Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Preservation of local linearity by neighborhood subspace scaling for solving the pre-image problem*

Sheng-kai YANG, Jian-yi MENG, Hai-bin SHEN‡
(*Institute of VLSI Design, Zhejiang University, Hangzhou 310027, China*)
E-mail: skyangzju@gmail.com; {mengjy, shb}@vlsi.zju.edu.cn

Received Sept. 6, 2013; Revision accepted Feb. 8, 2014; Crosschecked Mar. 17, 2014

Abstract: An important issue involved in kernel methods is the pre-image problem. However, it is an ill-posed problem, as the solution is usually nonexistent or not unique. In contrast to direct methods aimed at minimizing the distance in feature space, indirect methods aimed at constructing approximate equivalent models have shown outstanding performance. In this paper, an indirect method for solving the pre-image problem is proposed. In the proposed algorithm, an inverse mapping process is constructed based on a novel framework that preserves local linearity. In this framework, a local nonlinear transformation is implicitly conducted by neighborhood subspace scaling transformation to preserve the local linearity between feature space and input space. By extending the inverse mapping process to test samples, we can obtain pre-images in input space. The proposed method is non-iterative, and can be used for any kernel functions. Experimental results based on image denoising using kernel principal component analysis (PCA) show that the proposed method outperforms the state-of-the-art methods for solving the pre-image problem.

Key words: Kernel method, Pre-image problem, Nonlinear denoising, Kernel PCA, Local linearity preserving
doi:10.1631/jzus.C1300248 **Document code:** A **CLC number:** TP391

1 Introduction

Kernel methods have been widely used in pattern recognition and machine learning in recent years. The basic idea is to implicitly map data into a reproducing kernel Hilbert space, known as the feature space, and then operate the data within that space. Many algorithms using kernel methods have been proposed, such as support vector machine (SVM), support vector regression (SVR), kernel principal component analysis (PCA), and kernel K-means clustering. The key aspect of kernel methods is the kernel trick. By introducing a kernel function, data can be implicitly mapped from input space to feature space, without the need to explicitly calculate the mapping. In some applications, such as classifications using SVM, all subsequent operations are carried out in feature space after the mapping. However, in some other applications, such as noise reduction, feature extraction, and kernel K-means clustering, after performing certain operations in feature space, subsequent processing needs to be completed back in input space. In such situations, given a point in the feature space, we need to calculate its corresponding point in the input space. This is known as the pre-image problem.

An informal definition of the pre-image problem is as follows: ϕ defines the mapping from input space to feature space, and $\boldsymbol{P}\phi(\boldsymbol{x})$ is some point in feature space. The pre-image problem is to find some point $\boldsymbol{t}$ within the input space which satisfies the following equation:

$$\phi(\boldsymbol{t}) = \boldsymbol{P}\phi(\boldsymbol{x}). \tag{1}$$

However, this is an ill-posed problem. The exact

‡ Corresponding author
* Project supported by the National Science and Technology Major Project of China (No. 2012EX01027001-002) and the Fundamental Research Funds for the Central Universities, China

pre-image of $\boldsymbol{P}\phi(\boldsymbol{x})$ may not exist, or may not be unique (Mika *et al.*, 1998). Many methods, direct and indirect, have been proposed over the years for solving the pre-image problem. They will be discussed further in Section 2.

The pre-image problem is involved in many kernel methods. A typical application is image denoising. The most widely used kernel method for image denoising is kernel PCA (Bakir *et al.*, 2003; Gruber *et al.*, 2006; Teixeira *et al.*, 2008). Briefly, training data is projected into feature space. The clean data is considered to be in the subspace obtained by PCA in feature space. After projecting the noisy signal into the subspace, we can obtain a clean signal in feature space. To obtain the representation in input space, the final step is to solve the pre-image problem. Other kernel methods have also been used for denoising, such as support vector data description (SVDD) based denoising (Park *et al.*, 2007) and kernel entropy component analysis (Jenssen, 2010). Another application is the visualization of kernel-based clustering (Kwok and Tsang, 2004; Huang *et al.*, 2011). Rathi *et al.* (2006) applied the pre-image problem to statistical shape analysis. Kim *et al.* (2005) used kernel PCA and a pre-image algorithm to restore high-resolution images from low-resolution images. Other applications include autolocalization in wireless sensor networks (Honeine and Richard, 2011b), interpolation of images of a walking person (Arif *et al.*, 2010), and expression normalization (Zheng *et al.*, 2010).

In this paper, an indirect method for solving the pre-image problem is proposed. In our algorithm, an inverse mapping process is constructed based on a novel framework which preserves local linearity. To preserve the local linearity between feature space and input space, we perform a local nonlinear transformation in input space, which is implicitly conducted by neighborhood subspace scaling. With a smoothness assumption, this transformation can be extended from training data to test data. Thereby an inverse mapping process for test data can be constructed, and the pre-image problem is solved.

The proposed algorithm has three advantages: (1) It is non-iterative and can be calculated in a closed form; (2) As the proposed method does not require kernel functions to be differentiable or reversible, it can be applied to all kinds of kernel functions; (3) The calculations are simple and the results are outstanding.

2 Related studies

With kernel methods demonstrating outstanding performance, the pre-image problem is of widespread concern. Depending on the approach used, pre-image algorithms can be divided into two categories: direct methods and indirect methods. In direct methods, the goal is to minimize expression (2) using different optimization approaches, while in indirect methods, approximate equivalent models are constructed to replace the original optimization problem.

$$\boldsymbol{t}=\arg\min_{\boldsymbol{t}}\|\phi(\boldsymbol{t})-\boldsymbol{P}\phi(\boldsymbol{t})\|^2. \tag{2}$$

2.1 Direct methods

The pre-image problem and a solution using a direct method were first described by Mika *et al.* (1998). They used a fixed-point iteration algorithm to solve the pre-image problem:

$$\boldsymbol{t}_{k+1}=\frac{\sum_{i=1}^{n}\widetilde{\gamma}_i\mathrm{e}^{-\frac{\|\boldsymbol{t}_k-\boldsymbol{x}_i\|^2}{c}}\boldsymbol{x}_i}{\sum_{i=1}^{n}\widetilde{\gamma}_i\mathrm{e}^{-\frac{\|\boldsymbol{t}_k-\boldsymbol{x}_i\|^2}{c}}}. \tag{3}$$

Because of the non-convexity of the optimization problem (3), the solution of the algorithm is unstable, and falls easily into a local minimum. In addition, the algorithm can be used only for RBF kernels of the form of $k(\boldsymbol{x},\boldsymbol{y})=\exp(\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{c})$.

Many improved algorithms have since emerged. Rathi *et al.* (2006) calculated $\|\boldsymbol{t}_k-\boldsymbol{x}_i\|^2$ in Eq. (3) directly, based on the relationship between the distance in feature space and the distance in input space, resulting in a non-iterative formula. They applied the same idea to obtain a formula given polynomial kernels with odd orders. Some modern optimization methods have been used to solve the optimization problem (2), such as particle swarm optimization (Li and Su, 2008) and the evolutionary strategy (Li *et al.*, 2011).

Inspired by Kwok and Tsang (2004), Teixeira *et al.* (2008) proposed a method to determine the initial value of Eq. (3), and thus enhance the stability of the solution. Firstly, the nearby points of $\boldsymbol{P}\phi(\boldsymbol{x})$ were found, and then the center of corresponding points in input space was used as the initial value of the algorithm (3).

Additional regularization and constraints have been considered for solving the pre-image problem using direct methods. Abrahamsen and Hansen (2011) enhanced the stability of the solution by adding Tikhonov and Lasso regularization. Nguyen and de la Torre (2008) added regularization functions to the original cost function, forcing the obtained pre-image close to the original test sample in input space. Taking into account that in many practical cases there exist certain actual constraints, Kallas *et al.* (2013) added non-negativity constraints to the cost function for image denoising.

Iteration is usually involved in direct methods since the optimization problem (2) is non-convex. This leads to a time-consuming and unstable solving process. However, in most cases the main concern is not the optimal solution of Eq. (2), but performance in practical applications, measured by classification accuracy, signal-to-noise ratio (SNR), and visualization. Therefore, solving problem (2) is unnecessary. Approximate equivalent models can be constructed to calculate the pre-image more effectively.

2.2 Indirect methods

Many indirect methods have been proposed recently. A priori knowledge obtained from training samples was used to construct more effective models, trying to solve the pre-image problem and improve performance.

A novel approach combined with linear manifold learning was proposed by Kwok and Tsang (2004), where problem (2) was converted to a multidimensional scaling (MDS) problem in input space. They used the relationship between Euclidean distance in feature space and in input space. After translating distance constraints between $\boldsymbol{P}\phi(\boldsymbol{x})$ and training samples in feature space into those between its pre-image $\boldsymbol{t}$ and training samples in input space, the MDS algorithm was adopted to determine $\boldsymbol{t}$. The idea of constructing a model using local information was first proposed in their paper. The key to the algorithm is the distance conversion from feature space to input space, which means that this algorithm is not suitable for a lot of kernels, like polynomial kernels with even orders and composite kernels.

Bakir *et al.* (2003) used kernel ridge regression in an attempt to construct the inverse mapping process from feature space to input space. However, this algorithm would work only with a reasonable distribution of samples. A new approach inspired by locally linear embedding (LLE) was proposed by Zheng and Lai (2006). They assumed that the local linearity of points in feature space and input space were matched, an assumption that need not hold given the nonlinearity of the kernel mapping. Honeine and Richard (2011a) proposed a conformal mapping method. They assumed there was a new set of coordinate functions in feature space, such that under that basis the dot product was invariable between the points in feature space and the corresponding points in input space. The inverse mapping process could thus be constructed and they could obtain the pre-image when a new test sample arrived. Actually, a global linear assumption was made in this paper, which did not hold in practice (Huang *et al.*, 2011).

Regularization can also be added in indirect methods. Zheng *et al.* (2010) added different regularization functions and constraints based on Zheng and Lai (2006). This led to an iterative solution process.

All approaches to the pre-image problem used indicators (like SNR or classification accuracy) to evaluate the algorithms. It is generally accepted that the actual performance is more important than minimizing the distance in feature space.

In this paper, we evaluate our proposed pre-image algorithm in a kernel PCA denoising application. Note that the proposed algorithm can be extended easily to other kernel methods.

3 Kernel PCA and denoising

As the kernel extension of PCA, kernel PCA has shown good performance in many applications. In this section, kernel PCA and denoising are reviewed. For a more detailed description, refer to Schölkopf *et al.* (1997).

Define a $d\times n$ matrix $\boldsymbol{X}=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\dots,\boldsymbol{x}_n\}$, containing n d-dimensional training samples. Kernel PCA maps the object from input space to feature space (usually high-dimensional space), where PCA is applied. Matrix representation of training samples in feature space is $\boldsymbol{\Phi_X}=\{\phi(\boldsymbol{x}_1),\phi(\boldsymbol{x}_2),\dots,\phi(\boldsymbol{x}_n)\}$, in which ϕ defines the mapping. Define the centered training sample matrix in feature space $\overline{\boldsymbol{\Phi}}_{\boldsymbol{X}}$. The kth orthogonal eigenvector $\boldsymbol{\nu}_k$ of centered covariance matrix $\boldsymbol{C}=\overline{\boldsymbol{\Phi}}_{\boldsymbol{X}}\overline{\boldsymbol{\Phi}}_{\boldsymbol{X}}^{\mathrm{T}}$ can be expressed by linear combinations of training samples, i.e., $\boldsymbol{\nu}_k=\overline{\boldsymbol{\Phi}}_{\boldsymbol{X}}\boldsymbol{\alpha}_i$, in

which the linear coefficient $\boldsymbol{\alpha}_i$ can be obtained by singular value decomposition (SVD) on the kernel matrix, as follows:

Define the centered kernel matrix $\overline{\boldsymbol{K}} = \overline{\boldsymbol{\Phi}}_{\boldsymbol{X}}^{\mathrm{T}}\overline{\boldsymbol{\Phi}}_{\boldsymbol{X}} = \boldsymbol{HKH}$, in which $\boldsymbol{H}$ is the centering matrix and $\boldsymbol{K} = \boldsymbol{\Phi}_{\boldsymbol{X}}^{\mathrm{T}}\boldsymbol{\Phi}_{\boldsymbol{X}}$. By applying SVD on $\overline{\boldsymbol{K}}$, $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n]$ and $\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ satisfying $\overline{\boldsymbol{K}} = \boldsymbol{U\Lambda U}^{\mathrm{T}}$ can be obtained. The linear coefficient is $\boldsymbol{\alpha}_i = \boldsymbol{u}_i/\sqrt{\lambda_i}$.

When a new test sample $\boldsymbol{x_t}$ arrives, it is first implicitly mapped to feature space by computing $\boldsymbol{K_t} = [\phi(\boldsymbol{x_t})^{\mathrm{T}}\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x_t})^{\mathrm{T}}\phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x_t})^{\mathrm{T}}\phi(\boldsymbol{x}_n)]^{\mathrm{T}}$. Thus, $\boldsymbol{P}\phi(\boldsymbol{x_t})$, which is obtained by projecting $\phi(\boldsymbol{x_t})$ to the first l eigenvectors, can be represented by linear combinations of $\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_n)$, as follows:

$$\boldsymbol{P}\phi(\boldsymbol{x_t}) = \boldsymbol{\Phi_X}\boldsymbol{\beta}, \tag{4}$$

in which $\boldsymbol{\beta} = \boldsymbol{HMH}(\boldsymbol{K_t} - \frac{1}{n}\boldsymbol{K}\cdot\boldsymbol{1}) + \frac{1}{n}\boldsymbol{1}$ and $\boldsymbol{M} = \sum_{i=1}^{l}\boldsymbol{u}_i\boldsymbol{u}_i^{\mathrm{T}}/\lambda_i$. We consider $\boldsymbol{P}\phi(\boldsymbol{x_t})$ to be the clean signal in feature space. The last key step is to compute the pre-image $\boldsymbol{t}$ in input space based on an algorithm.

According to the representer theorem (Schölkopf *et al.*, 2001), many problems can be written in the form of Eq. (4). Therefore, in solving these problems we can take advantage of the pre-image algorithm to obtain solutions in the input space.

4 The proposed pre-image method

In this section, the proposed pre-image method is described. Zheng and Lai (2006) borrowed the idea from the LLE algorithm, and assumed that the local linearity of the points in feature space and input space was matched. However, this assumption may not hold given the nonlinearity of the kernel mapping. In this study, the assumption is relaxed, and a more general framework to retain this local linearity is proposed. The proposed algorithm consists of two steps. In step one, a local inverse mapping process preserving the local linearity from feature space to input space for training samples is established using our novel framework. In this framework, a local nonlinear transformation is implicitly conducted by neighborhood subspace scaling transformation to preserve the local linearity between feature space and input space. This intrinsically nonlinear transformation is consistent with the characteristics of the inverse mapping process, and results can be derived in a much simpler fashion. In step two, this inverse mapping process is extended to test samples to obtain the pre-image under the smoothness assumption. Thus, the pre-image can be calculated.

4.1 Step one: establish the local inverse mapping process for training samples

In step one, two tasks need to be completed: calculating the locally linear representation (LLR) in feature space and then constructing the local inverse mapping process. Calculating the LLR is similar to the process for LLE algorithms, and may be described as follows:

Following the definition in Section 3, for every $\phi(\boldsymbol{x}_i)$ in feature space, the nearest k points $\phi(\boldsymbol{s}_1)$, $\phi(\boldsymbol{s}_2)$, …, $\phi(\boldsymbol{s}_k)$ should first be found, where $\boldsymbol{s}_i \subseteq \boldsymbol{X}$. Here, different distance metrics can be used, such as the Euclidean distance or geodesic distance. In this study, the Euclidean distance is used. Thus, the distance between two points in feature space can be shown as

$$d^2(\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)) = K(i,i) + K(j,j) - 2K(i,j).$$

After finding the nearest k points, compute the LLR of $\phi(\boldsymbol{x_i})$ as follows:

$$\boldsymbol{\Phi_S}\boldsymbol{\omega}_i \cong \phi(\boldsymbol{x}_i) \text{ subject to } \boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{1} = 1,$$

in which $\boldsymbol{\Phi_S} = \{\phi(\boldsymbol{s}_1), \phi(\boldsymbol{s}_2), \ldots, \phi(\boldsymbol{s}_k)\}$ is the neighborhood matrix of $\phi(\boldsymbol{x}_i)$, and $\boldsymbol{\omega}_i$ is the linear coefficient determined by the neighborhood. The constraint $\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{1} = 1$ ensures that the LLR is invariable to translations, and can characterize the locally linear relationship better (Saul and Roweis, 2003).

The above problem can be expressed as the following constrained optimization problem:

$$\boldsymbol{\omega}_i = \arg\min_{\boldsymbol{\omega}_i} \|\phi(\boldsymbol{x}_i) - \boldsymbol{\Phi_S}\boldsymbol{\omega}_i\|^2, \quad \text{subject to } \boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{1} = 1. \tag{5}$$

The solution to problem (5) is (Izenman, 2008)

$$\boldsymbol{\omega}_i = \frac{\boldsymbol{G}^{-1}\boldsymbol{1}}{\boldsymbol{1}^{\mathrm{T}}\boldsymbol{G}^{-1}\boldsymbol{1}}.$$

The matrix $\boldsymbol{G}$ in the above formula is calculated as follows:

$$\begin{aligned}\boldsymbol{G}_{pq} &= (\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{s}_p))^{\mathrm{T}}(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{s}_q)) \\ &= \boldsymbol{K}_{ii} - \boldsymbol{K}_{i\hat{f}(p)} - \boldsymbol{K}_{\hat{f}(q)i} + \boldsymbol{K}_{\hat{f}(p)\hat{f}(q)} \\ &\text{for } p, q = 1, 2, \ldots, k,\end{aligned}$$

in which $\hat{f}:\mathbb{N}\mapsto\mathbb{N}$ maps the local indices p and q to global indices. By solving the optimization problem (5), we can obtain the LLR of training samples in feature space.

To construct the local inverse mapping process, a novel local linearity preserving framework is proposed and described here. Define the neighborhood subspace of one point as the subspace expanded by the nearest k points of that point. Given a point $\boldsymbol{x}_i$ in input space, the local linearity from feature space to input space can be retained by a transformation of the neighborhood subspace. Define a $d\times d$ diagonal scaling matrix $\boldsymbol{D}_i$, which is the scaling matrix of the neighborhood subspace of $\boldsymbol{x}_i$. Thus, the local linearity can be retained through this scaling transformation. It is expected that the inverse mapping information is encoded in this scaling matrix. This transformation is computationally linear and intrinsically nonlinear. For point $\boldsymbol{x}_i$, define the neighborhood matrix of $\boldsymbol{x}_i$ in input space as $\boldsymbol{S}_i=\{\boldsymbol{s}_1,\boldsymbol{s}_2,\dots,\boldsymbol{s}_k\}$, in which $\boldsymbol{s}_j$ is the corresponding point in input space of $\phi(\boldsymbol{s}_j)$. Thus, the following model is constructed:

$$\boldsymbol{D}_i\boldsymbol{S}_i\boldsymbol{\omega}_i=\boldsymbol{x}_i+\boldsymbol{\epsilon}_i, \tag{6}$$

in which $\boldsymbol{\epsilon}_i$ represents the fitness of the model. Note that this mapping holds for all samples, including training samples and test samples. To derive the diagonal scaling matrix $\boldsymbol{D}_i$, we need to solve the following optimization problem:

$$\boldsymbol{D}_i=\arg\min_{\boldsymbol{D}_i}\|\boldsymbol{D}_i\boldsymbol{S}_i\boldsymbol{\omega}_i-\boldsymbol{x}_i\|^2+\lambda\mathcal{R}(\boldsymbol{D}_i). \tag{7}$$

To prevent over-fitting and maintain the smoothness, the regularization function $\mathcal{R}(\boldsymbol{D}_i)$ is added to the original cost function. In this study, $\mathcal{R}(\boldsymbol{D}_i)=\|\boldsymbol{D}_i\|_2^2$. The solution of problem (7) is

$$\boldsymbol{D}_i(j,j)=\frac{\boldsymbol{x}_i(j)\boldsymbol{S}_i(j,:)\boldsymbol{\omega}_i}{(\boldsymbol{S}_i(j,:)\boldsymbol{\omega}_i)^2+\lambda}\text{ for }j=1,2,\dots,d. \tag{8}$$

4.2 Step two: extend the inverse mapping process to test samples

In step one, the LLR of training samples is calculated, and the local inverse mapping process for training samples is constructed. In step two, we need to calculate the LLR of test samples and construct their local inverse mapping process.

We know that $\boldsymbol{P}\phi(\boldsymbol{x_t})$ can be described by the linear combination of $\phi(\boldsymbol{x}_1),\phi(\boldsymbol{x}_2),\dots,\phi(\boldsymbol{x}_n)$, i.e., $\boldsymbol{P}\phi(\boldsymbol{x_t})=\boldsymbol{\Phi_X}\boldsymbol{\beta}$. Similar to step one, its nearest k points in feature space should first be found. The distance between $\boldsymbol{P}\phi(\boldsymbol{x_t})$ and $\boldsymbol{x}_i$ in feature space can be calculated as follows:

$$d^2(\boldsymbol{P}\phi(\boldsymbol{x_t}),\boldsymbol{x}_i)=\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{\beta}-2\boldsymbol{K}(i,:)\boldsymbol{\beta}+\boldsymbol{K}(i,i).$$

Then, we need to solve an optimization problem similar to (5), computing its locally linear coefficient $\boldsymbol{\omega_t}$. The specific formulas are not shown here since the solution procedure is similar to that in step one.

Using model (6) we establish a local inverse mapping process for training samples. However, the mapping process for test samples cannot be constructed directly using this model. In this step, according to the smoothness assumption, the neighborhood subspace scaling transformation can be extended smoothly to test samples. Here, the smoothness assumption means that the scaling matrix of one neighborhood subspace can be obtained by averaging the scaling matrices of the adjacent neighborhood subspaces. Many average algorithms could be used here, such as arithmetic average, geometric average, harmonic average, and weighted average. Similar to Huang *et al.* (2011), the weighted arithmetic average is used in this study. Thus, the scaling matrix of a neighborhood subspace for a test sample can be calculated as follows:

$$\boldsymbol{D_t}=\frac{1}{\sum_{i=1}^k\alpha_i}\sum_{j=1}^k\alpha_j\boldsymbol{D}_j, \tag{9}$$

in which $\boldsymbol{D}_j$ is the neighborhood subspace scaling matrix of neighbors of the test sample, and α_j is the weighting factor. In this paper, α_j is computed as $\alpha_j=\exp(-d^2(\boldsymbol{P}\phi(\boldsymbol{x_t}),\phi(\boldsymbol{s}_j))/\sigma)$, in which $\sigma=\sum_{j=1}^k d^2(\boldsymbol{P}\phi(\boldsymbol{x_t}),\phi(\boldsymbol{s}_j))/k$.

Finally the pre-image can be computed as follows:

$$\boldsymbol{t}=\boldsymbol{D_t}\boldsymbol{S_t}\boldsymbol{\omega_t}, \tag{10}$$

in which $\boldsymbol{\omega_t}$ is the locally linear coefficient of $\boldsymbol{P}\phi(\boldsymbol{x_t})$, and $\boldsymbol{S_t}$ is the neighborhood matrix of $\boldsymbol{t}$ in input space.

5 Experiments and discussion

5.1 Verification of the assumption

To obtain the neighborhood subspace scaling matrix $\boldsymbol{D_t}$ of test samples, we assumed that the

scaling of neighborhood subspace can be extended smoothly to test samples. The actual scaling matrix $\boldsymbol{D}_i$ of each neighborhood subspace is calculated using Eq. (8). On the other hand, the approximate scaling matrix $\widetilde{\boldsymbol{D}_i}$ of each neighborhood subspace can be calculated using Eq. (9). With the smoothness assumption, these two versions of the scaling matrix should be pairwise matched.

In this subsection, this smoothness assumption is verified experimentally. The USPS data set consisting of 16×16 handwriting digits (http://www.kernel-machines.org/) was used in this experiment. For simplicity, the images were down-sampled to 8×8, and 500 samples of number '8' were randomly chosen as the experimental data set. The RBF kernel $k(i,j)=\exp(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\tau})$ with $\tau=\sum_{i,j=1}^{n}\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2/n^2$ was used. Since there was no noise in the data set, all the eigenvectors were retained in kernel PCA. Using the approximate scaling matrix $\widetilde{\boldsymbol{D}_i}$, the reconstructed sample can be calculated using Eq. (10). Denote the original sample as $\boldsymbol{x}_i$ and the reconstructed sample as $\widetilde{\boldsymbol{x}_i}$. The reconstruction error e_{rec} can be calculated as $e_{\mathrm{rec}}=\sum_{i=1}^{n}\|\widetilde{\boldsymbol{x}_i}-\boldsymbol{x}_i\|^2/n$. To obtain a minimum value of e_{rec}, we set $k=60$ and $\lambda=1\times10^{-5}$. Since the scaling matrix is a diagonal matrix, we deal only with the diagonal of $\widetilde{\boldsymbol{D}_i}$ and $\boldsymbol{D}_i$. Denote $\boldsymbol{v}^D=\mathrm{diag}\{\boldsymbol{D}\}$ as the vector of diagonal. The Pearson correlation coefficient ρ of $\boldsymbol{v}^{D_i}$ and $\boldsymbol{v}^{\widetilde{D_j}}$ was calculated as a metric of matching between $\boldsymbol{D}_i$ and $\widetilde{\boldsymbol{D}_j}$, using the following equation:

$$\rho(i,j)=\frac{\sum_r(\boldsymbol{v}_r^{D_i}-\overline{\boldsymbol{v}^{D_i}})(\boldsymbol{v}_r^{\widetilde{D_j}}-\overline{\boldsymbol{v}^{\widetilde{D_j}}})}{\sqrt{\sum_r(\boldsymbol{v}_r^{D_i}-\overline{\boldsymbol{v}^{D_i}})^2\sum_r(\boldsymbol{v}_r^{\widetilde{D_j}}-\overline{\boldsymbol{v}^{\widetilde{D_j}}})^2}}.$$

The overall average correlation coefficient calculated from $\sum_{i,j=1}^{n}\rho(i,j)/n^2$ is 0.34, and by computing $\sum_{i=1}^{n}\rho(i,i)/n$, the average correlation coefficient of the actual scaling vector $\boldsymbol{v}^{D_i}$ and the corresponding approximate scaling vector $\boldsymbol{v}^{\widetilde{D_i}}$ is 0.62. The results show that the actual scaling matrix and the approximate scaling matrix are highly correlated. To visually describe this matching, two pairs of scaling matrices were chosen randomly. Their values on the diagonal are shown in Fig. 1. The actual scaling matrix and the approximate scaling matrix are very similar.

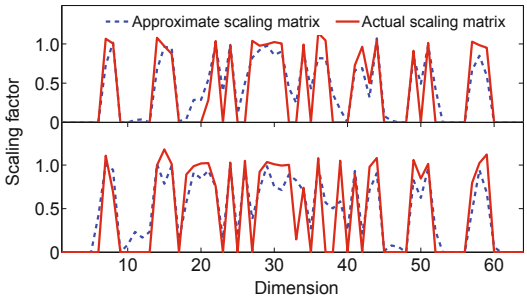


Fig. 1 Comparison of the actual scaling matrix and the corresponding approximate scaling matrix

5.2 Image denoising

5.2.1 Experimental database

In this study, the extended Yale Face Database B (Georghiades *et al.*, 2001; Lee *et al.*, 2005) was used. The database contains images of 38 human subjects under 64 illumination conditions. All the images were resized to 64×56, and the gray level was scaled to [0, 1]. Each image was reshaped to a vector column-wise. After removing 18 corrupted images, the remaining 2414 images were included in the data set. Five hundred images were randomly selected as the training set, and the remaining images comprised the test set. Random selection ensured data independence and reliability of the results.

5.2.2 Parameter optimization

Two parameters need to be determined in the proposed algorithm: the regularization parameter λ and the number of nearest points k. One hundred test samples were chosen from the test set, and Gaussian noise $N\sim(0,\sigma^2)$ with $\sigma^2=0.5$ was added to the test samples. The RBF kernel $k(i,j)=\exp(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\tau})$ with $\tau=\sum_{i,j=1}^{n}\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2/n^2$ was used, and 95% of the variance was retained in PCA in feature space. The grid search method was used to determine the optimal parameters $\{\lambda,k\}$. The mean square error (MSE) was used as the metric, as follows:

$$\mathrm{MSE}=\|\boldsymbol{t}-\boldsymbol{x}_c\|^2,$$

in which $\boldsymbol{t}$ was the pre-image obtained, and $\boldsymbol{x}_c$ was the original clean sample. The $\overline{\mathrm{MSE}}$, which was the average MSE over 100 samples, was used as the cost function. Experimental results are shown in Fig. 2.

Fig. 2 shows that this is a convex optimization problem, and the global minimum is obtained when k and λ are relatively small. Note that the curvature of

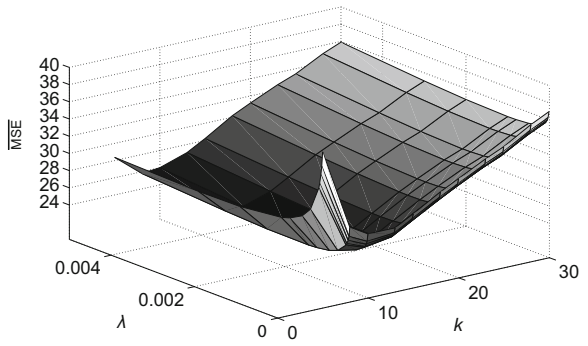
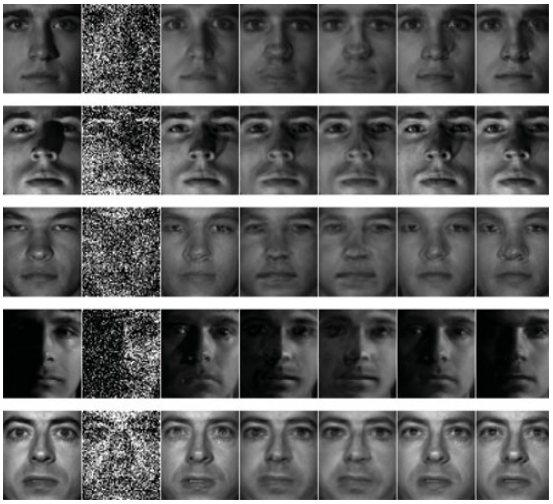


Fig. 2 $\overline{\text{MSE}}$ with different λ and k

the two sides along the direction of λ at the minimum point varies a lot. With further reduction in the value of λ, the $\overline{\text{MSE}}$ increases sharply. In contrast, when the value of λ increases, $\overline{\text{MSE}}$ increases slowly. This suggests that when making a trade-off between smoothness and fitness, more smoothness is preferred so as to obtain more satisfying results. For k, a relatively small value usually achieves better results; i.e., the local information can fit better the inverse mapping process from feature space to input space.

5.2.3 Denoising results

In this experiment, 100 test samples were randomly chosen. Two different types of noise were added to the samples: Gaussian noise ($N \sim (0, \sigma^2)$) and salt-and-pepper noise (p), with different noise intensities. For fair comparisons, the optimal parameters were used for all algorithms. The most commonly used kernel, RBF kernel, was used, and 95% of the variance was retained. Some clean images, noisy images, and the corresponding denoised images are shown in Figs. 3 and 4. In these figures, the first columns are the original clean figures; the second columns are the noisy figures with different noise intensities; the third to sixth columns are the denosied pre-images using Kwok's method (Kwok and Tsang, 2004), Zheng's method (Zheng and Lai, 2006), Honeine's method (Honeine and Richard, 2011a), and Huang's method (Huang *et al.*, 2011), respectively; the last columns are the pre-images denosied using the proposed method. As can be seen, our approach obtained better visualization. For Gaussian noise, when σ was small, all the algorithms listed showed good performance. However, as σ increased, the performance of some algorithms deteriorated sharply, while our algorithm



(a)

(b)

(c)

Fig. 3 Images with Gaussian noise: (a) $\sigma^2 = 0.3$; (b) $\sigma^2 = 0.4$; (c) $\sigma^2 = 0.5$

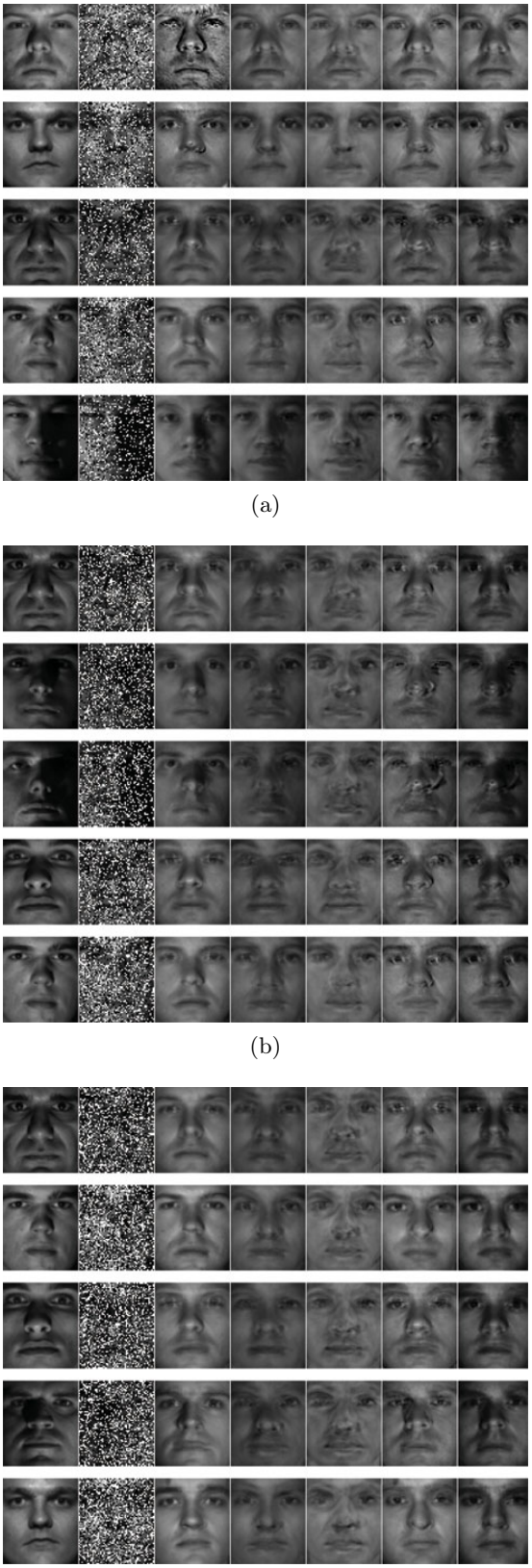


Fig. 4 Images with salt-and-pepper noise: (a) $p = 0.3$; (b) $p = 0.4$; (c) $p = 0.5$

performed only slightly worse. Thus, the proposed algorithm showed a more stable performance under different noise intensities. For salt-and-pepper noise, all the algorithms tended to show 'average face' as p increased. Comparing the results from the two experiments, RBF kernels performed well for Gaussian noise, but were not suitable for salt-and-pepper noise.

Quantitative comparisons are shown in Tables 1 and 2, where the $\overline{\text{MSE}}$s with different noise types and noise intensities of different algorithms are shown. The optimal value of k under Gaussian noise is also listed for comparison. Our proposed method performed better in all conditions. Observe in Table 1 that, based on parameter k, the algorithms listed can be divided into two categories: those using global information and those using local information. Kwok's method, Huang's method, and our method use local information, and show better performance than those using global information, including Zheng's method and Honeine's method. This suggests that by using local information the pre-image problem can be solved more effectively. Note that although k is included in Zheng's method, experimental results show that a large value of k is needed to achieve the best performance. This means that approximate global information is used in Zheng's method.

Results shown in Table 2 were consistent with the visualizations. Performance under salt-and-pepper noise was worse compared to that under Gaussian noise, although our method still performed best. A possible reason is that the image corrupted by salt-and-pepper noise was more noisy than that corrupted by Gaussian noise with the same σ^2 and p value. The average MSE of images corrupted by Gaussian noise with $\sigma^2 = 0.3$ was 261.9, while under salt-and-pepper noise with $p = 0.3$, the average MSE was 7.1554×10^4. Different kernel functions might also affect the performance under different types of noise. Further experiments are needed to determine the corresponding appropriate kernel function for different noise types.

Note that the optimal k value under salt-and-pepper noise was relatively large compared to that under Gaussian noise. Different categories of noise result in different optimal values of k. For both categories of noise, as the noise increased, the optimal value of k decreased.

Table 1 MSE comparison with Gaussian noise

Method	$\overline{\text{MSE}}$			Optimal k		
	$\sigma^2=0.3$	$\sigma^2=0.4$	$\sigma^2=0.5$	$\sigma^2=0.3$	$\sigma^2=0.4$	$\sigma^2=0.5$
Kwok	22.3979	24.6594	26.9366	10	5	5
Zheng	24.3877	43.1526	65.8261	205	215	215
Honeine	36.1106	69.6174	109.3635	500*	500*	500*
Huang	19.5596	23.9229	28.2292	15	10	10
This paper	**18.7613**	**21.8739**	**24.6841**	20	10	10

* Since global mapping was used in Honeine's method, the value of k was equal to the number of training samples

Table 2 MSE comparison with salt-and-pepper noise

Method	$\overline{\text{MSE}}$			Optimal k		
	$p=0.3$	$p=0.4$	$p=0.5$	$p=0.3$	$p=0.4$	$p=0.5$
Kwok	58.7487	97.1309	135.8279	10	10	5
Zheng	49.2949	70.0785	92.5192	105	140	160
Honeine	86.7215	120.1639	150.6930	500*	500*	500*
Huang	57.4134	92.5925	134.6848	35	35	15
This paper	**35.5646**	**61.3593**	**87.8186**	60	35	20

* Since global mapping was used in Honeine's method, the value of k was equal to the number of training samples

5.3 Time complexity analysis

Usually, training time is not a concern since the training process is performed only once. However, the time complexity of the test process (in this context, the time it takes to calculate the pre-image when a new test sample arrives), is usually of great concern. Table 3 shows the time complexity (TC) of the proposed algorithm and of other algorithms. The TC is related to parameters d, k, and n. To simplify the analysis, the TC of different algorithms under condition $d \gg n \gg k$ is shown in Table 4. The TC of our algorithm is $O(k^3+dn)$. This result is the same as that from using Zheng's method. Because of the locality of these two algorithms, if the value of k is extremely small so that $k^2 \ll d$, the TC can be further simplified as $O(dn)$, which is the same as that of Honeine's method. The TCs of Kwok's method and Huang's method are the largest, since d^2k and n^3 are contained in the expressions, respectively. In Kwok's algorithm, SVD was performed on a $d \times k$ matrix, with d the dimension of samples and k the number of neighbors. In applications where the dimension of samples is very large, such as image processing, Kwok's method is time-consuming. If the number of training samples is very large, Huang's method is time-consuming.

To compare intuitively the computing time of different algorithms, the execution time of calculating the pre-image 100 times was measured using the optimal parameters for different algorithms. Experimental results and comparisons are shown in Table 5. The test procedure was run in the Matlab R2012a environment on a laptop with a dual-core 2.53 GHz processor and 8 GB main memory. The experimental results are consistent with theoretical analysis. Note that although the TC of Zheng's method and of our method were the same, Zheng's method was more time-consuming, since the optimal value of k was relatively large according to previous experiments.

Table 3 The time complexity of different algorithms

Method	Time complexity
Kwok	$O(k^3+d^2k+n^2+dn)$
Zheng	$O(k^3+n^2+dn)$
Honeine	$O(n^2+dn)$
Huang	$O(n^3+k^3+k^2n+kn^2+dk^2+dn)$
This paper	$O(k^3+n^2+dn)$

Table 4 The time complexity of different algorithms with $d \gg N \gg k$

Method	Time complexity
Kwok	$O(d^2k)$
Zheng	$O(k^3+dn)$
Honeine	$O(dn)$
Huang	$O(n^3+dk^2+dn)$
This paper	$O(k^3+dn)$

Table 5 Execution time of different algorithms in seconds

Method	Gaussian noise			Salt-and-pepper noise			Average
	$\sigma^2=0.3$	$\sigma^2=0.4$	$\sigma^2=0.5$	$p=0.3$	$p=0.4$	$p=0.5$	
Kwok	45.34	25.98	26.58	46.17	45.82	27.61	36.25
Zheng	6.52	6.41	6.48	5.63	5.79	5.98	6.14
Honeine	3.98	3.68	4.22	5.03	4.62	4.60	4.36
Huang	7.40	7.41	7.22	14.30	14.24	9.58	10.03
This paper	5.03	4.54	4.53	7.06	5.61	5.19	5.33

6 Conclusions

A novel pre-image algorithm for kernel methods is proposed in this paper. By neighborhood subspace scaling transformation, we constructed a local linearity preserving process for mapping from feature space to input space for training samples. Assuming that this transformation could be extended to test samples smoothly, we could obtain the mapping for test samples, thus solving the pre-image problem. The proposed method is non-iterative, and can be used for any kernel function. The effectiveness of the algorithm was demonstrated in a common scenario: image denoising using kernel PCA. Compared with other algorithms, the proposed algorithm achieved the lowest MSE, and also had obvious advantages in reduced computational complexity. Experimental results showed that the proposed algorithm outperforms state-of-the-art methods for solving the pre-image problem. In the future, we will apply the proposed method to other kernel applications. Depending on the solutions, alternative kernel functions will be used to achieve the best possible results.

References

Abrahamsen, T.J., Hansen, L.K., 2011. Regularized pre-image estimation for kernel PCA de-noising. *J. Signal Process. Syst.*, **65**(3):403-412. [doi:10.1007/s11265-010-0515-4]

Arif, O., Vela, P.A., Daley, W., 2010. Pre-image problem in manifold learning and dimensional reduction methods. Proc. 9th Int. Conf. on Machine Learning and Applications, p.921-924. [doi:10.1109/icmla.2010.146]

Bakir, G.H., Weston, J., Schölkopf, B., 2003. Learning to find pre-images. *Adv. Neur. Inform. Process. Syst.*, **16**(7):449-456.

Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J., 2001. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell.*, **23**(6):643-660. [doi:10.1109/34.927464]

Gruber, P., Stadlthanner, K., Böhm, M., *et al.*, 2006. Denoising using local projective subspace methods. *Neurocomputing*, **69**(13-15):1485-1501. [doi:10.1016/j.neucom.2005.12.025]

Honeine, P., Richard, C., 2011a. A closed-form solution for the pre-image problem in kernel-based machines. *J. Signal Process. Syst.*, **65**(3):289-299. [doi:10.1007/s11265-010-0482-9]

Honeine, P., Richard, C., 2011b. Preimage problem in kernel-based machine learning. *IEEE Signal Process. Mag.*, **28**(2):77-88. [doi:10.1109/msp.2010.939747]

Huang, D., Tian, Y.D., de la Torre, F., 2011. Local isomorphism to solve the pre-image problem in kernel methods. Proc. IEEE Conf. on Computer Vision and Pattern Recognition, p.2761-2768. [doi:10.1109/cvpr.2011.5995685]

Izenman, A.J., 2008. Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning. Springer, New York, USA.

Jenssen, R., 2010. Kernel entropy component analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, **32**(5):847-860. [doi:10.1109/tpami.2009.100]

Kallas, M., Honeine, P., Richard, C., *et al.*, 2013. Non-negativity constraints on the pre-image for pattern recognition with kernel machines. *Pattern Recog.*, **46**(11):3066-3080. [doi:10.1016/j.patcog.2013.03.021]

Kim, K.I., Franz, M.O., Schölkopf, B., 2005. Iterative kernel principal component analysis for image modeling. *IEEE Trans. Pattern Anal. Mach. Intell.*, **27**(9):1351-1366. [doi:10.1109/TPAMI.2005.181]

Kwok, J.T.Y., Tsang, I.W., 2004. The pre-image problem in kernel methods. *IEEE Trans. Neur. Networks*, **15**(6):1517-1525. [doi:10.1109/tnn.2004.837781]

Lee, K.C., Ho, J., Kriegman, D., 2005. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell.*, **27**(5):684-698. [doi:10.1109/TPAMI.2005.92]

Li, J.W., Su, L., 2008. Combining KPCA and PSO for pattern denoising. Proc. Chinese Conf. on Pattern Recognition, p.1-6. [doi:10.1109/ccpr.2008.10]

Li, J.W., Su, L., Cheng, C., 2011. Finding pre-images via evolution strategies. *Appl. Soft Comput.*, **11**(6):4183-4194. [doi:10.1016/j.asoc.2011.03.011]

Mika, S., Schölkopf, B., Smola, A., *et al.*, 1998. Kernel PCA and de-noising in feature spaces. *Adv. Neur. Inform. Process. Syst.*, **11**:536-542.

Nguyen, M.H., de la Torre, F., 2008. Robust kernel principal component analysis. *Adv. Neur. Inform. Process. Syst.*, **21**:1185-1192.

Park, J., Kang, D., Kim, J., *et al.*, 2007. SVDD-based pattern denoising. *Neur. Comput.*, **19**(7):1919-1938. [doi:10.1162/neco.2007.19.7.1919]

Rathi, Y., Dambreville, S., Tannenbaum, A., 2006. Statistical shape analysis using kernel PCA. *SPIE*, **6064**:60641B. [doi:10.1117/12.641417]

Saul, L.K., Roweis, S.T., 2003. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *J. Mach. Learn. Res.*, **4**:119-155.

Schölkopf, B., Smola, A., Müller, K.R., 1997. Kernel principal component analysis. Proc. 7th Int. Conf. on Artificial Neural Networks, p.583-588. [doi:10.1007/bfb0020217]

Schölkopf, B., Herbrich, R., Smola, A.J., 2001. A generalized representer theorem. Proc. 14th Annual Conf. on Computational Learning Theory and 5th European Conf. on Computational Learning Theory, p.416-426. [doi:10.1007/3-540-44581-1_27]

Teixeira, A.R., Tomé, A.M., Stadlthanner, K., *et al.*, 2008. KPCA denoising and the pre-image problem revisited. *Digit. Signal Process.*, **18**(4):568-580. [doi:10.1016/j.dsp.2007.08.001]

Zheng, W.S., Lai, J.H., 2006. Regularized locality preserving learning of pre-image problem in kernel principal component analysis. Proc. 18th Int. Conf. on Pattern Recognition, p.456-459. [doi:10.1109/icpr.2006.991]

Zheng, W.S., Lai, J.H., Yuen, P.C., 2010. Penalized preimage learning in kernel principal component analysis. *IEEE Trans. Neur. Networks*, **21**(4):551-570. [doi:10.1109/tnn.2009.2039647]