dortet-Bernadet, J.L., Wicker, N.: Model-based clustering on the unit sphere with an illustration using gene expression profiles. Biostatistics **9**(1), 66–80 (2008)
28. Drineas, P., Frieze, A., Kannan, R., Vempala, S., Vinay, V.: Clustering large graphs via the singular value decomposition. Machine learning **56**(1), 9–33 (2004)
29. Duda, R.O., Hart, P.E., et al.: Pattern classification and scene analysis, vol. 3. Wiley New York (1973)
30. Fiedler, M.: Algebraic connectivity of graphs. Czechoslovak mathematical journal **23**(2), 298–305 (1973)
31. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval research logistics quarterly **3**(1-2), 95–110 (1956)
32. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. science **315**(5814), 972–976 (2007)
33. Gao, M., Lähner, Z., Thunberg, J., Cremers, D., Bernard, F.: Isometric multi-shape matching. In: CVPR (2021)
34. Gaujoux, R., Seoighe, C.: A flexible r package for nonnegative matrix factorization. BMC Bioinformatics **11**(1), 367 (2010). https://doi.org/10.1186/1471-2105-11-367, http://www.biomedcentral.com/1471-2105/11/367
35. Golub, G.H., Van Loan, C.F.: Matrix computations, vol. 4. JHU press (2013)
36. Golzy, M., Markatou, M., Shivram, A.: Algorithms for clustering on the sphere: Advances & applications. In: Proceedings of the World Congress on Engineering and Computer Science. vol. 1, pp. 420–425 (2016)
37. Hein, M., Bühler, T.: An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse pca. Advances in Neural Information Processing Systems **23**, 847–855 (2010)
38. Hornik, K., Feinerer, I., Kober, M., Buchta, C.: Spherical k-means clustering. Journal of Statistical Software **50**(10), 1–22 (2012). https://doi.org/10.18637/jss.v050.i10
39. Huang, Q., Liang, Z., Wang, H., Zuo, S., Bajaj, C.: Tensor maps for synchronizing heterogeneous shape collections. ACM Transactions on Graphics (TOG) **38**(4), 1–18 (2019)
40. Huang, R., Ren, J., Wonka, P., Ovsjanikov, M.: Consistent zoomout: Efficient spectral map synchronization. In: Computer Graphics Forum. vol. 39, pp. 265–278. Wiley Online Library (2020)
41. Huang, X., Liang, Z., Zhou, X., Xie, Y., Guibas, L.J., Huang, Q.: Learning transformation synchronization. In: CVPR (2019)

42. Jaggi, M.: Revisiting frank-wolfe: Projection-free sparse convex optimization. In: International Conference on Machine Learning. pp. 427–435. PMLR (2013)
43. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern recognition letters **31**(8), 651–666 (2010)
44. John, C.R., Watson, D., Barnes, M.R., Pitzalis, C., Lewis, M.J.: Spectrum: fast density-aware spectral clustering for single and multi-omic data. Bioinformatics **36**(4), 1159–1166 (2020)
45. Journée, M., Nesterov, Y., Richtárik, P., Sepulchre, R.: Generalized power method for sparse principal component analysis. Journal of Machine Learning Research **11**(2) (2010)
46. Kannan, R., Vempala, S., Vetta, A.: On clusterings: Good, bad and spectral. Journal of the ACM (JACM) **51**(3), 497–515 (2004)
47. Kent, J.T.: The fisher-bingham distribution on the sphere. Journal of the Royal Statistical Society: Series B (Methodological) **44**(1), 71–80 (1982)
48. Koopmans, T.C., Beckmann, M.: Assignment Problems and the Location of Economic Activities. Econometrica **25**(1), 53 (Jan 1957)
49. Lawler, E.L.: The quadratic assignment problem. Management science **9**(4), 586–599 (1963)
50. Lee, J.S., Lee, D.D., Choi, S., Park, K.S., Lee, D.S.: Non-negative matrix factorization of dynamic images in nuclear medicine. In: 2001 IEEE Nuclear Science Symposium Conference Record. vol. 4, pp. 2027–2030 (2001)
51. Lee, J.D., Panageas, I., Piliouras, G., Simchowitz, M., Jordan, M.I., Recht, B.: First-order methods almost always avoid strict saddle points. Mathematical programming **176**(1), 311–337 (2019)
52. Lei, Q., Zhong, K., Dhillon, I.S.: Coordinate-wise power method. In: Advances in Neural Information Processing Systems (2016)
53. Lu, H., Fu, Z., Shu, X.: Non-negative and sparse spectral clustering. Pattern Recognition (2014)
54. Maset, E., Arrigoni, F., Fusiello, A.: Practical and Efficient Multi-View Matching. In: ICCV (2017)
55. Michael Christoph, Stier, Q.: Fundamental clustering algorithms suite. SoftwareX **13**, 100642 (2021). https://doi.org/10.1016/j.softx.2020.100642, https://www.sciencedirect.com/science/article/pii/S2352711020303551
56. Munkres, J.: Algorithms for the Assignment and Transportation Problems. Journal of the Society for Industrial and Applied Mathematics **5**(1), 32–38 (Mar 1957)
57. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in neural information processing systems. pp. 849–856 (2002)
58. Pachauri, D., Kondor, R., Singh, V.: Solving the multi-way matching problem by permutation synchronization. In: NIPS (2013)
59. Shaham, U., Stanton, K., Li, H., Nadler, B., Basri, R., Kluger, Y.: Spectralnet: Spectral clustering using deep neural networks. arXiv preprint arXiv:1801.01587 (2018)
60. Shen, Y., Huang, Q., Srebro, N., Sanghavi, S.: Normalized Spectral Map Synchronization. In: NIPS (2016)
61. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on pattern analysis and machine intelligence **22**(8), 888–905 (2000)
62. Shi, X., Ling, H., Hu, W., Xing, J., Zhang, Y.: Tensor power iteration for multigraph matching. In: CVPR (2016)
63. Sra, S., Karp, D.: The multivariate watson distribution: Maximum-likelihood estimation and other aspects. Journal of Multivariate Analysis **114**, 256–269 (2013)

64. Steinhaus, H., et al.: Sur la division des corps matériels en parties. Bull. Acad. Polon. Sci **1**(804), 801 (1956)
65. Stella, X.Y., Shi, J.: Multiclass spectral clustering. In: Computer Vision, IEEE International Conference on. vol. 2, pp. 313–313. IEEE Computer Society (2003)
66. Swoboda, P., Kainmüller, D., Mokarian, A., Theobalt, C., Bernard, F.: A convex relaxation for multi-graph matching. In: CVPR (2019)
67. Thrun, M.C., Ultsch, A.: Clustering benchmark datasets exploiting the fundamental clustering problems. Data in brief **30**, 105501 (2020)
68. Tron, R., Zhou, X., Esteves, C., Daniilidis, K.: Fast multi-image matching via density-based clustering. In: ICCV (2017)
69. Von Luxburg, U.: A tutorial on spectral clustering. Statistics and computing **17**(4), 395–416 (2007)
70. Wang, B., Mezlini, A.M., Demir, F., Fiume, M., Tu, Z., Brudno, M., Haibe-Kains, B., Goldenberg, A.: Similarity network fusion for aggregating data types on a genomic scale. Nature methods **11**(3), 333–337 (2014)
71. Wang, B., Zhu, J., Pierson, E., Ramazzotti, D., Batzoglou, S.: Visualization and analysis of single-cell rna-seq data by kernel-based similarity learning. Nature methods **14**(4), 414–416 (2017)
72. Wang, Q., Zhou, X., Daniilidis, K.: Multi-Image Semantic Matching by Mining Consistent Features. In: CVPR (2018)
73. Yan, J., Yin, X.C., Lin, W., Deng, C., Zha, H., Yang, X.: A short survey of recent advances in graph matching. In: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval. pp. 167–174 (2016)
74. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering in: Advances in neural information processing systems (nips) (2004)
75. Zhang, X., Li, J., Yu, H.: Local density adaptive similarity measurement for spectral clustering. Pattern Recognition Letters **32**(2), 352–358 (2011)
76. Zhang, Z., Jordan, M.I.: Multiway spectral clustering: A margin-based perspective. Statistical science **23**(3), 383–403 (2008)
77. Zhong, S.: Efficient online spherical k-means clustering. In: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. vol. 5, pp. 3180–3185. IEEE (2005)
78. Zhong, S., Ghosh, J.: A comparative study of generative models for document clustering. In: Proceedings of the workshop on clustering high dimensional data and its applications in SIAM data mining conference. Citeseer (2003)
79. Zhou, X., Zhu, M., Daniilidis, K.: Multi-image matching via fast alternating minimization. In: ICCV (2015)

A   Proof of Proposition 1

1. It holds, by definition of U_{t+1} that **a)**

$$\text{tr}(U_t^T V(\alpha) U_{t+1}) \geq \text{tr}(U_t^T V(\alpha) U_t). \tag{13}$$

Moreover, since $V(\alpha)$ is positive definite, it holds that **b)**

$$0 \leq \text{tr}((U_{t+1} - U_t)^T V(\alpha)(U_{t+1} - U_t)) \tag{14}$$

$$= \text{tr}(U_{t+1}^T V(\alpha) U_{t+1}) + \text{tr}(U_t^T V(\alpha) U_t) - 2\text{tr}(U_t^T V(\alpha) U_{t+1}). \tag{15}$$

Combining **a)** and **b)** yields the desired inequality.

2. Since $\mathcal{S}$ is compact and $f(U_t) = \text{tr}(U_t^T V(\alpha) U_t)$ is increasing (see 1. above), $f(U_t)$ converges to a point as t goes to ∞.

3. For symmetric and positive definite matrices in $\mathbb{R}^{m \times m}$ we define the inner product as $\langle X, Y \rangle_{V(\alpha)} = \text{tr}(X^T V(\alpha) Y)$, and the norm $\| \cdot \|_{V(\alpha)} = \sqrt{\text{tr}((\cdot)^T V(\alpha)(\cdot))}$. We prove the statement with this alternative norm, whereby it holds for the Frobenius norm due to the equivalence of the two norms.

We know from 1) and 2) that $\|U_t\|_{V(\alpha)}^2$ monotonically converges to a $\delta > 0$ from below, as t goes to infinity. Thus, for $\epsilon > 0$ there is a T such that $\delta - \|U_t\|_{V(\alpha)}^2 \leq \epsilon$ for $t \geq T$. We note that, as a consequence of the relations (13) and (14), the following is true (for all $t \geq 0$):

$$\|U_{t+1} - U_t\|_{V(\alpha)}^2 \leq \|U_{t+1}\|_{V(\alpha)}^2 - \|U_t\|_{V(\alpha)}^2. \tag{16}$$

For $t > T \geq 0$ we define the partial sum $s_{T,t}$ as below. On the right-hand side, we use (16) (repeatedly) to obtain the inequality.

$$s_{T,t} := \sum_{l=T}^{t-1} \|U_{l+1} - U_l\|_{V(\alpha)}^2 \leq \|U_t\|_{V(\alpha)}^2 - \|U_T\|_{V(\alpha)}^2 \leq \epsilon. \tag{17}$$

Now, since $s_{T,t}$ is monotonically increasing for increasing t and bounded from above by ϵ, there is $s^* \leq \epsilon$ s.t. $\lim_{t \to \infty} s_{T,t} = s^*$. $\qquad \square$