Pattern Recognition ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Non-negativity constraints on the pre-image for pattern recognition with kernel machines[☆]

Maya Kallas [a,c], Paul Honeine [a,*], Cédric Richard [b], Clovis Francis [c], Hassan Amoud [d]

[a] Institut Charles Delaunay (CNRS), LM2S, Université de technologie de Troyes, Troyes, France
[b] Laboratoire Lagrange (CNRS, OCA), Université de Nice Sophia-Antipolis, Nice, France
[c] Laboratoire d'analyse des systèmes (LASYS), Faculté de Génie 1, Université Libanaise, Lebanon
[d] Azm Center for Research in Biotechnology and its Applications, Doctoral School, Lebanese University, Lebanon

ARTICLE INFO

Article history:
Received 4 April 2012
Received in revised form
8 March 2013
Accepted 29 March 2013

Keywords:
Kernel machines
Machine learning
SVM
Kernel PCA
Pre-image problem
Non-negativity constraints
Nonlinear denoising
Pattern recognition

ABSTRACT

Rules of physics in many real-life problems force some constraints to be satisfied. This paper deals with nonlinear pattern recognition under non-negativity constraints. While kernel principal component analysis can be applied for feature extraction or data denoising, in a feature space associated to the considered kernel function, a pre-image technique is required to go back to the input space, e.g., representing a feature in the space of input signals. The main purpose of this paper is to study a constrained pre-image problem with non-negativity constraints. We provide new theoretical results on the pre-image problem, including the weighted combination form of the pre-image, and demonstrate sufficient conditions for the convexity of the problem. The constrained problem is considered with the non-negativity, either on the pre-image itself or on the weights. We propose a simple iterative scheme to incorporate both constraints. A fortuitous side-effect of our method is the sparsity in the representation, a property investigated in this paper. Experimental results are conducted on artificial and real datasets, where many properties are investigated including the sparsity property, and compared to other methods from the literature. The relevance of the proposed method is demonstrated with experimentations on artificial data and on two types of real datasets in signal and image processing.

1. Introduction

Many applications in real-life engineering problems require a constrained solution, including pattern recognition problems. For instance, denoising or deblurring a gray-level image should result into an image of the same type [1]. In unmixing signals or images, e.g., deconvolution, as well as in estimating some spectral features, one may require the non-negativity of the extracted features [2]. This paper deals with a constrained nonlinear pattern recognition problem. Here, pattern recognition includes applications such as feature extraction and data denoising, where the well-known (kernel) principal component analysis (kernel PCA) is considered [3]. One may also consider other applications, such as dimensionality reduction or manifold learning. Nevertheless, it turns out that these applications can be regarded as either feature extraction or data denoising. Therefore, for clarity of presentation, we will only distinguish the latter cases.

It turns out that the non-negative constraint is very essential in many optimization problems [4]. This incorporates the mathematical equivalence between non-negative constrained optimization problems and non-positive ones. Only iterative methods can be used to solve general constrained optimization problems. Moreover, an iterative scheme for non-negativity can serve as the building block for more complex constrained optimization problems, such as the box-constrained optimization. Since the 1980s, this was studied for signal deconvolution by Thomas [5] and Prost et al. [6]. In the beginning of the 1990s, image deconvolution and deblurring were studied respectively by Thomas et al. [7] and Snyder et al. [8]. In the last decade, a general method for iterative optimization under non-negativity constraints has been investigated, initiated by Lantéri et al. [9], and more recently for online learning [10], system identification [11] and distributed regression [12]. Recently, such non-negativity has been introduced in [13] for feature extraction of Event-Related Potential signals, and in [14,15] to denoise images.

Most investigations in constrained solutions for pattern recognition have been geared towards linear algorithms, such as the PCA in [16–19]. In the last couple of decades or so, kernel machines

* Corresponding author. Tel.: +33 3 25 71 56 25; fax: +33 3 25 71 56 49.
E-mail addresses: maya.kallas@utt.fr (M. Kallas), paul.honeine@utt.fr (P. Honeine), cedric.richard@unice.fr (C. Richard), cfrancis@ul.edu.lb (C. Francis), hassan.amoud@gmail.com (H. Amoud).

☆This work was partly supported by the Lebanese University and the French-Lebanese Research Program CEDRE no. 10 SCI F15/L5.

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.03.021

have been increasingly used to solve nonlinear learning problems, popularized since Vapnik's Support Vector Machines (SVM) [20]. While applied successfully to solve nonlinear classification, regression, and detection problems, it was not the case regarding pattern recognition. This is essentially due to the concept of the *kernel trick*, a “double-edged sword”. In fact, the kernel trick provides a way to implicitly map data into some high-dimensional nonlinear feature space, which allows us to construct nonlinear decision rules with essentially the same computational cost as linear ones. Nevertheless, one does not have access to most elements of the feature space, e.g., features or denoised elements computed using kernel PCA [21]. This is related to the fact that the implicit map derived by the kernel is non-surjective, with most elements of the feature space that do not have exact pre-images, and thus cannot be exactly represented in the input space.

The *pre-image problem* consists of mapping the pattern back from the feature space to the input space. Although the exact pre-image may seldom exists, an approximate solution is constructed. To this end, many methods have been presented in the literature, starting with a fixed-point iterative algorithm proposed by Mika et al. [22]. However, this technique was shown to be unstable, and suffers from local minima. In [23], Kwok and Tsang presented a pre-image technique based on a relationship between distances in both input and feature space, using multi-dimensional scaling. More recently, a regularized pre-image estimation with kernel PCA has been introduced in [24]. Honeine et al. [25,26] proposed a more direct method using relationship between inner products. See [27] for a recent review of the pre-image estimation problem. However, none of the aforementioned methods provide constrained solutions.

This paper deals with two types of non-negativity constraints, by providing a unified framework. On the one hand, the non-negativity is applied to the pre-image, and on the other hand, it is considered regarding the weights in the model. In fact, the pre-image can be written as a weighted combination of the training data and thus the weights can be estimated under some constraints. A first attempt to constrain the weights is given in [28] where a penalized problem is considered with a Laplacian penalty, yielding a computationally expensive problem. In a general setting, the linear combination includes both positive and negative weights. Therefore, such weights represent contributions, without any restrictions on the signs. However, many applications cannot be interpreted by including subtracted parts within the model. This is motivated by the rules of physics, with models involving purely additive components, as illustrated for instance in [16,17] for the PCA.

One of the useful properties of constraining the weights of the model is the sparsity property. In fact, the unconstrained solution can combine additive and subtractive contributions, a large part of them neutralizing others in the linear combination. By setting non-negativity constraints to these weights, it turns out that such a balance will lead to a large number of inactive components, i.e., weights close to zero. This is the property of sparsity, contributing to the widespread of Support Vector Machines algorithms [20] and compressed sensing literature [29]. We emphasize on the fact that this is a fortuitous side-effect of the non-negativity constraints, as opposed to a main sparsity objective function, where one controls the degree of sparsity of the solution. It is worth noting that including explicitly the sparsity constraint, such as minimizing an ℓ_0 or an ℓ_1 cost function, is computationally expensive (see for instance [18] and references therein).

In this paper, we study a constrained solution to the pre-image problem, for nonlinear pattern recognition. To the best of our knowledge, pre-image techniques have only been applied for denoising purpose. We propose a unified framework to solve the pre-image problem for both feature extraction and denoising. We provide new theoretical results on the pre-image problem, including the weighted combination form, and provide sufficient conditions for the convexity of the problem. The constrained problem is considered with the non-negativity, either on the pre-image or on the weights. We propose a simple iterative scheme to address both constraints, with expressions for a wide range of kernel functions. Experiment results are conducted on artificial and real datasets, where many properties are investigated including the sparsity property, and compared to other methods from the literature.

The rest of the paper is organized as follows. In the next section, we present the main idea behind kernel machines, and describe the kernel PCA technique where a unified framework for pattern recognition is proposed. Section 3 describes the pre-image problem and provides new theoretical results. In Section 4, we solve the pre-image problem under non-negativity constraints, either on the pre-image or on the weights. Finally, Section 5 gives the experimental results illustrating the efficiency of the proposed method on both artificial and real datasets.

2. Kernel machines and kernel PCA for pattern recognition

In recent years, kernel methods have been progressively used more due, on the one hand, to the development of the statistical learning theory, and on the other hand to the computational efficiency of the corresponding algorithms. This is illustrated here with the kernel PCA, the nonlinear version of the principal component analysis.

2.1. Kernel machines

Let $\mathcal{X} \subset \mathbb{R}^d$ be an input space with the canonical (Euclidean) dot product $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ for any $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. Let $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ be a symmetric and continuous function, i.e., a kernel. A kernel is positive definite if and only if any matrix $\boldsymbol{K}$ with entries $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for any finite subset of $\mathcal{X}$ is positive definite, that is $\sum_{i,j} \alpha_i \alpha_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$ for all $\alpha_i, \alpha_j \in \mathbb{R}$ and all $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$. Based on the Moore–Aronszajn theorem [30], any positive definite kernel guarantees the existence of a unique[1] feature space (or reproducing kernel Hilbert space) $\mathcal{H}$ where κ defines an inner product. In other words, there exists a map $\Phi : \mathcal{X} \mapsto \mathcal{H}$, from the input space to the feature space, such that

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

for any $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the corresponding inner product in $\mathcal{H}$.

Therefore, the positive definite kernel, henceforth called (reproducing) kernel, corresponds to a generalization of the canonical dot product, and thus is a nonlinear measure of similarity between data. It turns out that most linear data processing algorithms can be easily recast in terms of dot product in input space. Substituting the dot product with a kernel offers nonlinear extensions of classical algorithms. This is referred to as the *kernel trick*, and can be done without the need to explicitly compute the map Φ. Table 1 summarizes the most commonly used kernel functions, grouped into two classes: projective kernels, of the form

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = f(\boldsymbol{x}_i \cdot \boldsymbol{x}_j), \tag{2}$$

and radial kernels, of the form

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = g(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2). \tag{3}$$

It is worth noting that some kernels induce infinite-dimensional feature spaces, such as the Gaussian kernel.

[1] Unique, up to an isometry.

Table 1
Commonly used reproducing kernels in machine learning, with parameters $c, \sigma > 0$, and $p \in \mathbb{N}_+$.

Type	General form
Projective	
Monomial	$\kappa_m(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)^p$
Polynomial	$\kappa_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = (c + \boldsymbol{x}_i \cdot \boldsymbol{x}_j)^p$
Exponential	$\kappa_E(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{1}{\sigma}(\boldsymbol{x}_i \cdot \boldsymbol{x}_j)\right)$
Sigmoid	$\kappa_S(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(c(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) + \sigma)$
Radial	
Laplacian	$\kappa_L(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{-1}{\sigma}\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert\right)$
Gaussian	$\kappa_G(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{-1}{2\sigma^2}\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert^2\right)$
Multiquadratic	$\kappa_{MQ}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert^2 + c}$
Rational	$\kappa_R(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - \frac{\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert^2}{\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert^2 + \sigma}$

The following two propositions will be considered in this paper to demonstrate new results, and are included here for completeness. Let $f^{(k)}(\zeta)$ be the k-th derivative of the function f with respect to ζ. The following result is due to [31] (see also [32, Proposition 7.2]).

Proposition 1 (*Radial kernels*). *A sufficient condition for a function of the form* $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = g(\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert^2)$ *to be a positive definite kernel is its complete monotonicity, i.e., its derivatives satisfies*

$$(-1)^k g^{(k)}(\zeta) \geq 0 \tag{4}$$

for any $\zeta > 0$ *and* $k \geq 0$.

This is the case of the Gaussian kernel $\kappa_G(\boldsymbol{x}_i, \boldsymbol{x}_j) = g(\lVert\boldsymbol{x}_i - \boldsymbol{x}_j\rVert^2)$ with

$$g^{(k)}(\zeta) = \left(-\frac{1}{2\sigma^2}\right)^k g(\zeta). \tag{5}$$

For the projective kernels, the following result is given in [32, Proposition 7.1].

Proposition 2 (*Projective kernels*). *Three necessary conditions for a function* $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = f(\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$ *to be a positive definite kernel are, for any non-negative* ζ:

$$f(\zeta) \geq 0$$
$$f^{(1)}(\zeta) \geq 0$$
$$f^{(1)}(\zeta) + \zeta f^{(2)}(\zeta) \geq 0 \tag{6}$$

2.2. Kernel PCA

The principal component analysis (PCA) is a powerful mathematical tool to reveal patterns within a set of data. It is a non-parametric approach, which does not incorporate any prior knowledge of the model, except its linearity. The PCA considers the most relevant eigenvectors of the data covariance matrix, i.e., eigenvectors associated to the largest eigenvalues. These eigenvectors constitute a set of orthonormal axes capturing most of the variance within data. Let us consider a set of n (column-wise) data $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathcal{X}\}$. Then, consider the eigen-problem $\lambda_k \boldsymbol{v}_k = \boldsymbol{C}\boldsymbol{v}_k$, where $\boldsymbol{C} = (1/n)\sum_{j=1}^n \boldsymbol{x}_j \boldsymbol{x}_j^T$ is the covariance matrix, data assumed to be centered around the origin. Then the m eigenvectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m \in \mathcal{X}\}$ are chosen from the largest eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_m$, where each λ_k gives the amount of captured variance in the direction of $\boldsymbol{v}_k$. Due to the linearity of the operations, each eigenvector lies in the span of the data.

The conventional PCA identifies only linear structures in a given dataset. A more generalized technique has been introduced to learn the nonlinearities using kernels, the so-called kernel PCA. The kernel PCA can reveal nonlinear kernel principal components that are more appropriate to complex and nonlinear data such as face images, handwritten digits and natural signals. For this purpose, data are (implicitly) mapped into a feature space, where PCA is applied. Although the resulting eigenvectors are linear in the feature space, they describe nonlinear relations in the input space. In order to solve this nonlinear problem, it is more likely to apply the kernel trick, and not to explicitly compute the map from the input to the feature space. The concept of kernel trick is illustrated here for kernel PCA [33,34].

To this end, the PCA algorithm is recast in terms of inner product of data in feature space. Let $\Phi : \mathcal{X} \mapsto \mathcal{H}$ be a nonlinear map, and $\{\Phi(\boldsymbol{x}_1), \Phi(\boldsymbol{x}_2), \ldots, \Phi(\boldsymbol{x}_n) \in \mathcal{H}\}$ the set of mapped data. We wish to solve the (kernel) PCA, in terms of inner products in the feature space, $\langle\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)\rangle_{\mathcal{H}}$, for $i, j = 1, 2, \ldots, n$. The covariance matrix[2] in $\mathcal{H}$ is $\boldsymbol{C}^{\Phi} = (1/n)\sum_{j=1}^n \Phi(\boldsymbol{x}_j)\Phi(\boldsymbol{x}_j)^T$. The principal axes, $\varphi_k \in \mathcal{H}$ for $k = 1, 2, \ldots, m$, correspond to the eigenvectors with the largest eigenvalues λ_k satisfying the expression

$$\lambda_k \varphi_k = \boldsymbol{C}^{\Phi} \varphi_k. \tag{7}$$

By analogy with the classical PCA, any solution φ_k lies in the span of the Φ-images of the data. This implies that there exist some coefficients $\alpha_1, \alpha_2, \ldots, \alpha_n$ such that

$$\varphi_k = \sum_{i=1}^n \alpha_{k,i} \Phi(\boldsymbol{x}_i). \tag{8}$$

This is a more general result known as the representer theorem [35,36] in kernel machines, where the solution of a (regularized) learning problem can be written in terms of a linear combination of the training data in the feature space.

Replacing the expression of $\boldsymbol{C}^{\Phi}$ and the representer (8) into the eigen-problem (7), we get the new eigen-problem in terms of inner product with

$$n\lambda_k \boldsymbol{\alpha}_k = \boldsymbol{K}\boldsymbol{\alpha}_k, \tag{9}$$

where $\boldsymbol{K}$ is the $n \times n$ matrix of entries $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ with (1) applied, and $\boldsymbol{\alpha}_k = [\alpha_{k,1}\alpha_{k,2}\ldots\alpha_{k,n}]^T$. Furthermore, two issues are considered in the final kernel PCA algorithm. First, as mentioned earlier, data should be centered in the feature space. This can be done by substituting the matrix $\boldsymbol{K}$ by $(\mathbf{I} - \mathbf{1}_n)\boldsymbol{K}(\mathbf{I} - \mathbf{1}_n)$, where $\mathbf{I}$ is the identity matrix and $\mathbf{1}_n$ is a $n \times n$ matrix of entries $1/n$. Second, we normalize as in PCA by requiring that the corresponding vectors in $\mathcal{H}$ be unit-norm, i.e., $\langle\varphi_k, \varphi_k\rangle_{\mathcal{H}} = 1$. This is done by rescaling the weight vectors $\boldsymbol{\alpha}_k$ such that $\lambda_k(\boldsymbol{\alpha}_k \cdot \boldsymbol{\alpha}_k) = 1$, for $k = 1, 2, \ldots, m$.

2.3. Kernel PCA for pattern recognition

Roughly speaking, two main applications can be given with conventional PCA: Either consider relevant principal axes as extracted features, or project some noisy observations onto (the subspace spanned by) these axes as a denoising scheme. Both techniques are illustrated here in the feature space, using kernel PCA.

2.3.1. Feature extraction

Kernel-PCA defines the set of most relevant axes in the feature space. Let $\{\varphi_1, \varphi_2, \ldots, \varphi_m \in \mathcal{H}\}$ be the set of these axes. Then each φ_k

[2] We assume that data are centered in $\mathcal{H}$; otherwise, we apply the algorithm by substituting each $\Phi(\boldsymbol{x}_j)$ with $\Phi(\boldsymbol{x}_j) - (1/n)\sum_{i=1}^n \Phi(\boldsymbol{x}_i)$.

takes the form (8), namely

$$\varphi_k = \sum_{j=1}^{n} \alpha_{k,j} \Phi(\boldsymbol{x}_j),$$

where $\alpha_{k,1}, \alpha_{k,2}, \ldots, \alpha_{k,n}$ are obtained from the eigenvector associated to the k-th eigenvalue in (9). We also define the relevant subspace of $\mathcal{H}$ as the one spanned by these axes. By analogy to the conventional PCA, these axes (as well as the associated subspace) capture most of the variance of the data. They can be regarded as features extracted from the data, capturing the largest variations and orthonormal to each others.

2.3.2. *Denoising*

Denoising is a technique applied in order to recognize patterns corrupted by noise. Let $\boldsymbol{x}_0 \in \mathcal{X}$ be a noisy sample. Then the associated image $\Phi(\boldsymbol{x}_0)$ is projected onto the relevant subspace (described above), resulting into the denoised pattern. The latter is expressed by the inner product of the mapped sample with the m principal axes, as

$$\varphi = \sum_{k=1}^{m} \langle \Phi(\boldsymbol{x}_0), \varphi_k \rangle_{\mathcal{H}} \varphi_k.$$

Expanding this expression by (8) and applying the equivalence between the inner product operator and the kernel function κ, we get

$$\begin{aligned}\varphi &= \sum_{k=1}^{m} \langle \Phi(\boldsymbol{x}_0), \sum_{i=1}^{n} \alpha_{k,i} \Phi(\boldsymbol{x}_i) \rangle_{\mathcal{H}} \sum_{j=1}^{n} \alpha_{k,j} \Phi(\boldsymbol{x}_j) \\ &= \sum_{k=1}^{m} \sum_{i=1}^{n} \alpha_{k,i} \kappa(\boldsymbol{x}_0, \boldsymbol{x}_i) \sum_{j=1}^{n} \alpha_{k,j} \Phi(\boldsymbol{x}_j).\end{aligned}$$

2.3.3. *A unifying view*

Now, we propose a unified view to tackle both the above pattern recognition problems. To this end, we write the extracted feature and the denoised pattern as a linear combination of the mapped training data, with $\varphi_k = \sum_{j=1}^{n} \alpha_{k,j} \Phi(\boldsymbol{x}_j)$ and $\varphi = \sum_{j=1}^{n} [\sum_{k=1}^{m} \sum_{i=1}^{n} \alpha_{k,i} \alpha_{k,j} \kappa(\boldsymbol{x}_0, \boldsymbol{x}_i)] \Phi(\boldsymbol{x}_j)$. Aggregating all these terms, we get a unifying view of both cases, with

$$\varphi = \sum_{j=1}^{n} \gamma_j \Phi(\boldsymbol{x}_j). \tag{10}$$

On the one hand, the feature extraction is given as $\varphi = \varphi_k$ with

$$\gamma_j = \alpha_{k,j},$$

and on the other hand, the denoising pattern with

$$\gamma_j = \sum_{k=1}^{m} \sum_{i=1}^{n} \alpha_{k,i} \alpha_{k,j} \kappa(\boldsymbol{x}_0, \boldsymbol{x}_i).$$

In the latter case, the coefficients γ_j depend on the noisy $\boldsymbol{x}_0$, which can be either a new observation or one of the training data. Summarized in Table 2, the unifying expression in (10) enables us to define a general form in the optimization problem for both feature extraction and denoising schemes.

Table 2
Unified view for the definition of γ_j in $\varphi = \sum_{j=1}^{n} \gamma_j \Phi(\boldsymbol{x}_j)$.

Application	γ_j
Feature extraction of φ_k	$\alpha_{k,j}$
Denoising of $\boldsymbol{x}_0$	$\sum_{k=1}^{m} \sum_{i=1}^{n} \alpha_{k,i} \alpha_{k,j} \kappa(\boldsymbol{x}_0, \boldsymbol{x}_i)$

3. The pre-image problem

Classically, the kernel PCA has shown its powerful ability in supervised learning, as a pre-processing stage followed by a discrimination rule. In these cases, for any given $\boldsymbol{x}_0$, the projection of $\Phi(\boldsymbol{x}_0)$ onto any $\varphi \in \mathcal{H}$ of the form (10), can be defined by $\langle \varphi, \Phi(\boldsymbol{x}_0) \rangle_{\mathcal{H}} = \sum_{j=1}^{n} \gamma_j \kappa(\boldsymbol{x}_j, \boldsymbol{x}_0)$, and comparing it to a threshold gives the decision rule. The problem can be easily solved, with the coefficients computed using the kernel trick. However, in pattern recognition such as feature extraction and denoising, we are interested in the feature itself. More likely, we seek its counterpart in the input space, the observation space. It is natural to have extracted patterns of the same type as the data, i.e., identical input space, since one often seeks for a signal as a pattern in signal processing, or a denoised image in image processing. The pre-image problem is illustrated in Fig. 1.

With the exception of the Φ-images of the training data, only a very few elements in the feature space have *pre-images*, i.e., data which maps into (10) for some given coefficients. In fact, this is an ill-posed problem since the exact pre-image may not exist, and even if it exists, it might be not unique. To solve this problem, we seek an approximate solution, i.e., a $\boldsymbol{x}^* \in \mathcal{X}$ whose image $\Phi(\boldsymbol{x}^*)$ is as close as possible to φ. The way back from the feature space to the input space is called *the pre-image problem*.

Many techniques have been introduced in the literature in order to solve this problem. The simplest one is the study of the gradient of the cost function defined by (12). This technique seeks for the minimum based on the opposite side of the gradient. An iterative fixed-point method proposed by Mika et al. in [22] is derived taking into consideration the kernel function used for denoising purpose. The implementation of this technique requires the choice of the starting point for the calculation, and a stopping criterion. The results vary widely depending on the starting point. Moreover, the fixed-point technique can be unstable and leads to local minima, and sometimes may not converge. Issues related to the iterative fixed-point technique are probably due to the absence of an adaptation step size parameter to enable control of the convergence of the algorithm.

In order to overcome these issues, many techniques have been introduced. First, Kwok et al. in [23] introduced a new technique based on a relationship between distances in both input and feature spaces, using multi-dimensional scaling. Ideally, the distances in the input space and the feature space are preserved. Few neighbors are taken into consideration in order to evaluate the solution in the same spirit as the locally linear embedding scheme in dimensionality reduction [37]. Another technique based on preserving the inner product measures is studied by Honeine et al. in [25]. This strategy consists of computing the solution by preserving the angular measures. Therefore, a coordinate system is created in the feature space having an isometry with the input space. This technique can identify the pre-images of a set of functions in the feature space.

Other techniques have been proposed in order to obtain a more stable estimation of the pre-image as in [24]. In this method the cost function is regularized by adding the distance in the input space multiplied by some non-negative regularization parameter. In the same spirit, another method has been introduced in [28] with two penalization functions. First of all, a convexity constraint is imposed during the training phase. After that, a penalized function is integrated as part of the optimization function during the learning process of the pre-images. In this paper, we propose another technique in order to solve the pre-image problem.

It consists of solving the optimization problem

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathcal{X}} \frac{1}{2} \|\varphi - \Phi(\boldsymbol{x})\|_{\mathcal{H}}^2, \tag{11}$$

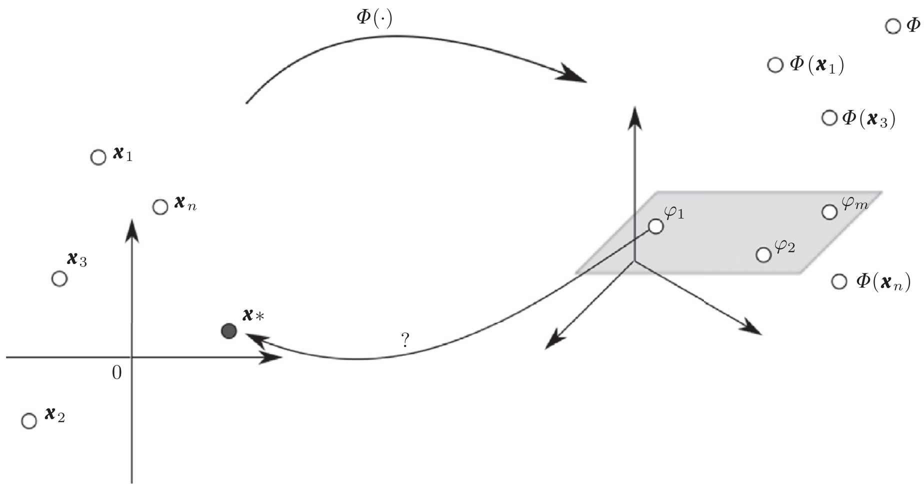


Fig. 1. Schematic illustration of the pre-image problem. Constructed in the feature space from some training data, principal axes are mapped back to the input space by solving the pre-image problem, pre-imaging φ_1 into $\boldsymbol{x}^*$ here. Not every feature has a unique pre-image that is why the mapping back is not given by the inverse of the function $\Phi(\cdot)$, and there is no explicit function to map back the evaluation in the feature space to input space, therefore it is represented by the question mark "?".

where $\|\cdot\|_{\mathcal{H}}$ denotes the norm in $\mathcal{H}$, and thus provides a measure of distance between elements in the feature space, with the norm of their residue. Thanks to the unifying view given in (10) with $\varphi=\sum_{i=1}^n \gamma_i \Phi(\boldsymbol{x}_i)$, we consider the same optimization problem for either feature extraction or denoising, with

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{X}} \frac{1}{2}\| \sum_{i=1}^n \gamma_i \Phi(\boldsymbol{x}_i) - \Phi(\boldsymbol{x})\|_{\mathcal{H}}^2$$

The general form used for the calculation is described by

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} J(\boldsymbol{x})$$

where $J(\boldsymbol{x})$ represents the resulting cost function, defined by

$$J(\boldsymbol{x}) = -\sum_{i=1}^n \gamma_i \kappa(\boldsymbol{x}_i,\boldsymbol{x}) + \frac{1}{2}\kappa(\boldsymbol{x},\boldsymbol{x}) \tag{12}$$

with γ_j given in Table 2. In this expression, the term $\frac{1}{2}\sum_{i=1}^n\sum_{j=1}^n \gamma_i\gamma_j\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)$ has been removed since it is independent of $\boldsymbol{x}$.

This is a highly nonlinear optimization problem. To solve this problem, one may study the gradient of the cost function $J(\boldsymbol{x})$ with respect to $\boldsymbol{x}$. At an optimum, the gradient with respect to $\boldsymbol{x}$ disappears, namely $\nabla_{\boldsymbol{x}} J(\boldsymbol{x})=0$. The resulting gradient is given as

$$\nabla_{\boldsymbol{x}} J(\boldsymbol{x}) = -\sum_{i=1}^n \gamma_i \nabla_{\boldsymbol{x}}\kappa(\boldsymbol{x}_i,\boldsymbol{x}) + \frac{1}{2}\nabla_{\boldsymbol{x}}\kappa(\boldsymbol{x},\boldsymbol{x}). \tag{13}$$

This is the general form for all kernels, including for instance the projective kernels of the form (2) such as the polynomial kernel. Expressions (12) and (13) can be further simplified for the wide class of radial kernels, of the form (3) such as the Gaussian kernel. In such cases, $\kappa(\boldsymbol{x},\boldsymbol{x})$ is independent of $\boldsymbol{x}$, for all $\boldsymbol{x}\in\mathcal{X}$, thus $\nabla_{\boldsymbol{x}}\kappa(\boldsymbol{x},\boldsymbol{x})$ equals to zero, and only the first term of (13) remains. See Table 1 for expressions of commonly used kernels.

Let us now write the pre-image using a linear combination of the available data, that is $\boldsymbol{x}^*=\sum_{i=1}^n \beta_i^* \boldsymbol{x}_i$. To the best of our knowledge, this is the first time that a proof of this statement is derived, while it has been exploited and validated by many pre-image techniques from the literature [23,25,28].

Theorem 1. *Any pre-image $\boldsymbol{x}^*$ can be written as a linear combination of the available data, namely*

$$\boldsymbol{x}^* = \sum_{i=1}^n \beta_i^* \boldsymbol{x}_i \tag{14}$$

where β_i^* are weights to be determined.

Proof. To prove this, we consider separately the two classes of kernels: projective and radial kernels (see Table 1). We consider the expression of the gradient of a composition of two functions. Let h_2 be a function of real values in $\mathbb{R}$ and h_1 a function of real values over $\mathcal{X}$ (we consider in particular either $h_1(\boldsymbol{x})=\langle \boldsymbol{x}_i\cdot\boldsymbol{x}\rangle$ or $h_1(\boldsymbol{x})=\|\boldsymbol{x}_i-\boldsymbol{x}\|^2$). Under mild conditions of differentiability of the function h_2 with respect to $h_1(x)$, we have

$$\nabla_{\boldsymbol{x}}(h_2\circ h_1)\boldsymbol{x} = h_2^{(1)}(h_1(\boldsymbol{x}))\nabla_{\boldsymbol{x}} h_1(\boldsymbol{x}),$$

where $h_2^{(1)}(\zeta)$ is the first derivative of the function h_2 over ζ, namely

$$h_2^{(1)}(\zeta) = \frac{\partial h_2(\zeta)}{\partial\zeta}.$$

Let us begin with the projective kernels, of the form (2). In this case

$$\nabla_{\boldsymbol{x}} f(\langle \boldsymbol{x}_i\cdot\boldsymbol{x}\rangle) = f^{(1)}(\boldsymbol{x}_i\cdot\boldsymbol{x})\nabla_{\boldsymbol{x}}(\boldsymbol{x}_i\cdot\boldsymbol{x})$$

where the gradient $\nabla_{\boldsymbol{x}}(\boldsymbol{x}_i\cdot\boldsymbol{x})$ is given by the vector whose k-th component is

$$\frac{\partial\langle \boldsymbol{x}_i\cdot\boldsymbol{x}\rangle}{\partial[\boldsymbol{x}]_k} = \sum_{j=1}^{\mathtt{dim}(\mathcal{X})} [\boldsymbol{x}_i]_j \frac{[\boldsymbol{x}]_j}{\partial[\boldsymbol{x}]_k} = \sum_{j=1}^{\mathtt{dim}(\mathcal{X})} [\boldsymbol{x}_i]_j \delta_{jk} = [\boldsymbol{x}_i]_k,$$

where δ_{jk} is the Kronecker symbol. In other words, $\nabla_{\boldsymbol{x}}(\boldsymbol{x}_i\cdot\boldsymbol{x})=\boldsymbol{x}_i$, and therefore

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_i\cdot\boldsymbol{x}) = f^{(1)}(\boldsymbol{x}_i\cdot\boldsymbol{x})\boldsymbol{x}_i.$$

On the other side, it is easy to demonstrate that

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}\cdot\boldsymbol{x}) = 2f^{(1)}(\boldsymbol{x}\cdot\boldsymbol{x})\boldsymbol{x},$$

since

$$\frac{\partial(\boldsymbol{x}\cdot\boldsymbol{x})}{\partial[\boldsymbol{x}]_k} = \frac{\partial\|\boldsymbol{x}\|^2}{\partial[\boldsymbol{x}]_k} = 2[\boldsymbol{x}]_k.$$

We have at the optimum $\nabla_{\boldsymbol{x}} J(\boldsymbol{x}^*)=0$, namely

$$\sum_{i=1}^n \gamma_i \nabla_{\boldsymbol{x}^*}\kappa(\boldsymbol{x}_i,\boldsymbol{x}^*) = \frac{1}{2}\nabla_{\boldsymbol{x}^*}\kappa(\boldsymbol{x}^*,\boldsymbol{x}^*). \tag{15}$$

Thus, the left-hand-side of this equation can be written as

$$\sum_{i=1}^n \gamma_i \nabla_{\boldsymbol{x}^*}\kappa(\boldsymbol{x}_i,\boldsymbol{x}^*) = \sum_{i=1}^n \gamma_i \nabla_{\boldsymbol{x}^*} f(\boldsymbol{x}_i\cdot\boldsymbol{x}^*)\boldsymbol{x}_i,$$

and its right-hand-side can be expressed as

$$\frac{1}{2}\nabla_{\boldsymbol{x}^*}\kappa(\boldsymbol{x}^*,\boldsymbol{x}^*)=\tfrac{1}{2}\nabla_{\boldsymbol{x}^*}f(\boldsymbol{x}^*\cdot\boldsymbol{x}^*)2\boldsymbol{x}^*.$$

The gradient (13) is thus given by

$$\nabla_{\boldsymbol{x}}J(\boldsymbol{x})=-\sum_{i=1}^{n}\gamma_i f^{(1)}(\boldsymbol{x}_i\cdot\boldsymbol{x})\boldsymbol{x}_i+f^{(1)}(\boldsymbol{x}\cdot\boldsymbol{x})\boldsymbol{x}.$$

Combining both expressions, Eq. (15) becomes

$$\boldsymbol{x}^*=\sum_{i=1}^{n}\gamma_i\frac{f^{(1)}(\boldsymbol{x}_i\cdot\boldsymbol{x}^*)}{f^{(1)}(\boldsymbol{x}^*\cdot\boldsymbol{x}^*)}\boldsymbol{x}_i, \qquad (16)$$

of the form $\boldsymbol{x}^*=\sum_{i=1}^{n}\beta_i^*\boldsymbol{x}_i$.

We now study the class of radial kernels, defined by expression (3). In such a case, the term $\partial\kappa(\boldsymbol{x},\boldsymbol{x})/\partial\boldsymbol{x}$ vanishes. Thus, we get

$$\nabla_{\boldsymbol{x}}g(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)=g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)\nabla_{\boldsymbol{x}}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2).$$

In this case, it is easy to demonstrate that the k-th component of the gradient $\nabla_{\boldsymbol{x}}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)$ is

$$\frac{\partial(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)}{\partial[\boldsymbol{x}]_k}=-2([\boldsymbol{x}_i]_k-[\boldsymbol{x}]_k),$$

thereby, we can write $\nabla_{\boldsymbol{x}}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)=-2(\boldsymbol{x}_i-\boldsymbol{x})$, and consequently

$$\nabla_{\boldsymbol{x}}g(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)=-2g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)(\boldsymbol{x}_i-\boldsymbol{x}).$$

The gradient (13) is thus given by

$$\nabla_{\boldsymbol{x}}J(\boldsymbol{x})=-\sum_{i=1}^{n}\gamma_i g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)(\boldsymbol{x}_i-\boldsymbol{x}).$$

At the optimum (15), the gradient can be written as

$$\sum_{i=1}^{n}\gamma_i\nabla_{\boldsymbol{x}^*}\kappa(\boldsymbol{x}_i,\boldsymbol{x}^*)=0,$$

with the left-hand-side given as

$$\sum_{i=1}^{n}\gamma_i\nabla_{\boldsymbol{x}^*}\kappa(\boldsymbol{x}_i,\boldsymbol{x}^*)=\sum_{i=1}^{n}\gamma_i\nabla_{\boldsymbol{x}^*}g(\|\boldsymbol{x}_i-\boldsymbol{x}^*\|^2)2(\boldsymbol{x}^*-\boldsymbol{x}_i).$$

The final result of (15) can thus be expressed as

$$\boldsymbol{x}^*=\sum_{i=1}^{n}\gamma_i\frac{g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}^*\|^2)}{\sum_{j=1}^{n}\gamma_j g^{(1)}(\|\boldsymbol{x}_j-\boldsymbol{x}^*\|^2)}\boldsymbol{x}_i, \qquad (17)$$

again of the form $\boldsymbol{x}^*=\sum_{i=1}^{n}\beta_i^*\boldsymbol{x}_i$. □

The following result provides new insight into the connection between the weights in both feature and input spaces.

Corollary 1. *When input data are non-negative, if the weights in the feature space are non-negative, i.e.,* $\gamma_1,\gamma_2,\ldots,\gamma_n\geq 0$, *then the weights of the corresponding pre-image are also non-negative, i.e.,* $\beta_1^*,\beta_2^*,\ldots,\beta_n^*\geq 0$. *Moreover, the non-negativity of the data is not required for radial kernels.*

Proof. For the projective kernels, we have from (16)

$$\beta_i^*=\gamma_i\frac{f^{(1)}(\boldsymbol{x}_i\cdot\boldsymbol{x}^*)}{f^{(1)}(\boldsymbol{x}^*\cdot\boldsymbol{x}^*)}.$$

When all input data are non-negative, the above derivatives are non-negative due to Proposition 2. The same proof can be applied for the radial kernels by applying Proposition 1 to (17) with

$$\beta_i^*=\gamma_i\frac{g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}^*\|^2)}{\sum_{j=1}^{n}\gamma_j g^{(1)}(\|\boldsymbol{x}_j-\boldsymbol{x}^*\|^2)}. \qquad \square$$

The above results are based on the first derivative of the cost function (12). Its second derivative provides a deeper insight on its convexity, as derived in the following theorem.

Theorem 2. *For the class of radial kernels, a sufficient condition for the convexity of the cost function is given by the non-negativity of the coefficients* $\gamma_1,\gamma_2,\ldots,\gamma_n$.

Proof. Taking the second derivative of the cost function (12) with respect to $\boldsymbol{x}$, we get

$$\nabla^2_{\boldsymbol{x}}J(\boldsymbol{x})=\nabla_{\boldsymbol{x}}\left[2\sum_{i=1}^{n}\gamma_i(\boldsymbol{x}_i-\boldsymbol{x})g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)\right]$$
$$=2\sum_{i=1}^{n}\gamma_i(-g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)+2(\boldsymbol{x}_i-\boldsymbol{x})^2g^{(2)}(\|\boldsymbol{x}_i-\boldsymbol{x}\|^2)).$$

The term between parentheses is positive, due to Proposition 1. Therefore, a sufficient condition for the second derivative to be non-negative, and thus for the convexity of (12), is that all the coefficients γ_i are non-negative. □

The non-negativity of the coefficients γ_i's is a condition imposed by the SVM for classification and regression, as well as some other machine learning methods. However, this is not the case in general, with the kernel PCA for instance. In this paper, we will not limit ourselves to the convex problem, but consider the more general non-convex problem.

From Theorem 1, the form $\boldsymbol{x}^*=\sum_{i=1}^{n}\beta_i^*\boldsymbol{x}_i$ provides a fixed-point iterative method to solve the pre-image problem, where the β_i's depend on $\boldsymbol{x}^*$. For the Gaussian kernel, we have

$$\kappa_G(\boldsymbol{x}_i,\boldsymbol{x}_j)=g(\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2)=\exp\left(\frac{-1}{2\sigma^2}\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2\right)$$

thus

$$g^{(1)}(\|\boldsymbol{x}_i-\boldsymbol{x}_j\|)=-\frac{1}{2\sigma^2}\kappa_G(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

From expression (17), we get the fixed-point iterative method for the Gaussian kernel

$$\boldsymbol{x}^*=\frac{\sum_{i=1}^{n}\gamma_i\kappa_G(\boldsymbol{x}_i,\boldsymbol{x}^*)\boldsymbol{x}_i}{\sum_{i=1}^{n}\gamma_i\kappa_G(\boldsymbol{x}_i,\boldsymbol{x}^*)}. \qquad (18)$$

Table 3
Gradient of the cost function (12) for most commonly used kernels, with respect to either $\boldsymbol{x}$ (second column) or $\boldsymbol{\beta}$ from $\boldsymbol{x}=\boldsymbol{X}^T\boldsymbol{\beta}$ (third column).

Type	$\nabla_{\boldsymbol{x}}J(\boldsymbol{x})$	$\nabla_{\beta}J(\boldsymbol{X}^T\boldsymbol{\beta})$
Polynomial	$-\sum_{i=1}^{n}\gamma_i p\kappa_{p-1}(\boldsymbol{x}_i,\boldsymbol{x})\boldsymbol{x}_i+p\kappa_{p-1}(\boldsymbol{x},\boldsymbol{x})\boldsymbol{x}$	$-\sum_{i=1}^{n}\gamma_i p\kappa_{p-1}(\boldsymbol{x}_i,\boldsymbol{X}^T\boldsymbol{\beta})\boldsymbol{X}\boldsymbol{x}_i+p\kappa_{p-1}(\boldsymbol{X}^T\boldsymbol{\beta},\boldsymbol{X}^T\boldsymbol{\beta})\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{\beta}$
Sigmoid	$-\sum_{i=1}^{n}\gamma_i(1-\kappa_S^2(\boldsymbol{x}_i,\boldsymbol{x}))c\boldsymbol{x}_i+c(1-\kappa_S^2(\boldsymbol{x},\boldsymbol{x}))\boldsymbol{x}$	$-\sum_{i=1}^{n}\gamma_i c(1-\kappa_S^2(\boldsymbol{x}_i,\boldsymbol{X}^T\boldsymbol{\beta}))\boldsymbol{X}\boldsymbol{x}_i+c(1-\kappa_S^2(\boldsymbol{X}^T\boldsymbol{\beta},\boldsymbol{X}^T\boldsymbol{\beta}))\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{\beta}$
Exponential	$-\frac{1}{\sigma}\sum_{i=1}^{n}\gamma_i\kappa_E(\boldsymbol{x}_i,\boldsymbol{x})\boldsymbol{x}_i+\frac{1}{\sigma}\kappa_E(\boldsymbol{x},\boldsymbol{x})\boldsymbol{x}$	$-\frac{1}{\sigma}\sum_{i=1}^{n}\gamma_i\kappa_E(\boldsymbol{x}_i,\boldsymbol{X}^T\boldsymbol{\beta})\boldsymbol{X}\boldsymbol{x}_i+\frac{1}{\sigma}\kappa_E(\boldsymbol{X}^T\boldsymbol{\beta},\boldsymbol{X}^T\boldsymbol{\beta})\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{\beta}$
Gaussian	$-\frac{1}{\sigma^2}\sum_{i=1}^{n}\gamma_i\kappa_G(\boldsymbol{x}_i,\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{x})$	$-\frac{1}{\sigma^2}\sum_{i=1}^{n}\gamma_i\kappa_G(\boldsymbol{x}_i,\boldsymbol{X}^T\boldsymbol{\beta})\boldsymbol{X}(\boldsymbol{x}_i-\boldsymbol{X}^T\boldsymbol{\beta})$

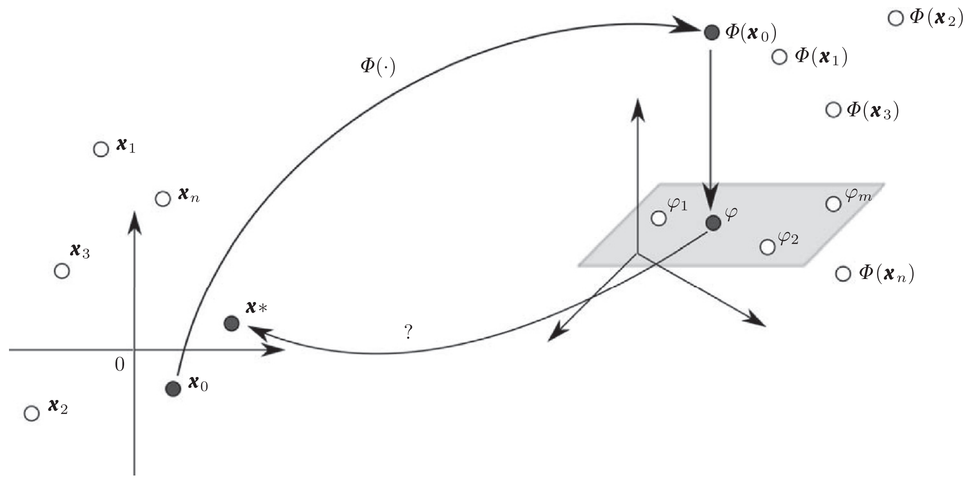


Fig. 2. Schematic illustration of the pre-image problem under the non-negativity constraints. A given noisy data $\boldsymbol{x}_0$ is mapped into $\Phi(\boldsymbol{x}_0)$, then projected into the subspace spanned by the most relevant principal axes $\varphi_1, \varphi_2, \ldots, \varphi_m$. The denoised pattern φ is mapped back to the input space, into $\boldsymbol{x}^*$. Not every feature has a unique pre-image that is why the mapping back is not given by the inverse of the function $\Phi(\cdot)$, and there is no explicit function to map back the evaluation in the feature space to input space, therefore it is represented by the question mark "?".

When the polynomial kernel is applied, with

$$\kappa_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = f(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) = (c + \boldsymbol{x}_i \cdot \boldsymbol{x}_j)^p,$$

then

$$f^{(1)}(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) = p\kappa_{p-1}(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

where $\kappa_{p-1}(\boldsymbol{x}_i, \boldsymbol{x}_j) = f(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) = (c + \boldsymbol{x}_i \cdot \boldsymbol{x}_j)^{p-1}$. From expression (16), we get the fixed-point iterative method for the polynomial kernel

$$\boldsymbol{x}^* = \frac{\sum_{i=1}^n \gamma_i \kappa_{p-1}(\boldsymbol{x}_i, \boldsymbol{x}^*)\boldsymbol{x}_i}{\kappa_{p-1}(\boldsymbol{x}^*, \boldsymbol{x}^*)}.$$

Table 3(second column) illustrates the diversity of the gradient expressions for different kernels. Such fixed-point iterative algorithm suffers from instabilities and even may not converge at all, as illustrated in [22] where only the Gaussian kernel was used. Moreover, results widely vary for different starting points in practice. These issues are likely due to two factors: First, the absence of a stepsize parameter, which allows us to control the convergence of the algorithm. Second, the unconstrained solution, as the hypothesis space corresponds to the whole input space. Both issues will be addressed in the next section.

4. The pre-image under non-negativity constraints

In many applications in pattern recognition, one seeks non-negativity in the solution. In image processing for instance, training data are images or patches within an image, i.e., data which are non-negative for gray-level images. To get a feature extracted or a denoised version of the same type (same input space with non-negativity of each pixel), one should impose non-negativity constraints on the pre-image. However, the constraints are applied either on the data itself, or on the weights model using the linear combination of (14).

4.1. Non-negativity constraint on the data itself

In this section, we consider the general problem of solving the pre-image problem under non-negativity constraint. With the cost function $J(\cdot)$ defined in (12), we study the constrained optimization problem

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} J(\boldsymbol{x}) \quad \text{subject to } \boldsymbol{x} \geq 0, \tag{19}$$

where expression $\boldsymbol{x} \geq 0$ refers to the non-negativity of all entries of the vector $\boldsymbol{x}$. The gradient of $J(\cdot)$ is given in Table 3 for several kernel types. Next, we derive an iterative updating rule that leads to the non-negativity of pre-image. Fig. 2 illustrates the concept of this constrained pre-image.

A general form of the pre-image problem under non-negativity constraints is defined in (19). Consider the Lagrangian function associated to this constrained optimization problem, with[3]

$$J(\boldsymbol{x}) - \boldsymbol{\mu}^T \boldsymbol{x},$$

where $\boldsymbol{\mu}$ represents the vector of non-negative Lagrange multipliers. At the optimum solution $\boldsymbol{x}^*$, corresponding to the optimal multiplier vector $\boldsymbol{\mu}^*$, the first-order (Karush–)Kuhn–Tucker optimality conditions are satisfied, with

$$\nabla_{\boldsymbol{x}}[J(\boldsymbol{x}^*) - \boldsymbol{\mu}^{*T}\boldsymbol{x}^*] = 0$$
$$\mu_i^* x_i^* = 0 \quad \text{for all } i = 1, 2, \ldots$$

where x_i^* (resp. μ_i^*) is the i-th component of $\boldsymbol{x}^*$ (resp. $\boldsymbol{\mu}^*$), and $\nabla_{\boldsymbol{x}}$ is the gradient with respect to $\boldsymbol{x}$. We can easily see that the first condition can be written as $\nabla_{\boldsymbol{x}}[J(\boldsymbol{x}^*)]_i - [\boldsymbol{\mu}^*]_i = 0$ for all i, where $[\cdot]_i$ denotes the i-th component. Combining all these equality conditions by removing the Lagrangian multipliers, we get for each $i = 1, 2, \ldots$, either $x_i^* = 0$ (active constraint) or $[\nabla_{\boldsymbol{x}} J(\boldsymbol{x}^*)]_i = 0$ (inactive constraint with $x_i^* > 0$).

In order to solve this equation, we consider an iterative scheme. The updating expression at iteration $t+1$ of all $x_i(t+1)$ from previous $x_i(t)$ is given by

$$x_i(t+1) = x_i(t) + \eta_i(t) x_i(t)[-\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i,$$

where $\eta_i(t)$ is a stepsize factor to control convergence and impose the non-negativity, and the minus sign illustrates a gradient descent scheme. A condition on $\eta_i(t)$ should be satisfied to insure the non-negativity of all components $x_i(t+1)$ of $\boldsymbol{x}(t+1)$. To this end, we write the above expression as

$$x_i(t+1) = x_i(t)(1 + \eta_i(t)[-\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i),$$

[3] A more general form can be given using a function expressing the constraints, $g(\boldsymbol{x})$, with the Lagrangian expression $J(\boldsymbol{x}) - \boldsymbol{\mu}^T g(\boldsymbol{x})$ [9]. For clarity of this paper, this function is substituted with its simplest form, $\boldsymbol{x}$.

and thus this translates into a condition on the non-negativity of $1+\eta_i(t)[-\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i$. Two cases can be distinguished: If $[\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i \le 0$, no restriction is applied on the value of the stepsize; otherwise, when $[\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i > 0$, then we have to crop the value of the stepsize such that

$$\eta_i(t) \le \frac{1}{[\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i}.$$

In practice, one may use a stepsize independent of i, which satisfies the following inequality:

$$\eta(t) \le \min_i \frac{1}{[\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))]_i}.$$

Written in a matrix form, the final updating rule is defined by

$$\boldsymbol{x}(t+1) = \boldsymbol{x}(t) - \eta(t)\, \mathtt{diag}[\boldsymbol{x}(t)] \nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t)), \tag{20}$$

where $\mathtt{diag}[\cdot]$ is the diagonal operator, namely $\mathtt{diag}[\boldsymbol{x}(t)]$ is the diagonal matrix whose entries are $x_i(t)$. In this expression, $-\mathtt{diag}[\boldsymbol{x}(t)]\nabla_{\boldsymbol{x}} J(\boldsymbol{x}(t))$ corresponds to the direction of descent.

4.2. Non-negativity constraint on the model weights

By virtue of Theorem 1, the pre-image can be expressed in terms of a linear combination of the available data, namely $\boldsymbol{x}^* = \sum_{i=1}^n \beta_i^* \boldsymbol{x}_i$, for some weights β_i^* to be determined. Therefore, we seek the optimal pre-image of the matrix form

$$\boldsymbol{x}^* = \boldsymbol{X}^T \boldsymbol{\beta}^*,$$

where $\boldsymbol{X} = [\boldsymbol{x}_1 \boldsymbol{x}_2 \cdots \boldsymbol{x}_n]^T$ and $\boldsymbol{\beta}^* = [\beta_1^* \beta_2^* \cdots \beta_n^*]^T$ is the vector of unknown coefficients. This allows us to present another strategy to tackle the pre-image problem, by imposing a constraint on the coefficients in the above expression. We define the constrained pre-image problem as

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} J(\boldsymbol{x}) \quad \text{subject to } \boldsymbol{\beta} \ge 0,$$

with $\boldsymbol{x} = \boldsymbol{X}^T \boldsymbol{\beta}$.

The corresponding cost function (12) can be written as

$$J(\boldsymbol{X}^T\boldsymbol{\beta}) = -\sum_{i=1}^n \gamma_i \kappa(\boldsymbol{x}_i, \boldsymbol{X}^T\boldsymbol{\beta}) + \frac{1}{2}\kappa(\boldsymbol{X}^T\boldsymbol{\beta}, \boldsymbol{X}^T\boldsymbol{\beta}). \tag{21}$$

Taking the gradient of the above expression with respect to $\boldsymbol{\beta}$, we get

$$\nabla_{\boldsymbol{\beta}} J(\boldsymbol{X}^T\boldsymbol{\beta}) = \boldsymbol{X} \nabla_{\boldsymbol{x}} J(\boldsymbol{x}), \tag{22}$$

where $\boldsymbol{x} = \boldsymbol{X}^T\boldsymbol{\beta}$. Table 3 (third column) gives the gradient with respect to $\boldsymbol{\beta}$ of the most commonly used kernels. The relationship between these expressions and the gradient with respect to $\boldsymbol{x}$ (second column in Table 3) is given in expression (22).

By deriving this analogy with the constrained optimization problem (19) (non-negativity on the pre-image), we revisit the latter in order to impose the non-negativity on the weights $\boldsymbol{\beta}^*$ in the expansion $\boldsymbol{x}^* = \boldsymbol{X}^T\boldsymbol{\beta}^*$. This yields the following optimization problem:

$$\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} J(\boldsymbol{X}^T\boldsymbol{\beta}) \quad \text{subject to } \boldsymbol{\beta} \ge 0.$$

In this expression $J(\cdot)$ is defined as in (21), with its gradient with respect to $\boldsymbol{\beta}$ given in (22). From (20), the updating rule of these weights is given as

$$\boldsymbol{\beta}(t+1) = \boldsymbol{\beta}(t) - \eta(t)\, \mathtt{diag}[\boldsymbol{\beta}(t)] \boldsymbol{X} \nabla_{\boldsymbol{x}} J(\boldsymbol{x}),$$

where $\boldsymbol{x} = \boldsymbol{X}^T\boldsymbol{\beta}$. The final weights $\beta_1^*, \beta_2^*, \ldots, \beta_n^*$ determine the pre-image with $\boldsymbol{x}^* = \sum_{i=1}^n \beta_i^* \boldsymbol{x}_i$. From this expression, we can see that in the case of non-negative training data, $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \ge 0$, the resulting pre-image $\boldsymbol{x}^*$ will be also non-negative.

By imposing non-negativity of the weights, we get a beneficial side-effect with the sparseness of the solution. This means that a large number of the weights is close to zero, or in other words, only a small number of training data contributes to the final solution. This property is probably due to the non-uniqueness of the unconstrained solution, where redundancy in data may result into additive and subtract components that neutralize their contributions. Sparseness is a very desirable property in pattern recognition and machine learning, contributing to a better understanding of the results, in bioinformatics for instance. It is illustrated in the next section on artificial and real datasets.

5. Experiments

Three applications of the proposed method are investigated in this section: two applications for data denoising and one on feature extraction. In the first application, two-dimensional artificial data are studied, providing an illustration of the behavior of the algorithm, for two cases: restricting the solution to be non-negative, or forcing the weights to be non-negative, and therefore studying the sparsity of the solution. In the second application, real images from the MNIST database are used to illustrate the efficiency of the proposed method with kernel PCA for denoising. In the third application, we study nonlinear feature extraction from real signals. Signals are based on the event-related potentials of the brain activity from the electroencephalograph.

5.1. Artificial datasets: denoising scheme

Let us start with the artificial datasets. For illustration purpose, we consider a two-dimensional space, and apply the denoising scheme separately on three different shapes[4]: a banana, a donut and a frame. For each example, a set of n samples, given in Fig. 3 (upper row), was generated to learn the m eigenvectors. With its quadratic form, we set $m=2$ for the banana dataset, while $m=4$ for the more complicated shape of the frame. Another set of N samples was generated from the same distributions, as given by the (very small) blue dots in Fig. 3. Values of the parameters used for each dataset are given in Table 4. It is obvious that these nonlinear shapes cannot be denoised properly using a linear approach, such as the conventional PCA.

First, we compared the non-negative pre-image approach with other unconstrained techniques, including the fixed-point technique defined by (18) and the regularized pre-image estimation [24]. To this end, we considered a setting where all algorithms should give comparable results: all the samples are non-negative. This was done by translating the samples into the positive quadrant, as illustrated in Fig. 3 (upper row). For all these algorithms, the noisy version of the data was used at initialization, i.e., $\boldsymbol{x}(t)$ for $t=0$, given by (very small) blue dots in Fig. 3. After different experiments, we found that 20 iterations were sufficient to denoise the samples. Therefore, we fixed the maximum number of iterations fixed to $t_{\max} = 20$ for iterative algorithms. The denoised samples obtained by these pre-image techniques are represented by red dots. The trajectories obtained at each iteration are represented by green lines (except for the regularized pre-image estimation which is not an iterative technique). As we can see with the length of these lines, the fixed-point algorithm (second row) has slower convergence as opposed to the proposed approach (last row).

[4] The banana dataset is defined by a parabola having $(x, x^2+\xi)$ as coordinates, where x on the x-axis is uniformly distributed within the interval $[0.5, 2.5]$, and ξ is normally distributed with a standard deviation of $\nu=0.2$. The donut dataset is given by data from a circle of radius 0.9, corrupted by a uniformly distributed noise on $[-\nu, \nu]$, with $\nu=0.4$. The frame dataset is defined by a square of four lines, each of length 2, where the length 2 is chosen randomly. The data were uniformly randomly drawn within these lines and corrupted by a uniformly distributed noise on $[-\nu, \nu]$, with $\nu=0.2$.

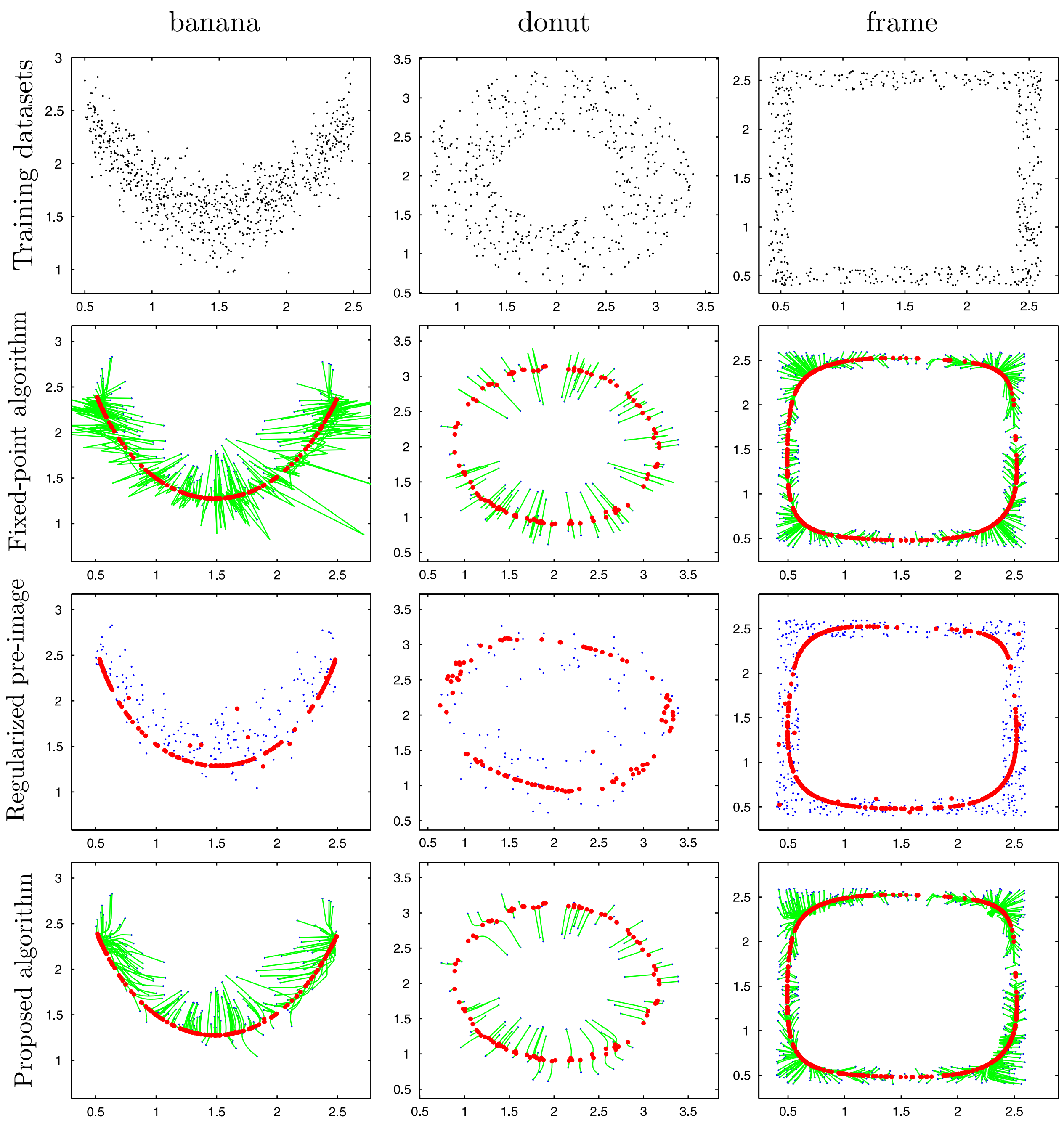


Fig. 3. Denoising artificial datasets, for the three shapes: banana (left column), donut (middle column) and frame (right column). A set of training data (▽ in the upper row) is used for constructing the relevant subspace using the kernel PCA with the Gaussian kernel. Another set of data (designated by •) is denoised (into ●) using either the fixed-point (second row), the regularized pre-image estimation method (third row) and the proposed (lower row) algorithms. The evolution of the solution for the iterative methods for the 20 iterations is given with the paths (shown with —). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 4
Values of the parameters for the three datasets.

Parameters		Datasets		
		Banana	Donut	Frame
Noise parameter	ν	0.2	0.4	0.2
Number of training data	n	800	500	550
Number of eigenvectors	m	2	4	4
Bandwidth of the Gaussian kernel	σ	0.7	0.8	0.5
Number of denoised data	N	200	100	510
Value of the stepsize parameter	η	0.3	0.3	0.3
Number of iterations	$t_{\max}$	20	20	20

This is mainly due to the use of the stepsize η, set here to a fixed value of $\eta = 0.3$ for the three datasets. One may take into consideration optimized stepsize values, either with an optimal value for each dataset using a line search technique, or with a stepsize value decreasing at each iteration, i.e., $\eta(t+1) < \eta(t)$. These optimization schemas are beyond the scope of this paper. It is worth noting that the results obtained from the regularized pre-image estimation show that one may get into local minima.

We turn now to the approach where the weights are constrained to be non-negative. No restrictions on the data were required in this case, and thus no translation was operated as given above, with samples having positive as well as negative values. We compared three types of kernels: the Gaussian kernel

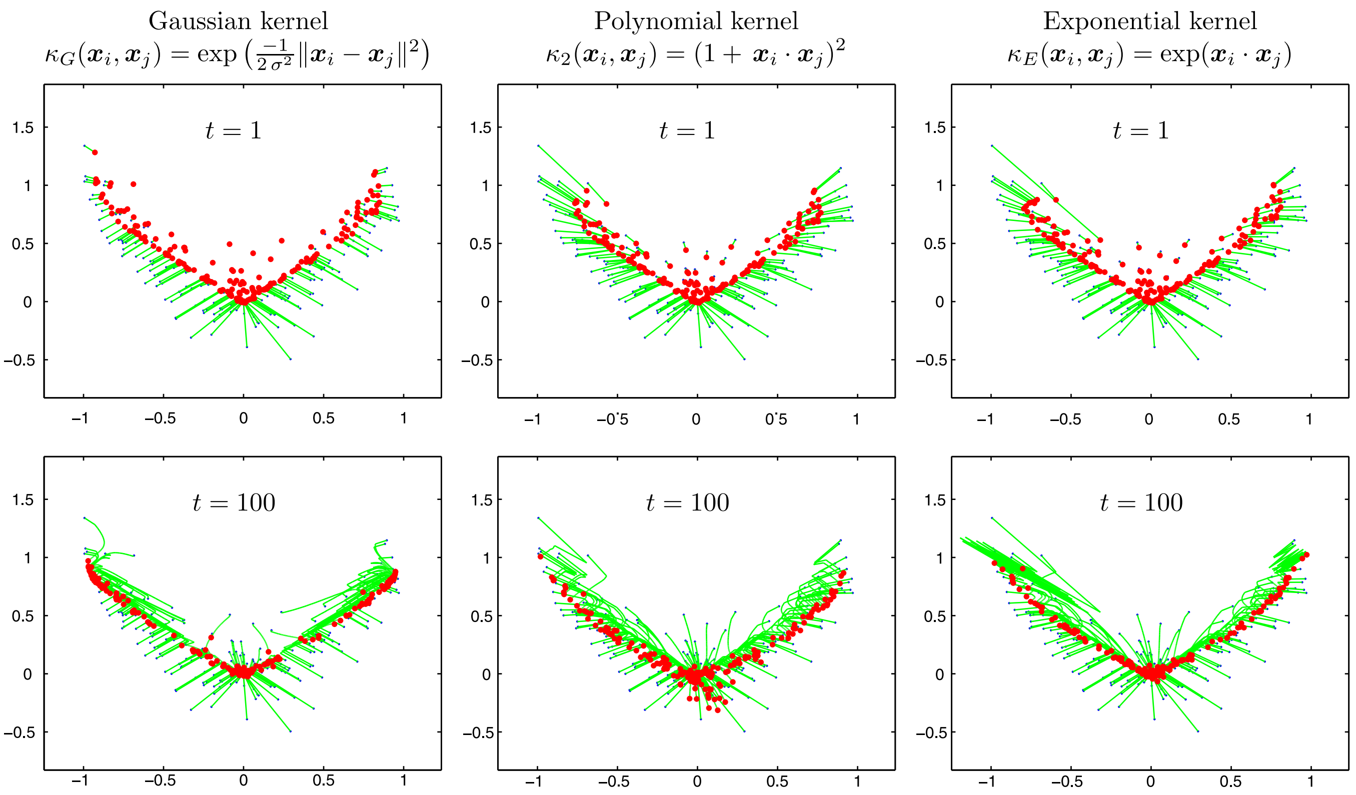


Fig. 4. Denoising with constraints on the model weights, of the banana dataset for a single iteration (upper row), and after $t=100$ iterations (lower row). Three different kernels are compared: Gaussian (left column), polynomial (middle column), and exponential (right column) kernels.

with bandwidth $\sigma=0.7$ as above, the polynomial quadratic kernel with $p=2$ and $c=1$, and the exponential kernel with $\sigma=1$ (see Table 1). For all these kernels, the initial value for $\boldsymbol{\beta}$ given a noisy data $\boldsymbol{x}_0$ was set to the solution of $\boldsymbol{x}_0=\boldsymbol{X}^T\boldsymbol{\beta}$ by retaining only non-negative weights, namely using a pseudo-inverse with

$$\boldsymbol{\beta}(0)=(\boldsymbol{X}\boldsymbol{X}^T)^{-1}\boldsymbol{X}\boldsymbol{x}_0, \tag{23}$$

for non-negative values; otherwise, it was set to zero. Even with only one iteration ($t=1$) and a small stepsize value of $\eta=0.1$, the proposed algorithm yielded a good denoised pre-image result, as shown in Fig. 4 (upper row). In this case, where the non-negative constraints are on the weights, we set the maximum number of iterations to $t=100$, the three kernels gave comparable results reflecting the shape of the banana manifold, as shown in Fig. 4 (lower row). This result is in opposition with the previous results observed in [23], where Kwok and Tsang claimed that only the Gaussian kernel can be pre-imaged with their work. By constraining the solution to only non-negative weights, as studied in this paper, we see that other kernels provide relevant results.

Now, we turn to the analysis of the model weights, and the sparsity of the solution. This experiment deals with the banana dataset. To this end, we consider the distribution of the weights $\beta_1,\beta_2,\ldots,\beta_n$ for each of the $N=200$ new samples to be denoised (given in Table 4). Fig. 5 shows the histogram of such distribution, where each color in the color bars corresponds to a denoised sample. While we represent here the results of a single iteration, similar results are obtained for larger number of iterations. These results illustrate the fact that the weights are non-negative as expected, lying between 0 and 0.018. Moreover, most of them are close to zero, namely below 0.002. This is the property of sparsity, well established and often required by a large class of algorithms in machine learning community.

5.2. Denoising images

We applied the denoising scheme on real handwritten digits, taken from the MNIST database.[5] From the dataset, we have chosen images of the digit “0”. Each image is defined by 28×28 gray-level pixels, i.e., pixels have values between 0 and 255. Thus, each image can be written as a 784-dimensional vector. The images were corrupted by adding a salt-and-pepper noise, with density $0.1, 0.25$ and 0.5. The images were denoised under the non-negativity of the data, as defined by (19) (see also [14]).

The relevance of the proposed method is now demonstrated for image denoising, and compared to different techniques: the fixed-point iterative method [22], the multi-dimensional scaling method [23], the regularized pre-image estimation [24], and the penalized pre-image [28]. For this purpose, we used the Gaussian kernel with a bandwidth set to $\sigma=500$, fixed for all pre-image techniques. A set of 500 images was used to train the kernel PCA with 50 eigenvectors. Another set of 10 images, shown in Fig. 6 (first row), corrupted by the same noise settings, was used for denoising (second row). Different techniques are applied in order to denoise the digits. As we can see in Fig. 6, where the density is set to 0.1, the proposed method presents the best denoising results among all the others. We evaluate the mean absolute error (MAE) for each denoised image using the several pre-image techniques with

$$\mathrm{MAE}=\frac{1}{28\times 28}\sum_{i=1}^{28}\sum_{j=1}^{28}\left|\boldsymbol{x}^*_{i,j}-\boldsymbol{x}_{i,j}\right|,$$

where $\boldsymbol{x}^*_{i,j}$ is the i,j–th pixel of the image $\boldsymbol{x}^*$ evaluated with a pre-image technique and $\boldsymbol{x}_{i,j}$ is the i,j–th pixel of the initial corresponding image without noise. Tables 5–7 show the MAE for every

[5] This MNIST database is available at http://yann.lecun.com/exdb/mnist/.

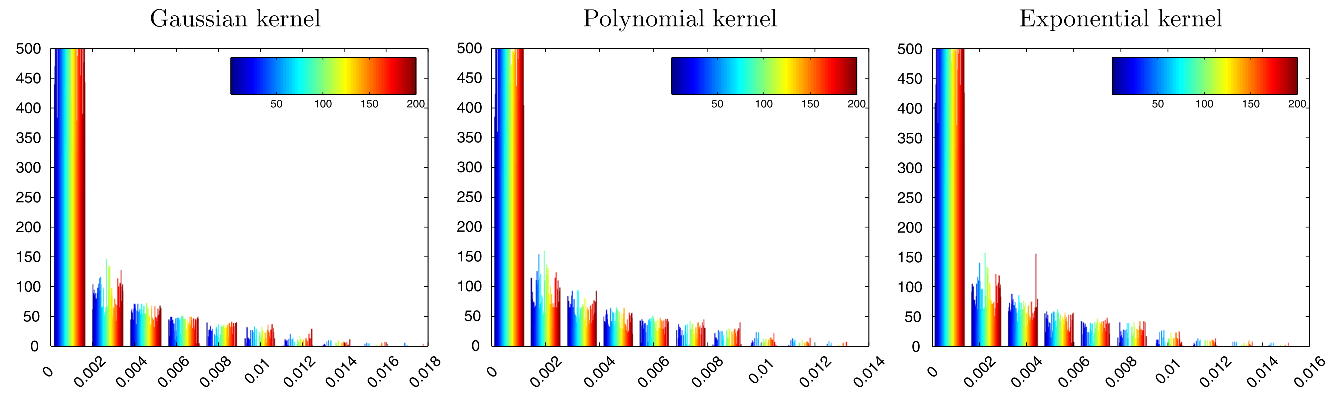


Fig. 5. Distribution of the model weights for each of the 200 noisy data from the banana dataset, after only one iteration of our algorithm, corresponding to results given in Fig. 4 (upper row). All denoised data (each represented by a color within the color bars) enjoy the sparsity property, with a large number of weights close to zero. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

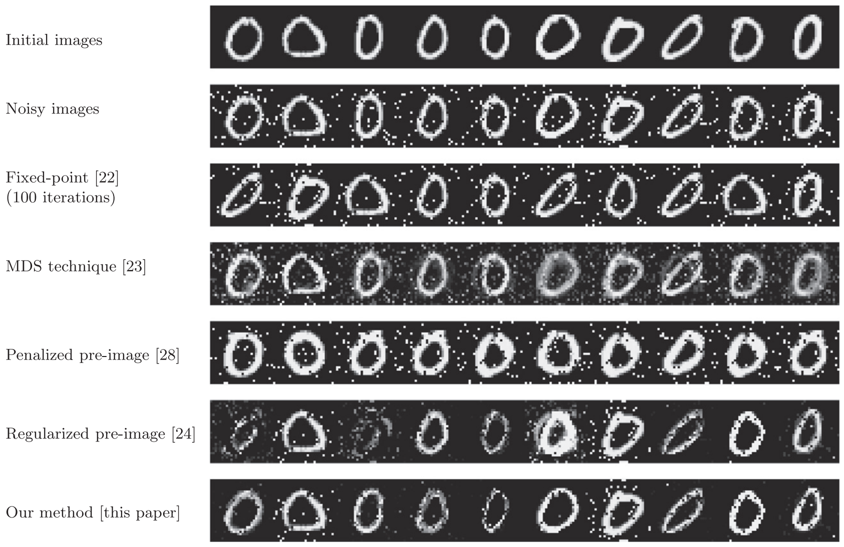


Fig. 6. A set of 10 “0”-digit images (first row) corrupted by a salt-and-pepper noise of density 0.1 (second row), on which we applied the kernel PCA for data denoising. The pre-image results using the fixed-point iterative algorithm [22] are illustrated (third row), the MDS technique [23] (fourth row), the penalized pre-image learning method [28] (fifth row), the regularized pre-image estimation technique [24] (sixth row), and the non-negative pre-image with the iterative schema (20) (last row).

Table 5
The mean absolute error for each technique evaluated on every denoised image where the density is 0.1.

Image number	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Mean
Fixed-point	52.75	**13.55**	21.77	**13.04**	**10.45**	50.95	**9.57**	54.67	11.02	57.23	29.50
MDS technique	31.31	30.49	25.15	20.83	27.89	25.87	24.92	**24.86**	**24.04**	25.85	**26.12**
Penalized pre-image	34.70	42.99	40.86	38.70	41.75	34.18	29.15	37.60	35.51	36.62	37.21
Regularized pre-image	176.43	**13.55**	144.77	145.07	42.00	284.03	17.23	236.62	54.63	20.85	113.52
Our method	**21.29**	**13.55**	**19.46**	145.07	42.00	**16.18**	71.59	126.24	54.63	**17.90**	52.79

Table 6
The mean absolute error for each technique evaluated on every denoised image where the density is 0.25.

Image number	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Mean
Fixed-point	**34.26**	76.81	**28.99**	**30.76**	**31.79**	70.89	54.25	67.97	54.55	67.05	51.73
MDS technique	47.56	47.73	41.34	44.13	63.94	44.81	52.43	43.91	58.12	47.39	49.13
Penalized pre-image	53.11	66.21	43.41	63.68	65.27	57.65	52.76	54.67	46.38	63.97	56.71
Regularized pre-image	**34.26**	213.43	**28.99**	**30.76**	**31.79**	175.21	124.61	314.11	148.77	156.04	125.80
Our method	**34.26**	**30.93**	**28.99**	**30.76**	**31.79**	**33.25**	**32.83**	**30.80**	**35.77**	**32.10**	**32.15**

Table 7
The mean absolute error for each technique evaluated on every denoised image where the density is 0.5.

Image number	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Mean
Fixed-point	77.8	93.51	72.51	**60.92**	72.86	80.57	77.25	90.47	75.78	**60.29**	76.20
MDS technique	75.37	73.82	78.55	88.09	65.35	69.75	116.35	71.39	70.86	87.98	79.75
Penalized pre-image	82.51	91.71	75.82	75.90	86.30	83.39	80.85	87.02	83.15	88.68	83.53
Regularized pre-image	142.89	272.87	**62.22**	**60.92**	158.47	**58.71**	109.55	**63**	403.75	**60.29**	139.27
Our method	**57.61**	**67.83**	**62.22**	**60.92**	**63.04**	**58.71**	**71.82**	**63**	**62.37**	**60.29**	**62.78**

Table 8
The peak signal-to-noise ratio for each technique evaluated on every denoised image where the density is 0.1.

Image number	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Mean
Fixed-point	12.91	12.89	7.45	8.94	7.61	7.37	13.44	14.21	9.23	7.99	10.20
MDS technique	12.30	11.48	**14.83**	13.37	14.78	13.90	13.95	12.26	**12.84**	**14.72**	**13.44**
Penalized pre-image	10.50	7.80	8.10	8.15	7.89	9.78	8.80	9.57	9.09	9.71	8.94
Regularized pre-image	14.04	12.95	−0.43	**15.12**	15.10	**16.60**	14.00	13.10	3.86	2.68	10.70
Our method	**15.27**	**12.97**	14.04	8.61	**15.16**	6.71	**14.22**	**14.52**	8.26	12.80	12.26

Table 9
The peak signal-to-noise ratio for each technique evaluated on every denoised image where the density is 0.25.

Image number	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Mean
Fixed-point	8.93	5.75	9.61	9.39	**9.19**	6.03	7.33	6.31	7.38	6.29	7.62
MDS technique	**9.98**	**9.86**	**11.90**	**11.33**	7.99	**10.39**	**10.28**	9.01	**9.40**	**10.25**	**10.04**
Penalized pre-image	7.63	6.49	8.40	6.60	6.39	7.09	7.54	7.41	8.22	6.49	7.23
Regularized pre-image	8.93	−1.05	9.61	9.39	**9.19**	0.63	3.39	−4.61	2.27	1.58	3.93
Our method	8.93	9.41	9.61	9.39	**9.19**	8.97	9.14	**9.32**	8.80	9.11	9.19

Table 10
The peak signal-to-noise ratio for each technique evaluated on every denoised image where the density is 0.5.

Image number	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	Mean
Fixed-point	5.58	4.75	5.87	**6.43**	5.85	5.40	5.61	4.87	5.72	**6.40**	5.65
MDS technique	**7.29**	**7.87**	**6.89**	6.36	**9.24**	**8.53**	3.98	**7.59**	**7.70**	6.22	**7.17**
Penalized pre-image	5.31	4.87	5.66	5.67	5.04	5.20	5.43	5.07	5.26	4.93	5.24
Regularized pre-image	2.88	−2.36	6.30	**6.43**	1.36	6.55	4.93	6.27	−5.92	**6.40**	3.28
Our method	6.70	5.99	6.30	**6.43**	6.24	6.55	**5.67**	6.27	6.32	**6.40**	6.29

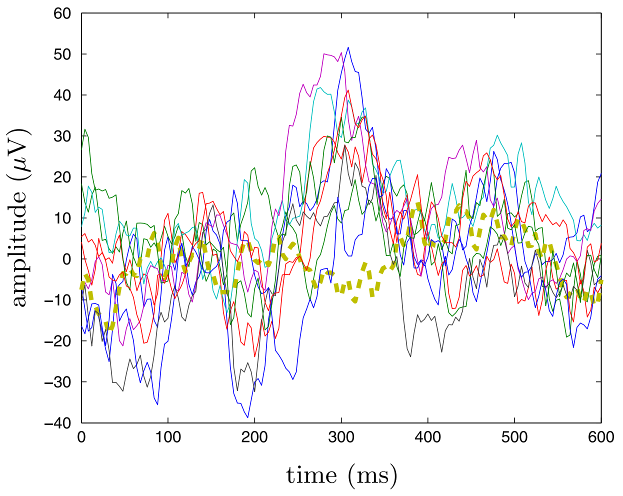


Fig. 7. Some ERP signals recorded from the Cz channel. The diversity of these signals is shown, with some signals not having a positive component around 300 ms (see for instance the signal shown with the dashed bold signal).

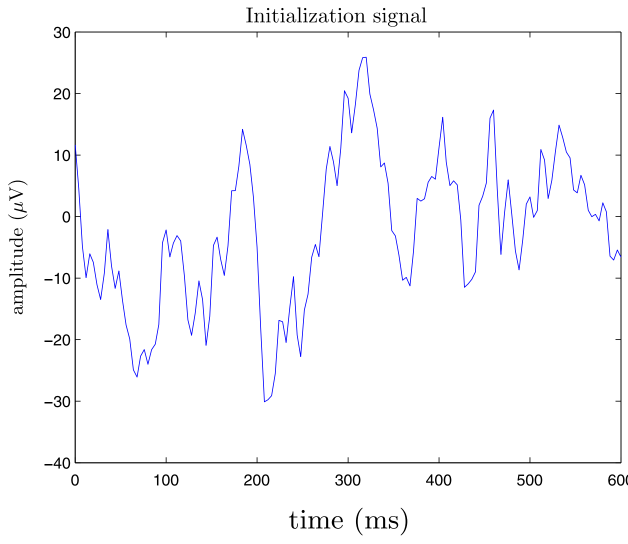
image using the several pre-image techniques with density equals to 0.1, 0.25 and 0.5 respectively. We also evaluate the peak signal-to-noise ratio (PSNR) for each denoised image using the several pre-image techniques with

$$\mathrm{PSNR} = 10 \times \log_{10}\left(\frac{255^2}{\frac{1}{28\times 28}\sum_{i=1}^{28}\sum_{j=1}^{28}|\boldsymbol{x}^*_{i,j} - \boldsymbol{x}_{i,j}|^2}\right),$$

where $\boldsymbol{x}^*_{i,j}$ is the i,j–th pixel of the image $\boldsymbol{x}^*$ evaluated with a pre-image technique and $\boldsymbol{x}_{i,j}$ is the i,j–th pixel of the initial corresponding image without noise. Tables 8–10 show the MAE for every image using the several pre-image techniques with density equals to 0.1, 0.25 and 0.5 respectively. It is worth noting that when the salt-and-pepper noise has a density of 0.5, it is difficult to denoise the images using any technique due to the presence of important noise.

5.3. *Feature extraction*

We considered feature extraction with an application to real signals, and more specifically recordings measuring brain activity. The feature extraction under non-negativity is the constraint applied on the weights of the model [13]. Event-related potentials (ERP) refer to the electrical activity in the brain due to a response to a specific stimulus, measured with electroencephalograph (EEG). There is a strong consensus on the components of an ERP recoding, independent of either the participants or the stimulus type. Such signal includes a negative wave deflection (called N200



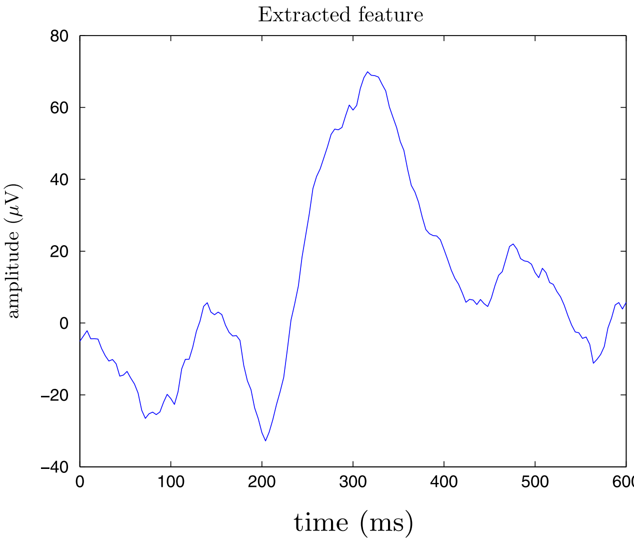


Fig. 8. Feature extraction of the ERP data, with the algorithm initialized to the initial signal (upper figure). By pre-imaging the first principal axe of kernel PCA, we get the feature (lower figure).

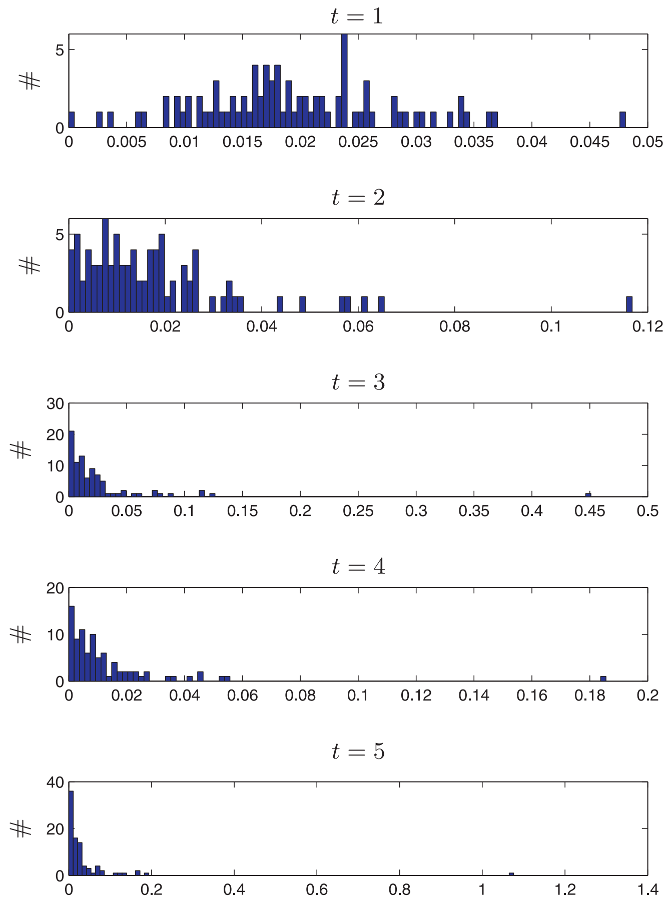


Fig. 9. Distribution of the model weights from the first iteration (upper figure) to the fifth iteration (lower figure). This illustrates the evolution of the weights towards a sparse distribution.

or N2) followed by a positive one (called P300 or P3), occurring respectively around 200 ms and 300 ms after stimulus onset. Within the brain activity, such a single response is not usually visible in these recordings. To circumvent this, many trials are often performed using the same stimulus. In practice, one takes the average of these responses, which gives a first-order moment statistic of the ERP recordings. In this paper, we give another statistic taking into account the variance of these signals, by combining kernel PCA with the Gaussian kernel on the one hand, and the proposed pre-image technique on the other.

For experimentations, we used the ERP signals available here[6]; for more information, see also [38,39]. The auditory stimulus is composed of a series of two alternating tone signals, randomly played with a time between stimuli (also called Inter-Stimulus Interval or ISI) of 1 s. These stimuli correspond to either a tone at the frequency 800 Hz or another tone at 560 Hz, played within the ratio 85% of the first signal and 15% of the second one. The ERP signals are recordings from a 64-channel EEG, where only the midline central channel Cz is used for its high reliability in potential detection. The recording captured within the Cz channel are segmented into signals in order to view the reaction of the subject to the stimulus by using a window [0, 600] ms, where 0 corresponds to the instance of onset stimulus. Such window is appropriate to extract both N200 and P300 components of the ERP. A set of 87 signals of length 600 ms is collected, with 151 samples each, as illustrated in Fig. 7 where only 10 randomly selected signals are shown to display the variety of these signals.

We applied the kernel PCA to extract the first principal axe of these data, in the feature space associated with the Gaussian kernel. The pre-image approach allowed us to go back to the initial space, that is, the signal space. Because signals have negative components,[7] we applied the pre-image technique with the non-negative constraints on the weights. Following some preliminary experimentations, the Gaussian kernel was used, with the bandwidth set to $\sigma = 500$, and the stepsize value to the value $\eta = 0.1$. Next, we study the influence of the initialization on the algorithm, based on two different initializations.

First, the algorithm was initialized using a random input data, namely $\boldsymbol{x}_1$ without loss of generality and shown in Fig. 8 (upper figure). Then $\boldsymbol{\beta}(0) = (\boldsymbol{X}\boldsymbol{X}^T)^{-1}\boldsymbol{X}\boldsymbol{x}_0$ for non-negative values, and zero

[6] The dataset of ERP recordings are available from the University of Kuopio, Finland and Mika Tarvainen's page http://venda.uku.fi/opiskelu/kurssit/LSA/.

[7] To be more precise, measurements of brain activity are always positive. However, practitioners calibrate these measurements, resulting into zero-mean signals.

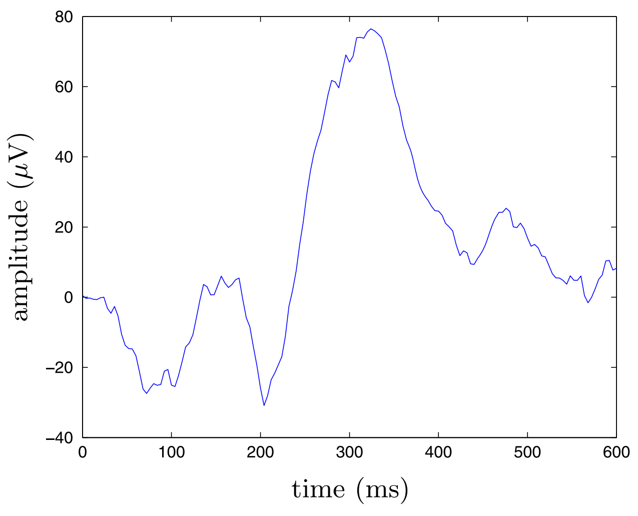


Fig. 10. Feature extraction of the ERP data, with the algorithm initialized to the uniform contribution of all available signals, corresponding to $t=5$ in Fig. 9 (lower figure).

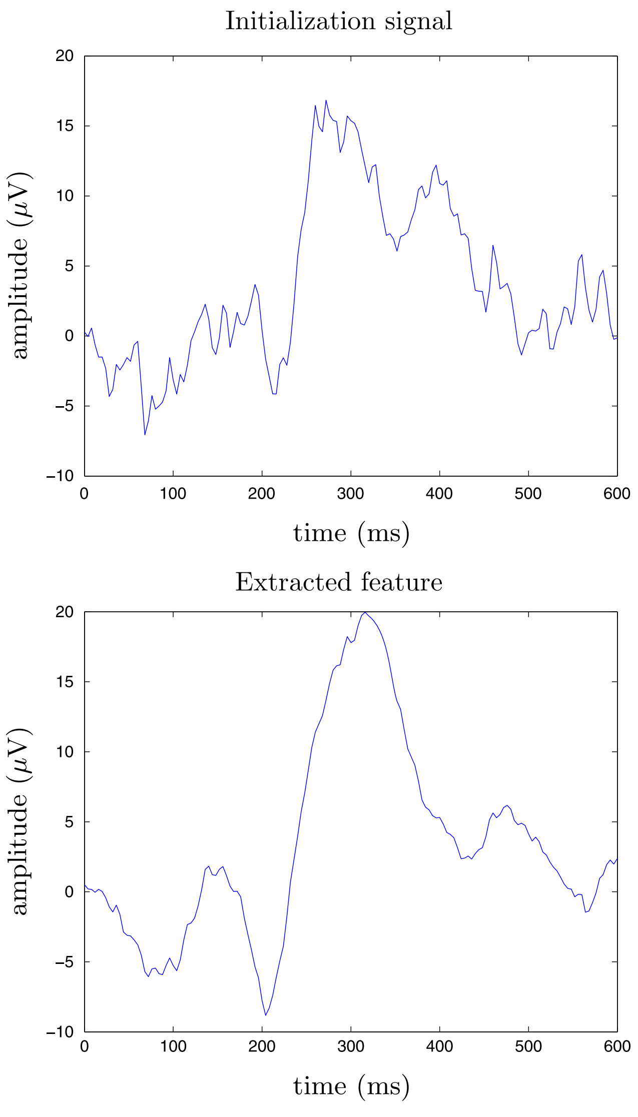


Fig. 11. The average of 10 signals (upper figure) as initialization signal and the corresponding feature extracted (lower figure).

otherwise. Applying the algorithm for $t=100$ iterations gave the feature illustrated in Fig. 8 (lower figure). We can easily see both important components of the ERP, the N200 and P300 waves. Moreover, variations of features of interest are opposed to the highly fluctuating initial signal.

In order to study the evolution of the weights at each iteration, we considered the initialization case where all the weights are equal, i.e., $\beta_k = 1/n$ for all $k = 1, 2, \ldots, n$. This corresponds to the average of the data, where the solution results from a uniform contribution of all the available data. The evolution of the distribution of these weights over the first five iterations is given in the histograms of Fig. 9. This shows that the proposed algorithm resulted into sparse representations, with sparsity increasing at each iteration. The resulting feature is given in Fig. 10, which shows both N200 and P300 components, even within the first few iterations. By comparing this technique to the average of some signals in Fig. 11 (upper figure) and to all signals in Fig. 11 (lower figure), we see that we need all the signals to find the N200 and P300, however, using our method, we only have to use a few signals.

6. Conclusion and future work

In this paper, we derived several new theoretical results, and proposed an iterative method to solve the pre-image problem with non-negativity constraints. These constraints were either on the pre-image itself, or on the weights of the model. In this case, we investigated experimentally the sparsity of the representation. Compared to other techniques, simulations showed the effectiveness of the proposed method.

As for future work, we would like to incorporate box constraints, where upper and lower bounds must be satisfied, such as processing gray-level images. We suggest further investigations on other methods that involve pre-image techniques, such as an autoregressive model.

Conflict of interest statement

None declared.

References

[1] G. Chen, B. Kégl, Image denoising with complex ridgelets, Pattern Recognition 40 (2) (2007) 578–585.
[2] W.-S. Zheng, J. Lai, S. Liao, R. He, Extracting non-negative basis images using pixel dispersion penalty, Pattern Recognition (2012).
[3] C. Twining, C. Taylor, The use of kernel principal component analysis to model data distributions, Pattern Recognition 36 (1) (2003) 217–227.
[4] Z. Yang, E. Oja, Quadratic nonnegative matrix factorization, Pattern Recognition 45 (4) (2012) 1500–1510.
[5] G. Thomas, A positive optimal deconvolution procedure, IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP 8 (April) (1983) 651–654.
[6] R. Prost, R. Goutte, Discrete constrained iterative deconvolution algorithms with optimized rate of convergence, Signal Processing 7 (December (3)) (1984) 209–230.
[7] G. Thomas, N. Souilah, Utilisation des multiplicateurs de lagrange pour la restauration d'image AVEC contraintes, Colloques sur le Traitement du Signal et des Images, 1991.
[8] D. Snyder, T. Schulz, J. O'Sullivan, Deblurring subject to nonnegativity constraints, IEEE Transactions on Signal Processing 40 (May) (1992) 1143–1150.
[9] H. Lantéri, M. Roche, O. Cuevas, C. Aime, A general method to devise maximum-likelihood signal restoration multiplicative algorithms with non-negativity constraints, Signal Processing 81 (May (5)) (2001) 945–974.
[10] J. Chen, C. Richard, P. Honeine, H. Snoussi, H. Lantéri, C. Theys, Techniques d'apprentissage non-linéaires en ligne avec contraintes de positivite, in: Actes de la VIème Conférence Internationale Francophone d'Automatique, Nancy, France, 2–4 June 2010.
[11] J. Chen, C. Richard, P. Honeine, H. Lantéri, C. Theys, System identification under non-negativity constraints, in: Proceedings of European Conference on Signal Processing (EUSIPCO), Aalborg, Denmark, Eurasip, 2010.

[12] J. Chen, C. Richard, P. Honeine, J.C.M. Bermudez, Non-negative distributed regression for data inference in wireless sensor networks, in: Proceedings of the 44th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove (CA), USA, 2010.

[13] M. Kallas, P. Honeine, C. Richard, H. Amoud, C. Francis, Nonlinear feature extraction using kernel principal component analysis with non-negative pre-image, in: Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Buenos Aires, Argentina, 31 August–4 September, 2010.

[14] M. Kallas, P. Honeine, C. Richard, C. Francis, H. Amoud, Non-negative pre-image in machine learning for pattern recognition, in: Proceedings of the 19th European Signal Processing Conference, Barcelona, Spain, 29 August–2.

[15] M. Kallas, H. Amoud, P. Honeine, C. Francis, Sur le problème de la pré-image en reconnaissance des formes avec contraintes de non-négativité, in: Colloque GRETSI'2011, Bordeaux, France, 5–8 Septembre 2011.

[16] H. Han, Nonnegative principal component analysis for mass spectral serum profiles and biomarker discovery, BMC Bioinformatics (2010).

[17] R. Zass, A. Shashua, Nonnegative sparse PCA, in: Neural Information Processing Systems, 2007.

[18] B. Moghaddam, Y. Weiss, S. Avidan, Spectral bounds for sparse PCA: exact and greedy algorithms, in: Advances in Neural Information Processing Systems, MIT Press, 2006, pp. 915–922.

[19] C.D. Sigg, J.M. Buhmann, Expectation–maximization for sparse and non-negative PCA, in: 25th International Conference on Machine Learning (ICML), ACM, 2008.

[20] V.N. Vapnik, Statistical Learning Theory, Wiley-Interscience, September 1998.

[21] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (July) (1998) 1299–1319.

[22] S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, G. Rätsch, Kernel PCA and de-noising in feature spaces, in: Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II, MIT Press, Cambridge, MA, USA, 1999, pp. 536–542.

[23] J.T. Kwok, I.W. Tsang, The pre-image problem in kernel methods, in: T. Fawcett, N. Mishra (Eds.), Proceedings of the 20th International Conference on Machine Learning, AAAI Press, 2003, pp. 408–415.

[24] T.J. Abrahamsen, L.K. Hansen, Regularized pre-image estimation for kernel PCA de-noising: input space regularization and sparse reconstruction, Journal of Signal Processing Systems 65 (3) (2011) 403–412.

[25] P. Honeine, C. Richard, Solving the pre-image problem in kernel machines: a direct method, in: Proceedings of the IEEE Workshop on Machine Learning for Signal Processing (MLSP), Grenoble, France, September 2009.

[26] P. Honeine, C. Richard, A closed-form solution for the pre-image problem in kernel-based machines, Journal of Signal Processing Systems 65 (December) (2011) 289–299.

[27] P. Honeine, C. Richard, Preimage problem in kernel-based machine learning, IEEE Signal Processing Magazine 28 (2) (2011) 77–88.

[28] W. Zheng, J. Lai, P.C. Yuen, Penalized preimage learning in kernel principal component analysis, IEEE Transaction Neural Networks 21 (April) (2010) 551–570.

[29] E.J. Candes, M.B. Wakin, An introduction to compressive sampling, IEEE Signal Processing Magazine 25 (March (2)) (2008) 21–30.

[30] N. Aronszajn, Theory of reproducing kernels, Transactions of the American Mathematical Society 68 (1950) 337–404.

[31] F. Cucker, S. Smale, On the mathematical foundations of learning, Bulletin of the American Mathematical Society 39 (2002) 1–49.

[32] C.J.C. Burges, Geometry and invariance in kernel based methods, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods, MIT Press, Cambridge, MA, USA, 1999, pp. 89–116.

[33] B. Schölkopf, A. Smola, K.-R. Müller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation 10 (5) (1998) 1299–1319.

[34] R. Rosipal, M. Girolami, L.J. Trejo, Kernel PCA for feature extraction and de-noising in non-linear regression, Neural Computing and Applications 10 (2000) 231–243.

[35] G.S. Kimeldorf, G. Wahba, Some results on Tchebycheffian spline functions, Journal of Mathematical Analysis and Applications 33 (1) (1971) 82–95.

[36] B. Schölkopf, R. Herbrich, A.J. Smola, A generalized representer theorem, in: Proceedings of the 14th Annual Conference on Computational Learning Theory and Fifth European Conference on Computational Learning Theory (COLT '01/EuroCOLT '01), Springer-Verlag, London, UK, 2001, pp. 416–426.

[37] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, in: SCIENCE, vol. 290, 2000, pp. 2323–2326.

[38] S.D. Georgiadis, State-Space Modeling and Bayesian Methods for Evoked Potential Estimation, Ph.D. Dissertation, Department of Applied Physics, University of Kuopio, Finland, May 2007.

[39] M.P. Tarvainen, Estimation Methods for Nonstationary Biosignals, Ph.D. Dissertation, Department of Applied Physics, University of Kuopio, Finland, June 2004.

Maya Kallas was born in Zahlé, Lebanon, on September 24, 1985. She received the Dipl.-Ing. degree in computer and communication engineering in 2008 from the Holy Spirit University, Lebanon, the M.Sc. degree in Industrial Control in 2009, from the Lebanese University, Lebanon, and the M.Sc. degree in Sciences, Technologies and Health in 2009, from the University of Technology of Compiegne, France. Since February 2010, she has been a Ph.D. student at the University of Technology of Troyes, France, and at the Lebanese University, Lebanon. Her research interests include nonstationary signal analysis, kernel machines, machine learning and pattern recognition.

Paul Honeine was born in Beirut, Lebanon, on October 2, 1977. He received the Dipl.-Ing. degree in mechanical engineering in 2002 and the M.Sc. degree in industrial control in 2003, both from the Faculty of Engineering, the Lebanese University, Lebanon. In 2007, he received the Ph.D. degree in Systems Optimization and Security from the University of Technology of Troyes, France, and was a Postdoctoral Research associate with the Systems Modeling and Dependability Laboratory, from 2007 to 2008.

Since September 2008, he has been an Assistant Professor at the University of Technology of Troyes, France. His research interests include nonstationary signal analysis, nonlinear adaptive filtering, sparse representations, machine learning, hyperspectral data analysis, and wireless sensor networks.

He is the co-author (with C. Richard) of the 2009 Best Paper Award at the IEEE Workshop on Machine Learning for Signal Processing.

Cédric Richard was born January 24, 1970 in Sarrebourg, France. He received the Dipl.-Ing. and the M.S. degrees in 1994 and the Ph.D. degree in 1998 from the University of Technology of Compiègne (UTC), France, all in electrical and computer engineering.

He joined the Côte d'Azur Observatory, University of Nice Sophia-Antipolis, France, in 2009. He is currently a Professor of electrical engineering. From 1999 to 2003, he was an Associate Professor at the University of Technology of Troyes (UTT), France. From 2003 to 2009, he was a Professor at the UTT, and the supervisor of a group consisting of 60 researchers and Ph.D. students. In winter 2009 and autumn 2010, he was a Visiting Researcher with the Department of Electrical Engineering, Federal University of Santa Catarina (UFSC), Florianópolis, Brazil. Cédric Richard is a junior member of the Institut Universitaire de France since October 2010. He was the General Chair of the XXIth francophone conference GRETSI on Signal and Image Processing that was held in Troyes, France, in 2007, and of the IEEE Statistical Signal Processing Workshop (IEEE SSP'11) that was held in Nice, France, in 2011. Since 2005, he is a Member of the board of the federative CNRS research group ISIS on Information, Signal, Images and Vision. He is a Member of GRETSI association board. Prof. Cédric Richard is the author of over 120 papers. His current research interests include statistical signal processing and machine learning.

Cédric Richard served as an Associate Editor of the IEEE Transactions on Signal Processing from 2006 to 2010, and of EURASIP Signal Processing since 2009. In 2009, he was nominated liaison local officer for EURASIP, and member of the Signal Processing Theory and Methods Technical Committee of the IEEE Signal Processing Society. Paul Honeine and Cédric Richard received Best Paper Award for "Solving the pre-image problem in kernel machines: a direct method" at the 2009 IEEE Workshop on Machine Learning for Signal Processing (IEEE MLSP'09).

Clovis Francis received his Engineering Diploma from the Lebanese University, Faculty of Engineering in 1990, the Diplôme d'Etudes Approfondies from the University Paul Sabatier – Toulouse in 1991, the Ph.D. degree from the University Paris XI – France in 1994 and the Accreditation to Supervise Research (HDR) in 2008. Currently he is a Professor at the Lebanese University. His main research activities cover the areas of Automatic Control, Signal Processing and Diagnosis.

Hassan Amoud was born on March 23, 1980 in Minieh, Lebanon. He received his engineering degree in Electrical Engineering in 2002 from the Lebanese University, Lebanon. He also received the Master degree in System Information and Technology in 2003 from the University of Technology of Compiègne, France and the Ph.D. degree in Signal Processing and Biomedical Engineering in 2006 from the University of Technology of Troyes (UTT), France. From 2006 to September 2009, he was in the Institut Charles Delaunay, Systems Modeling and Dependability team, at UTT. Since October 2009, he is an associate professor in the Azm Center for Research in Biotechnology and its Applications, at the Lebanese University. His research interests include statistical signal processing and machine learning.