Interpretable time series kernel analytics by pre-image estimation

Thi Phuong Thao Tran, Ahlame Douzal-Chouakria, Saeed Varasteh Yazdi, Paul Honeine, Patrick Gallinari

Interpretable time series kernel analytics by pre-image estimation

Thao Tran Thi Phuong, Ahlame Douzal, Saeed Varasteh Yazdi,
Paul Honeine, and Patrick Gallinari

Abstract

Kernel methods are known to be effective to analyse complex objects by implicitly embedding them into some feature space. To interpret and analyse the obtained results, it is often required to restore in the input space the results obtained in the feature space, by using pre-image estimation methods. This work proposes a new closed-form pre-image estimation method for time series kernel analytics that consists of two steps. In the first step, a time warp function, driven by distance constraints in the feature space, is defined to embed time series in a metric space where analytics can be performed conveniently. In the second step, the time series pre-image estimation is cast as learning a linear (or a nonlinear) transformation that ensures a local isometry between the time series embedding space and the feature space. The proposed method is compared to the state of the art through three major tasks that require pre-image estimation: 1) time series averaging, 2) time series reconstruction and denoising and 3) time series representation learning. The extensive experiments conducted on 33 publicly-available datasets show the benefits of the pre-image estimation for time series kernel analytics.

1. Introduction

Kernel methods [24] are well known to be effective in dealing with nonlinear machine learning problems in general, and are often required for machine learning tasks on complex data as sequences, time series or graphs. The main idea behind kernel machines is to map the data from the input space to a higher dimension feature space (i.e., kernel space) via a nonlinear map, where the mapped data can be then analysed by linear models. While the mapping from input space to the feature space is of primary importance in kernel methods, the reverse mapping of the obtained results from the feature space back to the input space (called the pre-image problem) is also very useful. Estimating pre-images is important in several contexts for interpretation and analysis purposes. From the beginning, it has been often considered to estimate denoised and compressed results of a kernel Principal Component Analysis (PCA). Other tasks are of great interest, since the pre-image estimation allows, for instance, to obtain the reverse mapping of the centroids of a kernel clustering, and of the atoms as well as of the sparse representations in kernel dictionary learning.

In view of the importance of the pre-image estimation issue and of its benefits in machine learning, several major propositions have been developed. First, in Mika et al. [23], the problem is formalised as a nonlinear optimisation problem and, for the particular case of the Gaussian kernel, a fixed-point iterative solution to estimate the reverse mapping is proposed. To avoid numerical instabilities of the latter approach, in Kwok et al.[18], the relationship between the distances in feature and input spaces is established for standard kernels, and then used to approximate pre-images by multidimensional scaling. In Bakir et al. [3], the pre-image estimation problem is cast as a regression problem between the input and the mapped data, the learned regression model is then used to predict pre-images. Honeine and Richard proposed in [15] an approach where the main idea is to estimate, from the mapped data, a coordinate system that ensures an isometry with the input space; this approach has the advantage to provide a closed-form solution, to be independent of the kernel nature and to involve only linear algebra. More recently, task-specific estimation has been studied, such as the resolution of the pre-image problem for nonnegative matrix factorisation in [32].

All the proposed methods for pre-image estimation are either based on optimisation schema, such as gradient descent or fixed-point iterative solution, or based on ideas borrowed from dimensionality reduction methods. In particular, these methods were developed for Euclidean input spaces, as derivations are straightforward owing to linear algebra (see [16] for a survey on the resolution of the pre-image problems in machine learning). A major challenge arises when dealing with non-Euclidean input spaces, that describe complex data as sequences, time series, manifolds or graphs. Some recent works address that pre-image problem on that challenging data. For instance, in Cloninger et al. [9] the pre-image problem is addressed for Laplacian Eigenmaps under L_1 regularisation and in Bianchi et al. [6] an encoder-decoder is used to learn a latent representation driven by the kernel similarities, where the pre-image estimation is explicitly given by the decoder side.

For temporal data, while kernel machinery has been increasingly investigated with success for time series analytics [13, 27, 31, 30, 29, 19], the pre-image problem for temporal data remains in its infancy. In addition, time series data, that involve varying delays and/or different lengths, are naturally lying in a non-Euclidean input space, making the above existing pre-image approaches for static data inapplicable. This work aims to fill this gap, by proposing a pre-image estimation approach for time series kernel analytics, that consists of two steps. In the first step, a time warp function, driven by distance constraints in the feature space, is defined to embed time series in a metric space where analytics can be performed conveniently. In the second step, the time series pre-image estimation is cast as learning a linear or a nonlinear transformation that ensures a local isometry between the time series embedding space and the feature space. The relevance of the proposed method is studied through three major tasks that require pre-image estimation: 1) time series averaging, 2) time series reconstruction and denoising under kernel PCA, and 3) time series representation learning and dictionary learning under kernel k-SVD. The benefits of

the proposed method are assessed through extensive experiments conducted on 33 publicly-available time series datasets, including univariate and multivariate time series that may include varying delays and be of the different lengths.

The main contributions of this paper are:

1. We propose a time warp function, driven by distance constraints in the feature space, that embeds time series into an Euclidean space
2. We cast the time series pre-image estimation approach as learning a linear or nonlinear transformations in the feature space
3. We propose a closed-form solution for pre-image estimation by preserving a local isometry between the temporal embedded space and the feature space
4. We conduct wide experiments to compare the proposed approach to the major alternative pre-image estimation methods under three crucial tasks: 1) time series averaging, 2) time series reconstruction and denoising and 3) time series representation and dictionary learning.

The reminder of the paper is organised as follows. Section 2 gives a brief introduction to kernel PCA and kernel k-SVD and Section 3 presents the major pre-image estimation methods. In Section 4, we formalise the pre-image estimation problem for time series and develop the proposed solution as well as the corresponding algorithm. We detail the experiments conducted and discuss the obtained results in Section 5.

2. Kernel PCA and kernel k-SVD

Kernel methods [24] rely on embedding samples $\boldsymbol{x}_i \in \mathbb{R}^d$ with $\Phi(\boldsymbol{x}_i)$ into a feature space $\mathcal{H}$, a Hilbert space of arbitrary large and possibly infinite dimension. The map function Φ needs not to be explicitly defined, since computations conducted in $\mathcal{H}$ can be carried out by a kernel function that measures the inner product in that space, namely $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_{i'}) \rangle$ for all $\boldsymbol{x}_i, \boldsymbol{x}_{i'}$. Given a set of input samples $\{\boldsymbol{x}_i\}_{i=1}^N$, $\boldsymbol{x}_i \in \mathbb{R}^d$, let K be the Gram matrix related to the kernel, namely $K_{ii'} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$. Let $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_N] \in \mathbb{R}^{d \times N}$ be the description of N input samples $\boldsymbol{x}_i \in \mathbb{R}^d$ and, with some abuse of notation, let $\Phi(X)$ be the matrix of entries $\Phi(\boldsymbol{x}_1), ..., \Phi(\boldsymbol{x}_N)$.

In the following, we describe two well-known kernel methods, kernel PCA and kernel k-SVD, as nonlinear extensions of the well-known PCA and k-SVD. While both methods estimate a linear combination for optimal reconstruction of the input samples, the former forces the orthogonality of the atoms that leads to an orthonormal basis, and the latter forces the sparsity while relaxing the orthogonality condition.

2.1. Kernel PCA

Kernel PCA extends standard PCA to find principal components that are nonlinearly related to the input variables. For that, the principal components

are rather determined in the feature space. For the sake of clarity, we assume for now that we are dealing with centred mapped data, namely $\sum_{i=1}^{N} \Phi(\boldsymbol{x}_i) = 0$. The covariance matrix in the feature space takes then the form

$$C = \frac{1}{N} \sum_{i=1}^{N} \Phi(\boldsymbol{x}_i)\, \Phi(\boldsymbol{x}_i)^T. \tag{1}$$

Similarly to standard PCA, the objective comes to find the pairs of eigenvalue $\lambda_j \geq 0$ and corresponding eigenvector $\boldsymbol{u}_j \in \mathcal{H} \setminus \mathbf{0}$ that satisfy

$$\lambda_j \boldsymbol{u}_j = C\boldsymbol{u}_j, \tag{2}$$

namely for each $\Phi(\boldsymbol{x}_i)$

$$\lambda_j \langle \boldsymbol{u}_j, \Phi(\boldsymbol{x}_i) \rangle = \langle C\boldsymbol{u}_j, \Phi(\boldsymbol{x}_i) \rangle. \tag{3}$$

As each eigenvector $\boldsymbol{u}_j$ lies in the span of $\Phi(\boldsymbol{x}_1), ..., \Phi(\boldsymbol{x}_N)$, there exist coefficients $\alpha_{ij}, ..., \alpha_{Nj}$ such that

$$\boldsymbol{u}_j = \sum_{i=1}^{N} \alpha_{ij}\, \Phi(\boldsymbol{x}_i). \tag{4}$$

From Eq. (3) and Eq. (4), and by simple developments, the problem in Eq. (2) remains to find the solution of the eigendecomposition problem:

$$\lambda_j \boldsymbol{\alpha}_j = \tfrac{1}{N} K \boldsymbol{\alpha}_j. \tag{5}$$

Let $\lambda_1 \geq ... \geq \lambda_p$ be the p highest non-zero eigenvalues of $\frac{1}{N}K$ and $\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_p$ their corresponding eigenvectors. The principal components in the feature space are then given by computing the projections $P_j(\Phi(\boldsymbol{x}))$ of the sample $\boldsymbol{x}$ onto the eigenvector $\boldsymbol{u}_j = \Phi(X)\, \boldsymbol{\alpha}_j$:

$$P_j(\Phi(\boldsymbol{x})) = \langle \boldsymbol{u}_j, \Phi(\boldsymbol{x}) \rangle = \sum_{i=1}^{N} \alpha_{ij} \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}) \rangle = \boldsymbol{k}_{\boldsymbol{x}} \boldsymbol{\alpha}_j \tag{6}$$

with $\boldsymbol{k}_{\boldsymbol{x}} = [\kappa(\boldsymbol{x}_1, \boldsymbol{x}), ..., \kappa(\boldsymbol{x}_N, \boldsymbol{x})]$. By denoting $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_p]$, the description $P(\Phi(\boldsymbol{x}))$ of $\Phi(\boldsymbol{x})$ into the subspace of the p first principal components is then

$$P(\Phi(\boldsymbol{x})) = (\boldsymbol{k}_{\boldsymbol{x}} \boldsymbol{\alpha})^T \tag{7}$$

Two considerations should be taken. First, the eigenvector expansion coefficients $\boldsymbol{\alpha_j}$ should be normalised by requiring $\lambda_j \langle \boldsymbol{\alpha_j}, \boldsymbol{\alpha_j} \rangle = 1$. Secondly, as $\Phi(\boldsymbol{x}_i)$ are assumed centred, both the Gram matrix K in Eq. (5) and $\boldsymbol{k_x}$ used in Eq. (6) and Eq. (7) need to be substituted with their centred counterparts, namely

$$\widetilde{K}_{ij} = (K - \mathbb{1}_N K - K\mathbb{1}_N + \mathbb{1}_N K \mathbb{1}_N)_{ij} \tag{8}$$

$$\widetilde{\boldsymbol{k}}_{\boldsymbol{x}} = \left(\boldsymbol{k}_{\boldsymbol{x}} - \tfrac{1}{N} \mathbf{1}_N^T K \right) (I_N - \mathbb{1}_N) \tag{9}$$

with $(\mathbb{1}_N)_{ij} = 1/N$ for all i, j, I_N the identity matrix and $\mathbf{1}_N \in \mathbb{R}^N$ the unit vector.

2.2. Kernel k-SVD

Sparse coding and dictionary learning become popular methods for a variety of tasks as feature extraction, reconstruction, denoising, compressed sensing and classification [17, 4, 5]. k-SVD [1] is among the most-known and tractable dictionary learning approach to learn a dictionary and to sparse represent the input samples as a linear combination of the dictionary atoms. When dealing with complex data, kernel k-SVD may be required to learn, in the feature space, the dictionary and the sparse representations of the mapped samples as a nonlinear combination of the dictionary atoms [28]. Let us introduce a brief description of kernel k-SVD.

Let $D = [\boldsymbol{d}_1, ..., \boldsymbol{d}_L] \in \mathbb{R}^{d \times L}$ be the dictionary composed of L atoms $\boldsymbol{d}_j \in \mathbb{R}^d$. The embedded dictionary $\Phi(D) = \Phi(X)\mathcal{B}$ is defined as a linear representation of $\Phi(X)$, since the atoms lie in the subspace spanned by the $\Phi(X)$. The kernel dictionary learning problem takes the form

$$\min_{\mathcal{B},\mathcal{A}} \|\Phi(X) - \Phi(X)\mathcal{B}\,\mathcal{A}\|_{\mathcal{F}}^2 \tag{10}$$

$$\text{s.t. } \|\boldsymbol{a}_i\|_0 \leq \tau \ \forall i = 1, ..., N, \tag{11}$$

where $\|.\|_{\mathcal{F}}$ is the Frobenius norm[1], the matrix $\mathcal{B} = [\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_L] \in \mathcal{R}^{N \times L}$ gives the representation of the embedded atoms into the base $\Phi(X)$ and $\mathcal{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_N] \in \mathbb{R}^{L \times N}$ gives the sparse representations of $\Phi(X)$, with the sparsity level τ imposed by the above constraint.

The kernel k-SVD algorithm iteratively cycles between two stages. In the first stage, the dictionary is assumed fixed with $\mathcal{B}$ known and a kernel orthogonal matching pursuit (OMP) technique [20] is deployed to estimate $\mathcal{A}$. As in standard OMP, given a sample $\boldsymbol{x}$, we select the first atom that best reconstructs $\phi(\boldsymbol{x})$; namely we select the j-th atom that maximises

$$\left(\boldsymbol{k}_{\boldsymbol{x}} - \boldsymbol{a}_{\boldsymbol{x}}^T \mathcal{B}^T K\right) \boldsymbol{\beta}_j. \tag{12}$$

The sparse codes are then updated by the projections onto the subspace of the yet selected atoms

$$\boldsymbol{a}_{\boldsymbol{x}} = \left(\mathcal{B}_\Omega^T K \mathcal{B}_\Omega\right)^{-1} \left(\boldsymbol{k}_{\boldsymbol{x}} \mathcal{B}_\Omega\right)^T, \tag{13}$$

where $\mathcal{B}_\Omega^T$ is the submatrix of $\mathcal{B}$ limited to the yet selected atoms. The procedure is reiterate until the selection of τ atoms.

Once the sparse codes $\mathcal{A}$ of the N samples estimated, the second stage of the kernel k-SVD is performed to update both $\mathcal{B}$ and $\mathcal{A}$. For that, the reconstruction

[1] $\|M\|_{\mathcal{F}} = \sqrt{\sum_{i=1}^{p} \sum_{j=1}^{q} (m_{ij})^2}$ is the Frobenius norm of the matrix $M \in \mathbb{R}^{p \times q}$ of general term m_{ij}

error is defined as

$$\min_{\mathcal{B},\mathcal{A}} \|\Phi(X) - \Phi(X) \sum_{j=1}^{L} \boldsymbol{\beta}_j \boldsymbol{a}_{j.}\|_{\mathcal{F}}^2, \tag{14}$$

with $\boldsymbol{a}_{j.} \in \mathbb{R}^N$ referencing the j-th row of $\mathcal{A}$, namely

$$\min_{\mathcal{B},\mathcal{A}} \|\Phi(X)E_k - \Phi(X)\boldsymbol{\beta}_k \boldsymbol{a}_{k.}\|_{\mathcal{F}}^2, \tag{15}$$

with $E_k = I_N - \sum_{j \neq k} \boldsymbol{\beta}_j \boldsymbol{a}_{j.}$ the error of reconstruction matrix when removing the k-th atom. An eigendecomposition is then performed to get

$$(E_k^R)^T K (E_k^R) = V \Sigma V^T, \tag{16}$$

where E_k^R is the error of reconstruction restricted to the samples that have involved the k-th atom. The sparse codes $\boldsymbol{\beta}_k$ and $\boldsymbol{a}_{k.}^R$ are updated by using the first eigenvector $\boldsymbol{v}_1$ with

$$\boldsymbol{a}_{k.}^R = \sigma_1 \boldsymbol{v}_1^T \text{ and } \boldsymbol{\beta}_k = \sigma_1^{-1} E_k^R \boldsymbol{v}_1. \tag{17}$$

3. Related works on pre-image estimation

From the representer theorem [22] any result $\boldsymbol{\varphi} \in \mathcal{H}$ obtained by some kernel method may be expressed in the form $\boldsymbol{\varphi} = \sum_{i=1}^{N} \gamma_i \Phi(\boldsymbol{x}_i)$; that is as a linear combination of the mapped training samples $\{\Phi(\boldsymbol{x}_i)\}_{i=1}^N$. In general, finding the exact pre-image $\boldsymbol{x}$ such that $\Phi(\boldsymbol{x}) = \boldsymbol{\varphi}$ is an ill-posed problem, that is often addressed by providing an approximate solution, namely by estimating $\boldsymbol{x}^*$ such that $\Phi(\boldsymbol{x}^*) \approx \boldsymbol{\varphi}$. In the following, we describe three major methods to estimate the pre-image $\boldsymbol{x}^*$ of a given $\boldsymbol{\varphi} \in \mathcal{H}$.

3.1. Pre-image estimation under distance constraints

The main idea proposed in Kwok et al. [18] is to use the distances between $\boldsymbol{\varphi}$ and $\Phi(\boldsymbol{x}_i)$ and their relation to the distances between the pre-image $\boldsymbol{x}^*$ and $\boldsymbol{x}_i$. The main steps of the proposed approach are:

1. Let $\widetilde{d}^2(\boldsymbol{\varphi}, \Phi(\boldsymbol{x}_i)) = \langle \boldsymbol{\varphi}, \boldsymbol{\varphi} \rangle - 2\langle \boldsymbol{\varphi}, \Phi(\boldsymbol{x}_i) \rangle + \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_i) \rangle$ be the square distance between $\boldsymbol{\varphi}$ and any $\Phi(\boldsymbol{x}_i)$. In practice, only neighbouring elements are considered. Let $\Phi(\dot{\boldsymbol{x}}_1), ...\Phi(\dot{\boldsymbol{x}}_n)$ denote the n-th closest elements to $\boldsymbol{\varphi}$.
2. For an isotropic kernel[2], the relation $d^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = g(\widetilde{d}^2(\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)))$ between the distances in the input and the feature spaces can be established.

[2] an isotropic kernel is a function of the form $k(\boldsymbol{x}, \boldsymbol{y}) = f(\|\boldsymbol{x} - \boldsymbol{y}\|)$ that depends only on the norm of the lag vector between two samples.

3. A solution is then deployed to determine the pre-image $\boldsymbol{x}^*$ such that

$$[d^2(\boldsymbol{x}^*, \dot{\boldsymbol{x}}_1), \ldots, d^2(\boldsymbol{x}^*, \dot{\boldsymbol{x}}_n)] = \left[g(\widetilde{d^2}(\varphi, \Phi(\dot{\boldsymbol{x}}_1))), ..., g(\widetilde{d^2}(\varphi, \Phi(\dot{\boldsymbol{x}}_n)))\right]. \quad (18)$$

For that, an SVD is deployed on the centred version of the submatrix $X_n = [\dot{\boldsymbol{x}}_1, ..., \dot{\boldsymbol{x}}_n]$, namely

$$X_n(I_n - \mathbb{1}_n) \quad = \quad U\Lambda V^T, \quad (19)$$

where $U = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_q]$ is the $d \times q$ matrix of the left-singular vectors. Let $Z = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n] = \Lambda V^T$ be the $q \times n$ matrix giving the projections of $\dot{\boldsymbol{x}}_i$ on the $\boldsymbol{u}_j$'s orthonormal vectors. The pre-image estimation $\boldsymbol{x}^*$ is then obtained as:

$$\boldsymbol{x}^* \quad = \quad U\boldsymbol{z}^* + \tfrac{1}{n} X_n \mathbf{1}_n, \quad (20)$$

where $\boldsymbol{z}^*$, the projection of $\boldsymbol{x}^*$ on the $[\boldsymbol{u}_1, ..., \boldsymbol{u}_q]$ coordinate system, is

$$\boldsymbol{z}^* = -\frac{1}{2}\,(Z\,Z^T)^{-1}\,Z\,(\boldsymbol{d}^2 - \boldsymbol{d}_0^2), \quad (21)$$

with $\boldsymbol{d}_0^2 = [\|\boldsymbol{z}_1\|^2, ..., \|\boldsymbol{z}_n\|^2]^T$, $\boldsymbol{d}^2 = [d_1^2, \ldots, d_n^2]^T$ and $d_i^2 = g(\widetilde{d^2}(\varphi, \Phi(\dot{\boldsymbol{x}}_i)))$.

3.2. Pre-image estimation by isometry preserving

To learn the pre-image $\boldsymbol{x}^*$ of a given $\boldsymbol{\varphi} = \sum_{i=1}^{N} \gamma_i\, \Phi(\boldsymbol{x}_i)$, the proposed approach in [15] proceeds in two steps. First, a coordinate system, spanned by the feature vectors $\{\Phi(\boldsymbol{x}_i)\}_{i=1}^{N}$ is learned to ensure an isometry with the input space; subsequently, the coordinate system is used to estimate the pre-image $\boldsymbol{x}^*$ of $\boldsymbol{\varphi}$. These two main steps are summarised in the followings:

1. Let $\Psi = \{\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_p\}$ $(p \leq N)$ be a coordinate system in the feature space, with $\boldsymbol{\psi}_k = \Phi(X)\boldsymbol{\alpha}_k$. The projection of $\Phi(X)$ onto the coordinate system is $P(\Phi(X)) = (KA)^T$, where $A = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_p]$. To estimate the coordinate system that is isometric with the input space, the following optimisation problem is solved

$$\arg\min_{A} \|X^T X - KAA^T K\|_{\mathcal{F}}^2 + \lambda \|KA\|_{\mathcal{F}}^2, \quad (22)$$

where the second term controls through the regularisation parameter λ the smoothness of the solution. The solution of this problem satisfies $AA^T = K^{-1}(X^T X - \lambda K^{-1})K^{-1}$.

2. Based on this result, the pre-image estimation $\boldsymbol{x}^*$ takes the form

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \|X^T\boldsymbol{x} \;-\; (X^T X - \lambda K^{-1})\,\boldsymbol{\gamma}\,\|_{\mathcal{F}}^2 \quad (23)$$

with $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_N)^T$. This problem defines a standard overdetermined equation system $(N \gg d)$ that can be resolved as a least-square minimisation problem (i.e., any technique such as the pseudo-inverse or the eigendecomposition). The pre-image estimation is then:

$$\boldsymbol{x}^* = (XX^T)^{-1} X (X^T X - \lambda K^{-1})\,\boldsymbol{\gamma}. \quad (24)$$

3.3. Pre-image estimation by kernel regression

In Bakir et al. [3], the pre-image estimation consists in learning a kernel regression function that maps all the $\Phi(\boldsymbol{x}_i)$ in the feature space $\mathcal{H}$ (related to the kernel κ) to $\boldsymbol{x}_i$ in the input space $\mathbb{R}^d$. For that, first a kernel PCA is deployed to embed $\Phi(X)$ into the subspace spanned by the eigenvectors $\boldsymbol{u}_1, ..., \boldsymbol{u}_p$ defined in (4), with $\boldsymbol{u}_j = \Phi(X)\boldsymbol{\alpha}_j$. Then, a kernel regression is learned from the set of the projections onto the kernel PCA subspace and X as

$$\arg\min_{B} \|X - B\widehat{K}\|_{\mathcal{F}}^2 + \lambda\|B\|_{\mathcal{F}}^2, \tag{25}$$

where $B \in \mathbb{R}^{d\times N}$ is the regression coefficient matrix and $\widehat{K}$ is the Gram matrix with entries $\widehat{K}_{ij} = \widehat{\kappa}(P(\Phi(\boldsymbol{x}_i)), P(\Phi(\boldsymbol{x}_j)))$. The solution to the problem (25) is then

$$B = X\widehat{K}(\widehat{K}^2 + \lambda I_N)^{-1} \tag{26}$$

For a given result $\boldsymbol{\varphi} = \Phi(X)\boldsymbol{\gamma} \in \mathcal{H}$, its pre-image $\boldsymbol{x}^*$ is then estimated as:

$$\boldsymbol{x}^* = B\widehat{\boldsymbol{k}}_{P(\boldsymbol{\varphi})}^T, \tag{27}$$

with

$$\widehat{\boldsymbol{k}}_{P(\boldsymbol{\varphi})} = [\widehat{\kappa}(P(\boldsymbol{\varphi}), P(\Phi(\boldsymbol{x}_1))), ..., \widehat{\kappa}(P(\boldsymbol{\varphi}), P(\Phi(\boldsymbol{x}_N)))], \tag{28}$$

where $\widehat{\kappa}(P(\boldsymbol{\varphi}), P(\Phi(\boldsymbol{x}_i))) = \widehat{\kappa}(\boldsymbol{\alpha}^T K\boldsymbol{\gamma}, \boldsymbol{\alpha}^T\boldsymbol{k}_{\boldsymbol{x}_i}^T)$ and $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_p]$.

3.4. Overview

To sum up, the three major methods presented above define three different approaches for pre-image estimation problem. First of all, all the methods involve only linear algebra and propose solutions that don't suffer from numerical instabilities. In Kwok et al. [18], the solution is mainly requiring the definition of a relation between the distances into the input and the kernel feature spaces. That requirement limite the Kwok et al. [18] approach to linear or isotropic kernels. Honeine et al. [15] alleviate that point by proposing a closed-form solution that is applicable to any type of kernels. Furthermore, while in Honeine et al. [15] the pre-image estimation is obtained by learning a linear transformation into the feature space that preserves the isometry between the input and the feature space, in Bakir et al. [3], the pre-image estimation is obtained by using a non linear kernel regression that predicts the input samples from their images into the feature space. Finally, while both [15] and [3] proposals involve the whole training samples for pre-image estimation, Kwok et al. [18] uses only the samples on the neighborhood of $\boldsymbol{\varphi}$, which offers a significant speed-up; highly valuable in the case of large scale data.

4. Proposed pre-image estimation for time series kernel analytics

Let $\{\boldsymbol{x}_i\}_{i=1}^N$ be a set of N time series, where each $\boldsymbol{x}_i \in \mathbb{R}^{d\times t_i}$ is a multivariate time series of length t_i that may involve varying delays. Let $\Phi(\boldsymbol{x}_i)$ be the Φ-mapping of the time series $\boldsymbol{x}_i$ into the Hilbert space $\mathcal{H}$ related to a temporal kernel κ that involves dynamic time alignments such as DTAK [25], KDTW [2] and KGA [10]. Let K be a the corresponding Gram matrix, with entries $K_{ii'} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$. Given $\boldsymbol{\varphi} = \sum_{i=1}^N \gamma_i\ \Phi(\boldsymbol{x}_i)$ a result generated in $\mathcal{H}$, the objective is to estimate the time series $\boldsymbol{x}^* \in \mathbb{R}^{d\times t^*}$ that is the pre-image of $\boldsymbol{\varphi}$. This problem is particularly challenging since, under varying delays, the time series are not longer lying in a metric space, which makes inapplicable the related work on pre-image estimation.

We tackle this problem in two parts. In the first part, we formalise the pre-image estimation problem as the estimation of a linear transformation in the feature space, that ensures an isometry between the input and the feature spaces. Moreover, this result is extended to the estimation of a nonlinear transformation in the feature space, shown powerful on challenging data in Section 5. Subsequently, we propose a local time-warp mapping function to embed time series into a vector space where the pre-image estimation can be estimated conveniently.

4.1. Learning linear and nonlinear transformations for pre-image estimation

Let $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_N] \in \mathbb{R}^{t\times N}$ be a matrix giving the description of N univariate[3] time series $\boldsymbol{x}_i$ that we assume first lying in the metric space $\mathbb{R}^t$; Section 4.2 addresses the general case of time series that are lying into nonEuclidean space. The proposed pre-image method relies on learning a linear transformation R in the feature space that ensures an isometry between X and $\Phi(X)$. We first describe the method as a linear transformation, and then extend it to nonlinear transformations.

Linear transformation

The main idea to solve the pre-image problem is the isometry preserving, in the same spirit as the method described in Section 3.2. For this purpose, we formalise the pre-image problem as the estimation of the square matrix R that establishes an isometry between X and $\Phi(X)$, by solving the optimisation problem

$$R^* \quad = \quad \arg\min_R \|X^T X - \Phi(X)^T R\, \Phi(X)\|_{\mathcal{F}}^2. \tag{29}$$

By using a kernel PCA where a relevant subspace is considered, an explicit description $P(\Phi(X)) \in \mathbb{R}^{p\times N}$ of $\Phi(X)$ is given and Eq. (29) can thus be rewritten

[3] For multivariate time series, simply define $X \in \mathbb{R}^{d\,t\times N}$.

as:

$$R^* \quad = \quad \arg\min_R \|X^T X - P(\Phi(X))^T \, R \, P(\Phi(X))\|_{\mathcal{F}}^2. \tag{30}$$

As $P(\Phi(X))\,P(\Phi(X))^T$ is invertible, a closed-form solution is given by:

$$R^* = \left(P(\Phi(X))P(\Phi(X))^T\right)^{-1} P(\Phi(X))X^T X P(\Phi(X))^T \left(P(\Phi(X))P(\Phi(X))^T\right)^{-1}. \tag{31}$$

The estimation of the time series $\boldsymbol{x}^*$, as the pre-image of $\boldsymbol{\varphi} = \Phi(X)\boldsymbol{\gamma}$, is then given by:

$$\begin{aligned} \boldsymbol{x}^* &= (XX^T)^{-1} X \, P(\Phi(X))^T R^* P(\boldsymbol{\varphi}) \\ &= (XX^T)^{-1} X \, P(\Phi(X))^T R^* \boldsymbol{\alpha}^T K \boldsymbol{\gamma}, \end{aligned} \tag{32}$$

with $P(\boldsymbol{\varphi}) = \boldsymbol{\alpha}^T K \boldsymbol{\gamma}$ and $\boldsymbol{\alpha}$ defined in Eq. (7).

One can easily include some regularisation terms in the optimisation problems (29) and (30), which can be easily propagated to the pre-image expression. For example, in the case of non-invertible XX^T, a regularisation term is introduced in Eq. (32) as:

$$\boldsymbol{x}^* \quad = \quad (XX^T + \lambda I_t)^{-1} X \, P(\Phi(X))^T R^* \boldsymbol{\alpha}^T K \boldsymbol{\gamma}, \tag{33}$$

for some positive regularisation parameter λ.

Nonlinear transformation

In the following, we propose to extend the above result to learn nonlinear transformations for pre-image estimation. Let $\widehat{\kappa}$ be a kernel defined on the feature space $\mathcal{H}$, and $\widehat{\Phi}$ the corresponding embedding function that maps any element of $\mathcal{H}$ into the Hilbert space defined by $\widehat{\kappa}$. With some abuse of notation, we denote $\widehat{\Phi}(\Phi(X))$ the matrix of all mapped elements $\widehat{\Phi}(\Phi(\boldsymbol{x}_i))$, for $i = 1, ..., N$. Let $\widehat{K}$ be the Gram matrix of general term $\widehat{\kappa}(\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j))$.

The pre-image estimation problem can be then defined as learning a nonlinear transformation that defines an isometry between X and $\widehat{\Phi}(\Phi(X))$ as:

$$R^* \quad = \quad \arg\min_R \|X^T X - \widehat{\Phi}(\Phi(X))^T \, R \, \widehat{\Phi}(\Phi(X))\|_{\mathcal{F}}^2. \tag{34}$$

Similarly, a closed-form solution for R^* can be obtained as:

$$\begin{aligned} R^* = \quad & (P(\widehat{\Phi}(\Phi(X)))\,P(\widehat{\Phi}(\Phi(X)))^T)^{-1} P(\widehat{\Phi}(\Phi(X))) \\ & X^T X P(\widehat{\Phi}(\Phi(X)))^T (P(\widehat{\Phi}(\Phi(X)))P(\widehat{\Phi}(\Phi(X)))^T)^{-1}, \end{aligned} \tag{35}$$

and

$$P(\widehat{\Phi}(\Phi(X))) = \widehat{\alpha}^T \, \widehat{K}. \tag{36}$$

To estimate $\widehat{K}$, an indirect manner is to use a kernel PCA, with $\widehat{\kappa}(\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)) \approx \widehat{\kappa}(P(\Phi(\boldsymbol{x}_i)), P(\Phi(\boldsymbol{x}_j)))$. A simpler way is possible when dealing with kernels that are radial basis functions. For example, for the well-known Gaussian kernel $\widehat{\kappa}$, $\widehat{K}$ is estimated directly from K as:

$$\begin{aligned}\widehat{\kappa}(\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)) &= \exp\left(-\frac{\| \Phi(\boldsymbol{x}_i) - \Phi(\boldsymbol{x}_j) \|^2}{2\sigma^2}\right) \\ &= \exp\left(-\frac{\langle\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_i)\rangle - 2\langle\Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)\rangle + \langle\Phi(\boldsymbol{x}_j), \Phi(\boldsymbol{x}_j)\rangle}{2\sigma^2}\right) \\ &= \exp\left(-\frac{\kappa(\boldsymbol{x}_i, \boldsymbol{x}_i) - 2\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + \kappa(\boldsymbol{x}_j, \boldsymbol{x}_j)}{2\sigma^2}\right) \end{aligned} \tag{37}$$

The estimation of the pre-image of $\boldsymbol{\varphi} = \sum_{i=1}^{N} \gamma_i \, \Phi(\boldsymbol{x}_i)$ is then given by the time series $\boldsymbol{x}^*$:

$$\boldsymbol{x}^* = (XX^T)^{-1} \, X \, P(\widehat{\Phi}(\Phi(X)))^T R^* P(\widehat{\Phi}(\boldsymbol{\varphi})), \tag{38}$$

with $P(\widehat{\Phi}(\boldsymbol{\varphi})) = (\widehat{\boldsymbol{k}}_{\boldsymbol{\varphi}}\widehat{\alpha})^T$, where $\widehat{\boldsymbol{k}}_{\boldsymbol{\varphi}}$ is the vector whose i-th entry is

$$\widehat{\kappa}(\boldsymbol{\varphi}, \Phi(\boldsymbol{x}_i)) = \exp\left(-\frac{\boldsymbol{\gamma}^T K \boldsymbol{\gamma} - 2\,\boldsymbol{\gamma}^T \boldsymbol{k}_{\boldsymbol{x}_i}^T + K_{ii}}{2\sigma^2}\right). \tag{39}$$

The above proposed formulations and results for pre-image estimation (Section 4.1) present some similarities and differences with the method proposed in [16] and presented in Section 3.2. First of all, both approaches propose formulations and solutions that only require linear algebra and are independent of the type of kernel. To establish the isometry, in [16] a linear transformation restricted to the form $R = \Phi(X)AA^T\Phi(X)^T$ is estimated, whereas in our proposal the estimated R may be linear Eq.(30) or non linear Eq.(34) and is importantly unconstrained, namely of general form which enlarges its potential to deal with complex structures. Finally, while in [16] the solution Eq.(24) involves the kernel information through the regularisation term, which may be canceled for lower values of λ, in the proposed solutions Eq.(33) and Eq.(38) the kernel information is entirely considered regardless of the regularisation specifications.

4.2. Learning a metric space embedding for time series pre-image estimation

In Section 4.1, time series are assumed of the same length and lying in a metric space. However, in general $X = \{\boldsymbol{x}_i\}_{i=1}^N$ is instead composed of time series $\boldsymbol{x_i}$ of different lengths t_i that are located in a non-metric space, rendering the previous results as well as the pre-image estimation related works not applicable.

To address the pre-image estimation for such challenging time series, we define an embedding function that allows to represent the time series in a metric

space, where the previous linear and nonlinear transformations method for pre-image estimation can be performed conveniently.

For this purpose, first we define $\mathcal{N}_{\boldsymbol{\varphi}}$ in $\mathcal{H}$ and $\mathcal{N}_{\boldsymbol{\varphi}}^{-1}$ as the set of the n-closest neighbours of $\boldsymbol{\varphi}$ and its pre-image, given as:

$$\mathcal{N}_{\boldsymbol{\varphi}} = \Big\{ \Phi(\boldsymbol{x}_i) \,\Big|\, \langle \Phi(\boldsymbol{x}_i), \boldsymbol{\varphi} \rangle = \sum_{j=1}^{N} \gamma_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \text{ is among the } n \text{ highest values} \Big\}, \tag{40}$$

$$\mathcal{N}_{\boldsymbol{\varphi}}^{-1} = \big\{ \boldsymbol{x}_i \mid \Phi(\boldsymbol{x}_i) \in \mathcal{N}_{\boldsymbol{\varphi}} \big\} \tag{41}$$

Let $\Phi(\boldsymbol{x}_r)$ be the representative of $\mathcal{N}_{\boldsymbol{\varphi}}$ with $\boldsymbol{x}_r \in \mathbb{R}^{t^*}$ defined as:

$$\Phi(\boldsymbol{x}_r) = \underset{\Phi(\boldsymbol{x}_i) \in \mathcal{N}_{\boldsymbol{\varphi}}}{\arg\max} \sum_{\Phi(\boldsymbol{x}_j) \in \mathcal{N}_{\boldsymbol{\varphi}}} \kappa(\boldsymbol{x_i}, \boldsymbol{x}_j). \tag{42}$$

To resorb the arising delays, a temporal alignment between each $\boldsymbol{x}_i$ and $\boldsymbol{x}_r$ is then performed by dynamic programming. An alignment $\boldsymbol{\pi}$ of length $|\boldsymbol{\pi}| = m$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_r$ is defined as the set of m increasing couples

$$\boldsymbol{\pi} = ((\pi_1(1), \pi_2(1)), (\pi_1(2), \pi_2(2)), ..., (\pi_1(m), \pi_2(m))),$$

where the applications π_1 and π_2 defined from $\{1, ..., m\}$ to $\{1, ..., t_i\}$ and $\{1, ..., t^*\}$ respectively obey to the following boundary and monotonicity conditions:

$$1 = \pi_1(1) \leq \pi_1(2) \leq ... \leq \pi_1(m) = t_i$$
$$1 = \pi_2(1) \leq \pi_2(2) \leq ... \leq \pi_2(m) = t^*$$

and $\forall\, l \in \{1, ..., m\}$, $\pi_1(l+1) \leq \pi_1(l) + 1$ and $\pi_2(l+1) \leq \pi_2(l) + 1$, $(\pi_1(l+1) - \pi_1(l)) + (\pi_2(l+1) - \pi_2(l)) \geq 1$.

Intuitively, an alignment $\boldsymbol{\pi}$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_r$ describes a way to associate each element of $\boldsymbol{x}_i$ to one or more elements of $\boldsymbol{x}_r$ and vice-versa. Such an alignment can be conveniently represented by a path in the $t_i \times t^*$ grid, as shown in Figure 1 (left), where the above monotonicity conditions ensure that the path is neither going back nor jumping. The optimal alignment $\boldsymbol{\pi}^*$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_r$ is then obtained as:

$$\boldsymbol{\pi}^* = \underset{\boldsymbol{\pi}=(\pi_1,\pi_2)}{\arg\min} \|\boldsymbol{x}_i^{\pi_1} - \boldsymbol{x}_r^{\pi_2}\|^2 \tag{43}$$

where $\boldsymbol{x}_i^{\pi_1} = (x_{i\,\pi_1(1)}, ..., x_{i\,\pi_1(m)})$ and $\boldsymbol{x}_r^{\pi_2} = (x_{r\,\pi_2(1)}, ..., x_{r\,\pi_2(m)})$ are $\boldsymbol{x}_i$ and $\boldsymbol{x}_r$ aligned through $\boldsymbol{\pi}$.

We define f_r, the temporal embedding function, to embed time series $\boldsymbol{x}_i \in \mathbb{R}^{d \times t_i}$ into a new temporal metric space as:

$$\begin{aligned} f_r : \quad X \quad &\longrightarrow \widetilde{X} \subset \widetilde{\mathcal{I}} = \mathbb{R}^{d \times t^*} \\ \boldsymbol{x}_i \quad &\longrightarrow f_r(\boldsymbol{x}_i) = \boldsymbol{x}_i \, W_{ir} \, N_{ir} \end{aligned} \tag{44}$$

where $W_{ir} \in \{0,1\}^{t_i \times t^*}$ is the binary matrix related to the optimal temporal alignment between $\boldsymbol{x}_i$ and $\boldsymbol{x_r}$, as shown in Figure 1 (right). The matrix

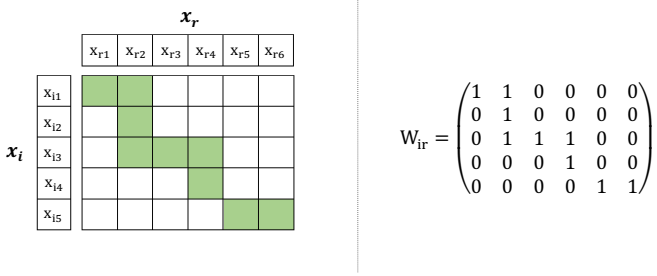


Figure 1: In the left, the temporal alignment between $\boldsymbol{x}_i$ ($t_i = 5$) and $\boldsymbol{x}_r$ ($t^* = 6$), the optimal alignment $\boldsymbol{\pi}^*$ is indicated in green. In the right, the adjacency binary matrix related to the optimal temporal alignment.

$N_{ir} = \text{diag}(W_{ir}^T \mathbf{1}_{t_i})^{-1}$ is the weight diagonal matrix of order t^*, of general term $\frac{1}{|N_t|}$, that gives the weight of the element t of $\boldsymbol{x}_r$, where $|N_t|$ is the number of time stamps of $\boldsymbol{x}_i$ aligned to t. In particular, note that $\mathbf{x}_r$ remains unchanged by f_r, as $W_{rr} = N_{rr} = \text{diag}([1, 1, \ldots, 1])$.

The set of embedded time series $\widetilde{X} = \{f_r(\boldsymbol{x}_1), ..., f_r(\boldsymbol{x}_N)\}$ is for now lying in a metric space $\widetilde{\mathcal{I}}$, where the delays are resorbed w.r.t. the representative time series $\boldsymbol{x}_r$. The pre-image solution provided in the method described in Section 4.1 can be developed to establish a linear or nonlinear transformations to preserve an isometry between $\widetilde{X}$ and $\Phi(X)$. The algorithm for the proposed solution TsPrima is summarised in Algorithm 1.

Algorithm 1 TsPrima: Pre-image estimation for time series

Input: $\{\boldsymbol{x}_i\}_{i=1}^N$ with $\boldsymbol{x}_i \in \mathbb{R}^{d \times t_i}$, κ (a temporal kernel), $\widehat{\kappa}$ (a Gaussian kernel),
$\boldsymbol{\gamma}$ (with $\boldsymbol{\varphi} = \sum_{i=1}^N \gamma_i \Phi(\boldsymbol{x}_i)$), n (the neighbourhood size)
Output: $\boldsymbol{x}^*$ the pre-image estimation of $\boldsymbol{\varphi}$

Define $\mathcal{N}_{\boldsymbol{\varphi}}$, $\mathcal{N}_{\boldsymbol{\varphi}}^{-1}$ and $\boldsymbol{x}_r$ using respectively (40), (41) and (42)
Embed $\mathcal{N}_{\boldsymbol{\varphi}}^{-1}$ into a temporal metric space by using Eq. (44), set $\widetilde{N}_{\boldsymbol{\varphi}}^{-1} = f_r(\mathcal{N}_{\boldsymbol{\varphi}}^{-1})$
Set $X = \widetilde{N}_{\boldsymbol{\varphi}}^{-1}$ and $\Phi(X) = \mathcal{N}_{\boldsymbol{\varphi}}$
Learn a linear (resp. nonlinear) transformation R by using Eq. (31) (resp. Eq. (35))
Estimate the pre-image $\boldsymbol{x}^*$ based on a linear (resp. nonlinear) transformation using Eq. (33) (resp. Eq. (38))

5. Experiments

In this section, we evaluate the efficiency of the proposed pre-image estimation method under three major time series analysis tasks: 1) time series averaging, 2) time series reconstruction and denoising and 3) time series representation learning. The proposed pre-image estimation method TsPrima is compared to three major alternative approaches introduced in Section 3, and referenced in the following as Honeine, Kwok and Bakir methods. The experiments are conducted on 33 public datasets (Table 1) including univariate and multivariate time series data, that may involve varying delays and be of the same or different lengths. The 25 first datasets in Table 1 are selected from the archive given in [8, 11] by using three selection criteria: a) have a reasonable number of classes (Nb. of Classes < 50), b) have a sufficient size for train and test samples (Train size $<= 500$ and Test size < 3000), c) avoid time series of extra large lengths (Time series length < 700). To obtain a manageable number of datasets, the 3 above selection criteria are applied on the top 40 datasets, in the order set out in [8]. The 25 obtained datasets are composed of univariate time series and half of the datasets include significant delays. We consider a dataset as including significant delays if the difference between the 1-NN Euclidean distance error and the 1-NN Dynamic time warping [21] error is greater than 5%. The 5 next datasets include univariate and multivariate time series covering local and noisy salient events as described in [30, 27, 14] and the three last datasets are related to handwritten digits and characters, they are described as multivariate time series of variable lengths [7]. In the following, we detail the evaluation process of the pre-image estimation methods then give and discuss the obtained results.

5.1. Time series averaging

Estimating the centroid of a set of time series is a major topic for many time series analytics as summarisation, prototype extraction or clustering. Time series averaging has been an active area in the last decade, where the propositions mainly focus on tackling the tricky problem of multiple temporal alignments [14, 26, 27]. A suitable way to circumvent the problem of multiple temporal alignments is to use a temporal kernel method to evaluate the time series centroid in the feature space. The pre-image of the centroid is then estimated to obtain the time series averaging in the input space.

In that context, let $\{\boldsymbol{x}_i\}_{i=1}^N$ and $\{\Phi(\boldsymbol{x}_i)\}_{i=1}^N$ be, respectively, a set of time series and their mapped images into the Hilbert space $\mathcal{H}$ related to the temporal kernel DTAK [25]. Let $\boldsymbol{\varphi} = \frac{1}{N}\sum_{i=1}^N \Phi(\boldsymbol{x}_i)$ be the centroid of the mapped time series in the feature space and $\boldsymbol{x}^*$ its pre-image in the input space. The quality of the obtained centroids is given by the within-class similarity $\sum_i \text{DTAK}(\boldsymbol{x}^*, \boldsymbol{x}_i)$; the higher the within-class similarity, the better is the estimated centroid.

To evaluate the efficiency of each pre-image estimation method, the time series centroid is estimated for each class of the studied datasets and the induced

Table 1: Data Description

Dataset	Nb. Classes	Train size	Test size	Time series length	Univariate
CC	6	300	300	60	✓
GunPoint	2	50	150	150	✓
CBF	3	30	900	128	✓
OSULeaf	6	200	242	427	✓
SwedishLeaf	15	500	625	128	✓
Trace	4	100	100	275	✓
FaceFour	4	24	88	350	✓
Lighting2	2	60	61	637	✓
Lighting7	7	70	73	319	✓
ECG200	2	100	100	96	✓
Adiac	37	390	391	176	✓
FISH	7	175	175	463	✓
Beef	5	30	30	470	✓
Coffee	2	28	28	286	✓
OliveOil	4	30	30	570	✓
DiatomSizeR	4	16	306	345	✓
ECG5Days	2	23	861	136	✓
FacesUCR	14	200	2050	131	✓
ItalyPowerD	2	67	1029	24	✓
MedicalImages	10	381	760	99	✓
MoteStrain	2	20	1252	84	✓
SonyAIBOII	2	27	953	65	✓
SonyAIBO	2	20	601	70	✓
Symbols	6	25	995	398	✓
TwoLeadECG	2	23	1139	82	✓
SPIRAL1	1	50	50	101	✗
SPIRAL2	1	50	50	300	✗
PowerCons	2	73	292	144	✓
BME	3	30	150	128	✓
UMD	3	36	144	150	✓
DIGITS	10	100	100	29∼218	✗
LOWER	26	130	260	27∼163	✗
UPPER	26	130	260	27∼412	✗

within-class similarity is evaluated. The average within-class similarity is then reported in Table 2 for each dataset and each pre-image estimation method; the best values are indicated in bold (t-test at 5% risk). In addition, a Nemenyi test [12] is performed to compare the significance of the obtained results, with the related critical difference diagram given in Figure 2. The estimated time series centroids for some challenging classes are shown in Figure 3, where we retain particularly SPIRAL1 and the handwritten digits and characters datasets

(DIGITS, LOWER and UPPER) as they are more intuitive to visually evaluate the quality of the estimated time series centroids.

Table 2: Average within-class similarity of the estimated time series centroids

DataSet	TsPrima	Honeine	Kwok	Bakir
CC	**0.744**	0.709	0.721	0.709
GunPoint	0.902	**0.910**	0.882	0.886
CBF	**0.798**	0.737	0.755	0.737
OSULeaf	0.985	**0.987**	0.986	**0.987**
SwedishLeaf	0.910	**0.920**	**0.920**	**0.920**
Trace	**0.998**	0.992	0.991	0.992
FaceFour	**0.981**	0.980	**0.981**	0.98
Lighting2	**0.918**	0.876	0.859	0.875
Lighting7	**0.964**	0.930	0.930	0.931
ECG200	**0.593**	0.565	0.567	0.566
Adiac	**0.997**	**0.997**	0.996	**0.997**
FISH	**0.996**	0.995	0.994	0.995
Beef	**0.900**	0.892	0.898	0.890
Coffee	**0.998**	**0.998**	**0.998**	**0.998**
OliveOil	**0.999**	**0.999**	0.998	**0.999**
DiatomSizeR	**0.997**	**0.997**	**0.997**	**0.997**
ECG5Days	**0.777**	0.746	0.417	0.746
FacesUCR	**0.721**	0.699	0.648	0.700
ItalyPowerD	**0.610**	0.552	0.420	0.542
MedicalImages	**0.671**	0.644	0.637	0.646
MoteStrain	0.776	**0.777**	0.701	**0.777**
SonyAIBOII	**0.749**	0.740	0.716	0.740
SonyAIBO	0.960	**0.962**	0.955	**0.962**
Symbols	**0.959**	0.949	0.904	0.951
TwoLeadECG	**0.980**	0.977	0.911	0.977
SPIRAL1	**0.831**	0.823	0.799	0.824
SPIRAL2	**0.947**	0.940	0.934	0.940
PowerCons	**0.458**	0.328	0.436	0.330
BME	**0.701**	0.572	0.638	0.555
UMD	**0.800**	0.765	0.724	0.755
DIGITS	**0.746**	0.575	0.657	0.581
LOWER	**0.713**	0.544	0.645	0.545
UPPER	**0.764**	0.572	0.570	0.573
Nb. Best	**28**	9	4	8
Avg. Rank	**1.5**	2.68	3.24	2.58

5.2. Time series reconstruction and denoising

The reconstruction and denoising tasks represent a standard application context for pre-image estimation. For the time series reconstruction task, a kernel

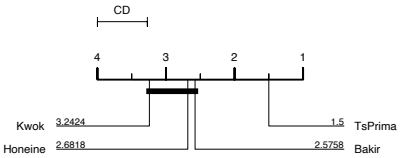


Figure 2: Nemenyi test: comparison of pre-image methods under centroid estimation task

PCA is performed on the training set, the reconstruction of a given test sample $\boldsymbol{x}$ is then defined as the pre-image $\boldsymbol{x}^*$ of its kernel PCA projection $P(\Phi(\boldsymbol{x}))$. The latter takes the form $\boldsymbol{\varphi} = \Phi(X)\boldsymbol{\gamma}$, with $\boldsymbol{\gamma}$ defined as:

$$\boldsymbol{\gamma} \quad = (I_N - \mathbb{1}_N)\, \boldsymbol{\alpha}\, \boldsymbol{\alpha}^T\, \widetilde{\boldsymbol{k}}_{\boldsymbol{x}}^T + \tfrac{1}{N}\mathbf{1}_N \tag{45}$$

The quality of the reconstruction is then measured as the similarity $\text{DTAK}(\boldsymbol{x}^*, \boldsymbol{x})$ between each test sample $\boldsymbol{x}$ and its reconstruction $\boldsymbol{x}^*$; the higher the criterion, the better is the reconstruction. Table 3 gives the average quality of reconstruction obtained for each dataset and each method. Figure 4 gives the critical difference diagram related to the Nemenyi test for the average ranking comparison of the studied methods. Figure 5 shows the reconstructions obtained for some challenging time series of DIGITS, LOWER and UPPER datasets.

For the time series denoising task, first a kernel PCA is performed on the training set, then a $(0, \sigma^2)$ Gaussian noise is added to the test samples $\boldsymbol{x}$ to generate noisy samples $\tilde{\boldsymbol{x}}$ with different variances σ^2. The denoised sample is obtained as the pre-image $\boldsymbol{x}^*$ of its kernel PCA projection $P(\Phi(\tilde{\boldsymbol{x}}))$, with $\boldsymbol{\gamma}$ defined as in Eq. (45). Similarly, the quality of the denoising is measured as the similarity $\text{DTAK}(\boldsymbol{x}^*, \boldsymbol{x})$ between $\boldsymbol{x}^*$ and the initial $\boldsymbol{x}$. Table 4 gives, for different values of σ^2, the average quality of the denoising for some datasets. Figure 6 illustrates the denoising results for some challenging times series of the noisy SPIRAL2 data and of the class "M" of UPPER dataset.

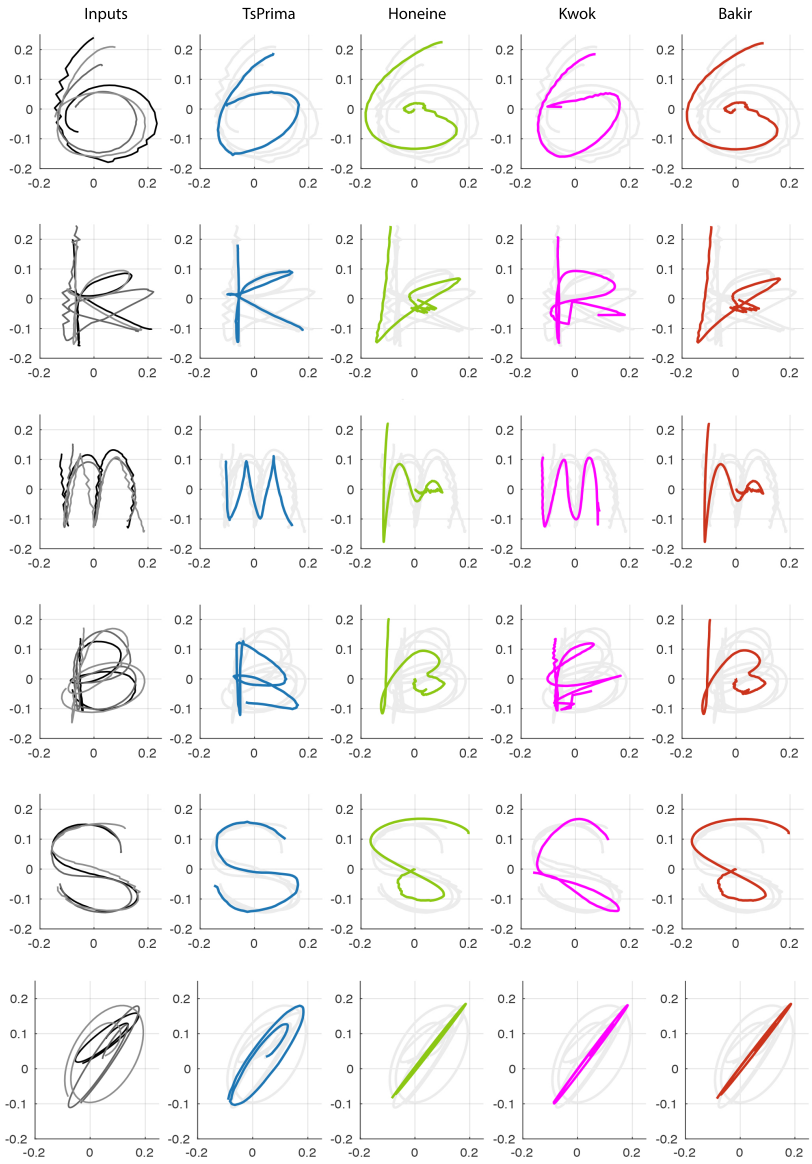


Figure 3: Time series centroids for some challenging classes of DIGITS, LOWER, UPPER and SPIRAL1 datasets

Table 3: Quality of the time series reconstruction under kernel PCA

DataSet	TsPrima	Honeine	Kwok	Bakir
CC	**0.798**	0.747	0.758	0.747
GunPoint	0.994	**0.996**	0.992	0.990
CBF	**0.916**	0.854	0.896	0.875
OSULeaf	0.997	**0.998**	0.995	**0.998**
SwedishLeaf	**0.798**	0.701	0.690	0.650
Trace	**0.689**	0.519	0.597	0.519
FaceFour	**0.981**	0.951	0.967	0.964
Lighting2	**0.993**	0.967	0.984	0.975
Lighting7	**0.954**	0.920	0.938	0.922
ECG200	0.965	**0.979**	0.959	0.962
Adiac	**0.194**	0.127	0.139	0.125
FISH	**0.779**	0.580	0.586	0.579
Beef	0.528	0.703	0.643	**0.704**
Coffee	0.584	**0.595**	0.570	0.559
OliveOil	**0.150**	0.125	0.141	0.121
DiatomSizeR	**0.330**	0.174	0.186	0.173
ECG5Days	**0.996**	**0.996**	0.995	0.995
FacesUCR	**0.939**	0.825	0.878	0.847
ItalyPowerD	0.831	**0.892**	0.023	0.851
MedicalImages	**0.946**	0.906	0.935	0.928
MoteStrain	0.971	**0.987**	0.970	0.979
SonyAIBOII	0.978	**0.989**	0.969	0.985
SonyAIBO	0.939	**0.98**	0.924	0.967
Symbols	**0.885**	0.822	0.724	0.761
TwoLeadECG	**0.825**	0.630	0.444	0.669
spiral1	**0.961**	0.939	0.933	0.911
spiral2	**0.966**	0.939	0.946	0.940
PowerCons	0.971	0.966	0.955	**0.977**
BME	**0.896**	0.800	0.858	0.666
UMD	0.885	0.855	**0.904**	0.797
digits	**0.840**	0.721	0.798	0.726
lower	**0.787**	0.696	0.747	0.685
upper	**0.856**	0.678	0.787	0.687
Nb. Best	**22**	9	1	3
Avg. Rank	**1.56**	2.67	2.71	3.06

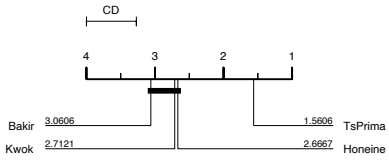


Figure 4: Nemenyi test: comparison of pre-image methods under kernel PCA reconstruction

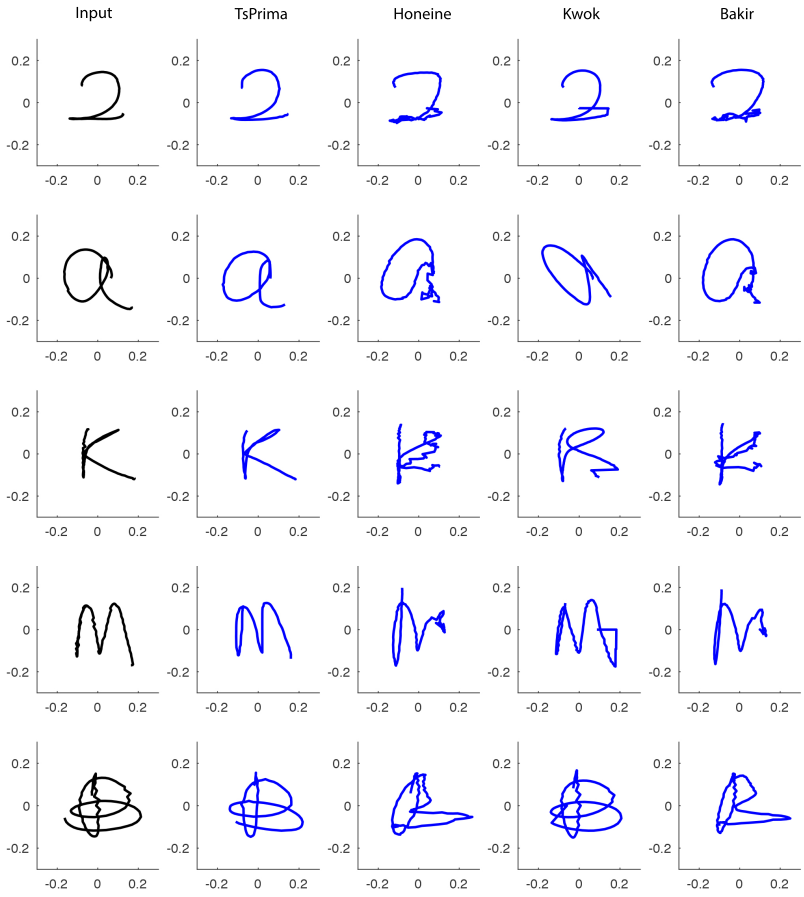


Figure 5: The time series reconstruction under kernel PCA of some samples of DIGITS, LOWER and UPPER datasets

Table 4: Quality of the denoising for several noise levels

DataSet	σ^2	TsPrima	Honeine	Kwok	Bakir
DIGITS	0.01	**0.832**	0.669	0.782	0.666
	0.05	**0.808**	0.619	0.742	0.627
	0.1	**0.791**	0.605	0.723	0.612
	0.15	**0.783**	0.598	0.719	0.606
LOWER	0.01	**0.766**	0.651	0.721	0.637
	0.05	**0.746**	0.614	0.689	0.606
	0.1	**0.736**	0.601	0.675	0.596
	0.15	**0.729**	0.594	0.670	0.591
UPPER	0.01	**0.837**	0.627	0.765	0.638
	0.05	**0.806**	0.579	0.712	0.600
	0.1	**0.789**	0.561	0.688	0.590
	0.15	**0.782**	0.554	0.679	0.586
Nb. Best		**12**	0	0	0
Avg. Rank		**1.00**	3.58	2.00	3.42

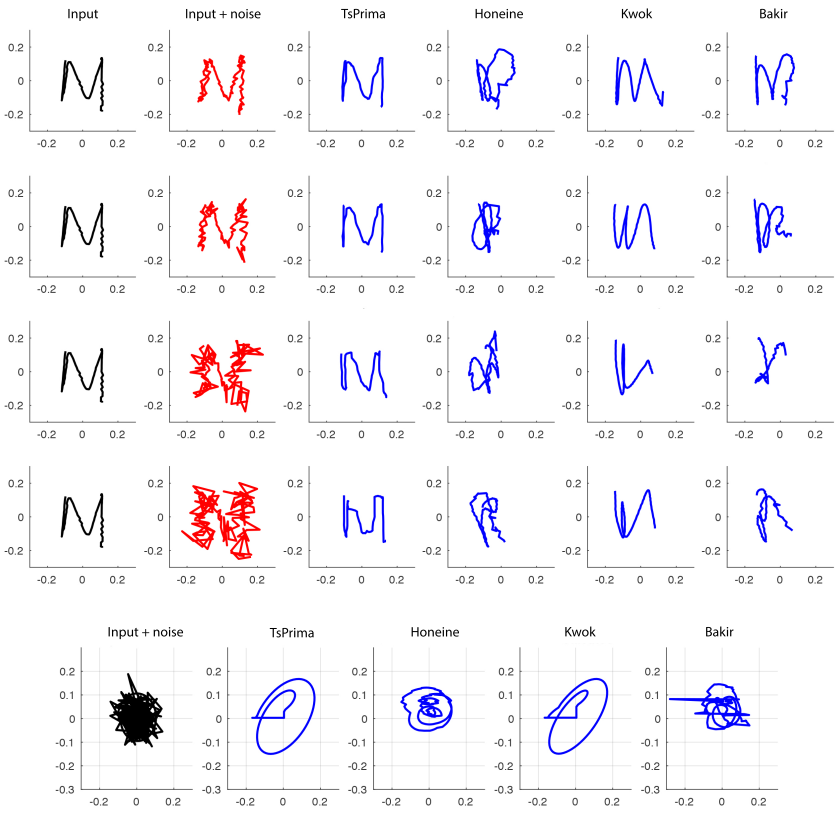


Figure 6: Time series denoising under kernel PCA of noisy samples of SPIRAL2 and of the class "M" of UPPER dataset.

5.3. Time series representation learning

For time series representation learning, the kernel k-SVD ($\tau = 5$) is used to learn, for each class of the considered datasets, the dictionary $\Phi(X)\mathcal{B}$ and the sparse representations $\mathcal{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_N]$ of its membership time series, as defined in Section 2.2. The pre-images D^* and X^* of the dictionary $\Phi(X)\mathcal{B}$ and of the sparse codes $\mathcal{A}$ are then obtained by considering $\boldsymbol{\gamma} = \mathcal{B}$ and $\boldsymbol{\gamma} = \mathcal{B}\mathcal{A}$, respectively. The quality of the learned sparse representations is then measured as the similarity $\text{DTAK}(\boldsymbol{x}_i, \boldsymbol{x}_i^*)$ between each time series $\boldsymbol{x}_i$ and the pre-image $\boldsymbol{x}_i^*$ of the sparse representation $\Phi(X)\mathcal{B}\,\boldsymbol{a}_i$. Table 5 gives the average quality of the learned representations for each dataset and each pre-image estimation method. Figure 7 gives the critical difference diagram related to the Nemenyi test for the average ranking comparison of the studied methods. Figure 8 shows the learned representations for some time series of DIGITS, LOWER and UPPER datasets and Figure 9 illustrates, for a challenging sample of the class "k" of LOWER dataset, the learned representations as well as the top 3 atoms involved in its reconstruction.

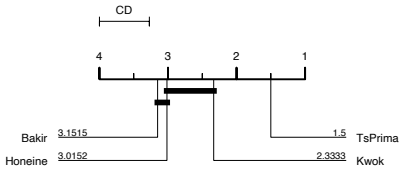


Figure 7: Nemenyi test: comparison of pre-image methods under kernel k-SVD representation learning

Table 5: Quality of the time series representation learning under Kernel k-SVD

DataSet	TsPrima	Honeine	Kwok	Bakir
CC	**0.788**	0.730	0.751	0.732
GunPoint	0.993	**0.994**	0.992	0.985
CBF	**0.917**	0.862	0.900	0.872
OSULeaf	**0.996**	**0.996**	0.995	**0.996**
SwedishLeaf	**0.789**	0.659	0.691	0.623
Trace	**0.687**	0.514	0.602	0.514
FaceFour	**0.971**	0.940	0.959	0.947
Lighting2	**0.991**	0.961	0.982	0.968
Lighting7	**0.961**	0.934	0.947	0.934
ECG200	0.953	**0.957**	0.950	0.941
Adiac	**0.184**	0.122	0.131	0.117
FISH	**0.757**	0.553	0.579	0.560
Beef	0.411	0.555	0.605	**0.621**
Coffee	0.596	**0.607**	0.586	0.560
OliveOil	0.145	0.133	**0.152**	0.120
DiatomSizeR	**0.287**	0.177	0.198	0.178
ECG5Days	**0.996**	**0.996**	0.995	0.994
FacesUCR	**0.917**	0.834	0.878	0.842
ItalyPowerD	**0.800**	0.781	0.034	0.728
MedicalImages	**0.937**	0.860	0.930	0.878
MoteStrain	0.969	0.970	**0.971**	0.970
SonyAIBOII	0.974	**0.975**	0.973	**0.975**
SonyAIBO	0.932	**0.938**	0.930	0.936
Symbols	**0.811**	0.785	0.794	0.755
TwoLeadECG	**0.810**	0.617	0.411	0.629
spiral1	**0.944**	0.913	0.920	0.914
spiral2	**0.964**	0.936	0.949	0.937
PowerCons	**0.968**	0.946	0.957	0.951
BME	**0.872**	0.734	0.843	0.622
UMD	0.888	0.842	**0.905**	0.788
digits	**0.822**	0.699	0.793	0.706
lower	**0.773**	0.678	0.738	0.671
upper	**0.840**	0.664	0.797	0.675
Nb. Best	**24**	7	3	3
Avg. Rank	**1.5**	3.02	2.33	3.15

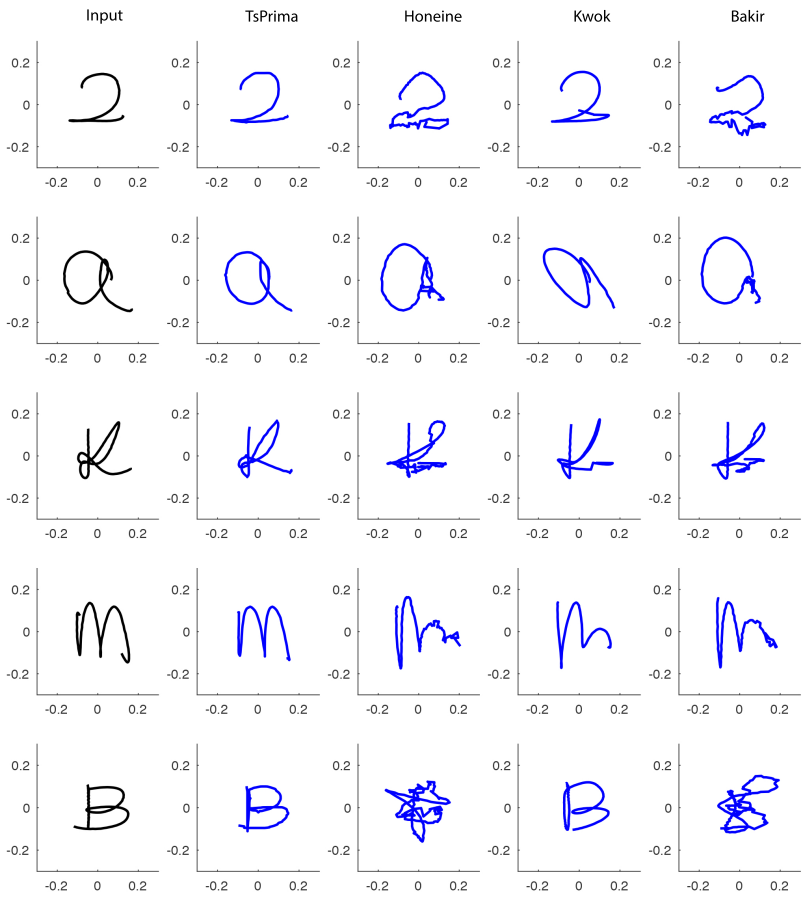


Figure 8: The learned time series representations under kernel k-SVD of some samples of DIGITS, LOWER, UPPER datasets

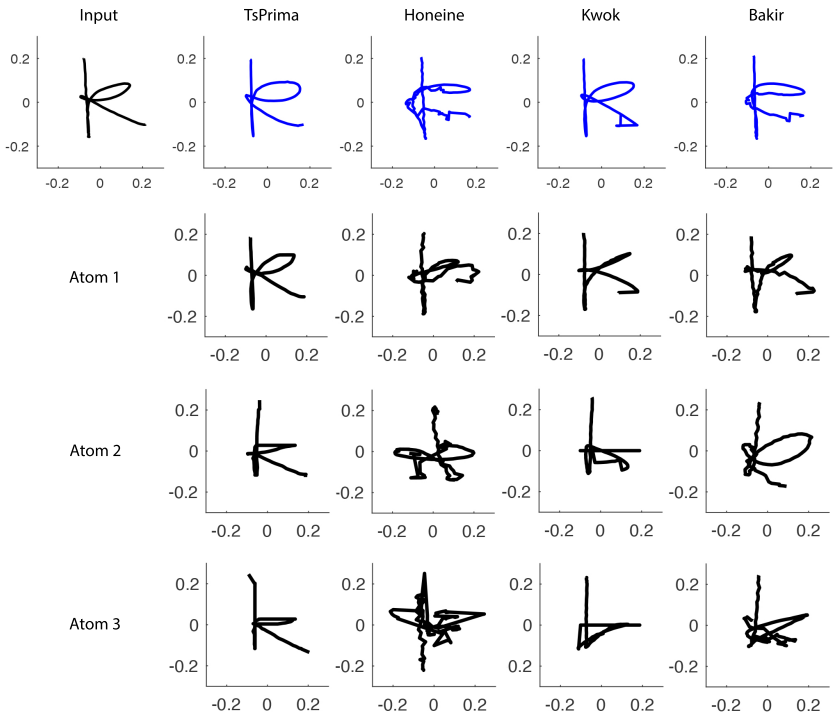


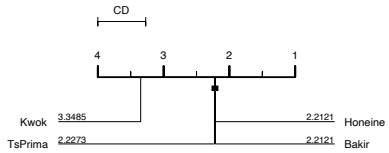
Figure 9: The sparse representation of a time series of the class "k" of LOWER dataset and the top 3 involved atoms for its reconstruction

5.4. Further comparison

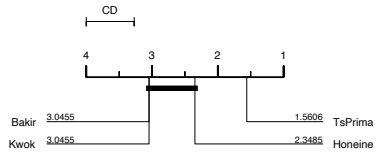
In the previous experiments (Sections 5.1 to 5.3), we have evaluated the performances of TsPrima that are mainly due to two major ingredients : 1) the defined temporal embedding function f_r (Section 4.2) and 2) the proposed transformation R to preserve an isometry between the time series embedding space and the feature space (Section 4.1). In this last part, the aim is to evaluate the efficiency of the proposed transformation R, regardless of the effect of f_r. For that, TsPrima is compared to the alternative methods Honeine, Kwok and Bakir once all the time series embedded into the same metric space; namely, all the pre-image estimation methods are performed between the time series embedding space and the feature space. Similar experiments are performed on the 33 public datasets (Table 1), the results obtained for the three tasks are summarised into Table 6 and the related Nemenyi tests are given in Figure 10.

Table 6: Further comparisons for pre-image estimation

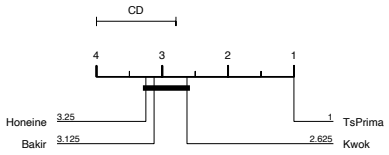
		TsPrima	Honeine	Kwok	Bakir
Averaging	Nb. Best	**19**	**20**	4	**19**
	Avg. Rank	**2.23**	**2.21**	3.35	**2.21**
Reconstruction	Nb. Best	**24**	10	0	1
(kernel PCA)	Avg. Rank	**1.56**	2.35	3.05	3.05
Denoising	Nb. Best	**12**	0	0	0
(kernel PCA)	Avg. Rank	**1.50**	3.25	2.62	3.12
Rep. Learning	Nb. Best	**25**	8	1	2
(kernel kSVD)	Avg. Rank	**1.44**	2.67	2.58	3.32



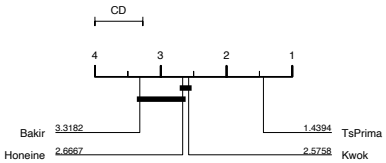
(a) Averaging



(b) Reconstruction (kernel PCA)



(c) Denoising (kernel PCA)



(d) Reconstruction (kernel k-SVD)

Figure 10: Nemenyi Tests.

5.5. Overall analysis

The experiments conducted in Sections 5.1 to 5.3 show that the proposed method TsPrima leads on almost all the datasets and through the three studied tasks to the best results. On the other hand, the performances obtained by the alternative methods seem slightly equivalent and lower than those obtained by TsPrima.

In particular, for time series averaging task, we can see in Table 2 that the centroids estimated by TsPrima lead to the highest within-class similarity on almost all the datasets; namely, each centroid obtained by TsPrima is in general the closest to the set of time series it represents. The analysis of the critical difference diagram given in Figure 2 indicates that the next best results are obtained respectively by Bakir, Honeine and Kwok methods. In addition, as the state of the art methods are connected by a solid bold line, their performances remain equivalent. From Figure 3, we can see that while all the methods succeed to restitute the centroids of some input classes (shown on the left column) as the class "6" of DIGITS and "S" of UPPER dataset, only TsPrima succeeds to estimate the centroids of the most challenging classes, as the "k" class of LOWER dataset and SPIRAL1.

For time series reconstruction, Table 3 shows that TsPrima leads to the highest reconstruction accuracies through almost all the datasets, followed by Honeine, Bakir and Kwok methods. Figure 4 indicates that there is no significant difference between the performances of the three state of the art methods (connected by a solid bold line). These results are assessed in Figure 5 that shows, for some input time series, the quality of the reconstructions obtained by TsPrima and the state of the art methods.

For the time series denoising task, we observe from Table 4 and for all the methods that the quality of the denoising decreases when the intensity of noise increases. This result is illustrated in Figure 6, that shows the denoising results of the time series "M" of UPPER dataset and of the highly noisy time series of SPIRAL2 dataset. In particular, note that that Kwok and TsPrima methods lead to the best results on SPIRAL2 data and seem less sensitive to noise than Honeine and Bakir methods.

Lastly, for time series representation learning task, Table 5 indicates that each studied method leads to the best sparse representations for at least some datatsets and that TsPrima performs better on almost all the datasets. Figure 8 shows the goodness of the sparse representations obtained. While all the methods succeed to sparse represent some input time series, the time series of "k" and "B" classes appear challenging for Honeine and Bakir methods. In Figure 9, we get a look on the quality of the learned atoms, that are involved into the reconstruction of the input samples. The first row gives for some input sample "k" (on the left), the sparse representations learned by each method. The

three next rows, provide the three first atoms involved into the reconstructions. We can see that while the first atom learned by TsPrima is nearly sufficient to sparse represent the "k" input sample, the state of the art methods need obviously more that one atom to sparse represent the input sample. Finally, the analysis of Figure 7 indicates that Honeine method performs equivalently than Kwok and Bakir, whereas the Kwok performances are significantly better that those of Bakir method.

Further comparisons (Table 6) are conducted in Section 5.4 to evaluate the efficiency of TsPrima related to the learned transformation R, regardless of the temporal embedding f_r. For averaging task, TsPrima, Honeine and Bakir lead equivalently to the best performances, followed by Kwok method (Figure 10 (a)). From these results we can conjecture that, linear transformations seem sufficient to achieve good pre-image estimations for averaging task on these datasets, as both linear and non linear approaches (TsPrima, Honeine, Bakir) perform equivalently. Furthermore, while Honeine and Bakir involve the whole datasets for the centroid pre-image estimations, Kwok uses a subset of samples into the neighbourhood of φ, which may explain the slightly lower performances of Kwok method. Note that, although TsPrima involves, similarly to Kwok method, fewer samples into the neighbourhood of φ, it succeeds to reach the best performances thanks to the efficiency of the learned transformation R. For the remaining tasks reconstruction, denoising and representation learning, TsPrima achieves the highest performances, followed by far by Honeine, Kwok and Bakir (Figure 10 (b), (c) and (d)), which assesses the crucial contribution of the learned transformations R of TsPrima. Lastly, of particular note is that Honeine and Bakir that involve the whole training samples induce much computations, specifically for the time series embedding process, than Kwok and TsPrima that require fewer samples into the neighbourhood of φ.

Finally, as all the studied methods propose closed-form solutions, they lead to comparable complexities. However, for large data, TsPrima and Kwok methods are expected to perform faster as requiring fewer samples on the neighborhood of φ than Honeine and Bakir that involve the whole samples for pre-image estimation. Note that the complexity of the proposed solutions is mainly related to the matrix inversion operator. In Kwok method, the inversion of ZZ^T required in Eq. (21), with Z of dimension $(q \times n)$ and n is the neighbourhood size, induces a complexity of $O(q^2n) + O(q^3)$; as q is in general small and fixed beforehand, the overall complexity is about $O(n)$. For Honeine method, Eq. (24) requires two inversions of XX^T and K, which induces, respectively, a complexity of $O(d^2N) + O(d^3)$ and $O(N^3)$, that leads to an overall complexity of $O(N^3)$. For Bakir method, Eq. (26), requires the inversion of the Gram matrix, which leads to a complexity of $O(N^3)$. For TsPrima, Eq. (32) involves the inversion of XX^T, with X is of dimension $(d \times n)$, d is the time series length and n is the neighbourhood size. The induced complexity is of $O(d^2n) + O(d^3)$. For the time series embedding part, the complexity is mainly related to the time warping function which is of order $O(d^2n)$. As d is in general higher that

the neighbourhood size n, the overall complexity for TSPrima is about $O(d^3)$. To sum up, as the neighbourhood size $n << N$ and $d << N$ (for not extra large time series), the complexity induced by both Kwok and TSPrima remains lower that the one of Honeine and Bakir. Note that, the Honeine method can be developed to consider only the neighbourhoods instead of all samples.

6. Conclusion

This work proposes TsPrima, a new closed-form pre-image estimation method for time series analytics under kernel machinery. The method consists of two stages. In the first step, we define a time warp embedding function, driven by distance constraints in the feature space, that allows to embed the time series in a metric space. In the second step, the time series pre-image estimation is cast as learning a linear (or a nonlinear) transformation to ensure a local isometry between the time series embedding space and the feature space. Extensive experiments show the efficiency and the benefits of TsPrima through three major tasks that require pre-image estimation: 1) time series averaging, 2) time series reconstruction and denoising and 3) time series representation and dictionary learning. Future work will focus on using pre-image estimation methods to enhance the interpretability and the computation of deep learning tasks for time series, sequence and graph analytics.

Acknowledgment

This work is supported by the French National Research Agency through the projects locust (ANR-15-CE23-0027-02) and APi (ANR-18-CE23-0014).

References

[1] Aharon, M., Elad, M., Bruckstein, A., 2006. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Transactions on signal processing 54, 4311–4322.

[2] Bahlmann, C., Haasdonk, B., Burkhardt, H., 2002. Online handwriting recognition with support vector machines-a kernel approach, in: Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, IEEE. pp. 49–54.

[3] Bakır, G.H., Weston, J., Schölkopf, B., 2004. Learning to find pre-images. Advances in neural information processing systems 16, 449–456.

[4] Balasubramanian, K., Yu, K., Lebanon, G., 2013. Smooth sparse coding via marginal regression for learning sparse representations, in: International Conference on Machine Learning, pp. 289–297.

[5] Bengio, S., Pereira, F., Singer, Y., Strelow, D., 2009. Group sparse coding, in: Advances in neural information processing systems, pp. 82–89.

[6] Bianchi, F.M., Livi, L., Mikalsen, K.Ø., Kampffmeyer, M., Jenssen, R., 2018. Learning representations for multivariate time series with missing data using temporal kernelized autoencoders. arXiv preprint arXiv:1805.03473 .

[7] Chen, M., AlRegib, G., Juang, B.H., 2012. 6dmg: A new 6d motion gesture database, in: Proceedings of the 3rd Multimedia Systems Conference, ACM. pp. 83–88.

[8] Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G., 2015. The ucr time series classification archive URL: https://www.cs.ucr.edu/~eamonn/time_series_data/.

[9] Cloninger, A., Czaja, W., Doster, T., 2017. The pre-image problem for laplacian eigenmaps utilizing l 1 regularization with applications to data fusion. Inverse Problems 33, 074006.

[10] Cuturi, M., Vert, J.P., Birkenes, O., Matsui, T., 2007. A kernel for time series based on global alignments, in: 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07, IEEE. pp. II–413.

[11] Dau, H.A., Keogh, E., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G., Hexagon-ML, 2018. The ucr time series classification archive. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

[12] Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine learning research 7, 1–30.

[13] Do, C.T., Douzal-Chouakria, A., Marié, S., Rombaut, M., Varasteh, S., 2017. Multi-modal and multi-scale temporal metric learning for a robust time series nearest neighbors classification. Information Sciences 418, 272–285.

[14] Frambourg, C., Douzal-Chouakria, A., Gaussier, E., 2013. Learning multiple temporal matching for time series classification, in: International Symposium on Intelligent Data Analysis, Springer. pp. 198–209.

[15] Honeine, P., Richard, C., 2011a. A closed-form solution for the pre-image problem in kernel-based machines. Journal of Signal Processing Systems 65, 289 – 299. URL: https://link.springer.com/article/10.1007/s11265-010-0482-9, doi:10.1007/s11265-010-0482-9.

[16] Honeine, P., Richard, C., 2011b. Preimage problem in kernel-based machine learning. IEEE Signal Processing Magazine 28, 77 – 88. URL: https://ieeexplore.ieee.org/document/5714388, doi:10.1109/MSP.2010.939747.

[17] Jenatton, R., Mairal, J., Obozinski, G., Bach, F.R., 2010. Proximal methods for sparse hierarchical dictionary learning., in: ICML, Citeseer. p. 2.

[18] Kwok, J.Y., Tsang, I.H., 2004. The pre-image problem in kernel methods. IEEE transactions on neural networks 15, 1517–1525.

[19] Paparrizos, J., Franklin, M.J., 2019. Grail: efficient time-series representation learning. Proceedings of the VLDB Endowment 12, 1762–1777.

[20] Pati, Y.C., Rezaiifar, R., Krishnaprasad, P.S., 1993. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, in: Proceedings of 27th Asilomar conference on signals, systems and computers, IEEE. pp. 40–44.

[21] Sakoe, H., Chiba, S., 1978. Dynamic programming algorithm optimization for spoken word recognition. IEEE transactions on acoustics, speech, and signal processing 26, 43–49.

[22] Schlegel, K., 2019. When is there a representer theorem? Journal of Global Optimization 74, 401–415. doi:10.1007/s10898-019-00767-0.

[23] Schölkopf, B., Mika, S., Smola, A., Rätsch, G., Müller, K.R., 1998. Kernel pca pattern reconstruction via approximate pre-images, in: International Conference on Artificial Neural Networks, Springer. pp. 147–152.

[24] Scholkopf, B., Smola, A.J., 2001. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press.

[25] Shimodaira, H., Noma, K.i., Nakai, M., Sagayama, S., 2002. Dynamic time-alignment kernel in support vector machine, in: Advances in neural information processing systems, pp. 921–928.

[26] Soheily-Khah, S., Chouakria, A.D., Gaussier, E., 2015. Progressive and iterative approaches for time series averaging., in: AALTD@ PKDD/ECML.

[27] Soheily-Khah, S., Douzal-Chouakria, A., Gaussier, E., 2016. Generalized k-means-based clustering for temporal data under weighted and kernel time warp. Pattern Recognition Letters 75, 63–69.

[28] Van Nguyen, H., Patel, V.M., Nasrabadi, N.M., Chellappa, R., 2012. Kernel dictionary learning, in: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 2021–2024.

[29] Wu, L., Yen, I.E.H., Yi, J., Xu, F., Lei, Q., Witbrock, M., 2018. Random warping series: A random features method for time-series embedding. arXiv preprint arXiv:1809.05259 .

[30] Yazdi, S.V., Douzal-Chouakria, A., 2018. Time warp invariant ksvd: Sparse coding and dictionary learning for time series under time warp. Pattern Recognition Letters 112, 1–8.

[31] Yazdi, S.V., Douzal-Chouakria, A., Gallinari, P., Moussallam, M., 2018. Time warp invariant dictionary learning for time series clustering: application to music data stream analysis, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer. pp. 356–372.

[32] Zhu, F., Honeine, P., 2016. Bi-objective nonnegative matrix factorization: Linear versus kernel-based models. IEEE Transactions on Geoscience and Remote Sensing 54, 4012 – 4022. URL: https://ieeexplore.ieee.org/document/7448928, doi:10.1109/TGRS.2016.2535298.