Preimage problem in kernel-based machine learning

Paul Honeine, Cédric Richard

The pre-image problem in kernel-based machine learning

Paul Honeine, and Cédric Richard

Kernel machines have gained considerable popularity during the last fifteen years, making a breakthrough in nonlinear signal processing and machine learning thanks to extraordinary advances. This increased interest is undoubtedly driven by the practical goal of being able to easily develop efficient nonlinear algorithms. The key principle behind this, known as the *kernel trick*, exploits the fact that a great number of data processing techniques do not depend explicitly on the data itself, but rather on a similarity measure between them, i.e., an inner product. To provide a nonlinear extension of these techniques, one can apply a nonlinear transformation to the data, mapping them into some feature space. According to the kernel trick, this can be achieved by simply replacing the inner product with a reproducing kernel (i.e., positive semi-definite symmetric function), the latter corresponds to an inner product in the feature space. One consequence is that the resulting nonlinear algorithms show significant performance improvements over their linear counterparts, with essentially the same computational complexity.

While the nonlinear mapping from the input space to the feature space is central in kernel methods, the reverse mapping from the feature space back to the input space is also of primary interest. This is the case in many applications, including kernel principal component analysis for signal and image denoising. Unfortunately, it turns out that the reverse mapping generally does not exist, and only a few elements in the feature space have a valid pre-image in the input space. The so-called pre-image problem consists of finding an approximate solution, by identifying data in the input space based on their corresponding features in the high-dimensional feature space. The pre-image problem is essentially a dimensionality reduction problem, and both have been intimately connected in their historical evolution, as studied in this paper.

I. An introductory example: kernel PCA for denoising

A. Linear denoising with PCA

In general, some correlations exist among data, thus techniques for dimensionality reduction or so-called feature extraction provide a way to confine the initial space to a subspace of lower dimensionality. The principal component analysis (PCA), also known as the Karhunen-Loève transformation, is one of the most widely used dimensionality reduction techniques. Conventional PCA seeks principal directions that capture the highest variance in the data. Mutually orthonormal, these directions define the subspace exhibiting information rather than noise, providing the optimal linear transformation. Here, the optimality is in the sense of least mean square reconstruction error. For instance, in data compression and manifold learning, much information is conserved by projecting onto the directions of highest variance, while in denoising, directions with small variance are dropped. These schemes are mathematically equivalent, we use here a denoising schema without loss of generality.

Consider an input space $\mathcal{X}$ endowed by the inner product $\langle \cdot, \cdot \rangle$, for instance a vectorial space with the Euclidean inner product $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle = \boldsymbol{x}_i^\top \boldsymbol{x}_j$. Let $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ denotes a set of available data (observations) from $\mathcal{X}$. PCA techniques seek the axes that maximize the mean variance of the projected data, under the unit-norm constraint, namely $\psi_1, \psi_2, \ldots, \psi_k$ by maximizing $\frac{1}{n} \sum_{i=1}^n |\langle \boldsymbol{x}_i, \psi_\ell \rangle|^2$ subject to $\langle \psi_\ell, \psi_{\ell'} \rangle = \delta_{\ell\ell'}$ for all $\ell, \ell' = 1, 2, \ldots, k$. In this expression, the Kronecker delta is defined as $\delta_{\ell\ell'} = 1$ if $\ell = \ell'$, and $\delta_{\ell\ell'} = 0$ otherwise. Solving this constrained optimization problem using the Lagrangian provides the following problem:

$$\lambda_\ell \, \psi_\ell = \boldsymbol{C} \, \psi_\ell, \tag{1}$$

where λ_ℓ defines the amount of variance captured by ψ_ℓ, and $\boldsymbol{C}$ is the covariance matrix of the data. In other words, $(\lambda_\ell, \psi_\ell)$ is the eigenvalue-eigenvector of the covariance matrix, data assumed zero-mean. Furthermore, eigenvectors lie in the span of the data, since for every $\ell = 1, 2, \ldots, k$ we have

$$\psi_\ell = \frac{1}{\lambda_\ell} \boldsymbol{C} \, \psi_\ell = \frac{1}{\lambda_\ell \, n} \sum_{i=1}^n \langle \boldsymbol{x}_i, \psi_\ell \rangle \, \boldsymbol{x}_i.$$

The eigenvectors associated to the largest eigenvalues provides a relevant low-dimensional subspace. As a consequence, we are interested in elements from this relevant subspace. This is the case, for instance, in data denoising, where the projection of a given noisy data onto this subspace provides

its *noise-free* counterpart. Therefore, the latter can be written as an expansion of the eigenvectors, namely for a noisy data $\tilde{\boldsymbol{x}}$ we get the denoised $\psi = \sum_{i=1}^{k} \langle \tilde{\boldsymbol{x}}, \psi_i \rangle \, \psi_i$, and from the above expression, as a linear expansion in terms of the available data, by taking the form

$$\psi = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{x}_i.$$

B. Kernel-PCA for nonlinear denoising

In order to provide a natural nonlinear extension of PCA, a nonlinear mapping is applied to the data as a pre-processing stage, prior to applying the PCA algorithm. Let $\phi(\cdot)$ be the nonlinear transformation, mapping data from the input space $\mathcal{X}$ to some feature space $\mathcal{H}$. Then problem (1) remains essentially the same, with the covariance matrix associated to the transformed data. From the linear expansion with respect to the latter, the resulting principal axes take the form

$$\psi_\ell = \sum_{i=1}^{n} \langle \phi(\boldsymbol{x}_i), \psi_\ell \rangle_{\mathcal{H}} \, \phi(\boldsymbol{x}_i), \tag{2}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product in the feature space $\mathcal{H}$. In this space, each feature ψ_ℓ lies in the span of the mapped input data, with the coefficients given by the ℓ-th eigenvector of the eigen-problem

$$n \, \lambda_\ell \, \boldsymbol{\alpha}_\ell = \boldsymbol{K} \, \boldsymbol{\alpha}_\ell, \tag{3}$$

where $\boldsymbol{K}$ is the so-called Gram matrix with entries $\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}$, for $i, j = 1, 2, \ldots, n$. As illustrated here, the expansion coefficients require only the evaluation of inner products. Without the need to exhibit the mapping function, this information can be easily exploited for a large class of nonlinearities, by substituting the inner product with a positive semi-definite kernel function. This argument is the kernel trick, which provides a nonlinear counterpart of the classical PCA algorithm, the so-called kernel-PCA [1].

Consider the denoising application using kernel-PCA. For a given $\tilde{\boldsymbol{x}}$, its nonlinear transformation $\phi(\tilde{\boldsymbol{x}})$ is projected onto the subspace spanned by the most relevant principal axes, providing the denoised pattern. The latter can be written as a linear expansion of the k principal axes, $\psi_1, \psi_2, \ldots, \psi_k$, with

$$\psi = \sum_{i=1}^{k} \langle \boldsymbol{x}, \psi_i \rangle \, \psi_i. \tag{4}$$

Equivalently, the denoised pattern can also be written as a linear expansion of the n images of the training data, namely $\psi = \sum_{i=1}^{n} \alpha_i \, \phi(\boldsymbol{x}_i)$, where the expansion in (2) is used. In practice, one is interested in representing the denoised pattern in the input space, as illustrated in Figure 2. It turns out that most elements of the feature space, including the denoised patterns, are not *valid images*, i.e., the result of applying the map to some input data. To get the denoised counterpart in the original input space, one needs to operate an approximation scheme, i.e., estimate $\boldsymbol{x}^*$ such that its image $\phi(\boldsymbol{x}^*)$ is as close as possible to ψ.

Beyond this kernel-PCA example, the kernel trick is well-known in the machine learning community. It provides flexibility to derive nonlinear techniques based on linear ones, data being implicitly mapped into a feature space. This space is given by the span of the mapped data, i.e., all the linear expansions of mapped data. The price to pay is that, in general, not each element of the space is necessary the image of some data. This is the case of most elements in the feature space, since they can be written as

$$\psi = \sum_{i=1}^{n} \alpha_i \, \phi(\boldsymbol{x}_i),$$

as illustrated above with either a principal axe ψ_ℓ or a denoised feature ψ. In order to give proper interpretation of these components, one should define the way back from the feature into the input space. This is the pre-image problem in kernel-based machine learning, as illustrated in Figure 1.

II. Kernel-based machine learning

In the past fifteen years or so, a novel breakthrough to artificial neural networks has been achieved in the field of pattern recognition and classification, within the framework of kernel-based machine learning. They have gained wide popularity, owing, on the one hand, to theoretical guarantees regarding performance, and on the other hand to low computational complexity in nonlinear algorithms. Pioneered by Vapnik's Support Vector Machines (SVM) for classification and regression [2], kernel-based methods are nonlinear algorithms that can be adapted to an extensive class of nonlinearities. As a consequence, they have found numerous applications, including classification [3], regression [4], time series prediction [5], novelty detection [6], image denoising [7], and bioengineering [8], to name just a few (see, e.g., [9] for a review).

A. Reproducing kernels and rkHs

Originally proposed by Aizerman *et al.* in [10], the kernel trick provides an elegant mathematical means to derive powerful nonlinear variants of classical linear techniques. Most well-known statistical (linear) techniques can be formulated as inner product between pairs of data. Thus, applying any nonlinear transformation to the data can only impact the values of the resulting inner products. Therefore, one does not need to compute such transformation explicitly for a large class of

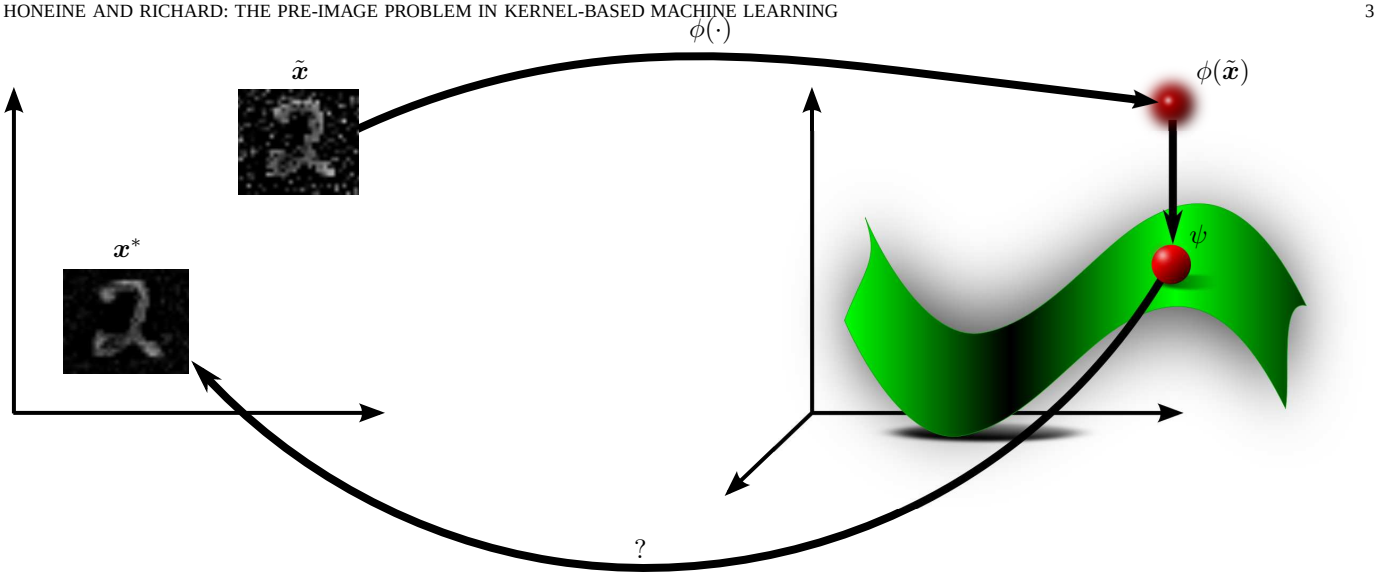

Fig. 1. Schematic illustration of the pre-image problem for pattern denoising with kernel-PCA. While dimensionality reduction through orthogonal projection is performed in the feature space (right panel), a pre-image technique is required to recover the denoised pattern in the input space (left panel).

nonlinearities. Instead, one only needs to replace the inner product operator with an appropriate kernel, i.e., a symmetric hermitian function. The only restriction is that the latter defines an inner product in some space. A sufficient condition for this is ensured by Mercer's theorem [11], which may be stated as follows: Any positive semi-definite kernel can be expressed as an inner product in some space, where the positive semi-definiteness of a kernel $\kappa\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is determined 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 $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. Furthermore, the Moore-Aronszajn theorem [12] states that to any positive semi-definite kernel κ corresponds a unique reproducing kernel Hilbert space (rkHs) whose inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, usually called reproducing kernel, is κ itself.

The one-to-one correspondence between rkHs and positive semi-definite functions has proved to be quite useful in numerous fields (see [13] and references therein). Since the pioneering work of Aronszajn [12], reproducing kernels and rkHs formalism have been increasingly used, especially after being selected for the resolution of interpolation problems by Parzen [14], Kailath [15] and Wahba [16]. A rkHs is a Hilbert space of functions for which point evaluations are bounded, and where the existence and uniqueness of the reproducing kernel is guaranteed by the Riesz representation theorem. In fact, let $\mathcal{H}$ be a Hilbert space of functions defined on some compact $\mathcal{X}$, for which the evaluation $\psi(\boldsymbol{x})$ of the function $\psi \in \mathcal{H}$ is bounded for all $\boldsymbol{x} \in \mathcal{X}$. By this theorem, there exists a unique function $\phi(\boldsymbol{x}) \in \mathcal{H}$ such as $\psi(\boldsymbol{x}) = \langle \psi, \phi(\boldsymbol{x}) \rangle_{\mathcal{H}}$. Also denoted $\kappa(\cdot, \boldsymbol{x})$, this function has the following popular property

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}, \tag{5}$$

for any $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. Moreover, distances can be easily evaluated using the kernel trick, since the distance between two elements can be given using only kernel values, with

$$\begin{aligned} \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)\|^2_{\mathcal{H}} &= \langle \phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j), \phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}} \\ &= \kappa(\boldsymbol{x}_i, \boldsymbol{x}_i) - 2\,\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + \kappa(\boldsymbol{x}_j, \boldsymbol{x}_j), \end{aligned} \tag{6}$$

where $\| \cdot \|_{\mathcal{H}}$ denotes the norm in the rkHs.

The inherent modularity of reproducing kernels allows scaling up linear algorithms into nonlinear ones, adapting kernel-based machines to tackle a large class of nonlinear tasks. Kernels are commonly defined on vectorial spaces, $\mathcal{X}$ endowed with the Euclidean inner product $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle = \boldsymbol{x}_i^\top \boldsymbol{x}_j$ and the associated norm $\|\boldsymbol{x}_i\|$. They can be easily adapted to operate on images, e.g., in face recognition or image denoising. They are not restricted to vectorial inputs, but can be naturally designed to measure similarities between sets, graphs, strings, and text documents [9]. As illustrated in Table I, most of the kernels used in the machine learning literature can be divided into two categories: projective kernels are functions of inner product, such as the polynomial kernel, and radial kernels (also known by isotropic kernels) are functions of distance, such as the Gaussian kernel. These kernels map implicitly the data into a high dimensional space, even infinite dimensional for the latter.

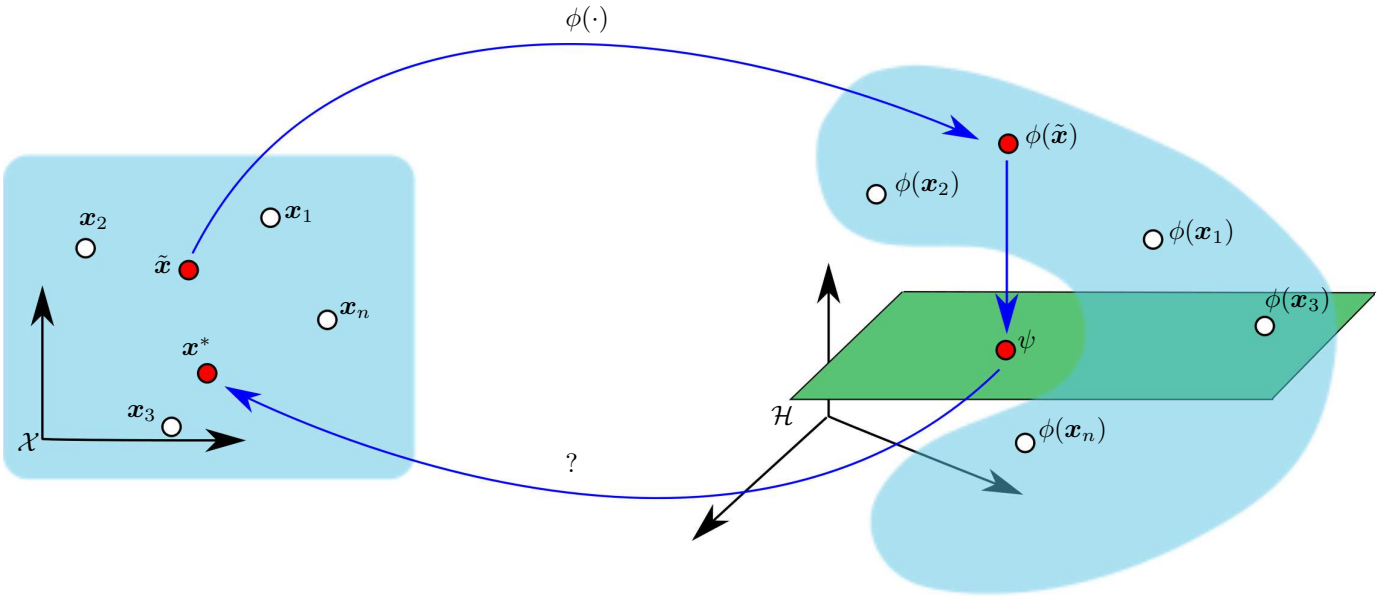

Fig. 2. Kernel machines map the input space (blue region in the left panel) into a higher-dimensional space (blue region in the right panel). The rkHs $\mathcal{H}$ is defined as the completion of the span of the mapped input data, with elements written as a linear expansion of mapped data. However, not each element of $\mathcal{H}$ is necessary the image of some input data. The pre-image problem consists of going back to the input space, e.g., to represent in the input space elements of the rkHs (e.g., the effect of projecting onto a subspace, as illustrated here).

TABLE I

COMMONLY USED KERNELS IN MACHINE LEARNING, WITH PARAMETERS $c > 0$, $p \in \mathbb{N}_+$, AND $\sigma > 0$.

	Kernels	Expressions
Projective	monomial	$(\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle)^p$
	polynomial	$(c + \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle)^p$
	exponential	$\exp(\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle / 2\sigma^2)$
	sigmoid (perceptron)	$\tanh(\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle / \sigma + c)$
Radial	Gaussian	$\exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / 2\sigma^2)$
	Laplacian	$\exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\| / 2\sigma^2)$
	multiquadratic	$\sqrt{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 + c}$
	inverse multiquadratic	$1/\sqrt{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 + c}$

B. The representer theorem

In machine learning, inferences are focused on the estimation of the structure of some data, based on a set of available data. Given n observations, $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$, and eventually the corresponding labels, $y_1, y_2, \ldots, y_n$, one seeks a function that minimizes a fitness error over the data, with some control of its complexity (i.e., functional norm). To this end, we consider the rkHs associated to the reproducing kernel as the hypothesis space from which the optimal is determined. The rkHs associated to κ can be identified, modulo certain details, with a space of functions defined by a linear combination of the functions $\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_n)$. Its flexibility allows to solve efficiently optimization problems, owing to the (generalized) representer theorem. Originally derived by Kimeldorf and Wahba for splines in [17], it was recently generalized to kernel-based machine learning in [18], including SVM and kernel-PCA, as follows:

Theorem 1 (Representer theorem): For any function $\psi \in \mathcal{H}$ minimizing a regularized cost function of the form

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

with $f(\cdot, \cdot)$ some loss function and $g(\cdot)$ a strictly monotonic increasing function on $\mathbb{R}_+$, can be written as an image expansion in terms of the available data, namely

$$\psi = \sum_{i=1}^{n} \alpha_i\, \phi(\boldsymbol{x}_i). \tag{7}$$

This theorem shows that, even in an infinite dimensional rkHs, one only needs to work in the subspace spanned by the n images of the training data.

Before we proceed further, we examine the effectiveness of this theorem on two machine learning techniques: First, consider the kernel-PCA, where the projected variance is maximized, namely $\psi_1, \psi_2, \ldots, \psi_k = \arg\max_\psi \frac{1}{n} \sum_i |\langle \boldsymbol{x}_i, \psi \rangle|^2$, under the orthonormality constraint, $\langle \psi_\ell, \psi_{\ell'} \rangle_{\mathcal{H}} = \delta_{\ell\ell'}$ for all $\ell, \ell' = 1, 2, \ldots, k$. As derived in the introductory example, one only needs to solve the eigen-problem (3), involving only n unknowns for each principal axe. These unknowns correspond to the weighting coefficients in the expansion (7). Second, we consider a regression problem, known as the ridge regression. In this case, the mean squared error is minimized, with

$$\min_{\psi} \frac{1}{n} \sum_{i=1}^{n} |y_i - \psi(\boldsymbol{x}_i)|^2 + \eta \|\psi\|_{\mathcal{H}}^2, \tag{8}$$

where the first term is the fitness error while the second one controls the complexity of the solution (known as Tikhonov regularization). By substituting (7) into (8), we get the optimization problem

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha}\|^2 + \eta\, \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha},$$

with $\boldsymbol{\alpha} = [\alpha_1 \ \alpha_2 \ \cdots \ \alpha_n]^\top$ and $\boldsymbol{y} = [y_1 \ y_2 \ \cdots \ y_n]^\top$. The optimal weighting coefficients are obtained by solving the linear system

$$(\boldsymbol{K} + \eta \boldsymbol{I})\, \boldsymbol{\alpha} = \boldsymbol{y}, \tag{9}$$

where $\boldsymbol{I}$ is the identity matrix.

Such models as a sum of basis functions have been extensively studied in the literature, for instance in interpolation problems [19] and more recently in machine learning [20]. To illustrate this theorem, take for instance the Gaussian kernel, investigated in [21] for interpolation in two-dimensions. For this kernel, we can think about the map $\phi(\boldsymbol{x}_i)\colon \boldsymbol{x}_i \mapsto \exp(-\|\cdot - \boldsymbol{x}_i\|^2/2\sigma^2)$ that transforms each input data into a Gaussian *bump* centered on that point. Clearly, the representer theorem (Theorem 1) states that the optimal solution is a linear combination of Gaussians centered on the available input data. However, it is well known that a sum-of-Gaussians centered at different points, cannot be written as a single Gaussian. Thus, the solution ψ in (7) cannot be a Gaussian sitting on some arbitrary data; in other words, it is not a valid image of some $\boldsymbol{x} \in \mathcal{X}$, using the map $\phi(\cdot)$ associated to the Gaussian kernel. Finding an input $\boldsymbol{x}^*$ whose image can approximate the function ψ is the pre-image problem.

III. Solving the pre-image problem

A problem is ill-posed if at least one of the following three conditions, which characterize well-posed problems in the sense of Hadamard, is violated: (i) a solution exists, (ii) it is unique, and (iii) it depends continuously on the data (also known as the stability condition). Unfortunately, identifying the pre-image is generally an ill-posed problem. This is an outcome of the higher dimensionality of the feature space compared to the input space. As a consequence, most elements ψ in the rkHs might not have a pre-image in the input space, i.e., there may not exist an $\boldsymbol{x}^*$ such that $\phi(\boldsymbol{x}^*) = \psi$. Moreover, even if $\boldsymbol{x}^*$ exists, it may not be unique. In order to circumvent this difficulty, one seeks an approximate solution, i.e., $\boldsymbol{x}^*$ whose map $\phi(\boldsymbol{x}^*)$ is as close as possible to ψ.

Consider a pattern ψ in the feature space $\mathcal{H}$, obtained by any kernel-based machine, e.g., a principal axe or a denoised pattern obtained from kernel-PCA. By virtue of the Theorem 1, let $\psi = \sum_{i=1}^n \alpha_i\, \phi(\boldsymbol{x}_i)$. The pre-image problem consists of the following optimization problem:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathcal{X}} \Big\| \sum_{i=1}^n \alpha_i\, \phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}) \Big\|_{\mathcal{H}}^2. \tag{10}$$

Equivalently, from the kernel trick, $\boldsymbol{x}^*$ minimizes the objective function

$$\Xi(\boldsymbol{x}) = \kappa(\boldsymbol{x}, \boldsymbol{x}) - 2 \sum_{i=1}^n \alpha_i\, \kappa(\boldsymbol{x}, \boldsymbol{x}_i), \tag{11}$$

where the term independent of $\boldsymbol{x}$ has been dropped.

As opposed to this functional formalism, one may also adopt a vector-wise representation, with elements in the rkHs given by their coordinates with respect to an orthogonal basis. Taking for instance the basis defined by the kernel-PCA, as given in (4), each $\psi \in \mathcal{H}$ is represented vector-wise with $[\langle \psi, \psi_1 \rangle \ \ \langle \psi, \psi_2 \rangle \ \cdots \ \langle \psi, \psi_k \rangle]^\top$, thus defining a k-dimensional representation. In such a case, the Euclidean distance between the latter and the one obtained from the image of $\boldsymbol{x}^*$ is minimized. This is essentially a classical dimensionality reduction problem, connecting the pre-image problem to the historical evolution of dimensionality reduction techniques. This is emphasized next, providing a survey on a large variety of methods.

A. The exact pre-image, when it exists

Suppose for now that there exists an exact pre-image of ψ, i.e., $\boldsymbol{x}^*$ such that $\phi(\boldsymbol{x}^*) = \psi$, then the optimization problem in (10) results into that pre-image. Furthermore, the pre-image can be easily computed when the kernel is an invertible function of $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$, such as some projective kernels including the polynomial kernel with odd degree and the sigmoid kernel (see Table I). Let $h\colon \mathbb{R} \to \mathbb{R}$ defines the inverse function such that $h(\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$. Then, given any orthonormal basis in the input space $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_N\}$, every element $\boldsymbol{x} \in \mathcal{X}$ can be written as

$$\boldsymbol{x} = \sum_{j=1}^N \langle \boldsymbol{e}_j, \boldsymbol{x} \rangle\, \boldsymbol{e}_j = \sum_{j=1}^N h\big(\kappa(\boldsymbol{e}_j, \boldsymbol{x})\big)\, \boldsymbol{e}_j.$$

As a consequence, the exact pre-image $\boldsymbol{x}^*$ of some pattern $\psi = \sum_{i=1}^n \alpha_i\, \phi(\boldsymbol{x}_i)$, namely $\phi(\boldsymbol{x}^*) = \psi$, can be expanded as

$$\boldsymbol{x}^* = \sum_{j=1}^N h\Big(\sum_{i=1}^n \alpha_i\, \kappa(\boldsymbol{e}_j, \boldsymbol{x}_i) \Big)\, \boldsymbol{e}_j.$$

Likewise, when the kernel is an invertible function of the distance, such as radial kernels, a similar expression can be derived by using the polarization identity $4\,\langle \boldsymbol{x}^*, \boldsymbol{e}_j \rangle = \|\boldsymbol{x}^* + \boldsymbol{e}_j\|^2 - \|\boldsymbol{x}^* - \boldsymbol{e}_j\|^2$ [22].

Clearly, such a simple derivation for the pre-image is only valid under the crucial assumption that the pre-image

$\boldsymbol{x}^*$ exists. Unfortunately, for a large class of kernels, there are no exact pre-images. Rather than seeking the exact pre-image, we consider an approximate pre-image by solving the optimization problem in (10). In what follows, we present several strategies for solving this problem. We first review techniques based on classical optimization schemes. We then present learning-based techniques, incorporating additional prior information.

B. Gradient descent techniques

Gradient descent is one of the simplest optimization techniques. It requires computing the gradient of the objective function (11), denoted $\nabla_{\boldsymbol{x}}\Xi(\boldsymbol{x}^*)$. In its simplest form, the current guess $\boldsymbol{x}_t^*$ is updated into $\boldsymbol{x}_{t+1}^*$ by stepping into the direction opposite to the gradient, with

$$\boldsymbol{x}_{t+1}^* = \boldsymbol{x}_t^* - \eta_t \, \nabla_{\boldsymbol{x}} J(\boldsymbol{x}_t^*)$$

where η_t is a step size parameter, often optimized using a line-search procedure. As an alternative to the gradient descent, one may use more sophisticated techniques, such as Newton's method. Unfortunately, the objective function is inherently nonlinear and clearly non-convex. Thus, a gradient descent algorithm must be run many times with several different starting values, in hope that a feasible solution will be amongst the local minima obtained over the runs.

C. Fixed-point iteration method

The structure of kernel functions provides useful insights to derive more appropriate optimization techniques, beyond classical gradient descent. More precisely, the gradient of expression (11) has a closed-form expression for most kernels. By setting this expression to zero, this greatly simplifies the optimization scheme, resulting into a fixed-point iterative technique. Taking for instance the Gaussian kernel [7], the objective function in (11) becomes

$$-2\sum_{i=1}^{n} \alpha_i \, \exp(-\|\boldsymbol{x} - \boldsymbol{x}_i\|^2/2\sigma^2),$$

with its gradient

$$\nabla_{\boldsymbol{x}}\Xi(\boldsymbol{x}) = -\frac{2}{\sigma^2}\sum_{i=1}^{n} \alpha_i \, \exp(-\|\boldsymbol{x} - \boldsymbol{x}_i\|^2/2\sigma^2)\,\big(\boldsymbol{x} - \boldsymbol{x}_i\big).$$

We get the pre-image by setting this gradient to zero, which results into the fixed-point iterative expression

$$\boldsymbol{x}_{t+1}^* = \frac{\sum_{i=1}^{n} \alpha_i \, \kappa(\boldsymbol{x}_t^*, \boldsymbol{x}_i) \; \boldsymbol{x}_i}{\sum_{i=1}^{n} \alpha_i \, \kappa(\boldsymbol{x}_t^*, \boldsymbol{x}_i)},$$

with $\kappa(\boldsymbol{x}_t^*, \boldsymbol{x}_i) = \exp(-\|\boldsymbol{x}_t^* - \boldsymbol{x}_i\|^2/2\sigma^2)$. Similar expressions can be derived for most kernels, such as the polynomial kernel of degree p [23] with

$$\boldsymbol{x}_{t+1}^* = \sum_{i=1}^{n} \alpha_i \left(\frac{\langle \boldsymbol{x}_t^*, \boldsymbol{x}_i\rangle + c}{\langle \boldsymbol{x}_t^*, \boldsymbol{x}_t^*\rangle + c}\right)^{p-1} \boldsymbol{x}_i.$$

Unfortunately, the fixed-point iterative technique still suffers from local minima and tends to be unstable. The numerical instability occurs especially when the value of the denominator decreases to zero. To prevent this situation, a regularized solution can be easily formulated, as studied in [24].

An interesting fact about the fixed-point iterative method is that the resulting pre-image lies in the span of the available data, taking the form $\boldsymbol{x}^* = \sum_i \beta_i \, \boldsymbol{x}_i$ for some coefficients $\beta_1, \beta_2, \ldots, \beta_n$ to be determined. Thus, the search space is controlled, as opposed to gradient descent techniques that explore the entire space. We further exploit information from available training data, and their mapped counterparts, as discussed next.

D. Learning the pre-image map

To find the pre-image map, a learning machine is constructed with training elements from the feature space and estimated values in the input space, as follows: we seek to estimate a function Γ^* with the property that $\Gamma^*(\phi(\boldsymbol{x}_i)) = \boldsymbol{x}_i$, for $i = 1, 2, \ldots, n$. Then, ideally, $\Gamma^*(\psi)$ should give $\boldsymbol{x}^*$, the pre-image of ψ. In order to make the problem computationally tractable, two issues are considered in [25], [26]. First, the function is defined on a vector space. This can be done by representing vector-wise any $\psi \in \mathcal{H}$ with $[\langle \psi, \psi_1\rangle \; \langle \psi, \psi_2\rangle \; \cdots \; \langle \psi, \psi_k\rangle]^\top$, using an orthogonal basis obtained from kernel-PCA. Second, the pre-image map Γ^* is decomposed into $\dim(\mathcal{X})$ functions to estimate each component of $\boldsymbol{x}^*$. From these considerations, we seek functions $\Gamma_1^*, \Gamma_2^*, \ldots, \Gamma_{\dim(\mathcal{X})}^*$, with $\Gamma_m^* \colon \mathbb{R}^k \to \mathbb{R}$. Each of these functions is obtained by solving the optimization problem

$$\Gamma_m^* = \arg\min_{\Gamma} \sum_{i=1}^{n} f\big([\boldsymbol{x}_i]_m, \Gamma(\psi)\big) + \eta \, g(\|\Gamma\|^2)$$

where $f(\cdot,\cdot)$ is some loss function, and $[\,\cdot\,]_m$ denotes the m-th component operator. By taking for instance the distance as a loss function, we get

$$\Gamma_m^* = \arg\min_{\Gamma} \frac{1}{n}\sum_{i=1}^{n} \big|[\boldsymbol{x}_i]_m - \Gamma(\psi)\big|^2 + \eta \, \|\Gamma\|^2.$$

This optimization problem can be easily solved by a matrix inversion scheme, in analogy to the ridge regression problem (8) and its linear system (9). This learning approach is further investigated in the literature, incorporating neighborhood

information [27] and regularization with a penalized learning [28]. All these methods are based on a set of available data in the input space and the associated images in the rkHs. The method discussed next carries this concept further, by exploring pairwise distances in both spaces.

E. MDS-based technique

As illustrated in the above pre-image learning approach, the pre-image map seeks data in the input space based on their associated images in the rkHs. Essentially, this is a low-dimensional embedding of *objects* from a high-dimensional space. This problem has received a lot of attention in multivariate statistics, under the framework of Multidimensional Scaling (MDS) [29]. MDS techniques mainly embed data in a low-dimensional space, by preserving pairwise distances. This approach has been applied with success to solve the pre-image problem [23]. Consider each distance in the rkHs $\delta_i = \|\psi - \phi(\boldsymbol{x}_i)\|_{\mathcal{H}}$, and its counterpart in the input space $\|\boldsymbol{x}^* - \boldsymbol{x}_i\|$. Ideally, these distances are preserved, namely

$$\|\boldsymbol{x}^* - \boldsymbol{x}_i\|^2 = \|\psi - \phi(\boldsymbol{x}_i)\|^2_{\mathcal{H}}, \tag{12}$$

for every $i = 1, 2, \ldots, n$. It is easy to verify that if there exists a i such that $\psi = \phi(\boldsymbol{x}_i)$, then we get the pre-image $\boldsymbol{x}^* = \boldsymbol{x}_i$.

One way to solve this problem is to minimize the mean square error between these distances, with

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \sum_{i=1}^{n} \left| \|\boldsymbol{x} - \boldsymbol{x}_i\|^2 - \|\psi - \phi(\boldsymbol{x}_i)\|^2_{\mathcal{H}} \right|^2 .$$

To solve this optimization problem, a fixed-point iteration method is proposed by setting the gradient of the above expression to zero, resulting into the expression

$$\boldsymbol{x}^* = \frac{\sum_{i=1}^{n} \left(\|\boldsymbol{x}^* - \boldsymbol{x}_i\|^2 - \delta_i^2 \right) \boldsymbol{x}_i}{\sum_{i=1}^{n} \left(\|\boldsymbol{x}^* - \boldsymbol{x}_i\|^2 - \delta_i^2 \right)}.$$

Another approach to solve this problem is to consider separately the identities (12), resulting into n equations

$$2\langle \boldsymbol{x}^*, \boldsymbol{x}_i \rangle = \langle \boldsymbol{x}^*, \boldsymbol{x}^* \rangle + \langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle - \delta_i^2,$$

for $i = 1, 2, \ldots, n$. In these expressions, the unknown appears also on the right-hand side, with $\langle \boldsymbol{x}^*, \boldsymbol{x}^* \rangle$. This unknown quantity can be easily identified in the case of centered data, since taking the average of both sides results into

$$\langle \boldsymbol{x}^*, \boldsymbol{x}^* \rangle = \frac{1}{n} \sum_{i=1}^{n} \left(\delta_i^2 - \langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle \right)$$

Let $\boldsymbol{\epsilon}$ be the vector having all its entries equal to $\frac{1}{n}\sum_{i=1}^{n}(\delta_i^2 - \langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle)$ then, in matrix form, we have

$$2\boldsymbol{X}^\top \boldsymbol{x}^* = \operatorname{diag}\big(\boldsymbol{X}^\top \boldsymbol{X}\big) - \begin{bmatrix} \delta_1^2 & \delta_2^2 & \cdots & \delta_n^2 \end{bmatrix}^\top + \boldsymbol{\epsilon},$$

where $\boldsymbol{X} = [\boldsymbol{x}_1 \ \boldsymbol{x}_2 \ \cdots \ \boldsymbol{x}_n]$ and $\operatorname{diag}(\cdot)$ is the diagonal operator with $\operatorname{diag}(\boldsymbol{X}^\top \boldsymbol{X})$ the column vector with entries $\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle$. The unknown pre-image is obtained using the least squares solution, namely

$$\boldsymbol{x}^* = \tfrac{1}{2}(\boldsymbol{X}\boldsymbol{X}^\top)^{-1}\boldsymbol{X} \left(\operatorname{diag}(\boldsymbol{X}^\top \boldsymbol{X}) - [\delta_1^2 \ \delta_2^2 \ \cdots \ \delta_n^2]^\top \right),$$

where the term $(\boldsymbol{X}\boldsymbol{X}^\top)^{-1}\boldsymbol{X}\boldsymbol{\epsilon}$ goes to zero thanks to the assumption of centered data.

To keep this technique tractable in practice, only a certain neighborhood is considered in the pre-image estimation, in the same spirit as the locally linear embedding scheme in dimensionality reduction [30]. This approach opened the door to a range of other techniques, borrowed from dimensionality reduction and manifold learning literature [31].

F. Conformal map approach

Besides the distance preserving method of MDS, one may also propose a pre-image method by preserving inner product measures. Using such a strategy, the angular measure is also preserved, since $\boldsymbol{x}_i^\top \boldsymbol{x}_j / \|\boldsymbol{x}_i\| \|\boldsymbol{x}_j\|$ defines the cosine of the angle between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in the Euclidean input space. For this reason, it is called the conformal map approach. A recent technique to solve the pre-image problem based on the conformal map has been presented in [32]. To this end, a coordinate system in the rkHs is constructed with an isometry with respect to the input space. We emphasize the fact that the model is not coupled with any constraint on the coordinate functions, as opposed to the orthogonality between the functions resulting from the kernel-PCA.

By virtue of Theorem 1, each of the n coordinate functions can be written as a linear expansion of the available images, namely $\Psi_\ell = \sum_{i=1}^{n} \theta_{\ell,i}\, \phi(\boldsymbol{x}_i)$, for $\ell = 1, 2, \ldots, n$, with unknown weights to be determined, rearranged in a matrix $\boldsymbol{\Theta}$. Therefore, the coordinates of any element of the rkHs can be obtained by a projection onto these coordinate functions, thus any $\phi(\boldsymbol{x}_i)$ can be represented with the n coordinates in $\boldsymbol{\Psi}_{\boldsymbol{x}_i} = [\langle \Psi_1, \phi(\boldsymbol{x}_i) \rangle \ \ \langle \Psi_2, \phi(\boldsymbol{x}_i) \rangle \ \cdots \ \langle \Psi_k, \phi(\boldsymbol{x}_i) \rangle]^\top$. Ideally, the inner products are preserved in both this coordinate system and the Euclidean input space, namely

$$\boldsymbol{\Psi}_{\boldsymbol{x}_i}^\top \boldsymbol{\Psi}_{\boldsymbol{x}_j} = \boldsymbol{x}_i^\top \boldsymbol{x}_j, \tag{13}$$

for all $i, j = 1, 2, \ldots, n$. This can be solved by minimizing the fitness error over all pairs,

$$\min_{\Psi_1, \ldots, \Psi_n} \sum_{i,j=1}^{n} \left| \boldsymbol{x}_i^\top \boldsymbol{x}_j - \boldsymbol{\Psi}_{\boldsymbol{x}_i}^\top \boldsymbol{\Psi}_{\boldsymbol{x}_j} \right|^2 + \eta \sum_{\ell=1}^{n} \|\Psi_\ell\|^2_{\mathcal{H}},$$

where the second term incorporates regularization. This can be written in matrix form as

$$\min_{\boldsymbol{\Theta}} \frac{1}{2} \|\boldsymbol{X}^\top \boldsymbol{X} - \boldsymbol{K}\,\boldsymbol{\Theta}^\top \boldsymbol{\Theta}\,\boldsymbol{K}\|_F^2 + \eta \operatorname{tr}(\boldsymbol{\Theta}^\top \boldsymbol{\Theta} \boldsymbol{K}),$$

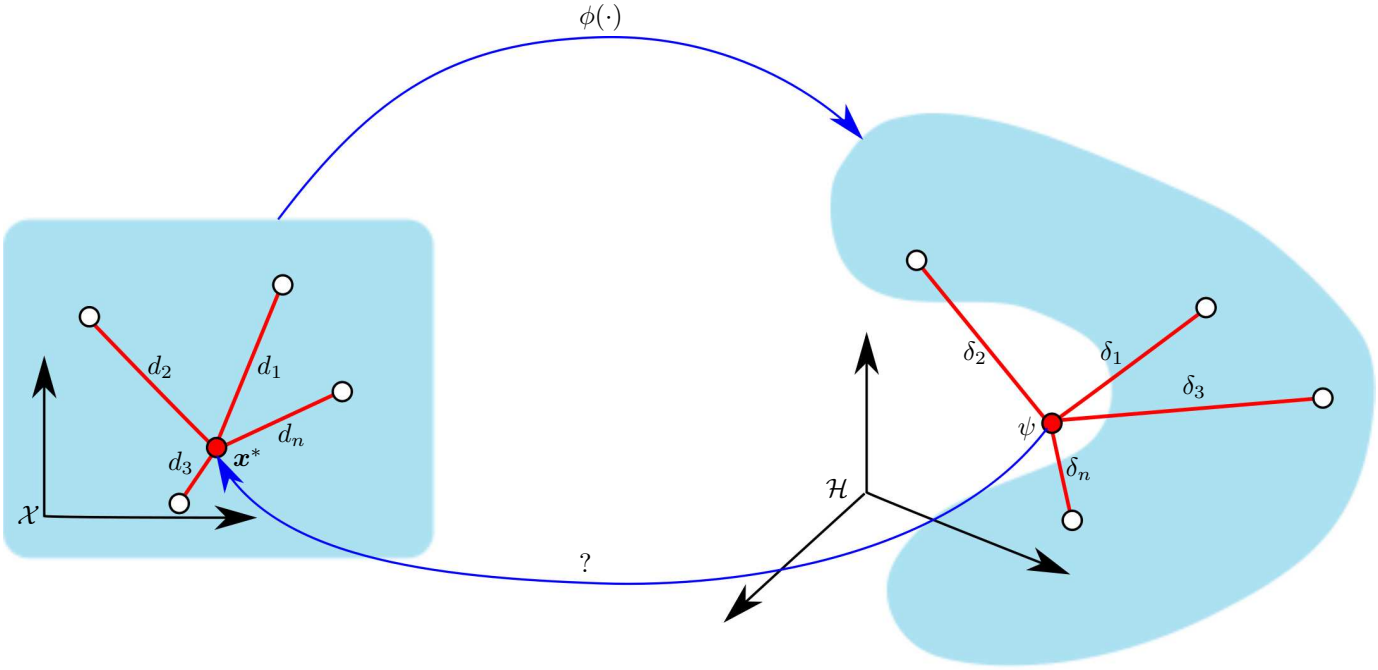

Fig. 3. Schematic illustration of the MDS-based technique where the pre-image is identified from pairwise distances in both input and feature spaces.

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix and $\|\cdot\|_F$ the Frobenius norm, i.e., the root of sum of squared (absolute) values of all its elements, or equivalently $\|\boldsymbol{M}\|_F^2 = \mathrm{tr}(\boldsymbol{M}^\top \boldsymbol{M})$. By taking the derivative of this expression with respect to $\boldsymbol{\Theta}^\top \boldsymbol{\Theta}$, one obtains

$$\boldsymbol{\Theta}^\top \boldsymbol{\Theta} = \boldsymbol{K}^{-1} \left(\boldsymbol{X}^\top \boldsymbol{X} - \eta\, \boldsymbol{K}^{-1} \right) \boldsymbol{K}^{-1}. \qquad (14)$$

Now we are in a position to determine the pre-image of some $\psi = \sum_{i=1}^n \alpha_i\, \phi(\boldsymbol{x}_i)$. Its coordinates associated to the system of coordinate functions $\Psi_1, \Psi_2, \ldots, \Psi_n$ are given by

$$\langle \psi, \Psi_\ell \rangle_{\mathcal{H}} = \sum_{i,j=1}^{n} \theta_{\ell,i}\, \alpha_j\, \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

for $\ell = 1, 2, \ldots, n$. By preserving the inner products in both spaces, ideally the model in (13) can be extended to ψ, resulting into

$$\boldsymbol{X}^\top \boldsymbol{x}^* = \boldsymbol{K} \boldsymbol{\Theta}^\top \boldsymbol{\Theta}\, \boldsymbol{K} \boldsymbol{\alpha}.$$

By combining this expression with (14), we get the simplified expression $\boldsymbol{X}^\top \boldsymbol{x}^* = (\boldsymbol{X}^\top \boldsymbol{X} - \eta\, \boldsymbol{K}^{-1})\, \boldsymbol{\alpha}$, whose least squares solution is

$$\boldsymbol{x}^* = (\boldsymbol{X} \boldsymbol{X}^\top)^{-1} \boldsymbol{X} (\boldsymbol{X}^\top \boldsymbol{X} - \eta\, \boldsymbol{K}^{-1})\, \boldsymbol{\alpha}.$$

It is worth noting that this expression is independent of the kernel type under investigation.

Furthermore, this technique can be easily extended to identify the pre-images of a set of elements in the rkHs, since the term between parentheses needs to be computed only once. In fact, this is a matrix completion scheme, as the one studied in [33]. This corresponds to completing an inner-product matrix based on another Gram matrix, here the matrix of kernel values.

IV. Scope of application of the pre-image problem

In this section, we present some application examples that involve solving the pre-image problem. Our first experiments are with kernel-PCA on toy data, and are mainly intended to illustrated the pre-image problem. Then, we provide a comparative study of several methods presented in this paper, on an image denoising problem. Finally, we show how the pre-image can be required in other applications, beyond kernel-PCA. To this end, we consider a problem of auto-localization of sensors in wireless sensor networks.

A. Some applications of kernel-PCA with pre-image

1. Feature extraction

A first illustration considered here is the use of kernel-PCA on a synthetic data to provide a visual illustration of PCA vs. kernel-PCA for feature extraction. The data distribution takes the form of a ring in 2D, with an inner diameter of 2 and an outer diameter of 3. Within this region, $n = 600$ training data were generated, as illustrated in Figure 4 with blue dots. In order to extract the most relevant feature, two methods were used: on the one hand the conventional PCA and on the other hand kernel-PCA with a pre-image step. The PCA technique provided linear axes by solving the eigenvector problem, and thus did not capture the circular shape of the data. This is illustrated by projecting data onto the first principal axis, given

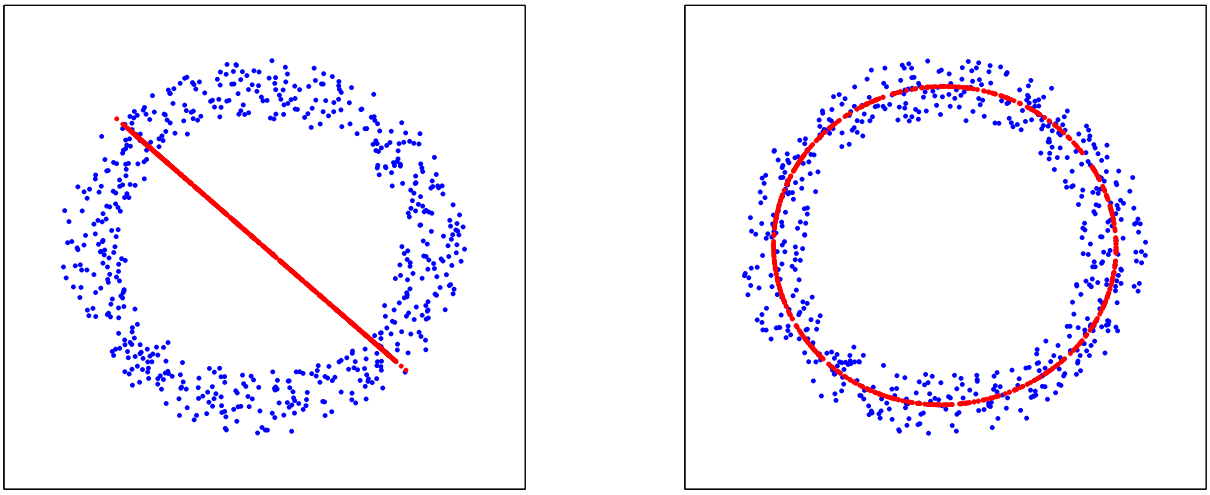

Fig. 4. Denoising data distributed on a ring, using classical PCA (left) and kernel-PCA with pre-image (right). The extracted feature is linear in the first case, and circular in the second.

by red dots in Figure 4 (left). The kernel-PCA was applied using a Gaussian kernel with bandwidth $\sigma = 2$, the principal axes being defined by a sum of n Gaussian functions in an infinite dimensional feature space. A pre-image method was required to derive the axes, or representations of these axes, within the input space. As shown in Figure 4 (right), this technique captured the nonlinear feature in the original space.

As described at the beginning of this paper, when we introduced the pre-image problem with the Gaussian kernel, each data is mapped into a Gaussian *bump* centered around it. By taking the sum of theses Gaussians, with some optimized weighting coefficients, we get the principal distribution whose *mean*, if it exists, provides the pre-image. It is worth noting that the definition of a mean only exists and makes sense for Gaussian-like curves, and not for a sum-of-Gaussians centered at different points. A schematic illustration of the pre-image problem is given in Figure 5, taking only a (one-directional) radial cut in the ring-distributed data. The data obtained by solving the pre-image problem can be interpreted as the center of the distribution Gaussian which best approximates the sum-of-Gaussians.

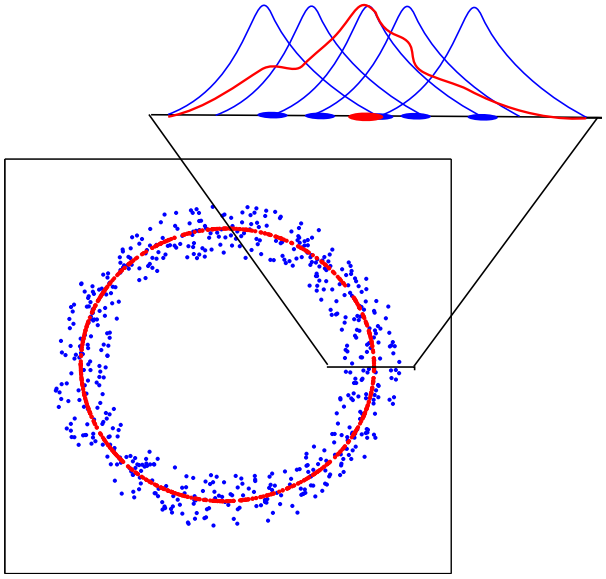

Fig. 5. Schematic illustration of the pre-image problem with the Gaussian kernel, where the profile corresponds to a radial cut in the ring-distributed data. From the sum-of-Gaussians (red curve), the pre-image corresponds to the mean value of the distribution (red dot).

In this application, the fixed-point iterative technique was used. Next, we give a comparative study of several techniques given in this paper, by considering an image denoising problem.

2. Image denoising

In this section, we illustrate the results obtained in a problem of real image denoising, using three techniques: the fixed-point iterative method, the MDS-based technique, and the conformal map approach. The images were consisting of the MINST database of handwritten digits [34], corresponding to handwritten digits, from "0" to "9", in (almost) binary 28-by-28 pixels. From a machine learning point of view, each image can be represented as a *point* in a 28×28 dimensional

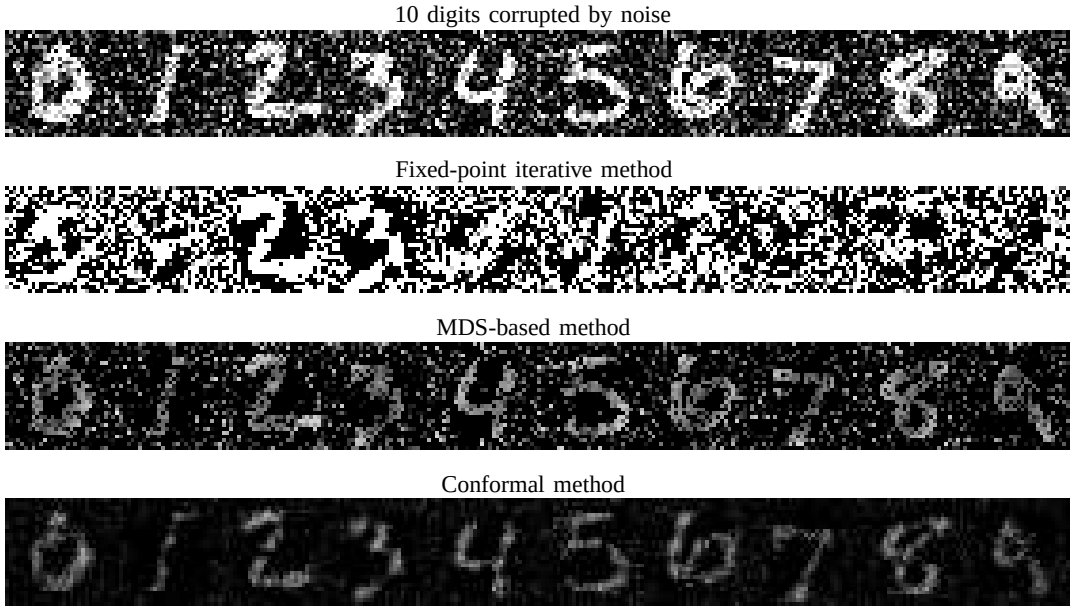

Fig. 6. Application to handwritten digit denoising with kernel-PCA, using several pre-image methods presented in this paper.

space. The original images were corrupted by adding a zero-mean white Gaussian noise with variance 0.2. In the training stage, a set of $1\,000$ images, 100 of each digit, were used to train the kernel-PCA, retaining only 100 leading principal axes. We used the Gaussian kernel for the three algorithms, with bandwidth set to $\sigma = 10^5$.

To illustrate the ability of this method for image denoising, another set of 10 images, one for each digit, was considered under the same noise conditions. These images are illustrated in Figure 6 (first row), with results obtained with the fixed-point iterative (second row), the MDS-based (third row) and the conformal (fourth row) methods. For such applications, the fixed-point iterative algorithm was found to be inappropriate, even with a large number of iterations (here $10\,000$ iterations were used). To take advantage of prior knowledge, the same training dataset was used for learning the reverse map. Realistic results were obtained using the MDS-based method. It is obvious that the conformal algorithm achieved better denoised results. For this simulation, the regularization parameter was set to $\eta = 10^{-9}$.

In an attempt to provide a measure of computational requirements, we considered the (average) total CPU time of each algorithm. These algorithms were implemented on a Matlab running on a MacBook Pro Duo Core, to offer a comparative study. With $10\,000$ iterations, the fixed-point iterative algorithm required a total CPU time of up to 1 hour. The MDS-based and the conformal algorithms required 5 minutes and 1.5 seconds, respectively.

B. Auto-localization in wireless sensor networks

With recent technological advances in both electronics and wireless communications, low-power and low-cost tiny sensors have been developed for monitoring physical phenomena and tracking applications. Densely deployed in the inspected environment with efficiently designed distributed algorithms, wireless ad-hoc networks seem to offer several opportunities. They were successfully employed in many situations, ranging from military applications such as battleground supervision, to civilian applications such as habitat monitoring and healthcare surveillance (see [35], [36] and references therein). While these sensors are often randomly deployed, e.g., for monitoring inhospitable habitats and disaster areas, information captured by each sensor remains obsolete as long as it stays unaware of its location. Implementing a self-localization device, such as a GPS receiver, at each sensor device may be too expensive and too power hungry for the desired application with battery-powered devices. As a consequence, only a small fraction of the sensors may be location aware, the so-called anchors or beacons. The other sensors have to estimate their locations by exchanging some information with its neighbors.

For this purpose, each sensor determines a ranging (distance) with other sensors, from inter-sensor measurements

such as the received signal strength indication (RSSI), the connectivity, the hop count, the time difference of arrival, ... Most methods used for auto-localization in sensor networks are based on either MDS techniques or semidefinite programming (for a survey, see [37], [38]), identifying a function that links the ranging between sensors to their locations. However, if the data are not inter-sensor distances or are linked to coordinates by an unknown nonlinear function, e.g., using the RSSI measurements or the estimated covariance sensor data [39], linear techniques such as MDS and PCA fail to accurately estimate the locations. Once again, the kernel machines provide an elegant way to overcome this drawback.

Here, we describe the method proposed in [40]. The main idea can be described in three stages. In the first stage, we construct the reproducing kernel and its associated rkHs which best describes the anchor pairwise similarities. In the second stage, a nonlinear manifold is designed from similarities between anchor-sensors measurements, by applying a kernel-PCA technique. The final stage consists of estimating the coordinates of non-anchor sensors by applying a pre-image technique on their projections onto the manifold. Next, we describe these three stages, before presenting experimental results.

Consider a network of N sensor nodes, with n location-aware anchors and $N-n$ sensors of unknown location, living in a p-dimensional space, e.g., $p = 2$ for localization in a plane. Let $\boldsymbol{x}_i \in \mathbb{R}^p$ be the coordinates of the i-th sensor, rearranged such that indices $i = 1, 2, \ldots, n$ correspond to anchors. Let $\tilde{\boldsymbol{K}}(i,j)$ be the inter-sensor similarity between sensors i and j, such as the RSSI.

Kernel selection from inter-anchor similarities

As a model of the similarity measurements, the appropriate reproducing kernel should be chosen, and tuned up, which allows a physical meaning of the results obtained from the kernel-PCA (next stage). The alignment criterion [41] provides a measure of similarity between a reproducing kernel and a target function, e.g., between a Gaussian kernel and the RSSI measurements. Maximizing the alignment $\mathcal{A}(\boldsymbol{K}, \tilde{\boldsymbol{K}})$ provides the optimal reproducing kernel, faithful to the inter-anchor measurements, where

$$\mathcal{A}(\boldsymbol{K}, \tilde{\boldsymbol{K}}) = \frac{\langle \boldsymbol{K}, \tilde{\boldsymbol{K}} \rangle_F}{\sqrt{\langle \boldsymbol{K}, \boldsymbol{K} \rangle_F \langle \tilde{\boldsymbol{K}}, \tilde{\boldsymbol{K}} \rangle_F}},$$

with $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product between two matrices. Taking for instance the Gaussian kernel, the optimization problem is reduced to finding the optimal bandwidth. In practice, this optimization problem is solved at each anchor, using only information from its neighborhood.

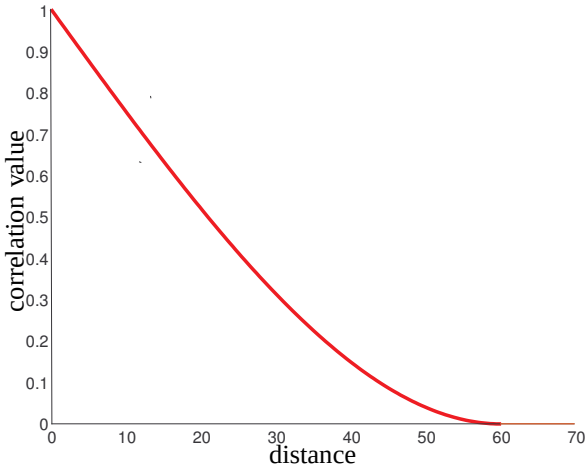

Fig. 7. Profile of the spherical model, as a function of the distance. The cut-off distance is set to $d = 60$.

Kernel-PCA upon anchors

After identifying the reproducing kernel adapted to the measurements, a kernel-PCA approach is applied to provide the most relevant subspace of the associated rkHs. Classical kernel-PCA is computed by a diagonalization scheme, which may be computational expensive for in-network processing. An alternative approach can be done using an iterative scheme, such as the *kernel-Hebbian algorithm* [42] (we refer the reader to [40] for its implementation in wireless sensor networks).

Pre-image for location estimation

For each sensor, we represent its image in the rkHs associated to the kernel maximizing the alignment criterion. The image is projected onto the manifold obtained using kernel-PCA with anchor pairwise similarities. The problem of estimating the coordinates from that representation is the pre-image problem.

Experimental results

A first batch of experiments was carried out on simulated measurements. For this purpose, we considered a network of sensors measuring some physical phenomena, e.g., temperature, atmospheric pressure or luminance. In a static field, we assumed that measurements were jointly generated from a normal distribution, with decreasing correlations between measurements as a function of the distance between sensors. This information was used as a local similarity measure between sensors [39]. More precisely, we considered the spherical model, commonly used in environmental and geological sciences [43], and defined by a covariance of the form

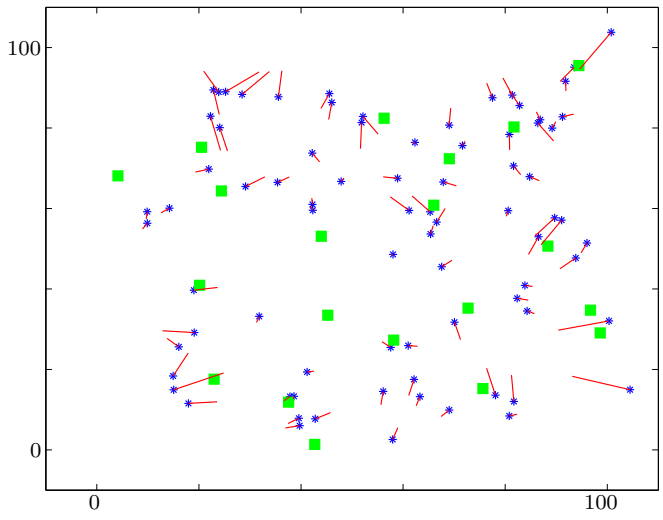

Fig. 8. Estimated locations of 80 sensors ($*$) based on 20 anchors of known positions ($\blacksquare$), with error to real position represented by a line ($-$).

$\zeta(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$ with

$$\zeta(u) = \begin{cases} 1 - \frac{3}{2d}u + \frac{1}{2d^3}u^3 & \text{for } 0 \leq u \leq d; \\ 0 & \text{for } d < u, \end{cases}$$

where d denotes the cut-off distance, and fixed to $d = 60$ in our experiments. The profile of the spherical model is illustrated in Figure 7. The experiments consisted of 100 sensors, from which 20 were anchors with known locations, randomly spread over a 100-by-100 square region. For each sensor, 200 measurements were collected, and the Gaussian kernel was considered. Figure 8 illustrates the localization results obtained with this method.

In a second experiment, real measurements of RSSI were collected from an indoor experiment, at the Motorola facility in Plantation, FL. The environment is a 14-by-13 meters office area, partitioned by cubicle walls (height = 1.8 meter). The network consisted of 40 unknown-location sensors, and 4 anchors near the corners. The experimental settings are described more in detail in [44] (see also http://www.eecs.umich.edu/∼hero/localize/). For each sensor i, we collected the RSSI associated to it in a 44-dimensional vector, denoted by $\boldsymbol{u}_i$. The inter-sensor similarity between sensors is given by the matrix $\tilde{\boldsymbol{K}}$, defined between sensor i and sensor j by

$$\tilde{\boldsymbol{K}}(i,j) = \exp(-\|\boldsymbol{u}_i - \boldsymbol{u}_j\|^2/200).$$

The Gaussian kernel was considered, with its bandwidth optimized by maximizing the alignment. The proposed method gives a root-mean-square location error, over the 40 sensors, of 2.13 meters. This should be compared to the maximum-likelihood estimator studied in [44] (that turned out to be biased), having a root-mean-square location error of 2.18 meters.

V. Final remarks

This article presented the pre-image problem in machine learning, providing an overview of the state-of-the-art methods and approaches for solving such a problem. Our aim was to show how this problem is intimately related to dimensionality reduction issues, borrowing and enhancing ideas derived from dimensionality reduction and manifold learning. Throughout this paper, we studied this problem for kernel-PCA, and provided a comparative study of several methods for image denoising. We extended the range of application of the pre-image problem to another context, sensor auto-localization in wireless sensor networks.

By interpreting in the original input space the processing performed in the feature space, this strategy opens the way to a range of diverse signal processing problems. These problems are nonlinear kernel-based formulations of classical signal processing methods, including the independent component analysis [45] and the Kalman filter [46]. Another area of application is the pre-image problem on structured spaces, including biological sequence analysis in bioinformatics [47] and string analysis in natural language [48]. In the latter, the authors derived a pre-image solution for a string kernel, using a graph-theoretical formulation. All these promising areas of application of the pre-image problem open an avenue for future work.

References

[1] 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.

[2] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[3] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K. Müller, "Fisher discriminant analysis with kernels," in *Advances in neural networks for signal processing*, Y. H. Hu, J. Larsen, E. Wilson, and S. Douglas, Eds. San Mateo, CA, USA: Morgan Kaufmann, 1999, pp. 41–48.

[4] R. Rosipal and L. Trejo, "Kernel partial least squares regression in reproducing kernel hilbert space," *Journal of Machine Learning Research*, vol. 2, pp. 97–123, 2002.

[5] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, March 2009.

[6] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," in *Advances in Neural Information Processing Systems*, vol. 12, 2000.

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

[8] G. Camps-Valls, J. L. Rojo-Alvarez, and M.-R. M. (Editors), *Kernel Methods in Bioengineering, Signal And Image Processing*. Hershey, PA, USA: IGI Publishing, 2007.

[9] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[10] M. Aizerman, E. Braverman, and L. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control*, vol. 25, pp. 821–837, 1964.

[11] J. Mercer, "Functions of positive and negative type and their connection with the theory of integral equations," *Royal Society of London Philosophical Transactions Series A*, vol. 209, pp. 415–446, 1909.

[12] N. Aronszajn, "Theory of reproducing kernels," *Trans. of the American Mathematical Society*, vol. 68, pp. 337–404, 1950.

[13] D. Alpay, Ed., *Reproducing kernel spaces and applications*, ser. Operator Theory: Advances and Applications. Birkhäuser, 2003, vol. 143.

[14] E. Parzen, "Statistical inference on time series by RKHS methods," in *Proc. 12th Biennial Seminar*, R. Pyke, Ed. Montreal, Canada: Canadian Mathematical Congress, 1970, pp. 1–37.

[15] T. Kailath, "RKHS approach to detection and estimation problems–I: Deterministic signals in gaussian noise," *IEEE Trans. on Information Theory*, vol. 17, no. 5, pp. 530–549, Sept. 1971.

[16] G. Wahba, *Spline Models for Observational Data*. Philadelphia: Society for Industrial and Applied Math (SIAM), 1990.

[17] G. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of Mathematical Analysis and Applications*, vol. 33, pp. 82–95, 1971.

[18] 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.

[19] C. A. Micchelli, "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constructive Approximation*, vol. 2, no. 1, pp. 11–22, December 1986.

[20] F. Girosi, M. Jones, and T. Poggio, "Priors stabilizers and basis functions: From regularization to radial, tensor and additive splines," Cambridge, MA, USA, Tech. Rep., 1993.

[21] I. P. Schagen, "Interpolation in two dimensions—a new technique," *Journal of the Institute of Mathematics and its Applications*, vol. 23, no. 1, pp. 53–59, 1979.

[22] B. Schölkopf, "Support vector learning," Ph.D. dissertation, Technischen Universität Berlin, Germany, 1997.

[23] J. T. Kwok and I. W. Tsang, "The pre-image problem in kernel methods," in *Proc. 20th International Conference on Machine Learning (ICML)*. Washington, DC, USA: AAAI Press, August 2003, pp. 408–415.

[24] T. J. Abrahamsen and L. K. Hansen, "Input space regularization stabilizes pre-images for kernel PCA de-noising," in *IEEE Workshop on Machine Learning for Signal Processing*, Grenoble, France, 2009.

[25] G. Bakir, J. Weston, and B. Schölkopf, "Learning to find pre-images," in *NIPS 2003*, L. S. Thrun, S. and B. Schölkopf, Eds., vol. 16. Cambridge, MA, USA: MIT Press, 2004, pp. 449–456.

[26] G. Bakir, "Extension to kernel dependency estimation with applications to robotics," Ph.D. dissertation, Technische Universitat Berlin, November 2005.

[27] W.-S. Zheng and J.-H. Lai, "Regularized locality preserving learning of pre-image problem in kernel principal component analysis," in *Proc. of the 18th International Conference on Pattern Recognition (ICPR)*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 456–459.

[28] W.-S. Zheng, J. H. Lai, and P. C. Yuen, "Penalized preimage learning in kernel principal component analysis," *IEEE Trans. on Neural Networks*, 2010.

[29] T. F. Cox and M. A. A. Cox, *Multidimensional Scaling*, 2nd ed., ser. Monographs on Statistics and Applied Probability. London: Chapman and Hall / CRC, September 2000.

[30] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[31] P. Etyngier, F. Ségonne, and R. Keriven, "Shape priors using manifold learning techniques," in *Proc. 11th IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2007, pp. 1–8.

[32] P. Honeine and C. Richard, "Solving the pre-image problem in kernel machines: a direct method," in *Proc. IEEE Workshop on Machine Learning for Signal Processing (MLSP)*, Grenoble, France, September 2009, winner of the Best Paper Award.

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

[34] Y. Lecun and C. Cortes, "The MNIST database of handwritten digits," 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[35] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 4. Los Alamitos, CA, USA: IEEE Computer Society, 2001, pp. 2033–2036.

[36] C. S. Raghavendra and K. Sivalingam, Eds., *Proc. of the 2nd ACM international workshop on Wireless sensor networks and applications*. San Diego, CA, USA: ACM, September 2003.

[37] J. Bachrach and C. Taylor, "Localization in sensor networks," in *Handbook of Sensor Networks*, I. Stojmenovic, Ed., 2005.

[38] G. Mao, B. Fidan, and B. Anderson, "Wireless sensor network localization techniques," *Comput. Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[39] N. Patwari and A. O. Hero, "Manifold learning algorithms for localization in wireless sensor networks," in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, May 2004, pp. 857–860.

[40] M. Essoloh, C. Richard, H. Snoussi, and P. Honeine, "Distributed localization in wireless sensor networks as a pre-image problem in a reproducing kernel hilbert space," in *Proc. of the European Conference on Signal Processing (EUSIPCO)*, Lausanne, Switzerland, August 2008.

[41] N. Cristianini, A. Elisseeff, J. Shawe-Taylor, and J. Kandola, "On kernel target alignment," *Proc. of the Neural Information Processing Systems (NIPS)*, pp. 367–373, 2002.

[42] K. Kim, M. Franz, and B. Schölkopf, "Iterative kernel principal component analysis for image modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 9, pp. 1351–1366, 2005.

[43] T. Gneiting, "Compactly supported correlation functions," Environmental Protection Agency, Technical report NRCSE-TRS No. 045, May 2000.

[44] N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'Dea, "Relative location estimation in wireless sensor networks," *IEEE Trans. on Signal Processing*, vol. 51, no. 8, pp. 2137–2148, August 2003.

[45] J. Yang, X. Gao, D. Zhang, and J.-Y. Yang, "Kernel ICA: An alternative formulation and its application to face recognition," *Pattern Recognition*, vol. 38, no. 10, pp. 1784–1787, October 2005.

[46] L. Ralaivola and F. D'Alché-Buc, "Time series filtering, smoothing and learning using the kernel Kalman filter," in *Proc. International Joint Conference on Neural Networks*, vol. 3, 2005, pp. 1449–1454.

[47] S. Sonnenburg, A. Zien, P. Philips, and G. Ratsch, "Poims: positional oligomer importance matrices–understanding support vector machine-based signal detectors," *Bioinformatics*, vol. 24, no. 13, pp. i6–14, July 2008.

[48] C. Cortes, M. Mohri, and J. Weston, "A general regression technique for learning transductions," in *Proc. of the 22nd international conference on machine learning (ICML)*. New York, NY, USA: ACM, 2005, pp. 153–160.