



HAL
open science

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

Maya Kallas, Paul Honeine, Cédric Richard, Clovis Francis, Hassan Amoud

► To cite this version:

Maya Kallas, Paul Honeine, Cédric Richard, Clovis Francis, Hassan Amoud. Non-negativity constraints on the pre-image for pattern recognition with kernel machines. *Pattern Recognition*, 2013, 46 (11), pp.3066 - 3080. 10.1016/j.patcog.2013.03.021 . hal-01965576

HAL Id: hal-01965576

<https://hal.science/hal-01965576v1>

Submitted on 26 Dec 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

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 H. Fizeau (CNRS, OCA), Université de Nice Sophia-Antipolis, Nice, France*

^c*Laboratoire d'analyse des systèmes (LASYS), Faculté de Génie 1, Université Libanaise, Liban*

^d*Azm Center for Research in Biotechnology and its Applications, Doctoral School, Lebanese University, Lebanon*

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. 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 relevance of the proposed method is demonstrated with experimentations on artificial data and on two types of real datasets in signal and image processing.

Keywords: kernel machines, machine learning, SVM, kernel PCA, pre-image problem, non-negativity constraints, nonlinear denoising, pattern recognition

^{*}This work is partly supported by the Lebanese University and the French-Lebanese research program CEDRE No. 10 SCI F15/L5.

^{*}Corresponding author. Address: Université de technologie de Troyes, 12 rue Marie Curie, 10010, Troyes. Phone: +33 3 25 71 56 25. Fax: +33 3 25 71 56 49.

Email addresses: maya.kallas@utt.fr (Maya Kallas), paul.honeine@utt.fr (Paul Honeine), cedric.richard@unice.fr (Cédric Richard), cfrancis@ul.edu.lb (Clovis Francis), hassan.amoud@gmail.com (Hassan Amoud)

1. Introduction

Many applications in real-life 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 feature, 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 eighties, this was studied for signal deconvolution by Thomas in [5] and Prost et al. in [6]. In the beginning of the nineties, image deconvolution and deblurring were studied respectively by Thomas et al. in [7] and Snyder et al. in [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 by the authors 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, 17, 18, 19]. In the last couple of decades or so, kernel machines 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 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

41 fact that the implicit map derived by the kernel is non-surjective, with most elements of the feature space
42 that do not have exact pre-images, and thus cannot be exactly represented in the input space.

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

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

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

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

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

87 2. Kernel machines and kernel PCA for pattern recognition

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

92 2.1. Kernel machines

93 Let $\mathcal{X} \subset \mathbb{R}^d$ be an input space with the canonical (Euclidean) dot product $\mathbf{x}_i \cdot \mathbf{x}_j$ for any $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$.
94 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
95 if and only if any matrix \mathbf{K} with entries $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ for any finite subset of \mathcal{X} is positive definite, that is
96 $\sum_{i,j} \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ for all $\alpha_i, \alpha_j \in \mathbb{R}$ and all $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$. Based on the Moore-Aronszajn theorem
97 [30], any positive definite kernel guarantees the existence of a unique¹ feature space (or reproducing kernel
98 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
99 input space to the feature space, such that

¹Unique, up to an isometry.

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

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

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

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i \cdot \mathbf{x}_j), \quad (2)$$

108 and radial kernels, of the form

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

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

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

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

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

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

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

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

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

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

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

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

$$\begin{aligned}
 f(\zeta) &\geq 0 \\
 f^{(1)}(\zeta) &\geq 0 \\
 f^{(1)}(\zeta) + \zeta f^{(2)}(\zeta) &\geq 0
 \end{aligned}$$

120 2.2. Kernel PCA

121 The principal component analysis (PCA) is a powerful mathematical tool to reveal patterns within a
 122 set of data. It is a non-parametric approach, which does not incorporate any prior knowledge of the model,
 123 except its linearity. The PCA considers the most relevant eigenvectors of the data covariance matrix, i.e.,
 124 eigenvectors associated to the largest eigenvalues. These eigenvectors constitute a set of orthonormal axes
 125 capturing most of the variance within data. Let us consider a set of n (column-wise) data $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in$
 126 $\mathcal{X}\}$. Then, consider the eigen-problem $\lambda_k \mathbf{v}_k = \mathbf{C} \mathbf{v}_k$, where $\mathbf{C} = \frac{1}{n} \sum_{j=1}^n \mathbf{x}_j \mathbf{x}_j^T$ is the covariance matrix,
 127 data assumed to be centered around the origin. Then the m eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m \in \mathcal{X}\}$ are chosen
 128 from the largest eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_m$, where each λ_k gives the amount of captured variance in the
 129 direction of \mathbf{v}_k . Due to the linearity of the operations, each eigenvector lies in the span of the data.

130 The conventional PCA identify only linear structures in a given dataset. A more generalized technique
 131 has been introduced to learn the nonlinearities using kernels, the so-called kernel PCA. The kernel PCA
 132 can reveal nonlinear kernel principal components that are more appropriate to complex and nonlinear data
 133 such as face images, handwritten digits and natural signals. For this purpose, data are (implicitly) mapped
 134 into a feature space, where PCA is applied. Although the resulting eigenvectors are linear in the feature
 135 space, they describe nonlinear relations in the input space. In order to solve this nonlinear problem, it is

136 more likely to apply the kernel trick, and not to explicitly compute the map from the input to the feature
 137 space. The concept of kernel trick is illustrated here for kernel PCA [33, 34].

138 To this end, the PCA algorithm is recast in terms of inner product of data in feature space. Let
 139 $\Phi: \mathcal{X} \mapsto \mathcal{H}$ be a nonlinear map, and $\{\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \dots, \Phi(\mathbf{x}_n) \in \mathcal{H}\}$ the set of mapped data. We wish to
 140 solve the (kernel) PCA, in terms of inner products in the feature space, $\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle_{\mathcal{H}}$, for $i, j = 1, 2, \dots, n$.
 141 The covariance matrix² in \mathcal{H} is $\mathbf{C}^{\Phi} = \frac{1}{n} \sum_{j=1}^n \Phi(\mathbf{x}_j) \Phi(\mathbf{x}_j)^T$. The principal axes, $\varphi_k \in \mathcal{H}$ for $k = 1, 2, \dots, m$,
 142 correspond to the eigenvectors with the largest eigenvalues λ_k satisfying the expression

$$\lambda_k \varphi_k = \mathbf{C}^{\Phi} \varphi_k. \quad (4)$$

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

$$\varphi_k = \sum_{i=1}^n \alpha_{k,i} \Phi(\mathbf{x}_i). \quad (5)$$

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

148 Replacing the expression of \mathbf{C}^{Φ} and the representer (5) into the eigen-problem (4), we get the new
 149 eigen-problem in terms of inner product with

$$n \lambda_k \boldsymbol{\alpha}_k = \mathbf{K} \boldsymbol{\alpha}_k, \quad (6)$$

150 where \mathbf{K} is the $n \times n$ matrix of entries $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ with (1) applied, and $\boldsymbol{\alpha}_k = [\alpha_{k,1} \ \alpha_{k,2} \ \dots \ \alpha_{k,n}]^T$. Further-
 151 more, two issues are considered in the final kernel PCA algorithm. First, as mentioned earlier, data should
 152 be centered in the feature space. This can be done by substituting the matrix \mathbf{K} by $(\mathbf{I} - \mathbf{1}_n)\mathbf{K}(\mathbf{I} - \mathbf{1}_n)$,
 153 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
 154 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
 155 the weight vectors $\boldsymbol{\alpha}_k$ such that $\lambda_k(\boldsymbol{\alpha}_k \cdot \boldsymbol{\alpha}_k) = 1$, for $k = 1, 2, \dots, m$.

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

156 *2.3. Kernel PCA for pattern recognition*

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

160 *2.3.1. Feature extraction*

161 Kernel-PCA defines the set of most relevant axes in the feature space. Let $\{\varphi_1, \varphi_2, \dots, \varphi_m \in \mathcal{H}\}$ be the
 162 set of these axes. Then each φ_k takes the form (5), namely

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

163 where $\alpha_{k,1}, \alpha_{k,2}, \dots, \alpha_{k,n}$ are obtained from the eigenvector associated to the k -th eigenvalue in (6). We
 164 also define the relevant subspace of \mathcal{H} as the one spanned by these axes. By analogy to the conventional
 165 PCA, these axes (as well as the associated subspace) capture most of the variance of the data. They can
 166 be regarded as features extracted from the data, capturing the largest variations and orthonormal to each
 167 others.

168 *2.3.2. Denoising*

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

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

173 Expanding this expression by (5) and applying the equivalence between the inner product operator and the
 174 kernel function κ , we get

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

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

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

2.3.3. A unifying view

Now, we propose a unified view to tackle both 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 $\varphi_k = \sum_{j=1}^n \alpha_{k,j} \Phi(\mathbf{x}_j)$ and $\varphi = \sum_{j=1}^n [\sum_{k=1}^m \sum_{i=1}^n \alpha_{k,i} \alpha_{k,j} \kappa(\mathbf{x}_0, \mathbf{x}_i)] \Phi(\mathbf{x}_j)$. Aggregating all these terms, we get a unifying view of both cases, with

$$\varphi = \sum_{j=1}^n \gamma_j \Phi(\mathbf{x}_j). \quad (7)$$

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

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

In the latter case, the coefficients γ_j depend on the noisy \mathbf{x}_0 , which can be either a new observation or one of the training data. Summarized in Table 2, the unifying expression in (7) enables us to define a general form in the optimization problem for both feature extraction and denoising schemes.

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 \mathbf{x}_0 , the projection of $\Phi(\mathbf{x}_0)$ onto any $\varphi \in \mathcal{H}$ of the form (7), can be defined by $\langle \varphi, \Phi(\mathbf{x}_0) \rangle_{\mathcal{H}} = \sum_{j=1}^n \gamma_j \kappa(\mathbf{x}_j, \mathbf{x}_0)$, 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

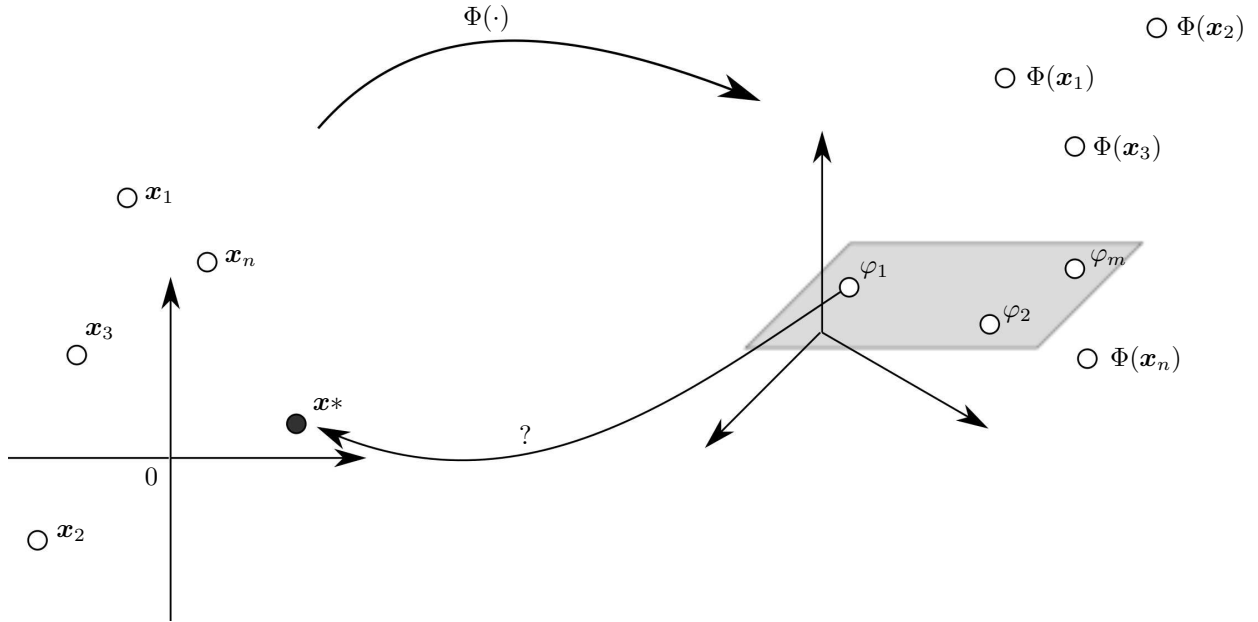


Figure 1: Schematic illustration of the pre-image problem. Constructed in the feature space from some training data, principal axes are mapped back to the input space by solving the pre-image problem, pre-imagining φ_1 into \mathbf{x}^* here.

191 feature itself. More likely, we seek its counterpart in the input space, the observation space. It is natural to
 192 have extracted patterns of the same type as the data, i.e., identical input space, since one often seeks for a
 193 signal as a pattern in signal processing, or a denoised image in image processing. The pre-image problem is
 194 illustrated in Figure 1.

195 With the exception of the Φ -images of the training data, only a very few elements in the feature space
 196 have *pre-images*, i.e., data which maps into (7) for some given coefficients. In fact, this is an ill-posed
 197 problem since the exact pre-image may not exist, and even if it exists, it might be not unique. To solve this
 198 problem, we seek an approximate solution, i.e., a $\mathbf{x}^* \in \mathcal{X}$ whose image $\Phi(\mathbf{x}^*)$ is as close as possible to φ .
 199 The way back from the feature space to the input space is called *the pre-image problem*. Initially studied
 200 by Mika et al. in [22] for denoising purpose, it consists of solving the optimization problem

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathcal{X}} \frac{1}{2} \|\varphi - \Phi(\mathbf{x})\|_{\mathcal{H}}^2, \quad (8)$$

201 where $\|\cdot\|_{\mathcal{H}}$ denotes the norm in \mathcal{H} , and thus provides a measure of distance between elements in the feature
 202 space, with the norm of their residue. Thanks to the unifying view given in (7) with $\varphi = \sum_{i=1}^n \gamma_i \Phi(\mathbf{x}_i)$, we
 203 consider the same optimization problem for either feature extraction or denoising, with

Table 3: Gradient of the cost function (9) for most commonly used kernels, with respect to either \mathbf{x} (second column) or $\boldsymbol{\beta}$ from $\mathbf{x} = \boldsymbol{\beta}^T \mathbf{X}$ (third column).

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

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

204 The general form used for the calculation is described by

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

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

$$J(\mathbf{x}) = -\sum_{i=1}^n \gamma_i \kappa(\mathbf{x}_i, \mathbf{x}) + \frac{1}{2} \kappa(\mathbf{x}, \mathbf{x}) \quad (9)$$

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

208 This is a highly nonlinear optimization problem. To solve this problem, one may study the gradient of
 209 the cost function $J(\mathbf{x})$ with respect to \mathbf{x} . At an optimum, the gradient with respect to \mathbf{x} disappears, namely
 210 $\nabla_{\mathbf{x}} J(\mathbf{x}) = 0$. The resulting gradient is given as

$$\nabla_{\mathbf{x}} J(\mathbf{x}) = -\sum_{i=1}^n \gamma_i \frac{\partial \kappa(\mathbf{x}_i, \mathbf{x})}{\partial \mathbf{x}} + \frac{1}{2} \frac{\partial \kappa(\mathbf{x}, \mathbf{x})}{\partial \mathbf{x}}. \quad (10)$$

211 This is the general form for all kernels, including for instance the projective kernels of the form (2) such as
 212 the polynomial kernel. Expressions (9)-(10) can be further simplified for the wide class of radial kernels, of
 213 the form (3) such as the Gaussian kernel. In such cases, $\kappa(\mathbf{x}, \mathbf{x})$ is independent of \mathbf{x} , for all $\mathbf{x} \in \mathcal{X}$, thus
 214 $\partial \kappa(\mathbf{x}, \mathbf{x}) / \partial \mathbf{x}$ equals to zero, and only the first term of (10) remains. See Table 1 for expressions of commonly
 215 used kernels.

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

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

$$\mathbf{x}^* = \sum_{i=1}^n \beta_i^* \mathbf{x}_i \quad (11)$$

220 where β_i^* are weights to be determined.

221 *Proof.* To prove this, we consider separately the two classes of kernels: projective and radial kernels (see
 222 Table 1). Using the expression of the gradient (10), we have at the optimum $\nabla_{\mathbf{x}} J(\mathbf{x}^*) = 0$, namely

$$\sum_{i=1}^n \gamma_i \frac{\partial \kappa(\mathbf{x}_i, \mathbf{x}^*)}{\partial \mathbf{x}^*} = \frac{1}{2} \frac{\partial \kappa(\mathbf{x}^*, \mathbf{x}^*)}{\partial \mathbf{x}^*}. \quad (12)$$

223 Let us begin with the projective kernels, of the form (2). Thus, the left-hand-side of this equation can
 224 be written as

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

225 and its right-hand-side can be expressed as

$$\frac{1}{2} \frac{\partial \kappa(\mathbf{x}^*, \mathbf{x}^*)}{\partial \mathbf{x}^*} = \frac{1}{2} \frac{\partial f(\mathbf{x}^* \cdot \mathbf{x}^*)}{\partial (\mathbf{x}^* \cdot \mathbf{x}^*)} 2\mathbf{x}^*.$$

226 Combining both expressions, the equation (12) becomes

$$\mathbf{x}^* = \sum_{i=1}^n \gamma_i \frac{f^{(1)}(\mathbf{x}_i \cdot \mathbf{x}^*)}{f^{(1)}(\mathbf{x}^* \cdot \mathbf{x}^*)} \mathbf{x}_i, \quad (13)$$

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

228 We now study the class of radial kernels, defined by expression (3). In such case, the term $\partial \kappa(\mathbf{x}, \mathbf{x}) / \partial \mathbf{x}$
 229 vanishes. The gradient at the optimum, (12), can be written as

$$\sum_{i=1}^n \gamma_i \frac{\partial \kappa(\mathbf{x}_i, \mathbf{x}^*)}{\partial \mathbf{x}^*} = 0,$$

230 with the left-hand-side given as

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

231 The final result of (12) can thus be expressed as

$$\mathbf{x}^* = \sum_{i=1}^n \gamma_i \frac{g^{(1)}(\|\mathbf{x}_i - \mathbf{x}^*\|^2)}{\sum_{j=1}^n \gamma_j g^{(1)}(\|\mathbf{x}_j - \mathbf{x}^*\|^2)} \mathbf{x}_i, \quad (14)$$

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

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

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

238 *Proof.* For the projective kernels, we have from (13):

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

239 When all input data are non-negative, the above derivatives are non-negative due to Proposition 2. The
 240 same proof can be applied for the radial kernels by applying Proposition 1 to (14) with

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

241 □

242 The above results are based on the first derivative of the cost function (9). Its second derivative provides
 243 a deeper insight on its convexity, as derived in the following theorem.

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

246 *Proof.* Taking the second derivative of the cost function (9) with respect to \mathbf{x} , we get

$$\begin{aligned} \nabla_{\mathbf{x}}^2 J(\mathbf{x}) &= \nabla_{\mathbf{x}} \left[2 \sum_{i=1}^n \gamma_i (\mathbf{x}_i - \mathbf{x}) g^{(1)}(\|\mathbf{x}_i - \mathbf{x}\|^2) \right] \\ &= 2 \sum_{i=1}^n \gamma_i \left(-g^{(1)}(\|\mathbf{x}_i - \mathbf{x}\|^2) + 2(\mathbf{x}_i - \mathbf{x})^2 g^{(2)}(\|\mathbf{x}_i - \mathbf{x}\|^2) \right). \end{aligned}$$

247 The term between parentheses is positive, due to Proposition 1. Therefore, a sufficient condition for the
 248 second derivative to be non-negative, and thus for the convexity of (9), is that all the coefficients γ_i are
 249 non-negative. □

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

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

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

256 thus

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

257 From expression (14), we get the fixed-point iterative method for the Gaussian kernel

$$\mathbf{x}^* = \frac{\sum_{i=1}^n \gamma_i \kappa_G(\mathbf{x}_i, \mathbf{x}^*) \mathbf{x}_i}{\sum_{i=1}^n \gamma_i \kappa_G(\mathbf{x}_i, \mathbf{x}^*)}. \quad (15)$$

258 When the polynomial kernel is applied, with

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

259 then

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

260 where $\kappa_{p-1}(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i \cdot \mathbf{x}_j) = (c + \mathbf{x}_i \cdot \mathbf{x}_j)^{p-1}$. From expression (13), we get the fixed-point iterative
261 method for the polynomial kernel

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

262 Table 3 (second column) illustrates the diversity of the gradient expressions for different kernels. Such
263 fixed-point iterative algorithm suffers from instabilities and even may not converge at all, as illustrated in
264 [22] where only the Gaussian kernel was used. Moreover, results widely vary for different starting points in
265 practice. These issues are likely due to two factor: First, the absence of a stepsize parameter, which allows
266 to control the convergence of the algorithm. Second, the unconstrained solution, as the hypothesis space
267 corresponds to the whole input space. Both issues will be addressed in next section.

268 4. The pre-image under non-negativity constraints

269 In many applications in pattern recognition, one seeks non-negativity in the solution. In image processing
270 for instance, training data are images or patches within an image, i.e., data which are non-negative for gray-
271 level images. To get a feature extracted or a denoised version of the same type (same input space with
272 non-negativity of each pixel), one should impose non-negativity constraints on the pre-image. However, the
273 constraints are applied either on the data itself, or on the weights model using the linear combination of
274 (11).

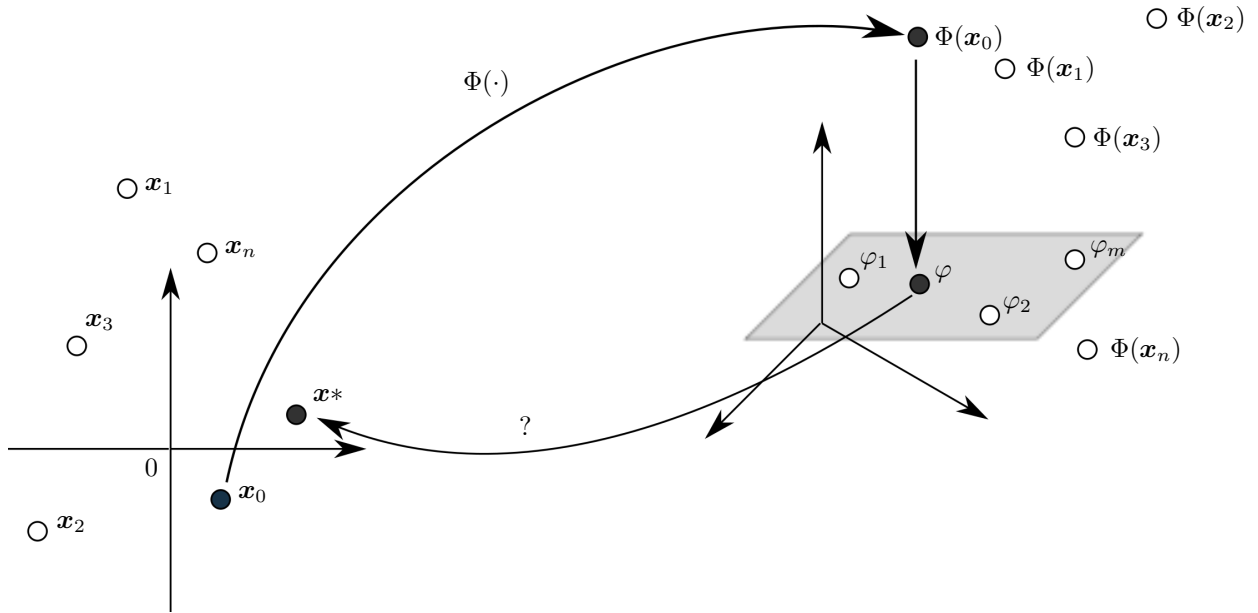


Figure 2: Schematic illustration of the pre-image problem under the non-negativity constraints. A given noisy data \mathbf{x}_0 is mapped into $\Phi(\mathbf{x}_0)$, then projected into the subspace spanned by the most relevant principal axes $\varphi_1, \varphi_2, \dots, \varphi_m$. The denoised pattern φ is mapped back to the input space, into \mathbf{x}^* .

275 *4.1. Non-negativity constraint on the data itself*

276 In this section, we consider the general problem of solving the pre-image problem under non-negativity
 277 constraint. With the cost function $J(\cdot)$ defined in (9), we study the constrained optimization problem

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} J(\mathbf{x}) \quad \text{subject to } \mathbf{x} \geq 0, \quad (16)$$

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

281 A general form of the pre-image problem under non-negativity constraints is defined in (16). Consider
 282 the Lagrangian function associated to this constrained optimization problem, with³

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

283 where $\boldsymbol{\mu}$ represents the vector of non-negative Lagrange multipliers. At the optimum solution \mathbf{x}^* , corre-
 284 sponding to the optimal multiplier vector $\boldsymbol{\mu}^*$, the first-order (Karush-)Kuhn-Tucker optimality conditions

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

285 are satisfied, with

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

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

290 In order to solve this equation, we consider an iterative scheme. The updating expression at iteration
291 $t + 1$ of all $x_i(t + 1)$ from previous $x_i(t)$ is given by

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

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

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

295 and thus this translates into a condition on the non-negativity of $1 + \eta_i(t) [-\nabla_{\mathbf{x}}J(\mathbf{x}(t))]_i$. Two cases can be
296 distinguished: If $[\nabla_{\mathbf{x}}J(\mathbf{x}(t))]_i \leq 0$, no restriction is applied on the value of the stepsize; Otherwise, when
297 $[\nabla_{\mathbf{x}}J(\mathbf{x}(t))]_i > 0$, then we have to crop the value of the stepsize such that

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

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

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

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

$$\mathbf{x}(t + 1) = \mathbf{x}(t) - \eta(t) \text{diag}[\mathbf{x}(t)] \nabla_{\mathbf{x}}J(\mathbf{x}(t)), \quad (17)$$

Table 4: Values of the parameters for the three datasets.

		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

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

4.2. Non-negativity constraint on the model weights

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

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

where $\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_n]^T$ 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

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

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

The corresponding cost function (9) can be written as

$$J(\boldsymbol{\beta}^T \mathbf{X}) = - \sum_{i=1}^n \gamma_i \kappa(\mathbf{x}_i, \boldsymbol{\beta}^T \mathbf{X}) + \frac{1}{2} \kappa(\boldsymbol{\beta}^T \mathbf{X}, \boldsymbol{\beta}^T \mathbf{X}). \quad (18)$$

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

$$\nabla_{\boldsymbol{\beta}} J(\boldsymbol{\beta}^T \mathbf{X}) = \nabla_{\mathbf{x}} J(\mathbf{x}) \mathbf{X}^T, \quad (19)$$

312 where $\mathbf{x} = \boldsymbol{\beta}^T \mathbf{X}$. Table 3 (third column) gives the gradient with respect to $\boldsymbol{\beta}$ of the most commonly used
 313 kernels. The relationship between these expressions and the gradient with respect to \mathbf{x} (second column in
 314 Table 3) is given in expression (19).

315 By deriving this analogy with the constrained optimization problem (16) (non-negativity on the pre-
 316 image), we revisit the latter in order to impose the non-negativity on the weights $\boldsymbol{\beta}^*$ in the expansion
 317 $\mathbf{x}^* = \boldsymbol{\beta}^{*T} \mathbf{X}$. This yields the following optimization problem

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

318 In this expression $J(\cdot)$ is defined as in (18), with its gradient with respect to $\boldsymbol{\beta}$ given in (19). From (17),
 319 the updating rule of these weights is given as

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

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

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

330 5. Experiments

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

337 extraction from real signals. Signals are based on the event-related potentials of the brain activity from the
338 electroencephalograph.

339 5.1. Artificial datasets: Denoising scheme

340 Let us start with the artificial datasets. For illustration purpose, we consider a two-dimensional space,
341 and apply the denoising scheme separately on three different shapes⁴: a banana, a donut and a frame. For
342 each example, a set of n samples, given in Figure 3 (upper row), was generated to learn the m eigenvectors.
343 With its quadratic form, we set $m = 2$ for the banana dataset, while $m = 4$ for the more complicated
344 shape of the frame. Another set of N samples was generated from the same distributions, as given by the
345 (very small) blue dots in Figure 3. Values of the parameters used for each dataset are given in Table 4.
346 It is obvious that these nonlinear shapes cannot be denoised properly using a linear approach, such as the
347 conventional PCA.

348 First, we compared the non-negative pre-image approach with other unconstrained techniques, including
349 the fixed-point technique defined by (15) and the regularized pre-image estimation [24]. To this end, we
350 considered a setting where all algorithms should give comparable results: all the samples are non-negative.
351 This was done by translating the samples into the positive quadrant, as illustrated in Figure 3 (upper row).
352 For all these algorithms, the noisy version of the data was used at initialization, i.e., $\mathbf{x}(t)$ for $t = 0$, given by
353 (very small) blue dots in Figure 3. With the number of iterations fixed to $t_{\max} = 20$ for iterative algorithms,
354 the denoised samples obtained by these pre-image techniques are represented by red dots. The trajectories
355 obtained at each iteration are represented by green lines (except for the regularized pre-image estimation
356 which is not an iterative technique). As we can see with the length of these lines, the fixed-point algorithm
357 (second row) has slower convergence as opposed to the proposed approach (last row). This is mainly due
358 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
359 consideration optimized stepsize values, either with an optimal value for each dataset using a line search
360 technique, or with a stepsize value decreasing at each iteration, i.e., $\eta(t + 1) < \eta(t)$. These optimization
361 schema are beyond the scope of this paper. It is worth noting that the results obtained from the regularized
362 pre-image estimation show that one may get into local minima.

363 We turn now to the approach where the weights are constrained to be non-negative. No restrictions
364 on the data were required in this case, and thus no translation was operated as given above, with samples

⁴The 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. The data were uniformly randomly drawn within these lines and corrupted by a uniformly distributed noise on $[-\nu, \nu]$, with $\nu = 0.2$.

365 having positive as well as negative values. We compared three types of kernels: the Gaussian kernel with
 366 bandwidth $\sigma = 0.7$ as above, the polynomial quadratic kernel with $p = 2$ and $c = 1$, and the exponential
 367 kernel with $\sigma = 1$ (see Table 1). For all these kernels, the initial value for β given a noisy data \mathbf{x}_0 was set
 368 to the solution of $\mathbf{x}_0 = \beta^T \mathbf{X}$ by retaining only non-negative weights, namely using a pseudo-inverse with

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

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

376 Now, we turn to the analysis of the model weights, and the sparsity of the solution. To this end, we
 377 consider the distribution of the weights $\beta_1, \beta_2, \dots, \beta_n$ for each of the $N = 200$ noisy samples. Figure 5
 378 shows the histogram of such distribution, where each color in the colorbar corresponds to a denoised sample.
 379 While we represent here the results of a single iteration, similar results are obtained for larger number of
 380 iterations. These results illustrate the fact that the weights are non-negative as expected, lying between 0
 381 and 0.018. Moreover, most of them are close to zero, namely below 0.002. This is the property of sparsity,
 382 well established and often required by a large class of algorithms in machine learning community.

383 5.2. Denoising images

384 We applied the denoising scheme on real handwritten digits, taken from the MNIST database⁵. From
 385 the dataset, we have chosen images of the digit “0”. Each image is defined by 28×28 gray-level pixels,
 386 i.e., pixels have values between 0 and 255. Thus, each image can be written as a 784-dimensional vector.
 387 The images were corrupted by adding a Salt-and-Pepper noise, with 0.1 density. The images were denoised
 388 under the non-negativity of the data, as defined by (16) (see also [14]).

389 The relevance of the proposed method is now demonstrated for image denoising, and compared to different
 390 techniques: the fixed-point iterative method [22], the multi-dimensional scaling method [23], the regularized
 391 pre-image estimation [24], and the penalized pre-image [28]. For this purpose, we used the Gaussian kernel

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

392 with a bandwidth set to $\sigma = 500$, fixed for all pre-image techniques. A set of 500 images was used to train
393 the kernel PCA with 50 eigenvectors. Another set of ten images, shown in Figure 6 (first row), corrupted
394 by the same noise settings, was used for denoising (second row). Different techniques are applied in order
395 to denoise the digits. As we can see in Figure 6, the proposed method presents the best denoising results
396 among all the others.

397 *5.3. Feature extraction*

398 We considered feature extraction with an application to real signals, and more specifically recordings
399 measuring brain activity. The feature extraction under non-negativity is the constraint applied on the
400 weights of the model [13]. Event-related potentials (ERP) refer to the electrical activity in the brain due to
401 a response to a specific stimulus, measured with electroencephalograph (EEG). There is a strong consensus
402 on the components of an ERP recoding, independent of either the participants or the stimulus type. Such
403 signal includes a negative wave deflection (called N200 or N2) followed by a positive one (called P300 or
404 P3), occurring respectively around 200 ms and 300 ms after stimulus onset. Within the brain activity,
405 such a single response is not usually visible in these recordings. To circumvent this, many trials are often
406 performed using the same stimulus. In practice, one takes the average of these responses, which gives a
407 first-order moment statistic of the ERP recordings. In this paper, we give another statistic taking into
408 account the variance of these signals, by combining kernel PCA with the Gaussian kernel on the one hand,
409 and the proposed pre-image technique on the other.

410 For experimentations, we used the ERP signals available here⁶; for more information, see also [37, 38].
411 The auditory stimulus is composed of a series of two alternating tone signals, randomly played with a time
412 between stimuli (also called Inter-Stimulus Interval or ISI) of one second. These stimuli correspond to either
413 a tone at the frequency 800 Hz or another tone at 560 Hz, played within the ratio 85% of the first signal
414 and 15% of the second one. The ERP signals are recordings from a 64-channel EEG, where only the midline
415 central channel Cz is used for its high reliability in potential detection. The recording captured within the
416 Cz channel are segmented into signals in order to view the reaction of the subject to the stimulus by using
417 a window $[0, 600]$ ms, where 0 corresponds to the instance of onset stimulus. Such window is appropriate to
418 extract both N200 and P300 components of the ERP. A set of 87 signals of length 600 ms is collected, with
419 151 samples each, as illustrated in Figure 7 where only ten randomly selected signals are shown to display
420 the variety of these signals.

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

421 We applied the kernel PCA to extract the first principal axe of these data, in the feature space associated
 422 with the Gaussian kernel. The pre-image approach allowed us to go back to the initial space, that is, the
 423 signal space. Because signals have negative components⁷, we applied the pre-image technique with the
 424 non-negative constraints on the weights. Following some preliminary experimentations, the Gaussian kernel
 425 was used, with the bandwidth set to $\sigma = 500$, and the stepsize value to the value $\eta = 0.1$. Next, we study
 426 the influence of the initialization on the algorithm, based on two different initializations.

427 First, the algorithm was initialized using a random input data, namely \mathbf{x}_1 without loss of generality and
 428 shown in Figure 8 (upper figure). Then $\boldsymbol{\beta}(0) = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{x}_0^T$ for non-negative values, and zero otherwise.
 429 Applying the algorithm for $t = 100$ iterations gave the feature illustrated in Figure 8 (lower figure). We
 430 can easily see both important components of the ERP, the N200 and P300 waves. Moreover, variations of
 431 features of interest are opposed to the highly fluctuating initial signal.

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

441 6. Conclusion and future work

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

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

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

449 **7. References**

- 450 [1] G. Chen and B. Kégl, “Image denoising with complex ridgelets,” *Pattern Recognition*, vol. 40, no. 2, pp. 578 – 585, 2007.
- 451 [2] W.-S. Zheng, J. Lai, S. Liao, and R. He, “Extracting non-negative basis images using pixel dispersion penalty,” *Pattern*
- 452 *Recognition*, no. 0, pp. –, 2012.
- 453 [3] C. Twining and C. Taylor, “The use of kernel principal component analysis to model data distributions,” *Pattern Recog-*
- 454 *nition*, vol. 36, no. 1, pp. 217 – 227, 2003.
- 455 [4] Z. Yang and E. Oja, “Quadratic nonnegative matrix factorization,” *Pattern Recognition*, vol. 45, no. 4, pp. 1500 – 1510,
- 456 2012.
- 457 [5] G. Thomas, “A positive optimal deconvolution procedure,” *IEEE International Conference on Acoustics, Speech, and*
- 458 *Signal Processing, ICASSP*, vol. 8, pp. 651 – 654, April 1983.
- 459 [6] R. Prost and R. Goutte, “Discrete constrained iterative deconvolution algorithms with optimized rate of convergence,”
- 460 *Signal Processing*, vol. 7, no. 3, pp. 209–230, Dec. 1984.
- 461 [7] G. Thomas and N. Souilah, “Utilisation des multiplicateurs de lagrange pour la restauration d’image avec contraintes,”
- 462 *Colloques sur le Traitement du Signal et des Images*, 1991.
- 463 [8] D. Snyder, T. Schulz, and J. O’Sullivan, “Deblurring subject to nonnegativity constraints,” *IEEE Transactions on Signal*
- 464 *Processing*, vol. 40, pp. 1143 – 1150, May 1992.
- 465 [9] H. Lantéri, M. Roche, O. Cuevas, and C. Aime, “A general method to devise maximum-likelihood signal restoration
- 466 multiplicative algorithms with non-negativity constraints,” *Signal Processing*, vol. 81, no. 5, pp. 945–974, May 2001.
- 467 [10] J. Chen, C. Richard, P. Honeine, H. Snoussi, H. Lantéri, and C. Theys, “Techniques d’apprentissage non-linéaires en
- 468 ligne avec contraintes de positivité,” in *Actes de la VI^{ème} Conférence Internationale Francophone d’Automatique*, Nancy,
- 469 France, 2 - 4 Juin 2010.
- 470 [11] J. Chen, C. Richard, P. Honeine, H. Lantéri, and C. Theys, “System identification under non-negativity constraints,” in
- 471 *Proc. of European Conference on Signal Processing (EUSIPCO)*. Aalborg, Denmark: Eurasip, 2010.
- 472 [12] J. Chen, C. Richard, P. Honeine, and J. C. M. Bermudez, “Non-negative distributed regression for data inference in
- 473 wireless sensor networks,” in *Proc. of the 44th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove
- 474 (CA), USA, 2010.
- 475 [13] M. Kallas, P. Honeine, C. Richard, H. Amoud, and C. Francis, “Nonlinear feature extraction using kernel principal com-
- 476 ponent analysis with non-negative pre-image,” in *Proc. 32nd Annual International Conference of the IEEE Engineering*
- 477 *in Medicine and Biology Society (EMBC)*, Buenos Aires, Argentina, 31 Aug. - 4 Sept. 2010.
- 478 [14] M. Kallas, P. Honeine, C. Richard, C. Francis, and H. Amoud, “Non-negative pre-image in machine learning for pattern
- 479 recognition,” in *19th European Signal Processing Conference*, Barcelona, Spain, 29 August - 2 September 2011 Submitted.
- 480 [15] M. Kallas, H. Amoud, P. Honeine, and C. Francis, “Sur le problème de la pré-image en reconnaissance des formes avec
- 481 contraintes de non-négativité,” in *Colloque GRETSI’2011*, Bordeaux, France, 5-8 Septembre 2011 Soumis.
- 482 [16] H. Han, “Nonnegative principal component analysis for mass spectral serum profiles and biomarker discovery,” *BMC*
- 483 *Bioinformatics*, 2010.
- 484 [17] R. Zass and A. Shashua, “Nonnegative Sparse PCA,” in *Neural Information Processing Systems*, 2007.
- 485 [18] B. Moghaddam, Y. Weiss, and S. Avidan, “Spectral bounds for sparse pca: Exact and greedy algorithms,” in *Advances*
- 486 *in Neural Information Processing Systems*. MIT Press, 2006, pp. 915–922.
- 487 [19] C. D. Sigg and J. M. Buhmann, “Expectation-maximization for sparse and non-negative pca,” in *25th International*
- 488 *Conference on Machine Learning (ICML)*. ACM, 2008.
- 489 [20] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, September 1998.
- 490 [21] B. Schölkopf, A. Smola, and K.-R. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” *Neural Com-*
- 491 *put.*, vol. 10, pp. 1299–1319, July 1998.
- 492 [22] S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, “Kernel PCA and de-noising in feature spaces,”
- 493 in *Proc. of the 1998 conference on advances in neural information processing systems II*. Cambridge, MA, USA: MIT
- 494 Press, 1999, pp. 536–542.
- 495 [23] J. T. Kwok and I. W. Tsang, “The pre-image problem in kernel methods,” in *Proc. of the Twentieth International*
- 496 *Conference on Machine Learning*, T. Fawcett and N. Mishra, Eds. AAAI Press, 2003, pp. 408–415.
- 497 [24] T. J. Abrahamsen and L. K. Hansen, “Regularized pre-image estimation for kernel pca de-noising: Input space regular-
- 498 ization and sparse reconstruction,” *Journal of Signal Processing Systems*, vol. 65, no. 3, pp. 403–412, 2011.
- 499 [25] P. Honeine and C. Richard, “Solving the pre-image problem in kernel machines: a direct method,” in *Proc. IEEE Workshop*
- 500 *on Machine Learning for Signal Processing (MLSP)*, Grenoble, France, September 2009.
- 501 [26] —, “A closed-form solution for the pre-image problem in kernel-based machines,” *Journal of Signal Processing Systems*,
- 502 vol. 65, pp. 289–299, December 2011.
- 503 [27] —, “Preimage problem in kernel-based machine learning,” *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 77–88,
- 504 2011.
- 505 [28] W. Zheng, J. Lai, and P. C. Yuen, “Penalized preimage learning in kernel principal component analysis,” *IEEE Transaction*
- 506 *Neural Networks*, vol. 21, pp. 551–570, April 2010.
- 507 [29] E. J. Candes and M. B. Wakin, “An introduction to compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25,
- 508 no. 2, pp. 21–30, March 2008.
- 509 [30] N. Aronszajn, “Theory of reproducing kernels,” *Trans. Amer. Math. Soc.*, vol. 68, pp. 337–404, 1950.
- 510 [31] F. Cucker and S. Smale, “On the mathematical foundations of learning,” *Bulletin of the American Mathematical Society*,
- 511 vol. 39, pp. 1–49, 2002.

- 512 [32] C. J. C. Burges, "Geometry and invariance in kernel based methods," in *Advances in kernel methods*, B. Schölkopf, C. J. C.
513 Burges, and A. J. Smola, Eds. Cambridge, MA, USA: MIT Press, 1999, pp. 89–116.
- 514 [33] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Com-*
515 *put.*, vol. 10, no. 5, pp. 1299–1319, 1998.
- 516 [34] R. Rosipal, M. Girolami, and L. J. Trejo, "Kernel PCA for feature extraction and de-noising in non-linear regression,"
517 *Neural Computing and Applications*, vol. 10, pp. 231–243, 2000.
- 518 [35] G. S. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *Journal of Mathematical Analysis and*
519 *Applications*, vol. 33, no. 1, pp. 82–95, 1971.
- 520 [36] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Proc. of the 14th Annual Confer-*
521 *ence on Computational Learning Theory and 5th European Conference on Computational Learning Theory (COLT*
522 *'01/EuroCOLT '01)*. London, UK: Springer-Verlag, 2001, pp. 416–426.
- 523 [37] S. D. Georgiadis, "State-space modeling and bayesian methods for evoked potential estimation," Ph.D. dissertation,
524 Department of Applied Physics, University of Kuopio, Finland, May 2007.
- 525 [38] M. P. Tarvainen, "Estimation methods for nonstationary biosignals," Ph.D. dissertation, Department of Applied Physics,
526 University of Kuopio, Finland, June 2004.

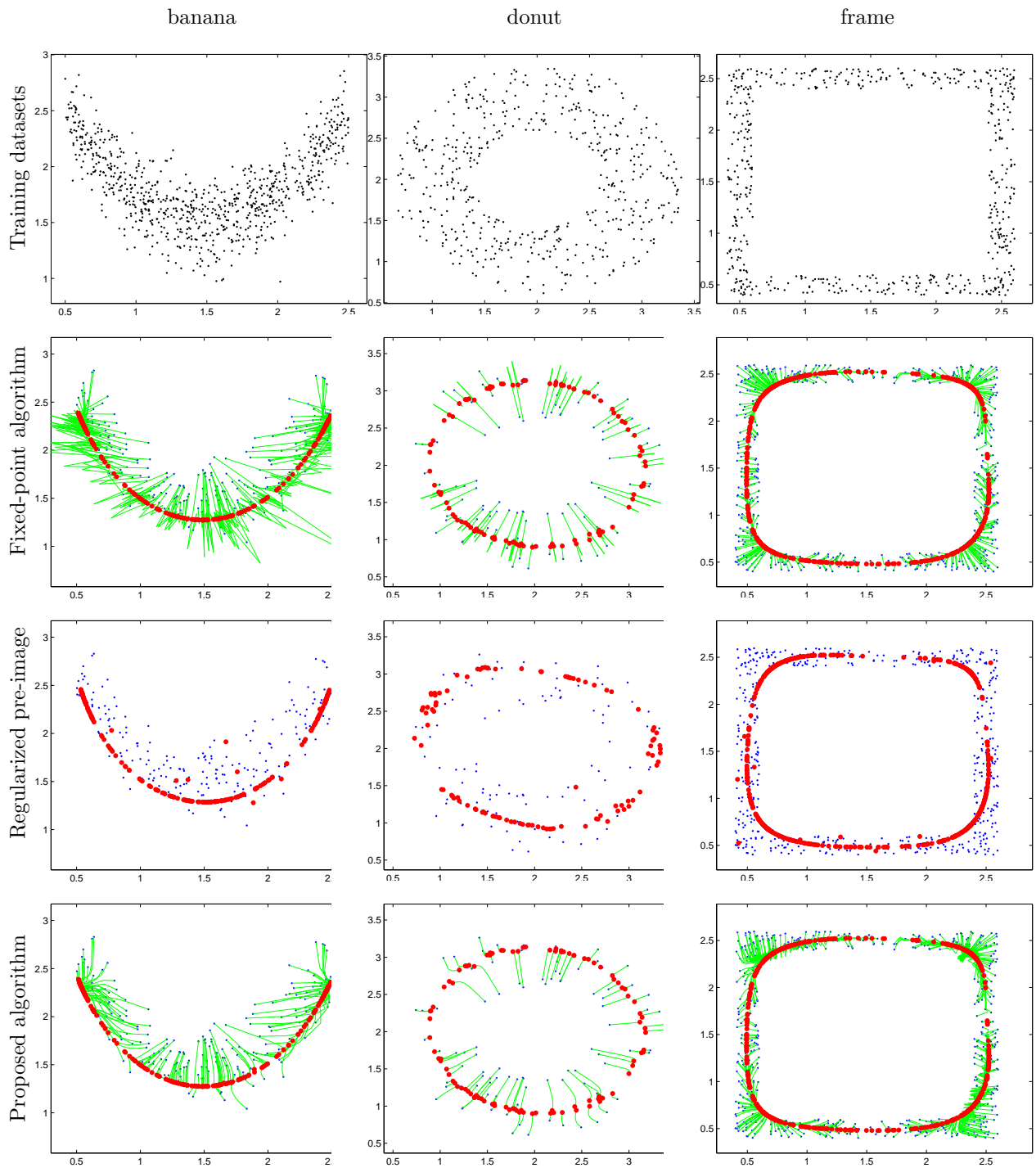


Figure 3: Denoising artificial datasets, for the three shapes: banana (left column), donut (middle column) and frame (right column). A set of training data (∇ in upper row) is used for constructing the relevant subspace using the kernel PCA with the Gaussian kernel. Another set of data (designated by \cdot) is denoised (into \bullet) 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 ---).

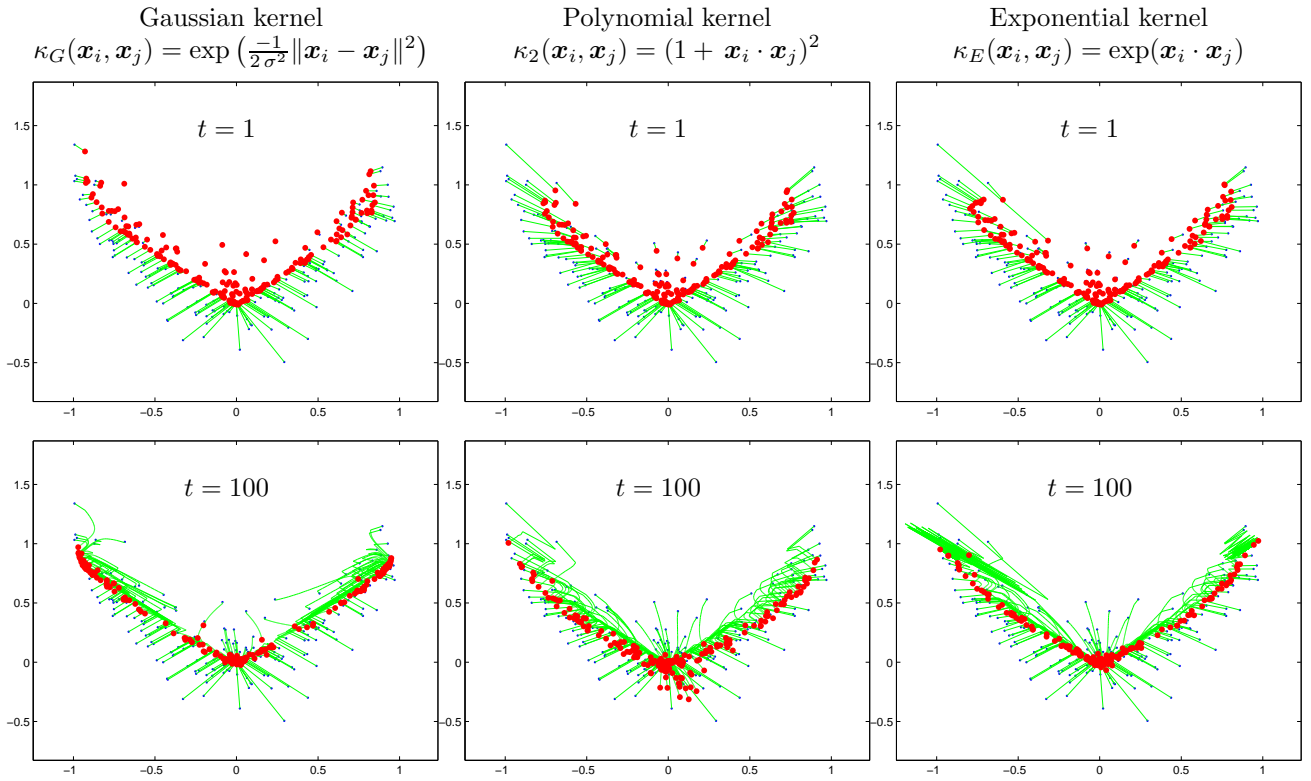


Figure 4: Denoising with constraints on the model weights, of the banana dataset for a single iteration (upper row), and after $t = 100$ iterations (lower row). Three different kernels are compared: Gaussian (left column), polynomial (middle column), and exponential (right column) kernels.

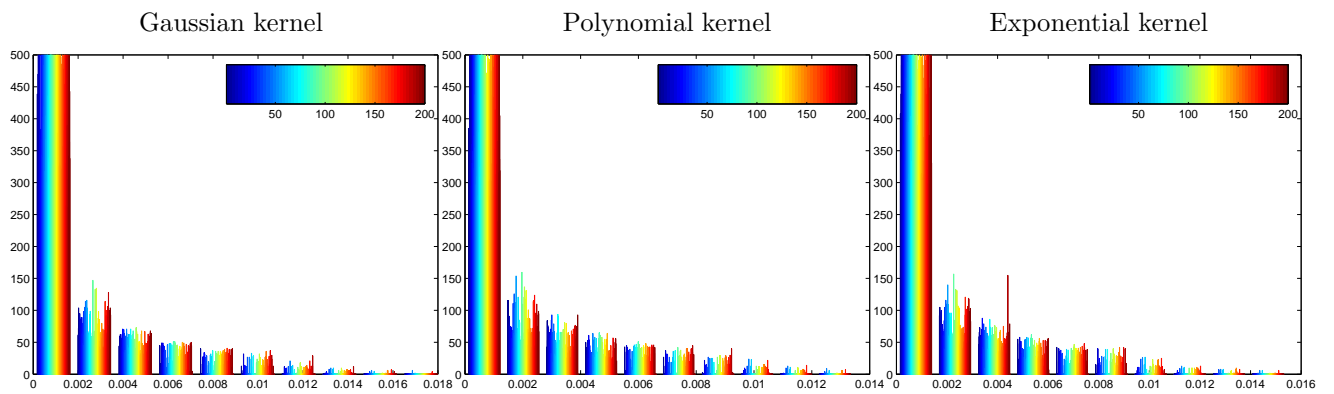


Figure 5: Distribution of the model weights for each of the 200 noisy data from the banana dataset, after only one iteration of our algorithm, corresponding to results given in Figure 4 (upper row). All denoised data (each represented by a color within the colorbar) enjoy the sparsity property, with a large number of weights close to zero.

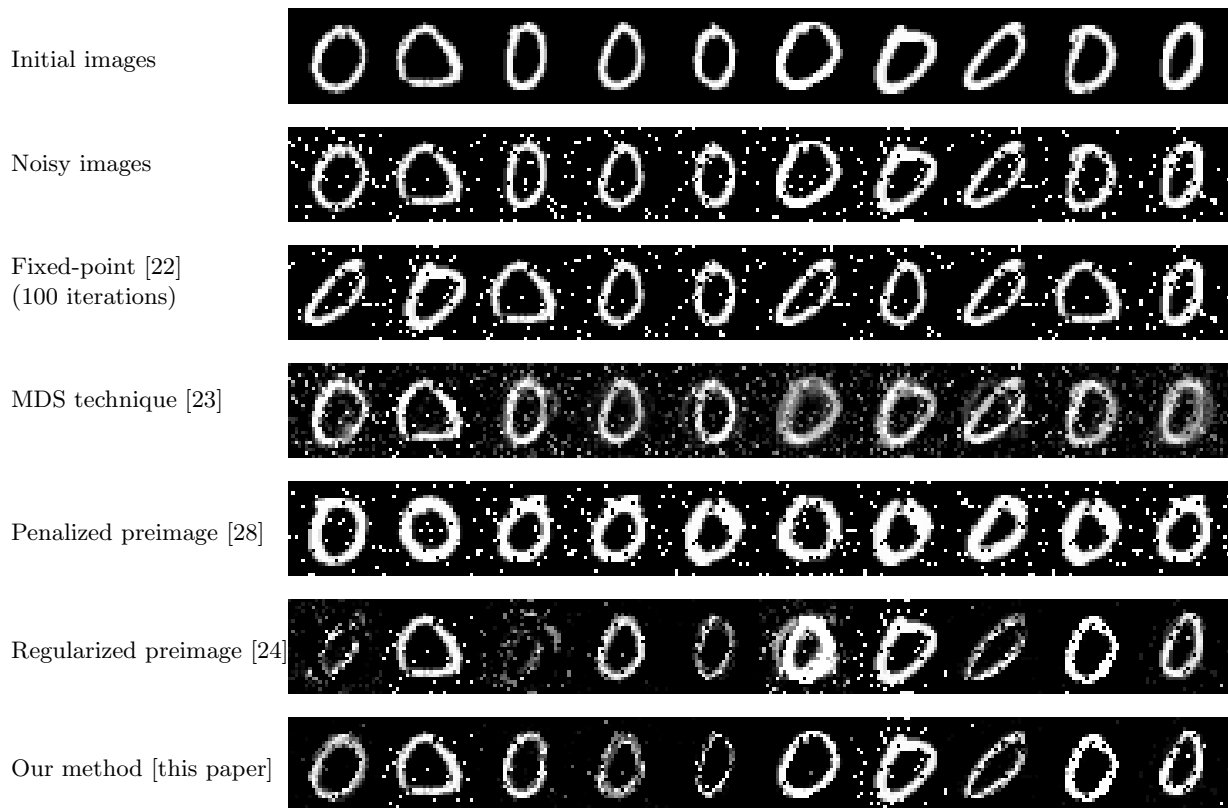


Figure 6: A set of ten “0”-digit images (first row) corrupted by a salt-and-pepper noise of density 0.1 (second row), on which we applied the kernel PCA for data denoising. The pre-image results using the fixed-point iterative algorithm [22] are illustrated (third row), the MDS technique [23] (fourth row), the penalized preimage learning method [28] (fifth row), the regularized preimage estimation technique [24] (sixth row), and the non-negative pre-image with the iterative schema (17) (last row).

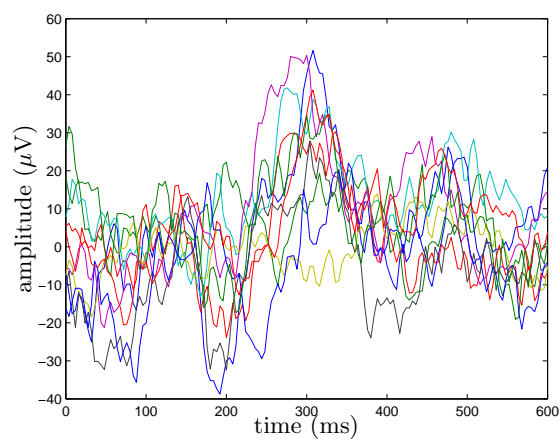


Figure 7: Some ERP signals recorded from the Cz channel. The diversity of these signals is shown, with some signals not having a positive component around 300 ms (see for instance —).

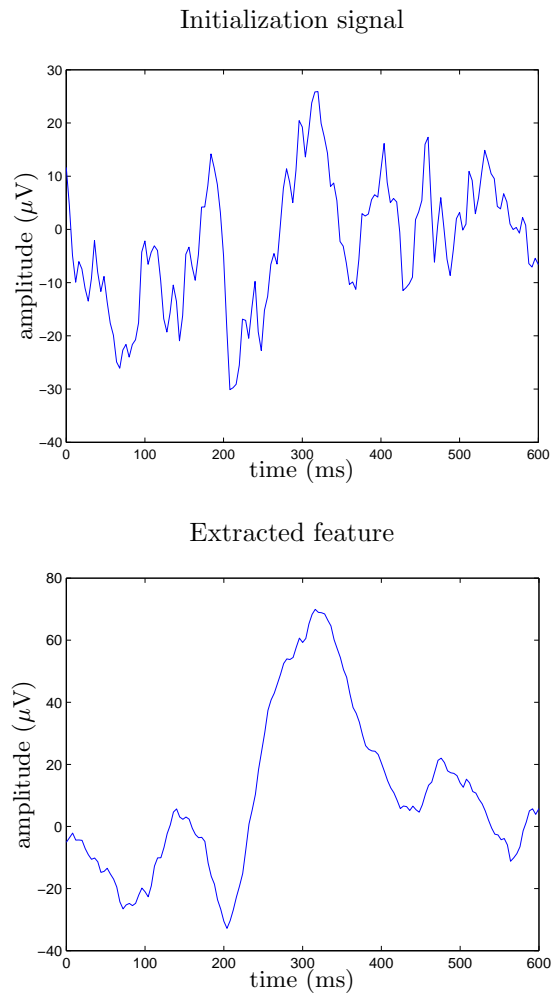


Figure 8: Feature extraction of the ERP data, with the algorithm initialized to the initial signal (upper figure). By pre-imaging the first principal axe of kernel PCA, we get the feature (lower figure).

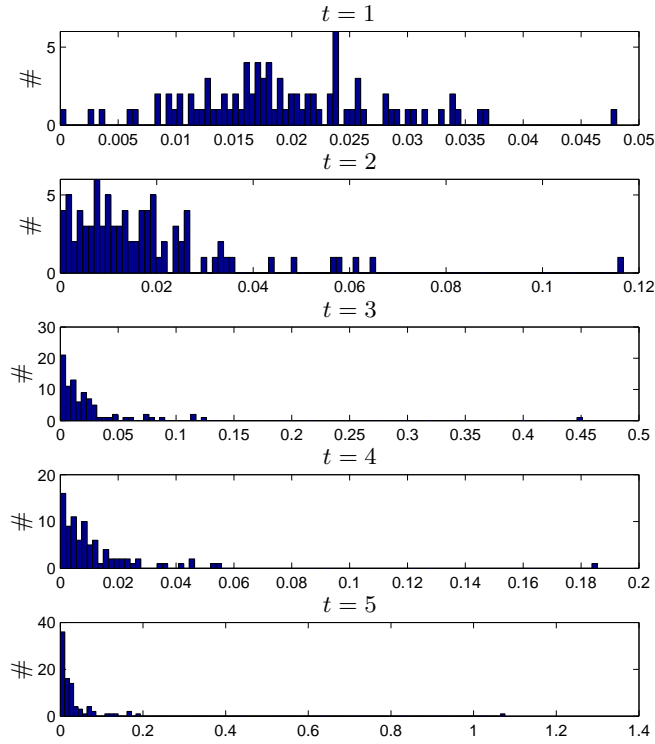


Figure 9: Distribution of the model weights from the first iteration (upper figure) to the fifth iteration (lower figure). This illustrates the evolution of the weights towards a sparse distribution.

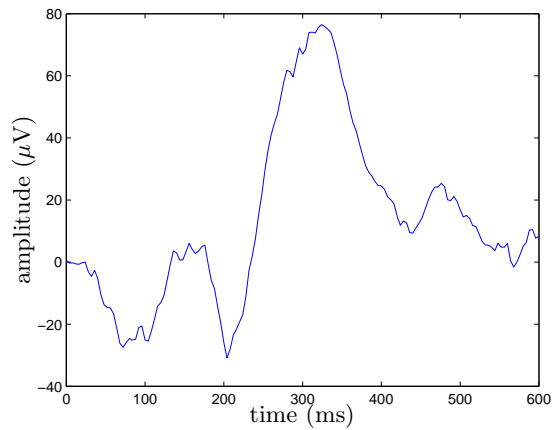


Figure 10: Feature extraction of the ERP data, with the algorithm initialized to the uniform contribution of all available signals, corresponding to $t = 5$ in Figure 9 (lower figure).

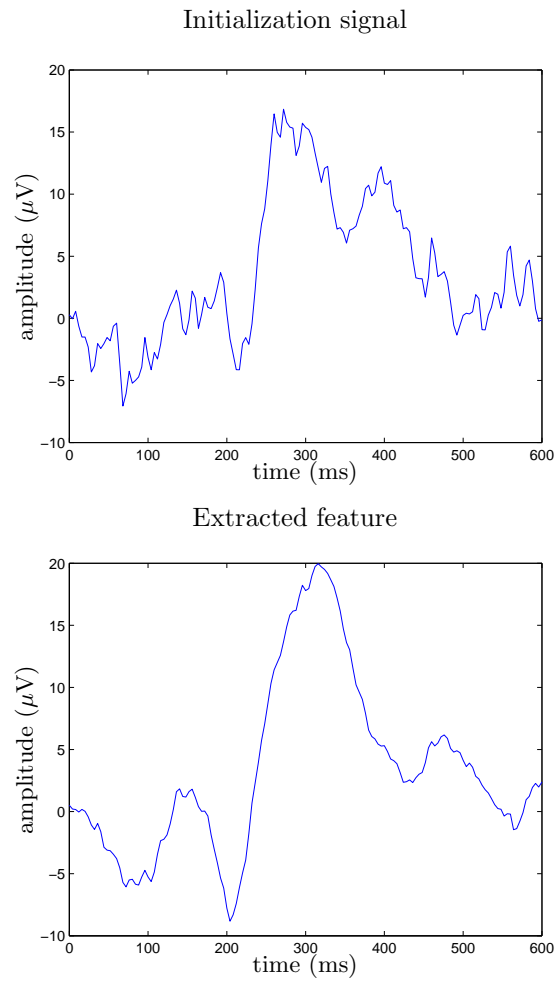


Figure 11: The average of 10 signals (upper figure) and the average of all signals (lower figure) of the ERP data.