SOLVING THE PRE-IMAGE PROBLEM IN KERNEL MACHINES: A DIRECT METHOD

Paul Honeine, Cédric Richard

Institut Charles Delaunay (FRE CNRS 2848), LM2S, Université de technologie de Troyes, 10010 Troyes, France

ABSTRACT

In this paper, we consider the pre-image problem in kernel machines, such as denoising with kernel-PCA. For a given reproducing kernel Hilbert space (RKHS), by solving the pre-image problem one seeks a pattern whose image in the RKHS is approximately a given feature. Traditional techniques include an iterative technique (Mika *et al.*) and a multidimensional scaling (MDS) approach (Kwok *et al.*). In this paper, we propose a new technique to learn the pre-image. In the RKHS, we construct a basis having an isometry with the input space, with respect to a training data. Then representing any feature in this basis gives us information regarding its pre-image in the input space. We show that doing a pre-image can be done directly using the kernel values, without having to compute distances in any of the spaces as with the MDS approach. Simulation results illustrates the relevance of the proposed method, as we compare it to these techniques.

Index Terms— kernel machines, pre-image problem, kernel matrix regression, denoising

1. INTRODUCTION

Kernel machines have gained wild popularity in the last decade, providing a breakthrough in statistical learning theory, together with low computational cost nonlinear algorithms, thanks to the *kernel trick*. Initiated by Vapnik's Support Vector Machines (SVM) [1], this led to the proliferation of nonlinear algorithms, including kernel Fisher discriminant analysis [2] and least-squares SVM [3] for supervised learning, and kernel principal component analysis (kernel-PCA) [4] and one-class SVM [5] for unsupervised learning. The kernel trick provides a mean to transform conventional linear algorithms into nonlinear ones, under the only requirement that the algorithm can be expressed only in terms of inner products between data. For this purpose, data from the input space are (nonlinearly) mapped into a feature space. One may not need to exhibit this map, since this action can be done implicitly by substituting the inner product by a positive definite kernel. This is the essence of the kernel trick. From a functional framework, this kernel is called the *reproducing kernel*, and the induced feature space is the so-called *reproducing kernel Hilbert space* (RKHS).

The extracted features, or functions to be more precise, are linear in the RKHS, resulting into non-linear features with the input data. This is the main idea behind using kernels. In most supervised problems, one seeks a decision from a statistic, given by the evaluation of the feature functions. However, this is not often the case for unsupervised problems in pattern recognition. For instance, while we can apply denoising or compression techniques in the RKHS with the virtues of the kernel trick, we need to go back into the input space for the final result. This is the case in denoising a signal (or image), the reconstructed signal belongs to the input space of the original signals. However, getting back to the input space from the RKHS is not so obvious, in general, as most elements of the latter may not have a pre-image in the former. This is the pre-image problem in kernel machines, as we seek an approximate solution. Solving this problem has received a growing amount of attention, with the most breakthrough given in [6] and [7]. In the former work, Mika *et al.* present the problem and its ill-posedness, and derive an iterative scheme to find an approximate solution. As always with iterative techniques, there is no guarantee that this leads to a global optimum, and may be unstable. In the latter work, Kwok *et al.* determine a relationship between the distances in the RKHS and the distances in the input data, based on a set of training data. Applying a multidimensional scaling technique (MDS) leads to the pre-image. This approach opens the door to a range of other techniques, such as manifold learning and out-of-sample methods [8, 9].

In this paper, we introduce a novel approach to find the pre-image. We learn a basis, not necessarily orthogonal, in the RKHS having an isometry with the input space, with respect to a set of training data. In other words, their inner products are (approximately) equal in both spaces. Thus, by representing any feature function of the RKHS in this basis, we get an estimate of the inner products between its counterpart in the input space and the training dataset. We show that setting the pre-image estimate follows easily from this information. It turns out that this approach is natural to kernel machines, and can be done using linear algebra. The proposed method is universal, in the sense of being independent, in its formulation, of both the type of the used kernel and of the feature under investigation. Moreover, once the basis constructed, more than one feature can be directly pre-imaged, using only linear

algebra. Comparing the proposed method to previous work, we have the following: It does not suffer from numerical instabilities or local minima as opposed to the iterative scheme in [6]. Compared to the MDS-based technique, we show that we don't need to compute and work on the distances in both spaces, inner products are sufficient. It is worth noting that the reproducing kernel gives us the inner products in the RKHS. This is the main idea behind the kernel trick.

The rest of this paper is organized as follows. In section 2, we begin by a brief review of the framework behind kernel machines, and derive the pre-image problem. The proposed method is presented in section 3, and its use for denoising with kernel-PCA illustrated. We conclude in section 4 with simulations.

2. KERNEL MACHINES AND THE PRE-IMAGE PROBLEM

2.1. Kernel machines

Let $\mathcal{X}$ be a compact of $\mathbb{R}^p$, endowed with the natural Euclidean inner product $\langle \cdot, \cdot \rangle$ defined by $\boldsymbol{x}_i^\top \boldsymbol{x}_j$ for any $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. Let $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite kernel on $\mathcal{X}$, where the positive definitness is defined by the property

$$\sum_{i,j} \alpha_i \alpha_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$$

for all $\alpha_i, \alpha_j \in \mathbb{R}$ and $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. The Moore-Aronszajn theorem [10] states that for every positive definite kernel, there exists a unique reproducing kernel Hilbert space (RKHS), and viceversa. Let $\mathcal{H}$ be the RKHS associated with κ, and let $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ be the inner product in this space. This means that we have the representer of evaluation at any $\boldsymbol{x}_j \in \mathcal{X}$, with

$$\psi(\boldsymbol{x}_j) = \langle \psi(\cdot), \kappa(\cdot, \boldsymbol{x}_j) \rangle_{\mathcal{H}}, \tag{1}$$

for all $\psi \in \mathcal{H}$. Replacing $\psi(\cdot)$ by $\kappa(\cdot, \boldsymbol{x}_i)$ yields

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \kappa(\cdot, \boldsymbol{x}_i), \kappa(\cdot, \boldsymbol{x}_j) \rangle_{\mathcal{H}}, \tag{2}$$

for all $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. This is the reproducing property from which the name of reproducing kernel is derived. Denoting by $\phi(\cdot)$ the map that assigns to each input $\boldsymbol{x} \in \mathcal{X}$ the kernel function $\kappa(\cdot, \boldsymbol{x})$, the reproducing property (2) implies that $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}$. The kernel then evaluates the inner product of any pair of elements of $\mathcal{X}$ mapped into $\mathcal{H}$, without any explicit knowledge of either the mapping function $\phi(\cdot)$ or the RKHS $\mathcal{H}$. This is the well-known kernel trick.

In combination with the kernel trick, the representer theorem [11] provides a powerful theoretical foundation for kernel machines. Classical applications to this theorem include SVM and kernel-PCA, where one seeks to maximize the margin or the output variance, respectively. This theorem states that any function $\varphi(\cdot)$ of a RKHS $\mathcal{H}$ minimizing a regularizes cost functional of the form

$$\sum_{i=1}^{n} J(\varphi(\boldsymbol{x}_i), y_i) + g(\|\varphi\|_{\mathcal{H}}^2),$$

with predicted output $\psi(\boldsymbol{x}_i)$ for input $\boldsymbol{x}_i$, and eventually the desired output y_i, and $g(\cdot)$ a strictly monotonically increasing function on $\mathbb{R}_+$, can be written as a kernel expansion in terms of available data

$$\varphi^*(\cdot) = \sum_{i=1}^{n} \gamma_i \, \kappa(\boldsymbol{x}_i, \cdot). \tag{3}$$

This theorem shows that even in an infinite dimensional RKHS, as with the Gaussian kernel, we only need to work in the subspace spanned by the n kernel functions of the training data, $\kappa(\cdot, \boldsymbol{x}_1), \ldots, \kappa(\cdot, \boldsymbol{x}_n)$.

2.2. The pre-image problem

By virtue of the representer theorem, evaluating the optimal function $\varphi^*(\cdot)$ at any $\boldsymbol{x} \in \mathcal{X}$ is given by $\sum_{i=1}^{n} \gamma_i \, \kappa(\boldsymbol{x}_i, \boldsymbol{x})$. This gives the prediction of $\boldsymbol{x}$, and comparing its value to a threshold yields a decision rule. This is done in supervised learning, such as regression and classification problems. However for pattern recognition with unsupervised learning, one might also be interested in $\varphi^*(\cdot)$, or more precisely in its counterpart in the input space. Since $\varphi^*(\cdot)$ might not have a pre-image, this is an ill-posed problem, where one seeks an approximate solution, i.e. $\boldsymbol{x}^*$ in $\mathcal{X}$ whose map $\kappa(\cdot, \boldsymbol{x}^*)$ is as close as possible to $\varphi^*(\cdot)$.

This is the pre-image problem. One may solve the optimization problem [9]

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathcal{X}} \|\varphi^*(\cdot) - \kappa(\cdot, \boldsymbol{x})\|_{\mathcal{H}}^2,$$

which minimizes the distance, or

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \left\langle \frac{\varphi^*(\cdot)}{\|\varphi^*(\cdot)\|_{\mathcal{H}}}, \frac{\kappa(\cdot, \boldsymbol{x})}{\|\kappa(\cdot, \boldsymbol{x})\|_{\mathcal{H}}} \right\rangle_{\mathcal{H}},$$

which maximizes the collinearity, where $\|\cdot\|_{\mathcal{H}}$ denotes the norm in the RKHS. In [6], Mika *et al.* propose an iterative scheme to solve the pre-image problem, which can only be applied to the Gaussian kernel, or any other radial basis kernel. Next, we show the relevance of solving such problem, in the case of denoising with kernel-PCA. It is worth noting that the proposed method is independent of such results, and can be applied to any kernel machine.

2.3. Kernel-PCA for denoising

The kernel-PCA [4] is an elegant nonlinear extension of the mostly used dimensional reduction and denoising technique, the principal component analysis (PCA). With PCA, we seek

principal axes that capture the highest variance in the data, that is, useful information as opposed to noise. These principal axes are the eigenvectors associated with the largest eigenvalues of the covariance matrix of data. There exists a dual formulation of PCA involving only the inner products of the training data. This can be applied implicitly in a RKHS, by substituting the inner products by kernel values.This is the kernel-PCA[1]. Each of the resulting principal functions takes the form (3), where the the weighting coefficients are obtained from the eigen-decomposition of the so-called Gram matrix, whose entries are $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for $i, j = 1, \ldots, n$.

In the same spirit of the conventional PCA, one constructs a subspace of $\mathcal{H}$ spanned by the most relevant principal functions. Using kernel-PCA for denoising any given $\boldsymbol{x} \in \mathcal{X}$, we project its kernel function $\kappa(\boldsymbol{x}, \cdot)$ onto that subspace. Let $\varphi^*(\cdot)$ be this projection which, by virtue of the PCA approach, is assumed to be noise-free. Thus, we need to get its counterpart in the input space, denoted $\boldsymbol{x}^*$, by solving the pre-image problem.

3. THE PROPOSED PRE-IMAGE METHOD

For any $\varphi^*(\cdot)$ of the RKHS $\mathcal{H}$, we learn the pre-image $\boldsymbol{x}^* \in \mathcal{X}$ from a set of available training data, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. To develop the proposed method, we proceed in two stages. In the first stage, we construct a basis in $\mathcal{H}$ having an isometry with the input space basis $\mathcal{X}$, where isometry is given with respect to the training data. In the second stage, $\varphi^*(\cdot)$ is represented in this basis, yielding the values of the inner products in $\mathcal{X}$ of its pre-image with the training data. From these inner products, we extract the pre-image $\boldsymbol{x}^*$.

Constructing the basis

The main idea of the proposed method is to construct a basis in the RKHS that is isometric with the input space. For this purpose, we use the training data, and by virtue of the representer theorem, we only have to consider the subspace spanned by the training kernel functions. Within this subspace, we construct a set of ℓ basis functions, each takes the form

$$\psi_k(\cdot) = \sum_{i=1}^{n} \alpha_{k,i}\, \kappa(\boldsymbol{x}_i, \cdot), \tag{4}$$

for $k = 1, 2, \ldots, \ell$, with at most $\ell = n$ basis functions, n being the number of the training data. The coordinate on $\psi_k(\cdot)$ of any kernel function $\kappa(\cdot, \boldsymbol{x})$ is given by

$$\langle \psi_k(\cdot), \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}} = \psi_k(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_{k,i}\, \kappa(\boldsymbol{x}_i, \boldsymbol{x}),$$

[1]The kernel-PCA algorithm requires a normalization and centering the kernel matrix. These details are omitted for the sake of simplicity; see [4].

where (1) is used. Therefore, its representation in this basis is given by the ℓ coordinates, written vector-wise

$$\Psi_{\boldsymbol{x}} = [\psi_1(\boldsymbol{x}) \;\; \psi_2(\boldsymbol{x}) \;\; \cdots \;\; \psi_\ell(\boldsymbol{x})]^\top,$$

where the k-th entry depends on the $\alpha_{k,i}$, for $i = 1, \ldots, n$.

In order to construct the basis of ℓ basis functions, we consider the model defined by

$$\Psi_{\boldsymbol{x}_i}^\top \Psi_{\boldsymbol{x}_j} = \boldsymbol{x}_i^\top \boldsymbol{x}_j + \epsilon_{ij}, \tag{5}$$

for all the training set, i.e. $i, j = 1, 2, \ldots, n$, and where ϵ_{ij} corresponds to the unfitness of the model. We don't impose any constraint in this model, such as the orthogonality between the basis functions. We only require the equivalence between the inner products in that basis, and their counterparts in the input space. Minimizing the variance of ϵ_{ij} yields the optimization problem

$$\min_{\psi_1, \ldots, \psi_\ell} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{x}_j - \Psi_{\boldsymbol{x}_i}^\top \Psi_{\boldsymbol{x}_j} \right)^2 + \lambda\, R(\psi_1, \ldots, \psi_\ell).$$

As preconized in machine learning literature, we introduce in this expression a regularization term, $\lambda\, R(\cdot)$, with λ a tunable parameter controlling the tradeoff between the fittness to the model (5) and the regularity of the solution. In this paper, we consider a more specific case with $R(\psi_1, \ldots, \psi_\ell) = \sum_{k=1}^{\ell} \|\psi_k\|_{\mathcal{H}}^2$, in order to penalize high norm functions.

From $\Psi_{\boldsymbol{x}}$, we collect the unknown and the known information into a matrix and a vector, respectively, and write

$$\Psi_{\boldsymbol{x}} = \boldsymbol{A}\, \boldsymbol{\kappa}_{\boldsymbol{x}},$$

where $\boldsymbol{\kappa}_{\boldsymbol{x}} = [\kappa(\boldsymbol{x}_1, \boldsymbol{x}) \;\; \kappa(\boldsymbol{x}_2, \boldsymbol{x}) \;\; \cdots \;\; \kappa(\boldsymbol{x}_n, \boldsymbol{x})]^\top$ and $\boldsymbol{A}$ is a $\ell \times n$ matrix of unknowns whose (k, i)-th entry is $\alpha_{k,i}$. Thus, the resulting optimization problem is

$$\widehat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} \; \frac{1}{2} \sum_{i,j=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{x}_j - \boldsymbol{\kappa}_{\boldsymbol{x}_i}^\top \boldsymbol{A}^\top \boldsymbol{A}\, \boldsymbol{\kappa}_{\boldsymbol{x}_j} \right)^2 + \lambda \sum_{k=1}^{\ell} \sum_{i,j=1}^{n} \alpha_{k,i} \alpha_{k,j}\, \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

By denoting $\|\cdot\|_F$ the Frobenius norm of a matrix, this yields

$$\widehat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} \frac{1}{2} \|\boldsymbol{P} - \boldsymbol{K}\, \boldsymbol{A}^\top \boldsymbol{A}\, \boldsymbol{K}\|_F^2 + \lambda\, \mathrm{tr}(\boldsymbol{A}^\top \boldsymbol{A} \boldsymbol{K}),$$

where $\boldsymbol{P}$and $\boldsymbol{K}$ are the Gram matrices with entries $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle = \boldsymbol{x}_i^\top \boldsymbol{x}_j$ and $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, respectively. Taking the derivative of the above cost function with respect to $\boldsymbol{A}^\top \boldsymbol{A}$, rather than $\boldsymbol{A}$, and setting it to zero, we get

$$\widehat{\boldsymbol{A}}^\top \widehat{\boldsymbol{A}} = \boldsymbol{K}^{-1} \left(\boldsymbol{P} - \lambda\, \boldsymbol{K}^{-1} \right) \boldsymbol{K}^{-1}. \tag{6}$$

In what follows, we show that only $\boldsymbol{A}^\top \boldsymbol{A}$ is required to find the pre-image, rather than $\boldsymbol{A}$. Thus we don't need to compute the coefficients defining the basis in the RKHS, since only their inner products are needed.

Back to the input space

Since the model (5) is valid for all the training data, we apply it to do the pre-image, as illustrated here. Let $\varphi^*(\cdot)$ be any optimal function resulting from a kernel machine, with $\varphi^*(\cdot) = \sum_{i=1}^n \gamma_i \kappa(\boldsymbol{x}_i, \cdot)$ as given in (3). By virtue of the representer theorem, it belongs to the subspace spanned by the training kernel functions, and therefore can be expressed in terms of the computed basis. The k-th coordinate of $\varphi^*(\cdot)$ is

$$\langle \varphi^*(\cdot), \psi_k(\cdot) \rangle_{\mathcal{H}} = \sum_{i,j=1}^{n} \alpha_{k,i} \gamma_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

Computed on each basis function, the ℓ coordinates are collected into one vector, denoted Ψ_{φ^*} with some abuse of notation. Thus, we write the model (5) as

$$\Psi_{\boldsymbol{x}_i}^\top \Psi_{\varphi^*} = \boldsymbol{x}_i^\top \boldsymbol{x}^*, \quad \text{for } i = 1, \ldots, n$$

where $\boldsymbol{x}^*$ is the resulting pre-image to be estimated. Matrix-wise, this is written as

$$\boldsymbol{K} \widehat{\boldsymbol{A}}^\top \widehat{\boldsymbol{A}}\, \boldsymbol{K} \boldsymbol{\gamma} = \boldsymbol{X}^\top \boldsymbol{x}^*$$

where $\boldsymbol{\gamma} = [\gamma_1\ \gamma_2\ \cdots\ \gamma_n]^\top$ and $\boldsymbol{X} = [\boldsymbol{x}_1\ \boldsymbol{x}_2\ \cdots\ \boldsymbol{x}_n]$. By injecting the constructed basis model (6) into this expression, we get

$$\boldsymbol{X}^\top \boldsymbol{x}^* = (\boldsymbol{P} - \lambda\, \boldsymbol{K}^{-1})\, \boldsymbol{\gamma}. \tag{7}$$

To find the pre-image $\boldsymbol{x}^*$ using this expression, different techniques may be considered. For instance, one can use an iterative scheme by solving the optimization problem

$$\min_{\boldsymbol{x}^*} \|\boldsymbol{X}^\top \boldsymbol{x}^* - (\boldsymbol{P} - \lambda\, \boldsymbol{K}^{-1})\, \boldsymbol{\gamma}\|^2.$$

Another non-iterative techniques may also be used to solve (7), such as the eigen-decomposition[2], in the spirit of the Nyström method, or the pseudo-inverse. Next, we use the pseudo-inverse, and show two interpretations of the proposed method

Interpretation 1

By using the pseudo-inverse from matrix algebra, we have the identity $(\boldsymbol{X}\boldsymbol{X}^\top)^{-1}\boldsymbol{X} = \boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{X})^{-1}$, which is only true for linearly independent training data. Thus, we can write

$$\boldsymbol{x}^* = \boldsymbol{X}\boldsymbol{P}^{-1}\big(\boldsymbol{P} - \lambda\, \boldsymbol{K}^{-1}\big)\, \boldsymbol{\gamma}. \tag{8}$$

Therefore, the resulting pre-image belongs to the span of the training data in the input space, in coherence with previous work on solving the pre-image problem [6, 7]. To show the impact of the regularization term, we set to zero the control parameter λ, which yields

$$\boldsymbol{x}^* = \boldsymbol{X}\boldsymbol{\gamma} = \sum_{i=1}^{n} \gamma_i\, \boldsymbol{x}_i. \tag{9}$$

This means that $\sum_i \gamma_i\, \kappa(\boldsymbol{x}_i, \cdot)$ has the pre-image $\sum_i\ \gamma_i\, \boldsymbol{x}_i$, thus having the same weighting coefficients in the RKHS and the input space. This is only true when no regularization is applied.

Interpretation 2

These expressions can be applied directly to a set of functions in the RKHS to get their pre-images in the input space. For this purpose, we write (7) as

$$\boldsymbol{X}^\top \boldsymbol{X}^* = (\boldsymbol{P} - \lambda\, \boldsymbol{K}^{-1})\, \boldsymbol{\Gamma},$$

where each column of matrix $\boldsymbol{\Gamma}$ represents the coefficient vector $\boldsymbol{\gamma}$, and each column of $\boldsymbol{X}^*$ the corresponding pre-image. From (8), we see that the matrix

$$\boldsymbol{M} = \boldsymbol{X}\boldsymbol{P}^{-1}\big(\boldsymbol{P} - \lambda\, \boldsymbol{K}^{-1}\big)$$

is computed only once, and then applied with $\boldsymbol{X}^* = \boldsymbol{M}\, \boldsymbol{\Gamma}$. This corresponds to a matrix completion scheme, or more specifically the kernel matrix regression approach, as given in [12, 13].

4. SIMULATION RESULTS

In this section, we compare the proposed method with two state-of-the art methods[3]: the iterative technique [6] and the MDS-based approach [7]. For this purpose, we consider four datasets, apply the kernel-PCA for denoising, with one of these three pre-image methods. While our method can operate on any positive definite kernel, the iterative method in [6] is limited to the Gaussian kernel. For this reason, we only consider the Gaussian kernel defined by

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{e}^{\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}},$$

where σ is its bandwidth.

For visualization, we consider a family of four datasets in 2-D (see [14] for more information), each having a geometric form corrupted by a noise with a bandwidth parameter ν. Within this area, data are uniformly randomly drawn. We generate n_{train} data to train the n_{eigen} eigenfunctions and to construct the basis. Then, we apply these results on another set of $n_{\text{pre-image}}$ generated data, in order to denoise using the

[2] Doing eigen-decomposition gives the pre-image relative to the eigen-basis in the input space. A post-processing is required to set the pre-image relative to the training data; this is called the procrustes problem.

[3] Matlab codes for these algorithms are available from the Statistical Pattern Recognition Toolbox, at the internet address `http://cmp.felk.cvut.cz/cmp/software/stprtool/`

Table 1. Values of the parameters for the different datasets.

	frame	banana	spiral	sine
n_{train}	350	300	70	420
$n_{\text{pre-image}}$	850	200	250	330
ν	0.1	0.2	0.3	0.5
n_{eigen}	5	3	10	10
σ	0.4	0.5	0.3	0.4

pre-image techniques. Values for these parameters as given in Table 1 for each dataset.

The frame dataset consists of a square of four lines of length 2. Data are uniformly randomly drawn within these lines and corrupted by a noise uniformly drawn from $[-\nu, \nu]$. The banana dataset is given by the parabola defined by the coordinates $(x, x^2 + \xi)$, with x uniformly randomly drawn from $[-1, 1]$, and ξ normally distributed with a standard deviation of ν. The spiral is defined by the coordinates $(A(\varphi)\cos(\varphi), A(\varphi)\sin(\varphi))$, with $A(\varphi) = 0.07\varphi + \xi$, where φ and ξ are uniformly randomly drawn from $[0, 6\pi]$ and $[0, \nu]$, respectively. The sine dataset is defined by the coordinates $(\varphi, 0.8\sin(2\varphi))$, where φ is uniformly randomly drawn from $[0, 2\pi]$, and where the data are corrupted by a uniformly random noise drawn from $[0, \nu]^2$.

For the iterative algorithm, the stopping criterion is set to a maximum of 100 iterations, which gives a reasonable cpu time. The required initial guess is set as in (9), with the weighting coefficients γ_i are uniformly randomly drawn from $[-1, 1]$. For the MDS-based algorithm, a global optimization scheme is used, as opposed to a neighborhood approach. As this algorithm is based on an eigen-decomposition technique, it yields a new basis in the input space. Thus we operate a procrustes technique to align this basis with the initial one, by minimizing the mean-squares error.

Figure 1 illustrate the denoising approach for the four datasets. In these figures, we show the training data with blue dots, and with red dots the denoised estimates obtained from another set (not shown explicitly, but given by the free ends of green lines). Green lines show the distance between the denoised and the initial noisy data. Consider for instance the frame dataset. Besides instability in many denoised data for the iterative technique, data within the upper border of the frame for instance (y-axis close to 1) are not denoised to the same area, as given by the proposed technique. It is obvious that the MDS is less adapted to any of the four given datasets. The iterative technique seems be sharper in denoising. However, as illustrated here, it suffers from numerical instabilities and local minima, as shown by long green lines, mostly in the frame and the sine applications. With all these datasets, the proposed method gives good results, which tend to fold at the tip of the dataset. This is illustrated for instance with the banana data, however, it folds less than the MDS results.

5. CONCLUSION

In this paper, we proposed a new method to solve the pre-image problem. The proposed method does not suffer from numerical instability, nor require computing the distances in the input and the RKHS. We showed that using only inner products between data in both spaces, the ones in the RKHS being defined by the kernel, we can construct a basis in the RKHS to make pre-image. We compared our method to state-of-the-art techniques. Perspectives include a more in-depth description of the regularization term, and applying this method on real data, for instance to denoise faces.

6. REFERENCES

[1] V. Vapnik, *Statistical Learning Theory*. New York, NY, USA: Wiley, September 1998.

[2] S. Mika, "Kernel fisher discriminants," PhD thesis, University of Technology, Berlin, October 2002.

[3] J. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. Singapore: World Scientific Pub. Co., 2002.

[4] B. Schölkopf, A. Smola, and K. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[5] D. Tax, "One-class classification; concept-learning in the absence of counter-examples," PhD thesis, Advanced School for Computing and Imaging – Delft University of Technology, June 2001.

[6] S. Mika, B. Schölkopf, A. Smola, K. Müller, M. Scholz, and G. Rätsch, "Kernel pca and de-noising in feature spaces," in *Proceedings of the 1998 conference on advances in neural information processing systems II*. Cambridge, MA, USA: MIT Press, 1999, pp. 536–542.

[7] J. T. Kwok and I. W. Tsang, "The pre-image problem in kernel methods," in *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003)*. Washington, DC, USA: AAAI Press, August 2003, pp. 408–415.

[8] Y. Bengio, J. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet, "Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[9] P. Arias, G. Randall, and G. Sapiro, "Connecting the out-of-sample and pre-image problems in kernel methods," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 18-23 jun 2007. [Online]. Available: http://ampere.iie.edu.uy/publicaciones/2007/ARS07

[10] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, pp. 337–404, 1950.

[11] B. Schölkopf, R. Herbrich, and R. Williamson, "A generalized representer theorem," Royal Holloway College, Univ. of London, UK, Tech. Rep. NC2-TR-2000-81, 2000.

[12] Y. Yamanishi and J.-P. Vert, "Kernel matrix regression," Tech. Rep. http://arxiv.org/abs/q-bio/0702054v1, 2007.

[13] P. Honeine, C. Richard, M. Essoloh, and H. Snoussi, "Localization in sensor networks - a matrix regression approach," in *5th IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, Darmstadt, Germany, July 2008.

[14] H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recognition*, vol. 40, pp. 863–874, 2007.

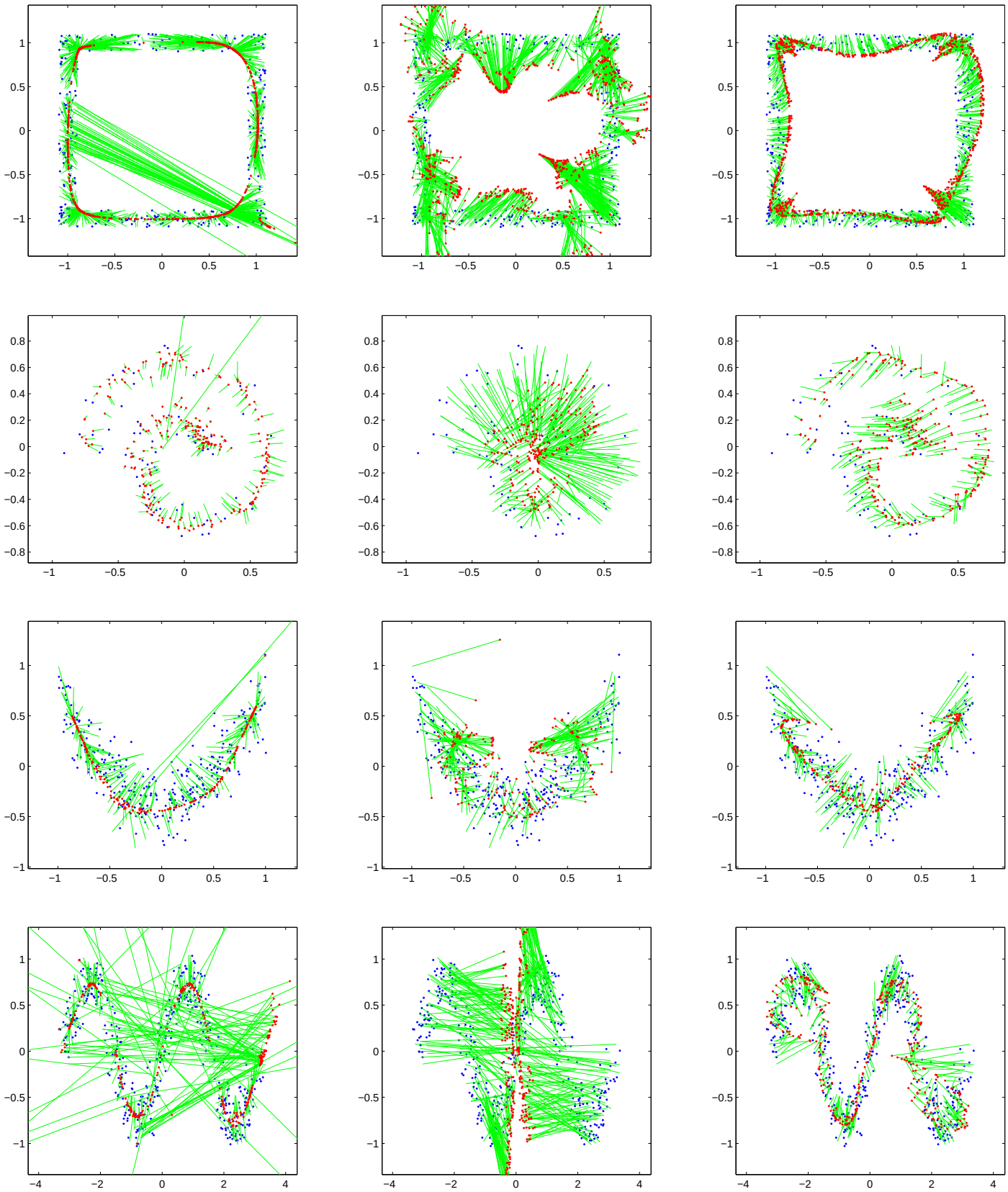

Fig. 1. Results obtained using the iterative (left), the MDS-based (middle), and the proposed (right) algorithm, for the frame (first row), the spiral (second row), the banana (third row), and the sine (fourth row) datasets. Training data are represented by blue dots, estimated pre-images by red dots, and green lines illustrate the distance between these denoised pre-images and the initial noisy data (not shown).