Semi-definite Manifold Alignment

Liang Xiong, Fei Wang, and Changshui Zhang

Dept. Automation, Tsinghua University,
Beijing, China

{xiong1, feiwang03}@mails.tsinghua.edu.cn,
zcs@mail.tsinghua.edu.cn

Abstract. We study the problem of *manifold alignment*, which aims at “aligning” different data sets that share a similar intrinsic manifold provided some supervision. Unlike traditional methods that rely on pairwise correspondences between the two data sets, our method only needs some relative comparison information like “A is more similar to B than A is to C”. This method provides a more flexible way to acquire the prior knowledge for alignment, thus is able to handle situations where corresponding pairs are hard or impossible to identify. We optimize our objective based on the graphs that give discrete approximations of the manifold. Further, the problem is formulated as a *semi-definite programming* (SDP) problem which can readily be solved. Finally, experimental results are presented to show the effectiveness of our method.

1 Introduction

In machine learning, we are often faced with data with very high dimensionality. Directly dealing with these data is usually intractable due to the computational cost and *the curse of dimensionality*. In recent years, researchers have realized that in many situations the samples are confined to a low-dimensional manifold embedded in the feature space [1,2]. This intrinsic structure is of great value to facilitate the analysis and learning. Consequently, many methods have been developed to reveal data manifolds, such as *Locally Linear Embedding* [2], *Laplacian Eigenmaps* [3] and *Maximum Variance Unfolding* [4]. However, all these algorithms are *unsupervised*. Therefore, their results usually fail to reflect samples’ underlying parameters (*e.g.* the pose parameters for head images). Fortunately, provided some *supervised* information, we are able to develop methods that can reveal these parameters.

In this paper, we will focus on the problem of *manifold alignment*. More concretely, given some data sets sharing the same manifold structure, we seek to learn the correspondences between samples from different data sets (*e.g.* Finding different persons’ face images with the same pose). Besides its usage in data analysis and visualization, this problem also have wide range of applications. For instance, in *facial expression recognition*, one may have a set of labeled images with known expressions. Then we can recognize the expressions of another person by aligning his/her facial images to the standard image set. One can refer to [5] for more details.

Fig. 1. An example of data sharing the same manifold. Above are facial expressions from *JAFFE* [6]. The top and bottom rows show pairs with the same underlying facial expression.

Fig. 2. Three facial expression images. (a) and (c) is *surprised* and (b) is *neutral*. It is hard to make a confident decision that (a) and (c) have the same expression. However, it is obvious that (c) is more similar to (a) than (b) is.

There have already been some methods to align manifolds in a *semi-supervised* way [7,8,5,9,10]. Specifically, they assume that some pairwise correspondences of samples between data sets are known, and then use those information to guide the alignment. However, in practice it might be difficult to obtain and use such information since:

1. The sizes of data sets can be very large, then finding high-quality correspondences between them can be very time consuming and even intractable.
2. There may exist some ambiguities in the images (see Fig. 2 for an example), which makes explicit matching a hard task. Brutally determine and enforce these unreliable constraints may lead to poor results;
3. There may not exist exact correspondences. For example, this situation may happen when the data is restricted and users can only access a small subset.

To solve these problems, we propose to apply another type of supervision to guide manifold alignment. In particular, we consider a relative and qualitative supervision of the form “*A is closer to B than A is to C*”. We believe that this type of information is more easily available than traditional correspondence information. With this information, we show that the manifold alignment problem can be formulated as a *Quadratically Constrained Quadratic Programming (QCQP)* [11] problem. To make the solution tractable, we further relax it to a *Semi-Definite Programming (SDP)* [11] problem, which can be readily solved. Besides, under this formulation we can incorporate both relative relations and correspondences to align manifolds in a very flexible way. Finally experimental results are presented to show the effectiveness of our method.

The rest of this paper is organized as follows. Section 2 will introduce some basic notations and related works. The detailed algorithm will be presented in section 3. The experimental results will be provided in section 4, followed by the discussions in section 5 and conclusions in section 6.

2 Notations and Related Works

We study the problem of *aligning* different data sets that share the same underlying manifold. For the convenience of presentation, first let us consider the case of two data sets $\mathcal{X}$ and $\mathcal{Y}$ in high-dimensional vector spaces

$$\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \subset \mathbb{R}^{d_x}, \quad \mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\} \subset \mathbb{R}^{d_y}. \quad (1)$$

Manifold learning methods such as *Laplacian eigenmaps* [3] can learn the low-dimensional embeddings by constructing an *undirected weighted graph* that captures the local structure of data. For example, for $\mathcal{X}$ we can construct a graph $\mathcal{G}_{\mathcal{X}} = (\mathcal{V}_{\mathcal{X}}, \mathcal{E}_{\mathcal{X}})$, where $\mathcal{V}_{\mathcal{X}} = \mathcal{X}$ are the vertices and $\mathcal{E}_{\mathcal{X}}$ are the edges. Generally there is a nonnegative weight W_{ij} associated with each edge $e_{ij} \in \mathcal{E}_{\mathcal{X}}$, and all the edge weights form an $N \times N$ *weight matrix* $\mathbf{W}_{\mathcal{X}}$ with its (i, j) -th entry $\mathbf{W}_{\mathcal{X}}(i, j) = W_{ij}$. The *degree matrix* $\mathbf{D}_{\mathcal{X}}$ is an $N \times N$ diagonal matrix with the i -th diagonal entry $\mathbf{D}_{\mathcal{X}}(i, i) = \sum_j W_{ij}$, and the *combinatorial graph Laplacian* is defined as $\mathbf{L}_{\mathcal{X}} = \mathbf{D}_{\mathcal{X}} - \mathbf{W}_{\mathcal{X}}$.

The low-dimensional embeddings of the data in $\mathcal{X}$, say $\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_N] \in \mathbb{R}^{d \times N}$ ($d \ll d_{\mathcal{X}}$) can be achieved by minimizing the criterion $\mathcal{S}_{\mathcal{X}} = \text{tr}(\mathbf{F}\mathbf{L}_{\mathcal{X}}\mathbf{F}^T)$, where $\text{tr}(\cdot)$ represents the trace of a matrix. According to [3], $\mathcal{S}_{\mathcal{X}}$ measures the smoothness of the embeddings of $\mathcal{X}$ over the its underlying manifold. Similarly, we can define a graph $\mathcal{G}_{\mathcal{Y}} = (\mathcal{V}_{\mathcal{Y}}, \mathcal{E}_{\mathcal{Y}})$ for $\mathcal{Y}$ with its combinatorial graph Laplacian $\mathbf{L}_{\mathcal{Y}} = \mathbf{D}_{\mathcal{Y}} - \mathbf{W}_{\mathcal{Y}}$. And the low-dimensional embeddings of $\mathcal{Y}$, say $\mathbf{G} = [\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_N] \in \mathbb{R}^{d \times N}$ can also be achieved by minimizing $\mathcal{S}_{\mathcal{Y}} = \text{tr}(\mathbf{G}\mathbf{L}_{\mathcal{Y}}\mathbf{G}^T)$. Moreover, we can minimize the following combined criterion to achieve the common embeddings of both $\mathcal{X}$ and $\mathcal{Y}$

$$\mathcal{S} = \text{tr}(\mathbf{F}\mathbf{L}_{\mathcal{X}}\mathbf{F}^T) + \text{tr}(\mathbf{G}\mathbf{L}_{\mathcal{Y}}\mathbf{G}^T). \quad (2)$$

Now let's return to our *manifold alignment* problem. Assuming that we have known some pairwise correspondences $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^l$, we can align $\mathcal{X}$ and $\mathcal{Y}$ in a common low-dimensional space by minimizing [8]

$$\mathcal{J} = \mu \sum_{i=1}^l \|\mathbf{f}_i - \mathbf{g}_i\|^2 + \text{tr}(\mathbf{F}\mathbf{L}_{\mathcal{X}}\mathbf{F}^T) + \text{tr}(\mathbf{G}\mathbf{L}_{\mathcal{Y}}\mathbf{G}^T), \quad (3)$$

where μ is a regularization parameter to balance the embedding smoothness and the matching precision. When $\mu = \infty$, the pairwise correspondences will become hard constraints which impose $\mathbf{f}_i = \mathbf{g}_i$ after embedding [7,10]. However, as we explained in the introduction, it may be difficult to obtain the pairwise correspondences. Hence we propose a novel scheme for manifold alignment in this paper, which is based on relative comparisons among the data points.

3 Manifold Alignment Via Semi-definite Programming

3.1 The Quadratic Formulation

In this section we show how to correctly embed the data from different data sets into a common low-dimensional space with the guidance of supervised information.

Co-embedding Without Prior Knowledge. Following [8], we adopt the graph based criterion as our optimization objective. We construct weighted undirected graphs $\mathcal{G}_{\mathcal{X}}, \mathcal{G}_{\mathcal{Y}}$ for data sets $\mathcal{X}$ and $\mathcal{Y}$ respectively, and then seek a embedding which minimize Eq.(2). To avoid the illness of this problem, we impose the scale and translational invariance constraints. Then the *co-embedding* problem can be formulated as

$$\begin{aligned} \min_{\mathbf{F}, \mathbf{G}} \quad & tr(\mathbf{F}\mathbf{L}_{\mathcal{X}}\mathbf{F}^T) + tr(\mathbf{G}\mathbf{L}_{\mathcal{Y}}\mathbf{G}^T) \\ \text{s.t.} \quad & tr(\mathbf{F}\mathbf{F}^T) = 1, tr(\mathbf{G}\mathbf{G}^T) = 1 \\ & \mathbf{F}\mathbf{e} = \mathbf{0}, \mathbf{G}\mathbf{e} = \mathbf{0} \end{aligned} \quad (4)$$

which is a co-dimensionality reduction problem without any prior knowledge about the relationship between $\mathcal{X}$ and $\mathcal{Y}$. For the choice of the *Laplacian* matrices $\mathbf{L}_{\mathcal{X}}$ and $\mathbf{L}_{\mathcal{Y}}$, we use the *iterated Laplacian* [3] $\mathbf{M}_{\mathcal{X}} = (\mathbf{I} - \mathbf{Q}_{\mathcal{X}})^T(\mathbf{I} - \mathbf{Q}_{\mathcal{X}})$, where $\mathbf{Q}_{\mathcal{X}}$ is an $N \times N$ matrix with its (i, j) -th entry q_{ij} being calculated by optimizing

$$\min_{q_{ij}} \left\| \mathbf{x}_i - \sum_{\mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i)} q_{ij} \mathbf{x}_j \right\|^2 \quad \text{s.t.} \quad \sum_j q_{ij} = 1,$$

where $\mathcal{N}(\mathbf{x}_i)$ is the neighborhood of $\mathbf{x}_i$ (e.g. *k-nearest neighborhood* or *ε -ball neighborhood*), and for $\mathbf{x}_j \notin \mathcal{N}(\mathbf{x}_i)$, $q_{ij} = 0$. Similarly, we can define the iterated Laplacian $\mathbf{M}_{\mathcal{Y}}$ for data set $\mathcal{Y}$.

Manifold Alignment by Incorporating the Prior Knowledge. Now let's take the relative comparison constraints into account. As we have introduced in section 2, the knowledge “ $\mathbf{y}_i$ is closer to $\mathbf{x}_j$ than $\mathbf{x}_k$ ” can be translated into the relative distance constraint

$$\|\mathbf{g}_i - \mathbf{f}_j\|^2 \leq \|\mathbf{g}_i - \mathbf{f}_k\|^2 \quad (5)$$

in the embedded space. In the rest of this paper, for notational convenience, we will denote the constraint shown in Eq.(5) as an ordered 3-tuple $t_c = \{\mathbf{y}_i, \mathbf{x}_j, \mathbf{x}_k\}$. We use $\mathcal{T} = \{t_c\}_{c=1}^C$ to denote the set of constraints. Let $\mathbf{H} = [\mathbf{F}, \mathbf{G}]$, $\mathbf{M} = \begin{bmatrix} \mathbf{M}_{\mathcal{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{\mathcal{Y}} \end{bmatrix}$. By incorporating those constraints, our optimization problem (4) can be formulated as

$$\begin{aligned} \min_{\mathbf{H}} \quad & tr(\mathbf{H}\mathbf{M}\mathbf{H}^T) \\ \text{s.t.} \quad & \forall \{\mathbf{y}_i, \mathbf{x}_j, \mathbf{x}_k\} \in \mathcal{T}, \|\mathbf{h}_{i+N} - \mathbf{h}_j\|^2 \leq \|\mathbf{h}_{i+N} - \mathbf{h}_k\|^2 \\ & tr(\mathbf{H}_F \mathbf{H}_F^T) = 1, tr(\mathbf{H}_G \mathbf{H}_G^T) = 1 \\ & \mathbf{H}_F \mathbf{e} = \mathbf{0}, \mathbf{H}_G \mathbf{e} = \mathbf{0}, \end{aligned} \quad (6)$$

where $\mathbf{H}_F$ and $\mathbf{H}_G$ are the sub-matrices of $\mathbf{H}$ corresponding to $\mathbf{F}$ and $\mathbf{G}$.

Now we have formulated our tuple-constrained optimization as a *Quadratically Constrained Quadratic Programming (QCQP)* [11] problem. However, since the relative distance constraints in Eq.(6) is not *convex*, then 1) computationally the solution is difficult to derive and 2) the solution is trapped in local minima. Therefore, a reformulation is needed to make this problem tractable.

3.2 A Semi-definite Approach

Now we present how to relax the QCQP problem Eq.(6) to a SDP problem. Note that

$$\|\mathbf{h}_{i+N} - \mathbf{h}_j\|^2 \leq \|\mathbf{h}_{i+N} - \mathbf{h}_k\|^2 \Leftrightarrow 2\mathbf{h}_{i+N}^T \mathbf{h}_j + 2\mathbf{h}_{i+N}^T \mathbf{h}_k + \mathbf{h}_j^T \mathbf{h}_j - \mathbf{h}_k^T \mathbf{h}_k \leq 0$$

$$\text{tr}(\mathbf{H}\mathbf{M}\mathbf{H}^T) = \text{tr}(\mathbf{M}\mathbf{H}^T \mathbf{H}).$$

These two facts motivate us to deal with the *Gram matrix* of data instead, which is defined as $\mathbf{K} = \mathbf{H}^T \mathbf{H}$. $\mathbf{K}$ can be divided into four blocks as

$$\mathbf{K} = \begin{bmatrix} \mathbf{F}^T \mathbf{F} & \mathbf{F}^T \mathbf{G} \\ \mathbf{G}^T \mathbf{F} & \mathbf{G}^T \mathbf{G} \end{bmatrix} = \begin{bmatrix} \mathbf{K}^{FF} & \mathbf{K}^{FG} \\ \mathbf{K}^{GF} & \mathbf{K}^{GG} \end{bmatrix}. \quad (7)$$

Using $\mathbf{K}$, we are able to convert the formulas in Eq.(6) into linear forms as follows:

- **The objective function** is

$$\min_{\mathbf{K}} \text{tr}(\mathbf{M}\mathbf{K}). \quad (8)$$

- **The relative distance constraints** is

$$\forall \{\mathbf{y}_i, \mathbf{x}_j, \mathbf{x}_k\} \in \mathcal{T}, -2\mathbf{K}_{i+N,j} + 2\mathbf{K}_{i+N,k} + \mathbf{K}_{j,j} - \mathbf{K}_{k,k} \leq 0. \quad (9)$$

- **The scale invariance** is achieved by constraining the traces of $\mathbf{K}^{FF}$ and $\mathbf{K}^{GG}$ i.e.

$$\text{trace}(\mathbf{F}\mathbf{F}^T) = \text{trace}(\mathbf{K}^{FF}) = 1, \text{trace}(\mathbf{G}\mathbf{G}^T) = \text{trace}(\mathbf{K}^{GG}) = 1. \quad (10)$$

- **The translation invariance** is achieved by constraints

$$\sum_{i,j} \mathbf{K}_{i,j}^{FF} = 0, \sum_{i,j} \mathbf{K}_{i,j}^{GG} = 0. \quad (11)$$

To see this, consider the following fact for $\mathbf{F}$ (and similar for $\mathbf{G}$)

$$\sum_i \mathbf{f}_i = 0 \Leftrightarrow \left| \sum_i \mathbf{f}_i \right|^2 = \sum_{i,j} \mathbf{f}_i^T \mathbf{f}_j = \sum_{i,j} \mathbf{K}_{i,j}^{FF} = 0 \quad (12)$$

Finally, $\mathbf{K}$ must be *positive semi-definite* i.e. $\mathbf{K} \succeq 0$ to be a valid Gram matrix. And to avoid the case of empty feasible set and to encourage the influence of prior knowledge, we introduce slack variables $\mathcal{E} = \{\varepsilon_c\}_{c=1}^C$ and write the problem as:

$$\begin{aligned} \min_{\mathbf{K}, \varepsilon} \quad & \text{tr}(\mathbf{M}\mathbf{K}) + \alpha \sum_{c=1}^C \varepsilon_c \\ \text{s.t.} \quad & \forall \{\mathbf{y}_i, \mathbf{x}_j, \mathbf{x}_k\} \in \mathcal{T}, -2\mathbf{K}_{i+N,j} + 2\mathbf{K}_{i+N,k} + \mathbf{K}_{j,j} - \mathbf{K}_{k,k} \leq \varepsilon_c \\ & \text{trace}(\mathbf{K}^{FF}) = 1, \text{trace}(\mathbf{K}^{GG}) = 1, \\ & \sum_{i,j} \mathbf{K}_{i,j}^{FF} = 0, \sum_{i,j} \mathbf{K}_{i,j}^{GG} = 0, \\ & \mathbf{K} \succeq 0, \\ & \forall \varepsilon_c \in \mathcal{E}, \varepsilon_c \leq 0, \end{aligned} \quad (13)$$

where α is a parameter to balance between the data's structure and the supervision.

Since Eq.(13) is a *Semi-Definite Programming (SDP)* problem [11], we call our method *Semi-Definite Manifold Alignment (SDMA)*. Clearly, Eq.(13) is convex and thus is free of local minima. Besides, various software packages are available for efficient solutions, and we have preferred the *Sedumi* [12] package in this paper. When the Gram matrix $\mathbf{K}$ is solved, the embedded coordinates $\mathbf{F}$, $\mathbf{G}$ can be recovered from $\mathbf{K}^{FF}$ and $\mathbf{K}^{GG}$'s dominant eigenvectors.

We emphasize that SDMA can serve as a very flexible framework for manifold alignment and embedding. More concretely, Eq.(13) can be generalized (or degenerated) in the following ways. 1) Flexible supervision. First, the form of tuple constraints can be changed from $t_c = \{\mathbf{y}_i, \mathbf{x}_j, \mathbf{x}_k\}$ to $t_c = \{\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_k\}$. This means that SDMA can accept relative distance constraints between *any* three samples (*e.g.* all from the same manifold, or from 3 different manifolds). Moreover, our formulation is able to incorporate the traditional correspondence information by adding constraints " $\mathbf{K}_{i,i} = \mathbf{K}_{i,j} = \mathbf{K}_{j,j}$ ". 2) Multi-manifold alignment. This can be done straightforwardly by adding more manifold components into $\mathbf{H}$ and $\mathbf{M}$ along with corresponding constraints. 3) Semi-supervised embedding. When there is only one manifold component, SDMA provides a way to embed it with the guide of flexible supervision.

4 Experiments

4.1 Data and Settings

- **Head pose** [13]. This data set contains head images of 15 sets, each of which has 2 series of 93 images of the same person at 93 different poses.
- **Facial expression** [6]. This data set contains 213 images of 7 facial expressions posed by 10 Japanese female models. The underlying parameters are unknown.

The relative distance constraints $\mathcal{T}$ are obtained as follows. First, samples are randomly drawn to form a tuple $T = \{\mathbf{y}_i, \mathbf{x}_j, \mathbf{x}_k\}$ meaning that " $\mathbf{y}_i$ is more similar to $\mathbf{x}_j$ than to $\mathbf{x}_k$ ". Then let a user judge if this tuple is valid based on relative similarity. Finally, valid tuples are collected into $\mathcal{T}$. Since only "yes/no" questions are involved, this procedure is very easy for the users. Specifically, for the head pose data, similarity is determined by the sum of horizontal and vertical angle differences. For the facial expression data, if y_i and x_j have the same expression that is different from x_k , then T is valid. This strategy gives a conservative yet reliable supervision.

The parameter α tunes the strength of relative distance constraints. In our experiments it is chosen manually from the grid $\{10^{-5}, \dots, 10^{-1}, 1\}$.

4.2 Results

Figure 3 shows the experimental result of SDMA on head pose data. We construct the graph using neighborhood size 7, and use 500 tuples, $\alpha = 10^{-3}$. 130 samples from 2 subjects are embedded onto a 2-D plane. It can be seen that both of the underlying manifold parameters are successfully captured and aligned.

Figure 4 shows the alignment of facial expression data. The graph is constructed with neighborhood size 5, and 50 tuples are used. Since its manifold structure is not evident,

Fig. 3. Alignment of head pose images. (a) and (b) shows the embedding by SDMA and the correspondences found. Points are colored according to horizontal angles in (a) and vertical angles in (b). (c) shows some samples of matched pairs.

Fig. 4. Alignment of facial expression images. (a) and (b) shows the embedding, with points colored according to the true expressions. (a) shows the true correspondences, while (b) shows those found by SDMA. (c) shows some matched pairs.

we set $\alpha = 10^{-1}$ to strengthen the influence of relative distance constraints. 40 samples are embedded onto a 2-D plane since only two eigenvalues of $\mathbf{K}$ are not zero.

5 Discussions

The idea of learning with relative comparisons has also been used in other problems. [14] treat relative distance relations as the character of data, and use *AdaBoost* to seek for an embedding where this character is preserved. However, they did not utilize the data's intrinsic structure. [15] and [16] propose to learn distance metrics from relative comparisons. They both seek a distance measure $d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \mathbf{A} (\mathbf{x} - \mathbf{y})}$ and use the relative relations to constrain the feasible region of $\mathbf{A}$ through *mathematical programming*. In spirit, our method is similar to [15]. They learn a distance measure that preserves global distance relations, and we learn an embedding that preserves local manifold structures.

SDMA are closely related to *kernel* methods. By the semi-definite relaxation, we first derive the data's Gram matrix (*a.k.a. Kernel matrix*), and then calculate the low-dimensional coordinates by eigen-decomposition. This procedure is similar to *kernel principal component analysis* (KPCA) [17] except that our kernel matrix is learnt by aligning manifolds. Therefore, SDMA can be considered a *kernel learning* method. From this perspective, SDMA is similar to [18]. The difference is that they only use a single manifold's structure, while we exploit multiple manifolds and their correspondences.

One drawback of SDMA is that its computational cost is high when dealing with large data sets. Although the semi-definite relaxation makes the problem tractable, it inevitably increases the number of variables. In the future we are aiming at finding more efficient solutions.

6 Conclusion

Traditional align algorithms rely on the knowledge of high-quality pairwise correspondence, which is difficult to acquire in many situations. In this paper, we study a new way of aligning manifold based on the smoothness on graphs. To achieve maximum applicability and minimum user effort, we introduce the novel *relative distance constraint* to guide the alignment. Alignment using this type of prior knowledge is first formulized as a *quadratically constrained quadratic programming* (QCQP) problem. Further, by manipulating the *Gram matrix* of data instead of the coordinates, we relax this problem to a *semi-definite programming* (SDP) problem, which can be solved readily. Besides, we show that this semi-definite formulation can serve as a general framework for semi-supervised manifold alignment and embedding. Experiments on aligning various data demonstrate the effectiveness of our method.

Acknowledgement

Funded by Basic Research Foundation of Tsinghua National Laboratory for Information Science and Technology (TNList).

References

1. Seung, H.S., Lee, D.D.: The manifold ways of perception. *Science* 290, 2268–2269 (2000)
2. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 2323–2326 (2000)
3. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation* 15, 1373–1396 (2003)
4. Weinberger, K.Q., Saul, L.K.: Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision* 70(1), 77–90 (2006)
5. Ham, J., Ahn, I., Lee, D.: Learning a manifold-constrained map between image sets: Applications to matching and pose estimation. In: *CVPR-06* (2006)
6. Lyons, M.J., Kamachi, M., Gyoba, J., Akamatsu, S.: Coding facial expressions with gabor wavelets. In: *Proceedings of the 3rd IEEE Aut. Face and Gesture Recog.*, IEEE Computer Society Press, Los Alamitos (1998)
7. Ham, J., Lee, D., Saul, L.: Learning high dimensional correspondence from low dimensional manifolds. In: *Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, ICML-03* (2003)
8. Ham, J., Lee, D., Saul, L.: Semisupervised alignment of manifolds. In: *Proceedings of the 8th International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)* (2005)
9. Verbeek, J., Roweis, S., Vlassis, N.: Non-linear cca and pca by alignment of local models. In: *Advances in NIPS-04* (2004)
10. Verbeek, J., Vlassis, N.: Gaussian fields for semi-supervised regression and correspondence learning. *Pattern Recognition* 39(10), 1864–1875 (2006)
11. Boyd, S.P., Vandenberghe, L.: *Convex Optimization*. Cambridge, UK (2004)
12. Sturm, J.F.: Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software* 11-12, 625–653 (1999)
13. Gourier, N., Hall, D., Crowley, J.L.: Estimating face orientation from robust detection of salient facial features. In: *Proceedings of Pointing 2004, ICPR, International Workshop on Visual Observation of Deictic Gestures*, Cambridge, UK (2004)

14. Athitsos, V., Alon, J., Sclaroff, S., Kollios, G.: Boostmap: A method for efficient approximate similarity rankings. In: CVPR-04 (2004)
15. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Advances in NIPS-03 (2003)
16. Rosales, R., Fung, G.: Learning sparse metrics via linear programming. In: KDD-06 (2006)
17. Schölkopf, B., Smola, A., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* 10, 1299–1319 (1998)
18. Weinberger, K., Fei, S., Saul, L.K.: Learning a kernel matrix for nonlinear dimensionality reduction. In: ICML-04 (2004)