SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

July 2022, Vol. 65 172103:1–172103:18
https://doi.org/10.1007/s11432-019-3019-3

# Extending generalized unsupervised manifold alignment

Xiaoyi YIN[1,2], Zhen CUI[4], Hong CHANG[1,2*], Bingpeng MA[2] & Shiguang SHAN[1,2,3]

[1]*Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190*, China;*
[2]*University of Chinese Academy of Sciences, Beijing* 100049*, China;*
[3]*CAS Center for Excellence in Brain Science and Intelligence Technology, Shanghai* 200031*, China;*
[4]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing* 210094*, China*

Received 11 December 2019/Revised 20 March 2020/Accepted 22 May 2020/Published online 23 June 2022

**Abstract** Building connections between different data sets is a fundamental task in machine learning and related application community. With proper manifold alignment, the correspondences between data sets will assist us with comprehensive study of data processes and analyses. Despite the several progresses in semi-supervised and unsupervised scenarios, potent manifold alignment methods in generalized and realistic circumstances remain in absence. Besides, theretofore unsupervised algorithms seldom prove themselves mathematically. In this paper, we devise an efficient method to properly solve the unsupervised manifold alignment problem and denominate it as extending generalized unsupervised manifold alignment (EGUMA) method. More specifically, an explicit relaxed integer programming method is adopted to solve the unsupervised manifold alignment problem, which reconciles three factors covering the updated local structure matching, the the feature comparability and geometric preservation. An additional effort is retained on extending the Frank Wolfe algorithm to tacking our optimization problem. Besides our previous endeavors we adopt a new strategy for neighborhood discovery in the manifolds. The main advantages over previous methods accommodate (1) simultaneous alignment and discovery of manifolds; (2) complete unsupervised learning structure without any prerequisite correspondence; (3) more concise local geometry for the embedding space; (4) efficient alternative optimization; (5) strict mathematical analysis on the convergence and efficiency issues. Experiments on real-world applications verify the high accuracy and efficiency of our proposed method.

**Keywords** unsupervised manifold alignment, relaxed integer programming, Frank Wolfe algorithm

**Citation** Yin X Y, Cui Z, Chang H, et al. Extending generalized unsupervised manifold alignment. Sci China Inf Sci, 2022, 65(7): 172103, https://doi.org/10.1007/s11432-019-3019-3

## 1 Introduction

Many studies in machine learning extend beyond studying disparate data sets but require building connections between such data sets. Applying the knowledge acquired from one data set to another is often useful. However, data sets must be connected to enable the transfer of knowledge, which requires the alignment of data sets, especially when there are no prerequisites. In real world scenarios, data sets often emerge with high diversity but share common features. Because data sets reside in different high dimensional spaces, it is challenging to find an alignment between them. Nevertheless, using the low intrinsic dimensions of the data sets makes alignment easier. Consequently, previous studies [1–3] have explored data latent structures, a characteristic of data sets that uses manifold models to fit and align. During the past decades, manifold alignment, which aims to discover the relationships between manifolds, has motivated many research interests. Wang et al. [4] proposed a visual domain adaptation method based on manifold embedding distribution alignment. Knowledge was transferred from various labeled source domains to one unlabeled target domain by aligning feature spaces [5]. Ji et al. [6] designed a manifold

* Corresponding author (email: changhong@ict.ac.cn)

 

alignment algorithm to cope with the multi-sensor coregistration problem. The manifold alignment problems are categorized into two different approaches. The first approach attempts to understand data by extracting manifold structures from the data sets; this approach is known as manifold learning. Manifold learning algorithms in [7–11] reveal the underlying data structure through dimensionality reduction. The algorithms are based on the assumption that data sets of similar semantic topics often reside in highly correlated embedded spaces. Based on the characteristic of the manifold to fit on manifold alignment, e.g., Ref. [12] devised a loss function to preserve the consistency of the local geometry structure to align manifolds. The other approach relates the manifold alignment problem to a classical 'assignment' task in combinatorial mathematics. Previous attempts often resort to graph theories and integer programming methods during the alignment process for a proper data match. Besides set matching, the convergence bound is also obtained through combinational analysis.

In machine learning and computer vision, the past decade witnessed an increased interest in manifold alignment methods, which is divided into two main categories: (semi-)supervised and unsupervised methods. Supervised or semi-supervised methods in [13–20] often require labeled data in both data sets, for example, labels or handcrafted correspondences, for learning the transformation. The drawback of the (semi-)supervised methods is that it fails to generalize to new circumstances where the counterpart labeled data are not available. On the other hand, unsupervised manifold alignment (UMA) methods discover information from manifold structures, which circumvents the problem of label deficiency. In [12], manifold structures are regarded as adjacent local relationships and specified the distance of the manifold points' as the corresponding weight matrices. With all possible permutations among neighbors of any two points taken into consideration, the method selects the minimum matching cost to fill in the distance connections. Consequently, the algorithm is computationally expensive and unscalable. To modify the problem, Ref. [21] devised a scheme of the B-spline curve to interpolate the gaps between the right and the next sorted adjacent weights. Then subsequences of the curves were aligned by calculating the matching scores of the curves across manifolds. Both (semi-)supervise and unsupervised methods divided the manifold alignment processes into the computation of matching similarities and discovering sequential counterparts. Nevertheless, the two step scheme is likely to be hampered by ignoring the development of neighborhood relationships. In other words, the neighbors belonging to a point in original space usually depart from it in the embedded space. To alleviate these problems, Ref. [22] modeled geometry structures as local reconstruction coefficients to be affine invariant. The mentioned method not only adjusts the neighborhood during the evolution of the matching matrix but simplifies the computation process for online alignment tasks. However, the reference set in the mentioned scheme is not a general case for UMA. Moreover, schemes mentioned above utilize local structures, which probably impinge the matching accuracy obliged to undersized scope. Global alignment is more likely to find reasonable matches with a broader consideration of all possible situations. Cui et al. [23] designed a global alignment method, where the alignment problem is modeled as an integer programming problem, and the geometry of one manifold is constrained before embedding with that of another.

In this paper, we generalize the problem as an extended global unsupervised manifold alignment (EGUMA) problem based on our previous paper [23] (GUMA). Distinct from previous studies, we implemented an explicit quadratic optimization model with relaxed 0-1 constraints. Relaxed constraints remedy the mathematical flaw in the original rounding after optimization algorithm structures, and solve the problem of different cardinalities across data sets. The local geometry is updated when optimizing the embedding function; in this case, we preserve the local structure of updated points as our previous work. In our formulation, we considered and balanced local geometry matching, global feature corresponding, and preserving the structure. We used an efficient alternative algorithm between matching and transformations for model optimization. We also extended the Frank Wolfe (FW) algorithm by [24] to handle the slack integer quadratic programming.

Note that the extended FW algorithm can be generalized to solve other linear programming problems. Theoretical deductions and extensive experiments prove the effectiveness of the proposed EGUMA method.

The proposed method differs from the traditional methods from the following aspects: (i) it does not predetermine the relationships of the manifold local neighborhood; (ii) it matches differing data sets globally instead of locally; (iii) it discovers the embedded feature spaces based on alignment process; (iv) it efficiently solves the general integer programming problems; and (v) it identifies strict convergence bound and mathematical precision of the whole algorithm structure. Although constraints in the proposed method are similar to that of kernel sorting in kernel sorting by [25] and latent variable model in [26].

However, kernel sorting by [25] and latent variable model in [26] does not consider geometry relations. Furthermore, we illustrate the superiority of our proposed methods over the other methods.

The rest of this paper is organized as follows. In Section 2, we define the problem in mathematical notations. We explain our model in Section 3, and Section 4 supplies the optimization schema. The rationale of the approach is stated in Section 5. We demonstrate experiments and present experimental result analysis in Section 6. Section 7 draws concluding remarks and proposes possible future works.

## 2 Problem description

Let $\boldsymbol{X}^1, \ldots, \boldsymbol{X}^N$ denote $N$ data sets, shaping up into $N$ different manifolds $\mathcal{M}_i$ for $\forall i = 1, \ldots, N$, where $\boldsymbol{X}_i \in \mathbb{R}^{D_i \times n_i}$, $d_i$ and $n_i$ are the dimension and cardinality of the data set respectively. The goal of UMA lies on building correspondence between every pair of data sets without any pre-specified correspondences. The output of the algorithm is defined as matrix representing the alignment results, with all the elements confined to the interval $[0, 1]$, i.e., $\boldsymbol{F}_{ij} \in [0, 1]^{n_i \times n_j}$, and subjecting to a sum-to-one constraint to record the correspondences between $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$. This alignment could be generalized to any other two data sets, so we simply use $\boldsymbol{F}$ to represent the alignment matrix. For an element, e.g., the $a$th row and $b$th column in $\boldsymbol{F}$ approaching to 1 represents the $a$th point of $\boldsymbol{X}_i$ verge upon the $b$th point of $\boldsymbol{X}_j$ while approaching to 0 depart. The matrix $\boldsymbol{F}$ is defined as

$$\Pi = \{\boldsymbol{F} | \boldsymbol{F} \in [0, 1]^{n_i \times n_j}, \mathbf{1}_{n_i}^{\mathrm{T}} \boldsymbol{F} = \mathbf{1}_{n_j}^{\mathrm{T}}, \boldsymbol{F} \mathbf{1}_{n_j} = \mathbf{1}_{n_i}\}. \tag{1}$$

$\mathbf{1}_{n_i}^{\mathrm{T}} \boldsymbol{F} = \mathbf{1}_{n_j}^{\mathrm{T}}$ is not trivial, not able to be derived from the other constraints.

Besides match learning in appearance geometry space, we project the data sets in linear or nonlinear methods to discover the lower dimensional intrinsic representations. $\boldsymbol{\Phi}_i(\backslash \boldsymbol{\Phi}_j)$ projects the data set $\boldsymbol{X}_i(\backslash \boldsymbol{X}_j)$ from the original (configuration) space to the mutual embedding (manifold) space $\mathcal{M}$, and $\boldsymbol{\Phi}_i(\boldsymbol{X}_i) \in \mathbb{R}^{d \times n_i}$ where $d \ll D_i$. To this end, the correspondence matrix $\boldsymbol{F}$ together with the embedding projections $\boldsymbol{\Phi}$ are to be learned from the model to accomplish the unsupervised manifold alignment.

## 3 The model

We build our model based on four key observations. First, data sets of similar categories, e.g., the same action performed by different persons, often drop into a certain slot of geometry appearance at every scales. Second, the underlying elements behind diverse manifolds aggregate in the embedded space. Third, the intrinsic characters for all data sets should be preserved to embed them properly. Fourth, the neighborhood structure alternates when data points projected to the alignment embedding spaces. Depending on these motivations, the following objective functions are proposed.

Overall objective function:

$$\begin{aligned} \min_{\boldsymbol{\Phi}_i, \boldsymbol{F}} \quad & E_s + \gamma_f E_f + \gamma_p E_p \\ \text{s.t.} \quad & \boldsymbol{F} \in \Pi, \quad \boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j \in \boldsymbol{\Theta}, \end{aligned} \tag{2}$$

where $\gamma_f$ and $\gamma_p$ are the tuning parameters; $\boldsymbol{\Theta}$ denotes the constraints to avoid trivial solutions for $\boldsymbol{\Phi}$; $E_s$, $E_f$ and $E_p$ are three terms respectively measuring the degree of local geometry matching, feature corresponding and geometry preserving criteria, which will be detailed respectively in the following text.

**$E_s$: Dynamically updated local geometry matching term.** Similarity in apperance is the most intuitive constraint to align different data sets. In the community of manifold alignment, the local geometry is the basis of all methodology. In our previous work we directly align the local structure in the original space, but in this work we would dynamically update the local structure in every steps. For this propose, graphs with weighted edges of the last step are exploited via graph adjacency matrices $\boldsymbol{K}_i^{(t-1)}$ ($\boldsymbol{K}_j^{(t-1)}$) of embedding of the data sets $\boldsymbol{\Phi}_i^{(t-1)}(\boldsymbol{X}_i)$ ($\boldsymbol{\Phi}_j^{(t-1)}(\boldsymbol{X}_j)$) where $t$ denotes the current step. To simplify the symbols we still denote them as $\boldsymbol{K}_i \in \mathbb{R}^{n_i \times n_i}$ ($\boldsymbol{K}_j \in \mathbb{R}^{n_j \times n_j}$). From the definition we formulate the local graph matrix as

$$[\boldsymbol{K}]_{ab} = d(\boldsymbol{Z}_{\cdot a}, \boldsymbol{Z}_{\cdot b}),$$

where $\boldsymbol{Z} = \Phi(X)$. In general cases $d$ denotes the geodesic distance, other formulations such as Euclidean distance are valid also. We adopt the former in this work. The geodesic distance algorithm constructed by graph theory mentioned by [7] listed in the appendix.

To reduce the scales, we normalize all original data sets to unit standard deviation. We then formulate manifold matching objective as the following function in global geometry scale:

$$E_s = \|\boldsymbol{K}_i - \boldsymbol{F}\boldsymbol{K}_j\boldsymbol{F}^{\mathrm{T}}\|_F^2 = \mathrm{tr}(\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{K}_i + \boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}_j\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{K}_j - 2\boldsymbol{F}^{\mathrm{T}}\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}_j), \tag{3}$$

where $\boldsymbol{F} \in \Pi$ denotes the (full or partial) coherences defined in Eq. (1).

$E_f$**: Feature matching term.** The aligned data points are expected to have little diversity of intrinsic feature representations in the mutual embedding space $\mathcal{M}$. Given $N$ data sets $\boldsymbol{X}_i$, $i = 1, 2, \ldots, N$, for any two data sets alignment, the featurematching term can be formulated as

$$E_f = \|\boldsymbol{\Phi}_i(\boldsymbol{X}_i) - \boldsymbol{\Phi}_j(\boldsymbol{X}_j)\boldsymbol{F}^{\mathrm{T}}\|_F^2 = \mathrm{tr}(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i + \boldsymbol{\Phi}_j\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Phi}_j^{\mathrm{T}} - 2\boldsymbol{\Phi}_j\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Phi}_i^{\mathrm{T}}), \tag{4}$$

where $\boldsymbol{\Phi}_i(\boldsymbol{X}_i)$ and $\boldsymbol{\Phi}_j(\boldsymbol{X}_j)$ are simplified from $\boldsymbol{\Phi}_i^{(t)}(\boldsymbol{X}_i)$ and $\boldsymbol{\Phi}_j^{(t)}(\boldsymbol{X}_j)$, represent embedding coordinates respectively for $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$. They are to either be linear projections or implicit nonlinear projections through extended kernel tricks.

$E_p$**: Geometry preserving term.** Unfolding manifolds to the mutual embedding space is not expected to deform the local neighborhood relationships, i.e., the local geometric structures ought to be preserved to avoid neighborhood impingements. Thus we consider a preserving term to reconcile the tension between the first two constrains. We construct the local adjacency weight matrices $\boldsymbol{W}_{X_i}$ for $i = 1, 2, \ldots, N$ respectively for the data sets $\boldsymbol{X}_i$, as several popular manifold learning algorithms of [9,27]. Then, we define the geometry preserving term as

$$E_p^i = \frac{1}{2}\sum_{a,b}\|\boldsymbol{\Phi}_i^{\mathrm{T}}(\boldsymbol{X}_{i\cdot a} - \boldsymbol{X}_{i\cdot b})\|^2 w_{ab}^x = \mathrm{tr}(\boldsymbol{\Phi}_i(\boldsymbol{X}_i)\boldsymbol{L}_i\boldsymbol{\Phi}_i(\boldsymbol{X}_i)^{\mathrm{T}}), \tag{5}$$

where $\boldsymbol{L}_i$ is the graph Laplacian matrices, with $\boldsymbol{L}_i = \mathrm{diag}([\sum_k w_{1k}^{x_i}, \ldots, \sum_k w_{n_i k}^{x_i}]) - \boldsymbol{W}_{X_i}$, and $w_{ab}^{x_i}$ is the weight between the $a$th point and the $b$th point in $\boldsymbol{X}_i$. We define $E_p = E_{pi} + E_{pj}$ to preserve the geometry structure of both $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$.

## 4 Efficient optimization

It is difficult to solve the objective function (2) owing to multiple indecomposable variables. When we derivate the arguments over the optimization function, they entangle with each other so a brutal force algorithm will fail. Here an efficient alternate optimization that approximates solution is proposed to solve the problem. Specifically, the objective function (2) is decomposed into two submodules, corresponding separately to the optimizations of the matching matrices $\boldsymbol{F}$ and the representations $\boldsymbol{\Phi}_i$, $\boldsymbol{\Phi}_j$. With $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Phi}_j$ clamped, we solve one submodule over the argument $\boldsymbol{F}$ through a non-convex quadratic relaxed integer programming procedure, where approximate solution is to find along the gradient-descent path of a convex model. We extend the Frank Wolfe algorithm [24] as an effective block coordinate descent (BCD) method, so as to find the solution at multinomial steps as the problem scale. When fixing $\boldsymbol{F}$, an analytic solution will be obtained over $\boldsymbol{\Phi}_i$ without looping steps. The two submodules are alternately optimized until convergence to the final.

### 4.1 Learning match

We set $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Phi}_j$ as constants during learning the matching matrix $\boldsymbol{F}$, so the primitive problem reduces to

$$\min_{\boldsymbol{F}\in\Pi} \Psi_0(\boldsymbol{F}) = E_s + \gamma_f E_f. \tag{6}$$

Through several derivations, we rewrite the objective function as

$$\min_{\boldsymbol{F}\in\Pi} \Psi_0(\boldsymbol{F}) = \{\|\boldsymbol{K}_i\boldsymbol{F} - \boldsymbol{F}\boldsymbol{K}_j\|_F^2 + \mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{1}\boldsymbol{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj}) + \mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{G})\}, \tag{7}$$

where $\boldsymbol{K}^{jj} = \boldsymbol{K}_j \odot \boldsymbol{K}_j$ and $\boldsymbol{G} = \gamma_f(\mathbf{1}\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Phi}_j \odot \boldsymbol{\Phi}_j) - 2{\boldsymbol{\Phi}_i}^{\mathrm{T}}\boldsymbol{\Phi}_j) - \mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{K}^{jj}$. The non-relaxed integer programming problem is NP-hard for an enumeration of $n!$ times if through the exhaustive search strategy. To get an efficient solution, we solve the relaxed optimization problem under the framework of FW algorithm [24].

As $\Psi_0$ is non-convex, its optimization easily degenerates owing to local optima. To avoid this, we incorporate an accessory function $\omega(\boldsymbol{F}) = \Lambda\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F})$, with $\Lambda = n_i \times \max\{-\min(\mathrm{eig}(\boldsymbol{K}^{jj})), 0\}$ added to $\Psi_0$ to obtain a new objective as

$$\Psi(\boldsymbol{F}) = \{\|\boldsymbol{K}_i\boldsymbol{F} - \boldsymbol{F}\boldsymbol{K}_j\|_F^2 + \mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj} + \Lambda\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}) + \mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{G})\}. \tag{8}$$

The first term of (8) is quadratic for variable $\mathrm{vec}(\boldsymbol{F})$, thus convex. As to the second, the Hessian is $2({\boldsymbol{K}^{jj}}^{\mathrm{T}} \otimes (\mathbf{1}\mathbf{1}^{\mathrm{T}}) + \Lambda\boldsymbol{I})$. Because we add an auxiliary term $\Lambda\boldsymbol{I}$, it is also positive definite, so the second term also convex. For the convexity is close under addition, the new objective function $\Psi$ is convex over the domain $\Pi$. The detailed proof is given in the appendix. It is worthy of a note that the solution from minimizing either $\Psi_0$ and $\Psi$ are the same because $\omega(\boldsymbol{F})$ is a constant.

In Algorithm B1 we list the algorithm of learning the match matrix $\boldsymbol{F}$, where it iteratively takes derivation of $\Psi$ and descendes along the derivation in the domain $\Pi$, as classical FW algorithm. The simplex algorithm in step 4 taken in the real domain will obtain a more proper $\boldsymbol{F}$ for the objective function $\Psi$. Formally we solve the problem with the use of MATLAB toolbox 'linprog' and exploit the 'interior-point' strategy.

$$\begin{aligned} &\max_{\boldsymbol{F}} \ \mathrm{tr}(\nabla\Psi(\boldsymbol{F}^k)^{\mathrm{T}}\boldsymbol{F}) \\ &\text{s.t. } 0 \leqslant \boldsymbol{F}_{ij} \leqslant 1, \\ &\quad \boldsymbol{F}\mathbf{1}_{n_j} = \mathbf{1}_{n_i}, \\ &\quad \mathbf{1}_{n_i}^{\mathrm{T}}\boldsymbol{F} = \mathbf{1}_{n_j}^{\mathrm{T}}, \end{aligned} \tag{9}$$

where

$$\nabla\Psi(\boldsymbol{F}^k) = 2(\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{K}_i\boldsymbol{F}^k + \boldsymbol{F}^k\boldsymbol{K}_j{\boldsymbol{K}_j}^{\mathrm{T}} - 2{\boldsymbol{K}_j}^{\mathrm{T}}\boldsymbol{F}^k\boldsymbol{K}_j + \mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}^k\boldsymbol{K}^{jj} + \Lambda\boldsymbol{F}^k) + \boldsymbol{G}. \tag{10}$$

Then alongside the descending path performed in steps 6–12 in Algorithm B1 we avoid local optima, where hard to tackle in the classical alignment methods performed in non-convex integer space. The objective value $\Psi(\boldsymbol{F}^k)$ is always non-increasing at each alternation, and thus $\{\boldsymbol{F}^1, \boldsymbol{F}^2, \ldots\}$ will converge into a fixed point if we loop enough times. After finding the match matrix $\boldsymbol{F}$, we find the best matching result using the Hungarian algorithm. The validation is detailed in Appendixes E and F.

### 4.2 Learning transformations

As $\boldsymbol{F}$ is fixed, by minimizing the following function we get the embedding transformations:

$$\underset{\boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j \in \boldsymbol{\Theta}}{\boldsymbol{\Upsilon}}(\boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j) = \gamma_f E_f + \gamma_p E_p. \tag{11}$$

Plugging in (4) and (5) we get

$$\begin{aligned} \boldsymbol{\Upsilon}(\boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j) &= \gamma_f\|\boldsymbol{\Phi}_i(\boldsymbol{X}_i) - \boldsymbol{\Phi}_j(\boldsymbol{X}_j)\boldsymbol{F}^{\mathrm{T}}\|_F^2 + \gamma_p\mathrm{tr}(\boldsymbol{\Phi}_i(\boldsymbol{X}_i)\boldsymbol{L}_i\boldsymbol{\Phi}_i(\boldsymbol{X}_i)^{\mathrm{T}} \\ &= \mathrm{tr}(\boldsymbol{\Phi}_i(\boldsymbol{X}_i)(\gamma_f\boldsymbol{I} + \gamma_p\boldsymbol{L}_i)\boldsymbol{\Phi}_i(\boldsymbol{X}_i)^{\mathrm{T}}) + \mathrm{tr}(\boldsymbol{\Phi}_j(\boldsymbol{X}_j)(\gamma_f\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \gamma_p\boldsymbol{L}_j)\boldsymbol{\Phi}_j(\boldsymbol{X}_j)^{\mathrm{T}}) \\ &\quad - 2\gamma_f\mathrm{tr}(\boldsymbol{\Phi}_i(\boldsymbol{X}_i)\boldsymbol{F}\boldsymbol{\Phi}_j(\boldsymbol{X}_j)^{\mathrm{T}}). \end{aligned} \tag{12}$$

Note that $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Phi}_j$ are easily substituted to non-linear features. To avoid pathological solutions of $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Phi}_j$, we centralize $\boldsymbol{X}_i, \boldsymbol{X}_j$ and reformulate the optimization problem by considering the following constraints:

$$\boldsymbol{\Theta_i} = \{\boldsymbol{\Phi}_i | \boldsymbol{X}_i\mathbf{1} = \mathbf{0}, \boldsymbol{\Phi}_i(\boldsymbol{X}_i)(\gamma_f\boldsymbol{I} + \gamma_p\boldsymbol{L}_i)\boldsymbol{\Phi}_i(\boldsymbol{X}_i)^{\mathrm{T}} = \boldsymbol{I}\}, \tag{13}$$

$$\boldsymbol{\Theta_j} = \{\boldsymbol{\Phi}_j | \boldsymbol{X}_j\mathbf{1} = \mathbf{0}, \boldsymbol{\Phi}_j(\boldsymbol{X}_j)(\gamma_f\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \gamma_p\boldsymbol{L}_j)\boldsymbol{\Phi}_j(\boldsymbol{X}_j)^{\mathrm{T}} = \boldsymbol{I}\}. \tag{14}$$

Then we convert (12) into the following equation:

$$\max_{\boldsymbol{\Phi}_i \in \boldsymbol{\Theta_i}, \boldsymbol{\Phi}_j \in \boldsymbol{\Theta_j}} \mathrm{tr}(\boldsymbol{\Phi}_i(\boldsymbol{X}_i)\boldsymbol{F}\boldsymbol{\Phi}_j(\boldsymbol{X}_j)^{\mathrm{T}}). \tag{15}$$

Following the method of canonical correlation analysis (CCA) [28], we obtain analytical solution by eigenvalue decomposition. According to a favorable proposition of the CCA, this value will be invariant to any affine transformation such as swapping, where it enables us to change the representations of $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ while preserving their intrinsic structures.

Hereinafter we model succinctly the projection in linear forms, a.k.a. $\boldsymbol{\Phi}_i(\boldsymbol{X}_i) = \boldsymbol{P}_i^{\mathrm{T}}\boldsymbol{X}_i$ and $\boldsymbol{\Phi}_j(\boldsymbol{X}_j) = \boldsymbol{P}_j^{\mathrm{T}}\boldsymbol{X}_j$. In light of the affine invariance of $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$, we re-model (15) as

$$\begin{aligned} \max_{\boldsymbol{P}_i,\boldsymbol{P}_j} \quad & \mathrm{tr}\left(\boldsymbol{P}_i^{\mathrm{T}}\boldsymbol{X}_i\boldsymbol{F}\boldsymbol{X}_j^{\mathrm{T}}\boldsymbol{P}_j\right) \\ \text{s.t.} \quad & \boldsymbol{P}_i^{\mathrm{T}}\boldsymbol{X}_i(\gamma_f\boldsymbol{I}+\gamma_p\boldsymbol{L}_i)\boldsymbol{X}_i^{\mathrm{T}}\boldsymbol{P}_i = \boldsymbol{I}, \\ & \boldsymbol{P}_j^{\mathrm{T}}\boldsymbol{X}_j(\gamma_f\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}+\gamma_p\boldsymbol{L}_j)\boldsymbol{X}_j^{\mathrm{T}}\boldsymbol{P}_j = \boldsymbol{I}. \end{aligned} \tag{16}$$

Let $\boldsymbol{C}_{ij} = \boldsymbol{X}_i\boldsymbol{F}\boldsymbol{X}_j^{\mathrm{T}}$, $\boldsymbol{A}_{ij} = \boldsymbol{X}_i(\gamma_f\boldsymbol{I}+2\gamma_p\boldsymbol{L}_i)\boldsymbol{X}_i{}^{\mathrm{T}}$, $\boldsymbol{B}_{ij} = \boldsymbol{X}_j(\gamma_f\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}+2\gamma_p\boldsymbol{L}_j)\boldsymbol{X}_j^{\mathrm{T}}$, and then we find the solution of the above equations by eigenvalue decompostion:

$$\boldsymbol{A}_{ij}^{-1}\boldsymbol{C}_{ij}\boldsymbol{B}_{ij}^{-1}\boldsymbol{C}_{ij}^{\mathrm{T}}\boldsymbol{P}_i = \lambda^2\boldsymbol{P}_i, \tag{17}$$

$$\boldsymbol{B}_{ij}^{-1}\boldsymbol{C}_{ij}\boldsymbol{A}_{ij}^{-1}\boldsymbol{C}_{ij}^{\mathrm{T}}\boldsymbol{P}_j = \lambda^2\boldsymbol{P}_j. \tag{18}$$

The solutions are usually requiring $\boldsymbol{A}_{ij}$ and $\boldsymbol{B}_{ij}$ invertible. In case singularity of the solutions, we add some small regularization term $\epsilon_0\boldsymbol{I}$ where $\epsilon_0$ is a small constant. Note that this term is able to be articulated in non-linear forms.

## 5 Algorithm analysis

### 5.1 Analysis on the EGUMA algorithm

As mentioned in Section 1, cultivating priors based on manifold hypothesis is crucial to construct meaningful alignment. To this end, we extract the similarities between the data sets in geometry appearances, and more importantly the underlying manifold representations. The adjacency matrices in the first objective term characterize the topological structures of manifolds. Different from many previous methods which mainly focus on local neighborhood relationships, we calculate the full adjacency matrix to capture the global manifold geometry structure. The third term in a CCA formulation is capable of preserving the geometry similarity and maintaining the consistency during the correspondence exchanges. Nevertheless, the second term is more important in the alignment scenario, which will be proved by experiments in Subsection 6.4. Besides the first term tries to match data in the original space, more accurate alignment is accomplished in the embedding space.

In our previous work [23] the alignment occurs in the integer space but the optimization of the objective is taken in real domain. Because the integer space is not complete, we make the remedy of this theoretical flaw by formulating the whole algorithm within the real domain and relaxing the 0-1 constraint in the first place.

### 5.2 Computational analysis of learning the match

Four operations in Algorithm B1 possibly consume computation resources including: matrix multiplication, Hardmard product, Kronecker product, and a nested simplex algorithm. For $\forall\boldsymbol{A}\in\mathbb{R}^{n\times d},\boldsymbol{B}\in\mathbb{R}^{n\times p}$, the computation complexity of their multiplication $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$ is $\boldsymbol{O}(dnp)$; Hardmard product is relatively lightweight with only a consumption along the quantity of elements. e.g., $\forall\boldsymbol{A},\boldsymbol{B}\in\mathbb{R}^{m\times n}$, we just take $\boldsymbol{O}(mn)$ multiplications to get $\boldsymbol{A}\odot\boldsymbol{B}$. $\forall\boldsymbol{S}\in\mathbb{R}^{p\times q},\boldsymbol{T}\in\mathbb{R}^{r\times s}$, their Kronecker product $\boldsymbol{S}\otimes\boldsymbol{T}$ contains $\boldsymbol{O}(pqrs)$ scaler multiplications from the definition. But in the special case where $\boldsymbol{T}$ is a matrix consisting of all ones, we only need to replicate every element of $\boldsymbol{S}$ for $r\times s$ times, and then the complexity depends only on the scale of $\boldsymbol{S}$, i.e., $\boldsymbol{O}(pq)$. The key technique over the whole algorithm revolves on adapting the FW algorithm to the EGUMA. FW is devised to solve the optimization problems within some compact convex set [24]. In implement, the calculation expense of the simplex algorithm is $\boldsymbol{O}(m^2n)$, where $m$ is the number of constraints, and $n$ the variables. Though not a polynomial method in the worst cases, Ref. [29] showed that it always finishes in a polynomial time along the input scale in practice. From the empirical experiments, the typical number of operations is proportional to $m$ but only logarithm of

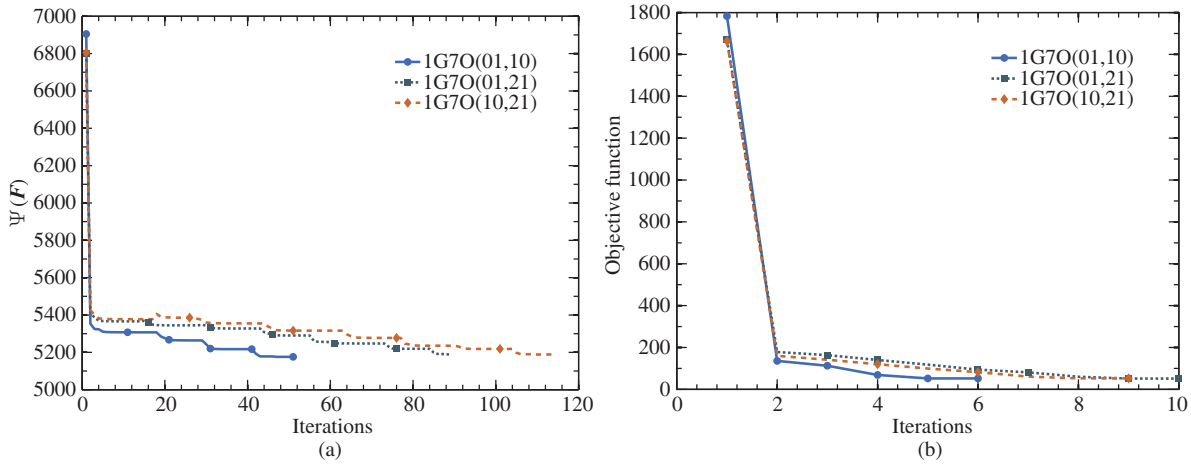


**Figure 1** (Color online) Convergence trend of learning (a) match and (b) alignment.

$n$ [30]. Based on the analysis above, the highest order of computation is cubic, i.e., $O(n_j^2 \times n_i)$ operations are needed to perform one iteration.

As the FW method converges at a rate of $O(1/k)$ along the alternations $k$, Algorithm B1 takes an expense of $O(\frac{1}{\epsilon_1} n_j^2 \times n_i)$, where $\epsilon_1$ is the controlling error parameter in step 14. In practice, steps 6 and 7 are supplied to cease the vibration of the curve on $\Psi(\boldsymbol{F}^k)$ along the increase of $k$, and save computations when the renew step happens. The detailed convergence proof is supplied at Appendix F.

### 5.3 Computational analysis of the overall alignment algorithm

We have analyzed the computational complexity of $\boldsymbol{F}$ is $O(\frac{1}{\epsilon_1} n_j^2 \times n_i)$, the left analysis focuses on $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$ and the cooperating effects.

The computation complexity to calculate $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$ depends on a eigenvalue decomposition operation. The eigenvalue decomposition of a matrix $\forall \boldsymbol{M} \in \mathbb{R}^{m \times n}$ has the complexity of $O(mn \min\{m, n\})$. Then in our case $O(n_i^2 \times n_j)$ is enough for learning transformation.

In implementation we loop steps 4–6 in Algorithm C1 until (1) $\boldsymbol{F}$ converges; or (2) object function converges; or (3) reaching maximum iterations. Because the linear transformation between normed space is bounded and continuous [31], we use this proposition to get a sub-linear convergence rate [32].

Through alternating the two steps mentioned above, we will finally get appropriate solutions as other BCD algorithms. The detailed convergence proof is supplied at Appendix G. The computation expense mainly draws at alignment learning submodule, i.e., the steps shown in Algorithm B1. The cost of the whole algorithm will be $O(\frac{1}{\epsilon} n_j^2 \times n_i)$, where $\epsilon = \max\{\epsilon_1, \epsilon_2\}$ and $\epsilon_2$ is the threshold of the object function. In many applications, very limited iterations are needed to get a relatively satisfying solution. We perform an experiment to show the convergence rate in Subsection 5.4.

### 5.4 Convergence trend of EGUMA

We perform more experiments on protein sequence alignment to show the convergence trend of matching function $\Psi(\boldsymbol{F})$ and objective function as shown in Figure 1. From the figures we see that the algorithm converges quite quickly within a few iterations, much faster than linear and akin to logarithm curves. In practice the EGUMA operates much faster than the previous UMA methods like [12] or [21].

It is noteworthy that for every update of $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$, $\Psi(\boldsymbol{F})$ leaps down a little from Figure 1(a). This phenomenon validates the CCA step in the overall loop.

### 5.5 Proof of the convergence

**Theorem 1.** The optimizing value of the function $\Psi(\boldsymbol{F}^k)$ is non-increasing at Algorithm B1.

*Proof.* Note that at steps 9 and 10, $\Psi(\boldsymbol{F}^{k+1}) = \Psi(\boldsymbol{H}) < \Psi(\boldsymbol{F}^k)$; at steps 6 and 7, for $\delta$ is the contemporary optimal stride, $\Psi(\boldsymbol{F}^{k+1}) = \Psi(\boldsymbol{F}^k + \delta(\boldsymbol{H} - \boldsymbol{F}^k)) \leqslant \Psi(\boldsymbol{F}^k)$. This proves that $\Psi(\boldsymbol{F}^k)$ is non-increasing.

Insomuch the domain of $\boldsymbol{F}^k$ to be a bounded and convex set, the object function is also bounded. A bounded and non-increasing sequence is convergent, which completes the proof.

While the above paragraph shows an existence proof, we are more interested in a constructive proof with a precise bound and a convergence rate. The following details of this part show the proof of the theorem tagging along.

**Theorem 2.** $\Psi(\boldsymbol{F}^k)$ in Algorithm B1 converges at the order of $\frac{1}{k}$, or $\boldsymbol{O}(\frac{1}{k})$.

**Definition 1** ([33])**.** Given a normed space $(\mathscr{S}, \|\cdot\|)$ and a function $f : \mathbb{R}^n \to \mathbb{R}$, we define the restricted smoothness property constant as

$$\boldsymbol{L}_{\|\cdot\|}(f;\mathscr{S}) := \sup_{\boldsymbol{X},\boldsymbol{y}\in\mathscr{S},\alpha\in(0,1]} \frac{f((1-\alpha)\boldsymbol{X}+\alpha\boldsymbol{y}) - f(\boldsymbol{X}) - \langle\nabla f(\boldsymbol{X}), \alpha(\boldsymbol{y}-\boldsymbol{X})\rangle}{\frac{1}{2}\alpha^2\|\boldsymbol{y}-\boldsymbol{X}\|^2}. \tag{19}$$

**Theorem 3.** For $\forall k \geqslant 1$, the iterates of steps 9–11 in Algorithm B1 lie in $\Pi$ satisfy

$$\Psi(\boldsymbol{F}^k) - \Psi(\boldsymbol{F}^{\star\star}) \leqslant \frac{4\boldsymbol{L}_{\|\cdot\|_F}(\Psi;\Pi)|||\Pi|||^2}{k}, \tag{20}$$

where $\boldsymbol{F}^{\star\star}$ denotes

$$\boldsymbol{F}^{\star\star} = \underset{\boldsymbol{F}\in\Pi}{\operatorname{argmin}} \operatorname{tr}(\nabla\Psi(\boldsymbol{F})\boldsymbol{F}^{\mathrm{T}}). \tag{21}$$

$\boldsymbol{L}_{\|\cdot\|_F}(\Psi;\Pi)$ is the restricted smoothness property constant of $\Psi(\boldsymbol{F})$, and $|||\Pi||| := \sup_{\boldsymbol{p}i\in\Pi}\|\operatorname{vec}(\boldsymbol{p}i)\|$.

We slightly make an amendment of the definition of restricted smoothness property constant (RSPC) and reduce the denominator by a half in [33], for a better link up to the theory of the Taylor expansion. The situation here is a special case of the original literature. This term reflects how much a function curls up by intuition. In the domain of a function, the steeper a curve is, the greater its second derivative is to claim, and the larger the RSPC will be. Now we prove the above theorem.

*Proof.* We define the gap between function value at present and the function value at minimum,

$$\begin{aligned}\eta_k &:= \Psi(\boldsymbol{F}^k) - \Psi(\boldsymbol{F}^{\star\star}) \leqslant \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{F}^k - \boldsymbol{F}^{\star\star}\rangle\\ &= \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{F}^k\rangle - \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{F}^{\star\star}\rangle\\ &\leqslant \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{F}^k\rangle - \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{H}\rangle.\end{aligned} \tag{22}$$

The last inequality holds because $\boldsymbol{H}$ is the minimizer of the inner product $\langle\nabla\Psi(\boldsymbol{F}^k), \cdot\rangle$ over domain $\Pi$ then $\langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{H}\rangle \leqslant \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{F}^{\star\star}\rangle$.

Let $\boldsymbol{L}$ denote $\boldsymbol{L}_{\|\cdot\|_F}(\Psi;\Pi)$ and $R$ denote $|||\Pi|||$, for $\forall\delta \in [0,1]$,

$$\begin{aligned}\Psi(\boldsymbol{F}^{k+1}) &= \Psi(\boldsymbol{F}^k + \delta(\boldsymbol{H}-\boldsymbol{F}^k))\\ &\leqslant \Psi(\boldsymbol{F}^k) + \delta\langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{H}-\boldsymbol{F}^k\rangle + \frac{1}{2}\delta^2\boldsymbol{L}\|\boldsymbol{H}-\boldsymbol{F}^k\|^2\\ &\leqslant \Psi(\boldsymbol{F}^k) + \delta\langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{H}-\boldsymbol{F}^k\rangle + \frac{1}{2}\delta^2\boldsymbol{L}(2\|\boldsymbol{H}\|^2 + 2\|\boldsymbol{F}^k\|^2)\\ &\leqslant \Psi(\boldsymbol{F}^k) + \delta(\langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{H}\rangle - \langle\nabla\Psi(\boldsymbol{F}^k), \boldsymbol{F}^k\rangle) + 2\delta^2R^2\boldsymbol{L}\\ &\leqslant \Psi(\boldsymbol{F}^k) - \delta\eta_k + 2\delta^2R^2\boldsymbol{L}.\end{aligned} \tag{23}$$

Subtract $\Psi(\boldsymbol{F}^{\star\star})$ in both sides, we obtain

$$\begin{aligned}\eta_{k+1} &\leqslant \eta_k - \delta\eta_k + 2\delta^2R^2\boldsymbol{L}\\ &= \eta_k - \frac{\eta_k^2}{8R^2\boldsymbol{L}} + \left(\delta - \frac{\eta_k}{4R^2\boldsymbol{L}}\right)^2\\ &\leqslant \eta_k - \frac{\eta_k^2}{8R^2\boldsymbol{L}}\\ &= -\frac{1}{8R^2\boldsymbol{L}}(\eta_k - 4R^2\boldsymbol{L})^2 + 2R^2\boldsymbol{L}.\end{aligned} \tag{24}$$

The second inequality holds when we set $\delta = \frac{\eta_k}{4R^2\boldsymbol{L}} \in [0, \frac{1}{2}]$. For the sake of succinctness we use $\gamma := 2R^2\boldsymbol{L}$, then

$$\begin{cases}\eta_{k+1} \leqslant \gamma - \frac{(\eta_k - 2\gamma)^2}{4\gamma},\\ \eta_1 \leqslant \gamma.\end{cases} \tag{25}$$

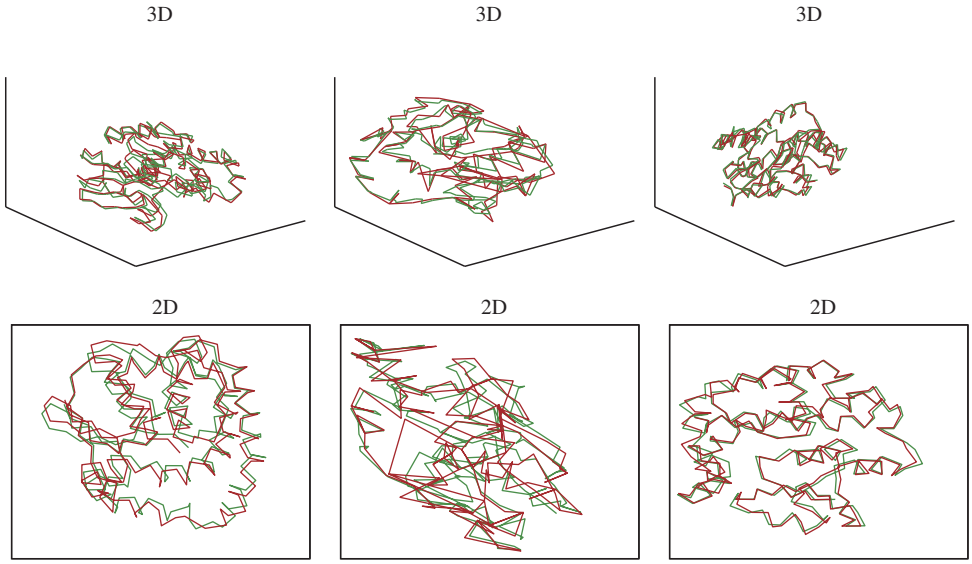


**Figure 2** (Color online) The alignment of protein sequences: 1G7O-01 and 1G7O-10. After alignment, the overlapping of two sequences in 3D and 2D space is shown by rows. The alignment results of Wang's, Pei's and EGUMA are shown by row from left to right.

Solving the recursion we get for $\forall k \geqslant 1, \eta_k \leqslant \frac{2\gamma}{k}$, which completes the proof.

**Theorem 4.** For $\forall k \geqslant 1$, the iterates over steps 4 and 5 in Algorithm B1:

$$\Psi(\boldsymbol{F}^{\star}) - \Psi(\boldsymbol{F}^{\star\star}) \leqslant \frac{4\boldsymbol{L}_{\|\cdot\|_F}(\Psi;\Pi)|||\Pi|||^2}{k}, \tag{26}$$

for $\forall \boldsymbol{F}^{\star}$ satisfies

$$\boldsymbol{F}^{\star} = \arg\min_{\boldsymbol{F}\in\Pi} \operatorname{tr}(\nabla\Psi(\boldsymbol{F}^k)^{\mathrm{T}}\boldsymbol{F}), \tag{27}$$

where $\boldsymbol{F}^{\star\star}$ has the same meaning of the previous theorem, and $\boldsymbol{F}^{\star}$ has a same definition as $\boldsymbol{H}$.

*Proof.* From the definition of $\boldsymbol{F}^{\star}$ we have $\Psi(\boldsymbol{F}^{\star}) - \Psi(\boldsymbol{F}^{\star\star}) \leqslant \Psi(\boldsymbol{F}^k) - \Psi(\boldsymbol{F}^{\star\star})$. Together with the previous theorem we easily validate the assertion.

The above two theorems guarantee that the alignment learning attains a linear convergence rate.

## 6 The experiments

We conducted experiments to evaluate the effectiveness of the proposed EGUMA method. The EGUMA was used for performing for matching on the protein data set. Moreover, we utilized applications for recognizing video faces and domain adaptation. These experiments demonstrate the general use of our method.

### 6.1 EGUMA for set matching

We choose manifold data from the bio information science to find structure matching of Glutaredoxin protein PDB 1G7O [18]. The protein molecule has 215 amino acids, and different structures originates from the same structure, which suggests manifold shapes different appearances in different configuration spaces. The alignment result verifies the algorithm performance in unveiling the intrinsic data representations. As shown in Figure 2, we get the alignment patterns in 1D, 2D and 3D subspaces of the two protein 1G7O-1 and 1G7O-10. Though Wang et al. [12] and Pei et al. [21] achieved adequate matching accuracy with only local geometry matching, the EGUMA method achieves better performance by discovering global underlying features.

### 6.2 EGUMA for video-based face verification

We assume that a set of varying face images belonging to the same person composes a high dimensional manifold. We use the above assumption to verify whether manifolds constructed by frames of two videos

**Table 1** The comparisons on YouTube faces data set (%)

| Method | Mean accuracy | Standard error |
|---|---|---|
| MSM | 62.54 | ±1.47 |
| MMD | 64.96 | ±1.00 |
| AHISD | 66.50 | ±2.03 |
| CHISD | 66.24 | ±1.70 |
| SANP | 63.74 | ±1.69 |
| DCC | 70.84 | ±1.57 |
| Unaligned | 61.80 | ±2.29 |
| GUMA(un) | 65.52 | ±1.92 |
| GUMA(su.) | 71.60 | ±1.57 |
| EGUMA(su.) | **72.20** | ±2.12 |

belonging to the same person coincide or not. When the elements are out of order, it is challenging to distinguish superficial mismatch of the same manifold or intrinsic variation of different manifolds. Aligning before comparison will alleviate this problem, and we use this method to check up the alignment algorithms vice versa. Two videos are aligned at first, and then the distance between the two given videos is calculated from the averaged distances of corresponding points from either videos. The computed distance is used to distinguish videos taken from the same or different identities. We demonstrate that our alignment process achieves an improved accuracy in this application.

We use YouTube faces dataset collected by [34] to verify our method. It contains 3425 clips extracted from the YouTube website. Many researchers use the database to demonstrate and validate video-based face recognition algorithms. We follow the set ups in [35] and then compress the images to $40\times24$ pixels with faces in the middle. Then histogram equalization is used to suppress illumination effects. After data preparation, we applied the EGUMA method on two different videos. We computed the Euclidean distances of the counterparts as their differences. To further demonstrate the effectiveness of EGUMA, we implement several classic video based face recognition methods as comparison, including MSM by [36], MMD by [37], AHISD by [38], CHISD by [38], SANP by [39] and DCC by [40]. To analyze the potentials of these methods, we adopted the implementations as released by their respective authors and selected the best results with proper tuning of the parameters as in the original papers. Table 1 shows that the unsupervised alignment process boosts the accuracy to 65.74% from 61.80%. In the case of the supervised alignment process, EGUMA outperformed related DCC and GUMA methods. The discovery of the manifold subspace helped in achieving high performance.

### 6.3 EGUMA for domain adaptation

Domain adaptation tasks are generally in need of the discovery of the correspondences between the source domain and the target domain. In unsupervised domain adaptation scenarios, we regard both source domain and target domain as manifolds. We apply the EGUMA method based on two observations. (1) The embedded space accommodate cross domains and their respective domain. We should preserve graph structures constructed by respective data. (2) Retrieval benefit from searching data neighborhood such as datum falling in the same query slot, whose neighbor data are more likely to be the same kind. We then use EGUMA to align two manifolds from cross domains and discover their respective embedded space. Using embeddeding information in the source domain, we classify the unlabeled examples and extract the category information of the target domain. In this subsection, we perform two group experiments; namely visual domain adaptation and cross model retrieval.

#### 6.3.1 *Visual domain adaptation*

We used four classical data sets for domain adaptation: Amazon, Webcam, DSLR collected in [41], and Caltech 256 by [42]. We set the data in the protocol of [41,43–45]. Furthermore, we extracted SURF [46] and listed 800-bin token frequency features from each image following the pretrained Amazon images codebook. Then we normalize the features and get them z-scored per dimension with zero mean and standard deviation of one. Each dataset is taken as the source domain at a time, so we made a total of 12 groups of domain adaptation experiments. We randomly selected 20 samples from each class in Amazon, Webcam, and DSLR, then eight samples in each class from Caltech-256. All the samples in the target

**Table 2** Comparisons of visual domain adaptation accuracy

| Models | OriFea | SGF(PCA) | SGF(PLS) | GFK(PCA) | GFK(PLS) | ITL | SA | GUMA | EGUMA |
|---|---|---|---|---|---|---|---|---|---|
| $C \to A$ | 27.79 | 37.90 | 44.34 | 37.32 | 40.65 | 37.85 | 36.59 | 46.22 | **49.19** |
| $C \to D$ | 25.73 | 38.41 | 39.20 | 36.40 | 43.41 | 39.33 | 38.28 | 44.52 | **46.11** |
| $A \to C$ | 24.58 | 33.62 | 36.93 | 31.13 | 34.10 | 33.82 | 33.01 | **36.08** | 36.03 |
| $A \to W$ | 26.97 | 33.81 | 33.59 | 30.95 | 34.20 | 33.56 | 33.47 | 35.90 | **37.36** |
| $A \to D$ | 22.64 | 34.81 | 34.46 | 32.32 | 37.07 | 34.33 | 32.32 | 37.48 | **38.09** |
| $W \to C$ | 19.29 | 28.88 | 29.10 | 30.45 | 31.59 | 30.57 | 31.98 | **34.04** | **34.04** |
| $W \to A$ | 20.10 | 32.58 | 32.76 | 34.74 | 35.97 | 33.05 | 32.17 | **36.92** | **36.92** |
| $W \to D$ | 50.35 | 74.20 | 65.67 | 72.74 | 73.57 | 76.82 | **77.73** | 72.83 | 72.45 |
| $D \to C$ | 24.81 | 31.80 | 31.62 | 31.26 | 32.37 | 31.51 | 32.39 | **35.33** | **35.33** |
| $D \to A$ | 28.21 | 33.56 | 32.91 | 33.99 | **36.66** | 33.40 | 34.40 | 35.19 | 35.19 |
| $D \to W$ | 61.10 | 82.22 | 73.42 | 82.85 | **84.85** | 83.92 | 84.30 | 82.44 | 83.00 |
| Average | 29.77 | 41.33 | 40.88 | 40.66 | 43.68 | 41.88 | 41.84 | 44.98 | **45.73** |

domain are unlabeled, and the task is to predict their labels as in [43,44]. In every group of experiments, we performed 20 rounds with different initial random samples.

Five baseline methods were used to compare the performance of the EGUMA, including OriFea (original features), sampling geodesic flow (SGF) [43], geodesic flow kernel (GFK) [44], information theoretical learning (ITL) [47], and subspace alignment (SA) [48]. The latter four methods are the state of the art unsupervised domain adaptation methods. SGF and its extended version GFK interpolate the intermediate domains between source and target to learn invariant features; ITL is an efficient unsupervised domain adaptation method, and SA formulates each domain as a subspace and tries to align its principal directions. We use the source codes published by the original authors except for ITL, which we implemented. To compare these methods without bias, we use the best tuned parameters to achieve peak performance. We use the NN classifier to predict the sample labels of the target domain. Note that we use SGF(PLS) and GFK(PLS) to label the methods nesting a partial least square (PLS) algorithm to get discriminant information. In the implementation of our method, we first perform clustering on the data of each class for the source domain. Then, we aggregate all unlabeled samples of the target domain, where the number of clusters is estimated using the Jump method [49]. The proposed method obtains stable sample points from high dimensional spaces, and these representative points benefit the successive manifold alignment.

A complete report of all experiment groups is demonstrated in Table 2. The EGUMA method performed better than the other methods, and it achieved the highest accuracy of 7 out of 12 cases. The accuracy of the EGUMA method is 45.73%, which is better than the next best result of 44.98% achieved by GUMA on average of all groups. Generally, all domain adaptation methods performed well based on original features. This phenomenon indicates manifold alignment can be applied to domain adaptation. Further, we assert that manifold structures rather than the linear subspaces are more capable of discovering correspondence between domains.

### 6.3.2 *Cross modal retrieval*

The more challenging task is to discover correspondence in cross media data sets. Given data from one domain, for example, a query image file, the challenge is to retrieve the corresponding data in another space, such as text file, to describe the query image. Though different modalities often lie on different feature spaces, their underlying semantic variants rely on similar manifolds. In this subsection, we present three experiments on different open source data sets to demonstrate the capability of the EGUMA to find common spaces of different manifolds.

Using the proposed method, we observed improved performance for both the image query texts and text query images tasks. For comparison, we also evaluate several popular ad hoc methods such as CCA from [28], SCM from [50], and GMLDA and GMMFA from [51]. The validation datasets include Wiki from [50][1)], Pascal VOC 2007 from [52][2)] and NUS WIDE from [53][3)]. These datasets are composed of the training and test data. The training data are labeled while the test data are not. We align the

1) http://www.svcl.ucsd.edu/projects/crossmodal.
2) http://host.robots.ox.ac.uk/pascal/VOC/.
3) http://lms.comp.nus.edu.sg/research/NUS WIDE.htm.

**Table 3** Cross modal retrieval performances on Wiki dataset

| Method | Task | | | | | |
|---|---|---|---|---|---|---|
| | $R = 50$ | | | $R = \text{all}$ | | |
| | Text query | Image query | Average MAP | Text query | Image query | Average MAP |
| CCA | 39.19 | 34.98 | 37.09 | 26.66 | 31.71 | 29.19 |
| LCFS | 34.41 | 26.11 | 30.26 | 19.68 | 25.85 | 22.76 |
| SCM | 48.56 | 36.40 | 42.48 | 31.67 | 35.37 | 33.52 |
| GMLDA | 45.69 | 35.63 | 40.66 | 29.84 | 35.05 | 32.44 |
| GMMFA | **50.10** | 35.41 | 42.76 | 30.71 | 33.72 | 32.21 |
| EGUMA(CM) | 39.40 | 35.08 | 37.24 | 26.91 | 31.91 | 29.41 |
| EGUMA(SCM) | 49.29 | **37.69** | **43.48** | **31.94** | **36.26** | **34.10** |

**Table 4** Cross modal retrieval performances on Pascal VOC dataset

| Method | Task | | | | | |
|---|---|---|---|---|---|---|
| | $R = 50$ | | | $R = \text{all}$ | | |
| | Text query | Image query | Average MAP | Text query | Image query | Average MAP |
| CCA | 32.02 | 26.20 | 29.11 | 17.73 | 20.10 | 18.92 |
| SCM | 43.18 | 30.96 | 37.08 | 25.47 | 28.24 | **26.86** |
| GMMFA | 28.92 | 19.32 | 24.12 | 19.60 | 22.33 | 20.96 |
| EGUMA(CM) | 32.68 | 26.35 | 29.51 | 17.97 | 20.43 | 19.20 |
| EGUMA(SCM) | **43.21** | **30.97** | **37.09** | **25.48** | **28.25** | **26.86** |

**Table 5** Cross modal retrieval performances on NUS_WIDE dataset

| Method | Task | | | | | |
|---|---|---|---|---|---|---|
| | $R = 50$ | | | $R = \text{all}$ | | |
| | Text query | Image query | Average MAP | Text query | Image query | Average MAP |
| CCA | 44.31 | 33.91 | 39.11 | 24.37 | 30.39 | 27.38 |
| GMLDA | **64.64** | 24.68 | 44.66 | 36.46 | 27.88 | 32.17 |
| SCM | 54.61 | 35.69 | 45.15 | 31.69 | 39.22 | 35.46 |
| EGUMA(CM) | 45.33 | 33.79 | 39.56 | 25.08 | 31.26 | 28.16 |
| EGUMA(SCM) | 54.58 | **35.97** | **45.28** | **31.69** | **39.36** | **35.54** |

training data and get a reliable embedding function, and apply it to the test part to perform the cross modal retrieval. During this process, we set the alignment matrix $\boldsymbol{F}$ fixed for the ordered sample. The unsupervised manner is marked as 'EGUMA(CM)' whereas 'EGUMA(SCM)' to represent the process with labels used. Because of the comparison, we set the same training samples and test samples on each data set for all the methods. We use the deep features of Wiki and NUS_WIDE datasets extracted from the AlexNet $fc-7$ layer, pretrained with ImageNet mentioned by [54].

For Wiki, 2173 samples were selected randomly as training data and 693 samples as test data to avoid extrema. On Pascal VOC, the scale is 2808 as training data and 2841 for test data, while in NUS_WIDE, the number is 4353 and 2925 for training and test data, respectively. The mean average precision (MAP) mentioned by [50] is used as the evaluation index. To get a comprehensive evaluation, a neighbor group that 50 of all samples are selected for each sample retrievals perching on the cosine distance at the test stage and marked $R = 50$. Then mark $R =$ all for MAP get from all retrieved samples. This evaluation method is mentioned by [55] to compare the retrieval accuracy tendency, where the difference between $R = 50$ and $R =$ all signifies the distribution of retrieval. In Wiki, a sample is considered retrieved if it falls into the semantic class as the query data. While in Pascal and NUS_WIDE a sample is retrieved if it touches at least one labeled concept among the classes of query data. From Tables 3–5, we can observe that the EGUMA method performed well on all three datasets. The baseline is provided by CCA and SCM, depicting a significant gain from labels. The MAP score of the EGUMA method is higher than non semantic methods such as 'CCA' and 'LCFS', and 'EGUMA(SCM)' is better than all other methods. Preserving the neighborhood relationship in the data graph helps the EGUMA method achieve improved performance, or the Laplacian matrix term from the mathematical point of view. We figure out the confusion matrix for our proposed method in Figure 3. The darker shadowed elements show closer correlations. Generally speaking, a projection considering graph structure is useful to bridge data from

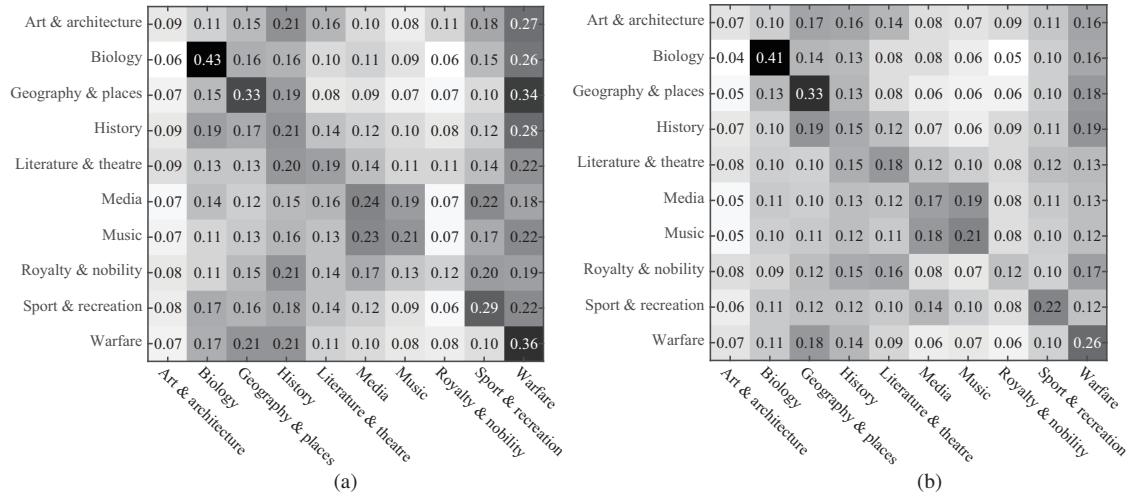


**Figure 3** Image-query-text and text-query-image MAP scores of our method. (a) Image-query-text and (b) text-query-image MAP scores of our method.

**Table 6** Comparisons of alignment accuracy of face image sets from Multi-PIE [56]

| Models | Wang's | Pei's | GUMA(F) | GUMA(S) | GUMA | EGUMA |
|---|---|---|---|---|---|---|
| Pose | 15.70 | 82.23 | 69.02 | 35.35 | 86.78 | **87.16** |
| Expression | 33.06 | 61.99 | 51.00 | 18.78 | 72.68 | **74.38** |
| Both | 5.20 | 52.58 | 37.09 | 6.81 | 66.23 | **67.86** |

different domains. Further, we assert that manifold structures rather than the linear subspaces are more capable of discovering correspondence between domains.

### 6.4 Ablation study

We perform ablation study through face image sets alignment, where the data sets to be aligned contain faces with different poses, expressions, and illuminations. The results on pose/expression matching are shown in Table 6. We quantitatively evaluate the accuracy of poses matching, expressions corresponding, and their combination. Our method is compared with two methods as Wang's [12] and Pei's [21]. We report that the accuracies of a single element affected both of feature and structure matching, and denote them as GUMA(F) and GUMA(S), respectively. Comparing GUMA(F), GUMA(S), GUMA, and EGUMA, the matching accuracy is increasing accordingly. This phenomenon suggests that alignment first benefits from matches of features and then matches of geometry structures. Their combination is better than either of them.

## 7 Conclusion

We raise a generalized unsupervised manifold alignment method in this paper, which aims at finding the coherences and discovering the embedding manifolds. The unsupervised manifold alignment problem is formulated as a relaxed 0-1 optimization, and take considerations of the match on global manifold structures and underlying features. The real domain where alignment happens guarantees the completeness in mathematics, which promises the validation of the gradient decent method during the process of the optimization. We adopt two submodules to get the final solution. One of them finds matches both in the appearance space and the embedding subspace, and the other is learning transformations to maintain consistency between the original data and the respective projection representatives. To perform the learning match submodule, FW algorithm is broadened to optimize the objective function, which suggests broad applications of the FW algorithm for optimizing quadratic programming problems. Strict mathematical proof is provided to ensure the convergence and the effectiveness of our algorithm. We perform experiments on matching data sets, recognizing video faces and adapting visual domains and the results manifest the effectiveness as well as practicability of the EGUMA. Some tasks are left for future endeavors. As the data representation is crucial in the manifold alignment issue, we would further study

the representation learning to improve the alignment accuracy. Also we will further generalize EGUMA into nonlinear situations to obtain wider applications.

**Acknowledgements** This work was partially supported by National Natural Science Foundation of China (Grant Nos. 61876171, 61976203) and Open Project Fund from Shenzhen Institute of Artificial Intelligence and Robotics for Society (Grant No. AC01202005015).

**References**

1 Li Z, Tang J. Unsupervised feature selection via nonnegative spectral analysis and redundancy control. IEEE Trans Image Process, 2015, 24: 5343–5355
2 Li Z, Tang J, He X. Robust structured nonnegative matrix factorization for image representation. IEEE Trans Neural Netw Learn Syst, 2018, 29: 1947–1960
3 Li Z C, Liu J, Yang Y, et al. Clustering-guided sparse structural learning for unsupervised feature selection. IEEE Trans Knowl Data Eng, 2014, 26: 2138–2150
4 Wang J, Feng W, Chen Y, et al. Visual domain adaptation with manifold embedded distribution alignment. In: Proceedings of ACM International Conference on Multimedia, 2018. 402–410
5 Peng X, Bai Q, Xia X, et al. Moment matching for multi-source domain adaptation. In: Proceedings of IEEE International Conference on Computer Vision, 2019. 1406–1415
6 Ji P, Qin D, Feng P, et al. Manifold alignment-based radio map construction in indoor localization. In: Proceedings of International Conference on Machine Learning and Intelligent Communications, 2018. 327–337
7 Tenenbaum J B, de Silva V, Langford J C. A global geometric framework for nonlinear dimensionality reduction. Science, 2000, 290: 2319–2323
8 Zhang K, Kwok J T. Clustered Nyström method for large scale manifold learning and dimension reduction. IEEE Trans Neural Netw, 2010, 21: 1576–1587
9 Roweis S T, Saul L K. Nonlinear dimensionality reduction by locally linear embedding. Science, 2000, 290: 2323–2326
10 van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res, 2008, 9: 2579–2605
11 Chen J, Ma Z M, Liu Y. Local coordinates alignment with global preservation for dimensionality reduction. IEEE Trans Neural Netw Learn Syst, 2013, 24: 106–117
12 Wang C, Mahadevan S. Manifold alignment without correspondence. In: Proceedings of International Joint Conference on Artificial Intelligence, 2009
13 Abeo T A, Shen X J, Ganaa E D, et al. Manifold alignment via global and local structures preserving PCA framework. IEEE Access, 2019, 7: 38123–38134
14 Ham J, Lee D D, Saul L K. Semisupervised alignment of manifolds. In: Proceedings of International Conference on Artificial Intelligence and Statistics, 2005. 120: 27
15 Shon A, Grochow K, Hertzmann A, et al. Learning shared latent structure for image synthesis and robotic imitation. In: Proceedings of Neural Information Processing Systems, 2006. 1233–1240
16 Lafon S, Keller Y, Coifman R R. Data fusion and multicue data matching by diffusion maps. IEEE Trans Pattern Anal Mach Intell, 2006, 28: 1784–1797
17 Xiong L, Wang F, Zhang C. Semi-definite manifold alignment. In: Proceedings of European Conference on Machine Learning, Berlin, 2007. 773–781
18 Wang C, Mahadevan S. Manifold alignment using procrustes analysis. In: Proceedings of International Conference on Machine Learning, 2008. 1120–1127
19 Wang C, Mahadevan S. Heterogeneous domain adaptation using manifold alignment. In: Proceedings of International Joint Conference on Artificial Intelligence, Barcelona, 2011
20 Wang C, Cao L, Fan J. Building joint spaces for relation extraction. In: Proceedings of International Joint Conference on Artificial Intelligence, New York, 2016. 2936–2942
21 Pei Y R, Huang F C, Shi F H, et al. Unsupervised image matching based on manifold alignment. IEEE Trans Pattern Anal Mach Intell, 2012, 34: 1658–1664
22 Cui Z, Shan S, Zhang H, et al. Image sets alignment for video-based face recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2012. 2626–2633
23 Cui Z, Chang H, Shan S, et al. Generalized unsupervised manifold alignment. In: Proceedings of Neural Information Processing Systems, 2014. 2429–2437
24 Frank M, Wolfe P. An algorithm for quadratic programming. Naval Res Logist, 1956, 3: 95–110
25 Quadrianto N, Song L, Smola A J. Kernelized sorting. In: Proceedings of Neural Information Processing Systems, 2009. 1289–1296
26 Haghighi A, Liang P, Berg-Kirkpatrick T, et al. Learning bilingual lexicons from monolingual corpora. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, 2008. 771–779
27 Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput, 2003, 15: 1373–1396
28 Hardoon D R, Szedmak S, Shawe-Taylor J. Canonical correlation analysis: an overview with application to learning methods. Neural Comput, 2004, 16: 2639–2664
29 Spielman D A, Teng S H. Smoothed analysis of algorithms. J ACM, 2004, 51: 385–463
30 Dantzig G B, Thapa M N. Linear Programming 1: Introduction. Berlin: Springer, 2006
31 Abraham R, Marsden J E, Ratiu T. Manifolds, Tensor Analysis, and Applications. Berlin: Springer, 2012
32 Beck A, Tetruashvili L. On the convergence of block coordinate descent type methods. SIAM J Optim, 2013, 23: 2037–2060
33 Tewari A, Ravikumar P K, Dhillon I S. Greedy algorithms for structurally constrained high dimensional problems. In: Proceedings of Neural Information Processing Systems, 2011. 882–890
34 Wolf L, Hassner T, Maoz I. Face recognition in unconstrained videos with matched background similarity. In: Proceedings of Computer Vision & Pattern Recognition, 2011
35 Cui Z, Li W, Xu D, et al. Fusing robust face region descriptors via multiple metric learning for face recognition in the wild. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Portland Oregon, 2013. 3554–3561
36 Yamaguchi O, Fukui K, Maeda K. Face recognition using temporal image sequence. In: Proceedings of the 3rd International Conference on Face & Gesture Recognition, Nara, 1998. 318–323
37 Wang R, Shan S, Chen X, et al. Manifold-manifold distance with application to face recognition based on image set. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2008. 1–8

38 Cevikalp H, Triggs B. Face recognition based on image sets. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, 2010. 2567–2573
39 Hu Y, Mian A S, Owens R. Sparse approximated nearest points for image set classification. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2011. 121–128
40 Kim T K, Kittler J, Cipolla R. Discriminative learning and recognition of image set classes using canonical correlations. IEEE Trans Pattern Anal Mach Intell, 2007, 29: 1005–1018
41 Saenko K, Kulis B, Fritz M, et al. Adapting visual category models to new domains. In: Proceedings of European Conference on Computer Vision, Berlin, 2010. 213–226
42 Griffin G, Holub A, Perona P. Caltech-256 Object Category Dataset. Technical Report. Pasadena: California Institute of Technology, 2007
43 Gopalan R, Li R, Chellappa R. Domain adaptation for object recognition: an unsupervised approach. In: Proceedings of IEEE International Conference on Computer Vision, Barcelona, 2011. 999–1006
44 Gong B, Shi Y, Sha F, et al. Geodesic flow kernel for unsupervised domain adaptation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2012. 2066–2073
45 Cui Z, Li W, Xu D, et al. Flowing on riemannian manifold: domain adaptation by shifting covariance. IEEE Trans Cybern, 2014, 44: 2264–2273
46 Bay H, Tuytelaars T, van Gool L. Surf: speeded up robust features. In: Proceedings of European Conference on Computer Vision, Berlin, 2006. 404–417
47 Shi Y, Sha F. Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. 2012. ArXiv:1206.6438
48 Fernando B, Habrard A, Sebban M, et al. Unsupervised visual domain adaptation using subspace alignment. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2013. 2960–2967
49 Sugar C A, James G M. Finding the number of clusters in a dataset. J Am Statistical Association, 2003, 98: 750–763
50 Rasiwasia N, Costa Pereira J, Coviello E, et al. A new approach to cross-modal multimedia retrieval. In: Proceedings of ACM International Conference on Multimedia, 2010. 251–260
51 Sharma A, Kumar A, Daume H, et al. Generalized multiview analysis: a discriminative latent space. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2012. 2160–2167
52 Everingham M, van Gool L, Williams C K I, et al. The pascal visual object classes (VOC) challenge. Int J Comput Vis, 2010, 88: 303–338
53 Chua T S, Tang J, Hong R, et al. NUS-WIDE: a real-world web image database from National University of Singapore. In: Proceedings of the ACM International Conference on Image and Video Retrieval, 2009. 48
54 Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. In: Proceedings of Neural Information Processing Systems, 2012. 1097–1105
55 Wu F, Lu X, Zhang Z, et al. Cross-media semantic representation via bi-directional learning to rank. In: Proceedings of ACM International Conference on Multimedia, 2013. 877–886
56 Gross R, Matthews I, Cohn J, et al. Multi-PIE. Image Vision Comput, 2010, 28: 807–813
57 Laub A J. Matrix Analysis for Scientists and Engineers. Philadelphia: SIAM, 2005
58 Boyd S, Vandenberghe L. Convex Optimization. Cambridge: Cambridge University Press, 2004
59 Tseng P. Convergence of a block coordinate descent method for nondifferentiable minimization. J Opt Theor Appl, 2001, 109: 475–494


## Appendix A Notations appeared in the main text

Bold lowercase (uppercase) letters denote vectors (matrix), while non-bold ones as scalars. $[a, b]$ refers to the interval between a scaler $a \in \mathbb{R}$ and a scaler $b \in \mathbb{R}$. $a!$ denotes the factorial of a scaler $a$. $\boldsymbol{X}_{i\cdot}$ ($/\boldsymbol{X}_{\cdot i}$) represents the $i$th row (/column) of matrix $\boldsymbol{X}$. $\boldsymbol{X}^k$ represents the $k$th update step of matrix $\boldsymbol{X}$. $\mathbf{1}_{m\times n}, \mathbf{0}_{m\times n} \in \mathbb{R}^{m\times n}$ are matrices of ones and zeros. $\boldsymbol{I}_n \in \mathbb{R}^{n\times n}$ is an identity matrix. $\mathrm{tr}(\cdot)$ means the trace. The superscript $^{\mathrm{T}}$ denotes the transpose of a vector or matrix. For a matrix $\boldsymbol{X}$ and a space $\mathscr{P}$, we use $\|\boldsymbol{X}\|_{\mathscr{P}}$ to denote induced norm, while $\|\boldsymbol{X}\|_F^2 = \mathrm{tr}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})$ designates the Frobenius norm. Note that it does not equal to induced norm of matrix $\boldsymbol{F}$. $\mathrm{vec}(\boldsymbol{X})$ represents the vectorization of matrix $\boldsymbol{X}$ in columns. $\mathrm{diag}(\boldsymbol{X})$ denotes the sequence retrieved from diagonal positions of the matrix $\boldsymbol{X}$, and $\mathrm{diag}(\boldsymbol{X})$ denotes a diagonal matrix of the diagonal elements $\boldsymbol{X}$. $\mathrm{eig}(\boldsymbol{X})$ retains all eigenvalues of $\boldsymbol{X}$. $\boldsymbol{X} \otimes \boldsymbol{Y}$ and $\boldsymbol{X} \odot \boldsymbol{Y}$ denote the Kronecker and Hadamard products, respectively. $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle$ represents the inner product of vector $\boldsymbol{X}$ and $\boldsymbol{Y}$, and $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle$ represents the inner product of the vectorized matrix $\boldsymbol{X}$ and $\boldsymbol{Y}$ after vectorization.

## Appendix B Matching algorithm

**Algorithm B1** Match on multiple manifolds

**Require:** $\boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j, \boldsymbol{F}^0$.
Initialize: $\boldsymbol{F}^\star = \boldsymbol{F}^0, k = 0$.
**repeat**
  (Re)calculate $\boldsymbol{K}_i, \boldsymbol{K}_j$ using the equation;
  Calculate the derivation of $\Psi$ over $\boldsymbol{F}^k$ and get $\nabla\Psi(\boldsymbol{F}^k)$;
  Minimize the Taylor expansion of the objective function $\Psi$ up to order one, $\boldsymbol{H} = \arg\min_{\boldsymbol{F}\in\Pi}\mathrm{tr}(\nabla\Psi(\boldsymbol{F}^k)^{\mathrm{T}}\boldsymbol{F})$ using the the simplex algorithm;
  **if** $\Psi(\boldsymbol{H}_{ij}) < \Psi(\boldsymbol{F}^k)$ **then**
    $\boldsymbol{F}^\star = \boldsymbol{F}^{k+1} = \boldsymbol{H}$;
  **else**
    $\delta = \underset{0\leqslant\delta\leqslant 1}{\operatorname{argmin}}\ \Psi(\boldsymbol{F}^k + \delta(\boldsymbol{H} - \boldsymbol{F}^k))$;
    $\boldsymbol{F}^{k+1} = \boldsymbol{F}^k + \delta(\boldsymbol{H} - \boldsymbol{F}^k)$;
    $\boldsymbol{F}^\star = \underset{\boldsymbol{F}\in\{\boldsymbol{H},\boldsymbol{F}^\star\}}{\operatorname{argmin}}\ \Psi(\boldsymbol{F})$;
  **end if**
  $k = k + 1$;
**until** $\|\Psi(\boldsymbol{F}^{k+1}) - \Psi(\boldsymbol{F}^k)\| < \epsilon$;
Find the best match result in $\boldsymbol{X}$ of every element in $\boldsymbol{Z}$ with Hungarian algorithm, denoted as $\boldsymbol{V}$;
**Ensure:** $\boldsymbol{F}^\star, \boldsymbol{V}$.

## Appendix C Manifold alignment process

**Algorithm C1** EGUMA algorithm

**Require:** $N$ data sets $\boldsymbol{X}_i$ for $i = 1, 2, \ldots, N$, dimension $d$ of embedding space.
Initialize: matching matrix $\boldsymbol{F} \in \Pi$ and linear transforms $\boldsymbol{P}_i, \boldsymbol{P}_j$;
**repeat**
  Compute $\boldsymbol{K}_i, \boldsymbol{K}_j$ with global geometry structures;
  Solve $\boldsymbol{F}$, $\boldsymbol{V}$ by Algorithm B1;
  Solve $\boldsymbol{P}_i, \boldsymbol{P}_j$ by eigenvalue decomposition on (16);
**until** convergence;
**Ensure:** $\boldsymbol{F}, \boldsymbol{V}$ and $\boldsymbol{P}_i, \boldsymbol{P}_j$.

## Appendix D The derivation algorithm of the geodesic distance

**Algorithm D1** Calculate the geodesic distance

**Require:**
Initialize: number of nearest neighbor $K$, $\boldsymbol{D} = \boldsymbol{D_0}, i = j = 0$, where $\boldsymbol{D}_0$ is the Euclidean distance of points in the graph $\boldsymbol{G}$.
**repeat**
  Find a pair of $i,\ j$ that not processed;
  **if** Point $i$ is one of the $K$ nearest neighbors of point $j$ **then**
    $\boldsymbol{D}(i,j) = \boldsymbol{D_0}(i,j)$;
  **else**
    $\boldsymbol{D}(i,j) = \infty$;
  **end if**
**until** Processed with all $i, j$ pairs;
Initialize $i, j = 0$;
**repeat**
  For each $i,\ j$ pairs;
  Replace all entries of $\boldsymbol{D}(i,j)$ by $\min\{\boldsymbol{D}(i,j), \boldsymbol{D}(i,k) + \boldsymbol{D}(k,j)\}$ for $\forall k$;
**until** Processed with all $i, j$ pairs;
**Ensure:** $\boldsymbol{D}$ as the geodesic distance matrix.

## Appendix E Derivation of (7)

Let $\hat{\boldsymbol{X}}_i = \boldsymbol{\Phi}_i(\boldsymbol{X}_i)$ and $\hat{\boldsymbol{X}}_j = \boldsymbol{\Phi}_j(\boldsymbol{X}_j)$ denote the features of data set $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ respectively, and $\hat{\boldsymbol{F}} = \boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} \in [0,1]^{n_j\times n_j}$. We derive (6) as the following:

$$\begin{aligned}\Psi_0(\boldsymbol{F}) =& E_s + \gamma_f E_f\\ =& \|\boldsymbol{K}_i - \boldsymbol{F}\boldsymbol{K}_j\boldsymbol{F}^{\mathrm{T}}\|_F^2 + \gamma_f\|\hat{\boldsymbol{X}}_i - \hat{\boldsymbol{X}}_j\boldsymbol{F}^{\mathrm{T}}\|_F^2\\ =& \mathrm{tr}(\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{K}_i + \hat{\boldsymbol{F}}\boldsymbol{K}_j\hat{\boldsymbol{F}}^{\mathrm{T}}\boldsymbol{K}_j - 2\boldsymbol{F}^{\mathrm{T}}\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}_j) + \gamma_f\mathrm{tr}(\hat{\boldsymbol{X}}_i\hat{\boldsymbol{X}}_i{}^{\mathrm{T}} + \hat{\boldsymbol{X}}_j\hat{\boldsymbol{F}}\hat{\boldsymbol{X}}_j^{\mathrm{T}} - 2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j\hat{\boldsymbol{F}}^{\mathrm{T}})\end{aligned}$$

$$
\begin{aligned}
&=\mathrm{tr}(\boldsymbol{K}_i\boldsymbol{K}_i^{\mathrm{T}}+\mathbf{1}^{\mathrm{T}}\boldsymbol{F}(\boldsymbol{K}_j\odot\boldsymbol{K}_j)\boldsymbol{F}^{\mathrm{T}}\mathbf{1}-2\boldsymbol{F}^{\mathrm{T}}\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}_j)+\gamma_f\mathrm{tr}(\hat{\boldsymbol{X}}_i\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}+\mathbf{1}^{\mathrm{T}}(\hat{\boldsymbol{X}}_j\odot\hat{\boldsymbol{X}}_j)\boldsymbol{F}^{\mathrm{T}}\mathbf{1}-2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j\hat{\boldsymbol{F}}^{\mathrm{T}})\\
&=\mathrm{tr}(\boldsymbol{K}_i\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{K}_i^{\mathrm{T}}+\mathbf{1}^{\mathrm{T}}\boldsymbol{F}(\boldsymbol{K}_j\odot\boldsymbol{K}_j)\boldsymbol{F}^{\mathrm{T}}\mathbf{1}-2\boldsymbol{F}^{\mathrm{T}}\boldsymbol{K}_i^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}_j)+\gamma_f\mathrm{tr}(\mathbf{1}^{\mathrm{T}}(\hat{\boldsymbol{X}}_j\odot\hat{\boldsymbol{X}}_j)\boldsymbol{F}^{\mathrm{T}}\mathbf{1}-2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j\hat{\boldsymbol{F}}^{\mathrm{T}})+\tau\\
&=\|\boldsymbol{K}_i\boldsymbol{F}-\boldsymbol{F}\boldsymbol{K}_j\|_F^2+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj}-\boldsymbol{K}_z^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}_j)+\gamma_f\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}(\mathbf{1}\mathbf{1}^{\mathrm{T}}\tilde{\boldsymbol{X}}_j-2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j))+\tau\\
&=\|\boldsymbol{K}_i\boldsymbol{F}-\boldsymbol{F}\boldsymbol{K}_j\|_F^2+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj}-\mathbf{1}^{\mathrm{T}}\boldsymbol{K}^{jj}\boldsymbol{F}^{\mathrm{T}}\mathbf{1})+\gamma_f\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}(\mathbf{1}\mathbf{1}^{\mathrm{T}}\tilde{\boldsymbol{X}}_j-2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j))+\tau\\
&=\|\boldsymbol{K}_i\boldsymbol{F}-\boldsymbol{F}\boldsymbol{K}_j\|_F^2+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj})+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}[\gamma_f(\mathbf{1}\mathbf{1}^{\mathrm{T}}\tilde{\boldsymbol{X}}_j-2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j)-\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{K}^{jj}])+\tau\\
&=\|\boldsymbol{K}_i\boldsymbol{F}-\boldsymbol{F}\boldsymbol{K}_j\|_F^2+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj})+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{G})+\tau,
\end{aligned}
\tag{E1}
$$

where $\boldsymbol{K}^{jj}=\boldsymbol{K}_j\odot\boldsymbol{K}_j$, $\tilde{\boldsymbol{X}}_j=\hat{\boldsymbol{X}}_j\odot\hat{\boldsymbol{X}}_j$, $\boldsymbol{G}=\gamma_f(\mathbf{1}\mathbf{1}^{\mathrm{T}}\tilde{\boldsymbol{X}}_j-2\hat{\boldsymbol{X}}_i{}^{\mathrm{T}}\hat{\boldsymbol{X}}_j)-\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{K}^{jj}$, and $\tau$ is a scaler. Then the optimization function (6) is identical to (7). Because we have assumed $n_x=n_z$, then $\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}=\boldsymbol{I}$, but not for $\hat{\boldsymbol{F}}=\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\neq\boldsymbol{I}$. Here we use a property of all-one-vectors that $\mathbf{1}^{\mathrm{T}}\boldsymbol{A}\mathbf{1}$ is equal to $\sum A_{i,j}$ for $\forall i,j$. Another math equation worth mentioning is $\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\hat{\boldsymbol{F}}\boldsymbol{A})=\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{A})=\mathbf{1}^{\mathrm{T}}(\boldsymbol{A}\odot\boldsymbol{A}\boldsymbol{F})\mathbf{1}$. A simple explanation is 'the trace of a matrix permuted by some order then squared is equal to the summation over the matrix squared then permuted by the order'.

## Appendix F Proof of the convexity of (8)

**Proposition 1** ([57]). Suppose $\boldsymbol{A}\in\mathbb{C}^{p\times p}$ with eigenvalues $\lambda_1,\lambda_2,\ldots,\lambda_p$ and $\boldsymbol{B}\in\mathbb{C}^{q\times q}$ with eigenvalues $\mu_1,\mu_2,\ldots,\mu_q$, then the eigenvalues of $\boldsymbol{A}\otimes\boldsymbol{B}$ are $\lambda_i\mu_j$, where $(i=1,2,\ldots,p;\ j=1,2,\ldots,q)$.

*Proof.* Let $\boldsymbol{X}$ be the eigenvector of $\boldsymbol{A}$ and $\boldsymbol{y}$ be the eigenvector of $\boldsymbol{B}$. Then

$$
(\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{X}\otimes\boldsymbol{y})=\boldsymbol{A}\boldsymbol{X}\otimes\boldsymbol{B}\boldsymbol{y}=\lambda\boldsymbol{X}\otimes\mu\boldsymbol{y}=\lambda\mu(\boldsymbol{X}\otimes\boldsymbol{y}).
\tag{F1}
$$

The eigenvector is $(\boldsymbol{X}\otimes\boldsymbol{y})$ and the eigenvalues are full permutations of elements in $\lambda$ and $\mu$, which completes the proof.

**Proposition 2** ([58]). We say a function $f$ convex iff. dom$f$ holds convexity, and the Hessian of the function is positive semi-definite:

$$
\forall x\in\mathrm{dom}f,\quad \nabla^2 f(x)\succeq 0.
$$

**Theorem 5.** Given $\Lambda=n_i\times\max\{-\min(\mathrm{eig}(\boldsymbol{K}^{jj})),0\}$, and denoted

$$
g(\boldsymbol{F})=\Lambda\mathrm{tr}(\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}),
\tag{F2}
$$

the new object function $\Psi(\boldsymbol{F})=\Psi_0(\boldsymbol{F})+g(\boldsymbol{F})$ is convex in the domain $\Pi$, and has the same optimal solution as $\Psi_0$.

*Proof.* Within the domain $\Pi$, $g(\boldsymbol{F})$ is a scaler for $\forall\boldsymbol{F}$. Then the new optimization object $\Psi(\boldsymbol{F})$ has the same optimal solution as $\Psi_0(\boldsymbol{F})$. Now we prove Eq. (8) is convex. In light of the above proposition, it is reasonable to leverage the convexity of a function by ensuring the positiveness of its Hessian matrix. The Hessian matrix of quadratic term $\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj})+\boldsymbol{\Lambda}\mathrm{tr}(\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}})$ is $\boldsymbol{K}^{jj\,\mathrm{T}}\otimes(\mathbf{1}\mathbf{1}^{\mathrm{T}})+\boldsymbol{\Lambda}\boldsymbol{I}_{n_i*n_j}$, where the term is positive definite when $\boldsymbol{\Lambda}=n_i\times\max\{-\min(\mathrm{eig}(\boldsymbol{K}^{jj})),0\}$. We derive this as follows. Suppose $\beta_1,\beta_2,\ldots,\beta_{n_j}$ are eigenvalues of $\boldsymbol{K}^{jj}$. We also calculate that $\underbrace{0,\ldots,0}_{n_i-1},n_i$ are eigenvalues of $\mathbf{1}\mathbf{1}^{\mathrm{T}}$, where $\mathbf{1}\in\mathbb{R}^{n_i\times 1}$. Then from proposition above, $\beta_1\times n_i,\beta_2\times n_i,\ldots,\beta_{n_j}\times n_i,\ \underbrace{0,\ldots,0}_{n_i\times n_j-n_i}$ are eigenvalues of $\boldsymbol{K}^{jj\,\mathrm{T}}\otimes(\mathbf{1}\mathbf{1}^{\mathrm{T}})$.

$\boldsymbol{\Lambda}$ is zero where $\mathrm{eig}(\boldsymbol{K}^{jj})$ are all positive; when some of $\mathrm{eig}(\boldsymbol{K}^{jj})$ are negative, suppose $\beta_p$ is the minimal eigenvalue of $\boldsymbol{K}^{jj}$, then $\boldsymbol{\Lambda}=-n_i\times\beta_p$, and $\mathrm{eig}(\boldsymbol{K}^{jj\,\mathrm{T}}\otimes(\mathbf{1}\mathbf{1}^{\mathrm{T}})+\boldsymbol{\Lambda}\boldsymbol{I}_{n_i\times n_j})$ are $(\beta_1-\beta_p)\times n_i,(\beta_2-\beta_p)\times n_i,\ldots,(\beta_p-\beta_p)\times n_i,\ldots,(\beta_{n_j}-\beta_p)\times n_i,\underbrace{-\beta_p\times n_i,\ldots,-\beta_p\times n_i}_{n_i\times n_j-nx}$, either equal or greater than zero, which means the Hessian matrix is positive semi-definite. That claims convexity of this term.

We use the definition to prove the convexity of the first term and define it as $h(\boldsymbol{F})=\|\boldsymbol{K}_i\boldsymbol{F}-\boldsymbol{F}\boldsymbol{K}_j\|_F^2$. $\forall\ \boldsymbol{F}^1,\boldsymbol{F}^2\in\Pi$, $\theta\in[0,1]$, we always have

$$
\begin{aligned}
h(\theta\boldsymbol{F}^1+(1-\theta)\boldsymbol{F}^2)&=\theta^2\|\boldsymbol{K}_i\boldsymbol{F}^1-\boldsymbol{F}^1\boldsymbol{K}_j\|_F^2\\
&\quad+(1-\theta)^2\|\boldsymbol{K}_i\boldsymbol{F}^2-\boldsymbol{F}^2\boldsymbol{K}_j\|_F^2\\
&\quad+2\theta(1-\theta)\|\boldsymbol{K}_i\boldsymbol{F}^1-\boldsymbol{F}^1\boldsymbol{K}_j\|_F\|\boldsymbol{K}_i\boldsymbol{F}^2-\boldsymbol{F}^2\boldsymbol{K}_j\|_F\\
&\leqslant\theta^2\|\boldsymbol{K}_i\boldsymbol{F}^1-\boldsymbol{F}^1\boldsymbol{K}_j\|_F^2+(1-\theta)^2\|\boldsymbol{K}_i\boldsymbol{F}^2-\boldsymbol{F}^2\boldsymbol{K}_j\|_F^2\\
&\quad+\theta(1-\theta)(\|\boldsymbol{K}_i\boldsymbol{F}^1-\boldsymbol{F}^1\boldsymbol{K}_j\|_F^2+\|\boldsymbol{K}_i\boldsymbol{F}^2-\boldsymbol{F}^2\boldsymbol{K}_j\|_F^2)\\
&=\theta\|\boldsymbol{K}_i\boldsymbol{F}^1-\boldsymbol{F}^1\boldsymbol{K}_j\|_F^2+(1-\theta)\|\boldsymbol{K}_i\boldsymbol{F}^2-\boldsymbol{F}^2\boldsymbol{K}_j\|_F^2
\end{aligned}
\tag{F3}
$$

which confirms the convexity of $h(\boldsymbol{F})$. The sign '$\leqslant$' is derived from fundamental inequality. For convexity is closed under addition, $\Psi(\boldsymbol{F})=h(\boldsymbol{F})+\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{K}^{jj})+\lambda\mathrm{tr}(\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}})$ is also convex. This completes the proof.

## Appendix G Proof of the convergence of the projection function in Algorithm C1

We have proved the convergence of $\{\boldsymbol{F}^1,\boldsymbol{F}^2,\ldots\}$ with a convergence rate of $\boldsymbol{O}(\frac{1}{k})$. For EGUMA takes a general BCD method as the rule of optimization, we will show a general theorem of BCD and then adapt to the EGUMA scenario. But before demonstrating an algorithm level proof we will begin with a mathematical deduction to guarantee $\P_i$ is bounded.

**Proposition 3** ([31]). If $\mathscr{P}$ is a finite dimensional normed space and $\boldsymbol{T}:\mathscr{P}\to\mathscr{Q}$ is linear, then $\boldsymbol{T}$ is continuous.

**Proposition 4** ([31]). Suppose $\boldsymbol{T}:\mathscr{P}\to\mathscr{Q}$ is linear between normed spaces $\mathscr{P}$ and $\mathscr{Q}$. Then $\boldsymbol{T}$ is continuous, iff. $\exists M>0$ s.t.,

$$\|\boldsymbol{T}\boldsymbol{p}\|_{\mathscr{Q}} \leqslant M\|\boldsymbol{p}\|_{\mathscr{P}} \text{ for } \forall \boldsymbol{p} \in \mathscr{P}.$$

**Theorem 6.** The operation in (16) is bounded.

*Proof.* The CCA in (16) is a linear transformation over $\boldsymbol{F}$, from the proposition above it is continuous in $\Pi$, thus $\exists M > 0$ s.t. $\mathrm{tr}\left(\boldsymbol{P}_i^{\mathrm{T}}\boldsymbol{X}\boldsymbol{F}\boldsymbol{X}_j^{\mathrm{T}}\boldsymbol{P}_j\right) \leqslant M\|\boldsymbol{F}\|$, which means the transformation is bounded.

**Algorithm G1** EGUMA algorithm as a standard BCD

1: Initialize: $\mathbb{U}^0 = (\boldsymbol{U}_1^0, \boldsymbol{U}_2^0)$, $t = 0$, compute auxiliary variables;
2: **repeat**
3: $\quad \boldsymbol{U}_1^{t+1} = \underset{\boldsymbol{U}_1}{\mathrm{argmin}}\mathrm{Obj}(\boldsymbol{U}_1, \boldsymbol{U}_2^t)$;
4: $\quad \boldsymbol{U}_2^{t+1} = \underset{\boldsymbol{U}_2}{\mathrm{argmin}}\mathrm{Obj}(\boldsymbol{U}_1^{t+1}, \boldsymbol{U}_2)$;
5: $\quad t = t + 1$;
6: **until** convergence
**Ensure:** $\mathbb{U}^{t-1}$.

Now we reformulate the EGUMA algorithm as a standard BCD method. Let $\mathbb{U}^t = (\boldsymbol{U}_1^t, \boldsymbol{U}_2^t)$ be the tuple of arguments at the $t$th iteration, where $\boldsymbol{U}_1 := \boldsymbol{F}$ and $\boldsymbol{U}_2 := (\boldsymbol{P}_i, \boldsymbol{P}_j)$. Let $\mathrm{Obj} := E_s + \gamma_f E_f + \gamma_p E_p$ be the object function. We notice that $\mathscr{U}^0 := \{\mathbb{U}|\mathrm{Obj}(\mathbb{U}) \leqslant \mathrm{Obj}(\mathbb{U}^0)\}$ will be bounded and Obj is lower semi-continuous (weaker than continuous and always hold for this situation), so it is compact. This minimization can be attained when Obj is convex, where is always true in our cases. The theorem below will be true from literature [59] but we will admit without proving owing to limitation of space.

**Theorem 7** ([32])**.** The level set $\mathscr{U}^0 = \{\mathbb{U}|\mathrm{Obj}(\mathbb{U}) \leqslant \mathrm{Obj}(\mathbb{U}^0)\}$ is compact. Obj is continuous on set $\mathscr{U}^0$. In use of the essentially cyclic rule, the BCD generating sequence $\{\mathbb{U}^t = (\boldsymbol{U}_1^t, \boldsymbol{U}_2^t)\}_{t=0,1,2,\ldots}$ is defined and bounded. Moreover, each point of $\{\mathbb{U}^t\}$ serves as a stationary point of Obj.

Then the EGUMA is to converge strictly. From literature [32] the BCD with conservative constant step size rule will converge at a sublinear rate. So the overall complexity will be $O(\frac{1}{\epsilon}n_j^2 \times n_i)$ using the assertion from Subsection 5.3.