Time warp invariant *k*SVD: Sparse coding and dictionary learning for time series under time warp

Saeed Varasteh Yazdi[a], Ahlame Douzal-Chouakria[a,*]

[a] *Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, F-38000 Grenoble France*

Pattern Recognition Letters, DOI: https://doi.org/10.1016/j.patrec.2018.05.017

ABSTRACT

Learning dictionary for sparse representing time series is an important issue to extract latent temporal features, reveal salient primitives and sparsely represent complex temporal data. Time series are challenging data, they are often of different durations, may be composed of local or global salient events, that may arise with varying delays at different time stamps. This paper addresses the sparse coding and dictionary learning for such challenging time series. For that, we propose a non linear time warp invariant *k*SVD (TWI-*k*SVD) where both input samples and dictionary atoms may have different lengths while involving varying delays. For the sparse coding problem, we propose an efficient time warp invariant orthogonal matching pursuit based on a new cosine maximisation time warp operator and induced sibling atoms. For the dictionary learning, thanks to a rotation transformation between each atom and its sibling atoms, a singular value decomposition is used to jointly approximate the coefficients and update the dictionary. The proposed method is confronted to major shift invariant, convolved and kernel dictionary learning methods on several challenging character and digit handwritten trajectories. The experiments conducted show the potential of TWI-*k*SVD to efficiently sparse represent time series and to extract latent discriminative primitives for time series classification.

1. Introduction

Sparse coding and dictionary learning become popular methods in machine learning and pattern recognition for a variety of tasks as feature extraction (Mailhé et al., 2008; Barthélemy et al., 2012), reconstruction, denoising, compressed sensing (Aharon et al., 2006; Donoho, 2006) and classification (Wright et al., 2009; Guha and Ward, 2012). The aim of sparse coding methods is to represent input samples as a linear combination of few basis functions called *atoms* composing a given dictionary. Sparse coding is generally formalised as an optimisation problem that minimises the error of the reconstruction under l_0 or l_1 sparsity constraint. The l_0 constraint, that controls the maximum number of involved atoms, leads to a non convex and NP-hard problem. This problem can however be solved efficiently by using the matching pursuit method (Mallat and Zhang, 1993) or its orthogonal variant (Pati et al., 1993). Relaxing the sparsity constraint from l_0 to l_1 norm yields a convex sparse coding problem, also known as

[*] Corresponding author: Ahlame.Douzal@imag.fr

LASSO problem (Tibshirani et al., 2013). In sparse coding, the used dictionary may be selected among pre-specified family of basis functions as, among others, Fourier, Wavelets (Mallat, 1999), Curvelets (Starck et al., 2002), Contourlets (Do and Vetterli, 2005) and Gabor functions (Yang and Zhang, 2010). Although these dictionaries allow fast transforms, their reconstruction potential is tightly related to the nature of the data. For instance, Wavelets show efficient reconstruction for natural images and textures (Ophir et al., 2011), Curvelets for edges (Yang et al., 2013) and Gabor for sounds (Ataee et al., 2010). The alternative to the above basis functions is to use a dictionary learning approach to learn, from the input data, a set of atoms to sparse represent the input samples. To solve that dictionary learning problem most approaches alternate between two steps: 1) keep the dictionary fixed and find the sparse representation using a sparse approximation algorithm, *e.g.*, orthogonal matching pursuit (OMP), 2) keep the representation fixed and update the dictionary, either all the atoms at once by using for instance MOD (method of optimal directions) (Engan et al., 1999) or one atom at a time as in *k*SVD (Aharon et al., 2006). In particular, *k*SVD uses a singular value decomposition to learn jointly the dictionary as well as the sparse coefficients. *k*SVD can be viewed as a generalisation of *k*-means algorithm that relaxes the *assignment* constraint to represent each input sample by a linear combination of few representative atoms (*i.e.*, the centroids) instead by using only one centroid.

Last decade was marked by the growing emergence of applications involving complex temporal data (*e.g.*, time series, sequences, traces). What makes temporal data particularly challenging is that the observations usually arise with varying delays, with the salient temporal features related to only a subset of the observations and localised at different time stamps. Sparse coding and dictionary learning approaches appears as an essential issue to extract latent temporal features, reveal salient primitives and sparsely represent complex temporal features.

Several works have been proposed to address shift invariances in sparse coding and dictionary learning. Without being exhaustive, in (Lewicki and Sejnowski, 1999) a few set of kernel basis functions of small lengths is used to learn a linear combination of kernels and the time stamps (*i.e.* translation parameters) where they should be involved for sparse representation. For that, the model is formalised in a convolutional form based on a circulant matrix to convolve the kernel functions at all translation positions. The convolved kernel functions are first zero padded to make them of the same length as the input samples. In (Barchiesi and Plumbley, 2011) a block coordinate descent method is proposed to learn a convolved dictionary based on learning the impulse of a convolutive Toplitz matrix. In (Elad and Aharon, 2006) and (Mailhé et al., 2008) a shift invariant *k*SVD is proposed based on a translation operator that allows to generate, for each basic atom, all its translated atoms. In the first step, standard sparse coding algorithm is used to estimate the sparse coefficients and the translated positions of the involved atoms. In the update dictionary step, the sparse codes and the translation parameters are kept fixed and the atoms are updated. In the same spirit, (Jost et al., 2006) formalise the translation-invariant dictionary learning as a convex optimisation problem to estimate atoms and their time-translation to maximise their correlation to the training data under uncorrelated atom constraints.

Challenging time series often involve varying delays through time that may be difficult to tackle with time shift and translation linear transformations. In this paper, we propose a non linear time warp invariant *k*SVD (TWI-*k*SVD) where both input samples and atoms define time series that may have different lengths and involve varying delays. For that, in the first sparse coding step, we propose a time warp invariant orthogonal matching pursuit (TWI-OMP) based on a new cosine maximisation time warp operator (COSTW) and the induced sibling atoms. In the second step, thanks to a rotation transformation between each atom and its sibling atoms, a singular value decomposition is used to jointly approximate the coefficients and update the dictionary. The potential of the proposed TWI-*k*SVD method is confronted to major alternative approaches on several character and digit handwritten trajectories, that define naturally time series of different lengths that include varying delays. The reminder of the paper is organised as follows. Section 2 formalises the sparse coding and dictionary learning problem and defines the orthogonal matching pursuit (OMP) and the *k*SVD methods. Sections 3 and 4 formalise, respectively, the problem of sparse coding and *k*SVD for time series under time warp invariances, then present and detail the proposed solutions TWI-OMP and TWI-*k*SVD. Section 5 presents the experiments conducted and discusses the obtained results.

2. Sparse coding and dictionary learning

This Section formalises standard sparse coding and dictionary learning problems and presents two standard efficient solutions that we focus on in this paper. In the following, lower case letters are used for scalars, bold lower case letters for vectors and upper case letters for matrices.

Sparse coding. Given the input sample $\boldsymbol{x} \in \mathbb{R}^p$ and a dictionary $D \in \mathbb{R}^{p\times K} : D = [\boldsymbol{d}_1, ..., \boldsymbol{d}_K]$ composed of K atoms $\boldsymbol{d}_j \in \mathbb{R}^p$, the objective of sparse coding problem is to represent sparsely $\boldsymbol{x}$ by a linear combination of a few atoms of D. This problem is formalised as minimising the error of reconstruction of $\boldsymbol{x}$ under a sparsity constraint:

$$\underset{\boldsymbol{\alpha}}{min}\, ||\boldsymbol{x} - D\,\boldsymbol{\alpha}||_2^2 \;\; s.t.||\boldsymbol{\alpha}||_0 \leq \tau. \tag{1}$$

where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_K)^t \in \mathbb{R}^K$ is the sparse code of $\boldsymbol{x}$ and D is in general a predefined (*e.g.*, Fourier, Wavelet, Gabor basis functions) and overcomplete (*i.e.*, $p << K$) dictionary. The l_0 sparsity constraint in Eq. (1) ensures to limit the maximum number of involved atoms to τ. Although the l_0-norm renders the problem formalised in Eq. (1) non convex and NP-hard, it can be efficiently solved via matching pursuit (Mallat and Zhang, 1993) or its orthogonal variant OMP (Pati et al., 1993). The main idea of OMP method is to select at each iteration the atom $\boldsymbol{d}_j$ that is highly correlated to the input sample or to its residual part. The coefficient α_j, obtained by an orthogonal projection on the sub-space defined by the yet selected atoms, defines the contribution of $\boldsymbol{d}_j$ to reconstruct $\boldsymbol{x}$. The process is reiterated until the maximum number τ of atoms is reached. Algorithm 1 gives the main steps of the OMP method. It

Algorithm 1 OMP

```
Input: $\boldsymbol{x}$, $D$, $\tau$
Output: $\alpha$
$\boldsymbol{r} = \boldsymbol{x}, \Omega = \{\phi\}$
while $|\Omega| \leq \tau$ do
    Select the atom $\boldsymbol{d}_j$ ($j \notin \Omega$) that maximises $\frac{\boldsymbol{r}^T \boldsymbol{d}_j}{||\boldsymbol{r}||_2 ||\boldsymbol{d}_j||_2}$
    Update the set of selected atoms: $\Omega = \Omega \cup \{j\}$
    Update the coefficients: $\boldsymbol{\alpha}_\Omega = (D_\Omega^T D_\Omega)^{-1}(D_\Omega^T \boldsymbol{x})$
    { $D_\Omega$ is the sub-dictionary of the selected atoms and $\boldsymbol{\alpha}_\Omega$ the related coefficients}
    Estimate the residual: $\boldsymbol{r} = \boldsymbol{x} - D_\Omega \boldsymbol{\alpha}_\Omega$
end while
```

is worth noting that although initialising the dictionary with a given family of basis functions (*e.g.*, Fourier, Wavelets) hastens the process, the sparse coding results remain in general more precise when the dictionary is learned. Several efficient approaches for dictionary learning are proposed in the literature, among them the kSVD method that we detail in the following.

Dictionary learning. Let $X \in \mathbb{R}^{p \times N} : X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_N]$ be the matrix giving the description of N input samples, with $\boldsymbol{x}_i \in \mathbb{R}^p$. The dictionary learning problem, that generalises the sparse coding given in Eq. (1), can be formalised as learning both the sparse codes and the dictionary D to minimise the error of reconstruction of a set of input samples:

$$\underset{A,D}{min} ||X - DA||_F^2 \tag{2}$$

$$\text{s.t. } \forall i \; ||\boldsymbol{\alpha}_i||_0 \leq \tau, \quad \forall j \;\; ||\boldsymbol{d}_j||_2 = 1$$

where $A \in \mathbb{R}^{K \times N} : A = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_N]$ gives the sparse codes $\boldsymbol{\alpha}_i \in \mathbb{R}^K$ of samples $\boldsymbol{x}_i$ and $\boldsymbol{d}_j \in \mathbb{R}^p$ is the jth atom of unit l_2-norm. The above optimisation problem is not convex in both A and D, that is resolved in general by using a block-coordinate-descent method. This method consists of alternating two phases: 1) keep D fixed and learn the sparse codes A and 2) keep A fixed and learn the dictionary D. We focus in the following on the highly used kSVD approach that follows the same iterative principle to learn A and D, as detailed in the following.

The first step of kSVD is purely a sparse coding process to learn A. Mainly, it keeps D fixed and decomposes the problem given in Eq. (2) into N distinct problems of the form given in Eq. (1), that can be solved separately to estimate $\boldsymbol{\alpha}_i$ for each $\boldsymbol{x}_i$ by using for instance the OMP approach (line 2 to 4 in Algorithm 2). In the second step and based on the learned A, the atoms $\boldsymbol{d}_j$ and its related coefficients $\boldsymbol{\alpha}_{j.}$ are updated iteratively, one at a time (line 5 to 11 in Algorithm 2). For that, the objective function given in Eq. (2) is formulated as:

$$||X - DA||_F^2 = ||X - \sum_{j=1}^{K} \boldsymbol{d}_j \boldsymbol{\alpha}_{j.}||_F^2 \tag{3}$$

$$||(X - \sum_{j \neq k} \boldsymbol{d}_j \boldsymbol{\alpha}_{j.}) - \boldsymbol{d}_k \boldsymbol{\alpha}_{k.}||_F^2 = ||E_k - \boldsymbol{d}_k \boldsymbol{\alpha}_{k.}||_F^2 \tag{4}$$

Algorithm 2 kSVD(X, D, τ)

```
     Input: $X, D, \tau$
     Output: $D, A$
 1:  repeat
 2:      for $i = 1, ..., N$ do
 3:          $\boldsymbol{\alpha}_i$ =OMP($\boldsymbol{x}_i, D, \tau$)
 4:      end for
 5:      for $k = 1, ..., K$ do
 6:          Estimate $E_k = X - \sum_{j \neq k} \boldsymbol{d}_j \boldsymbol{\alpha}_{j.}$
 7:          $\Omega_k = \{i \,/\, \alpha_{ki} \neq 0, \quad i = 1, ..., N\}$
 8:          Define $E_k^{\Omega_k}$ as the restriction of $E_k$ to $\Omega_k$
 9:          Apply an SVD on $E_k^{\Omega_k} = U \Sigma V^t$
10:          Update $\boldsymbol{d}_k = \boldsymbol{u}_1$ and $\alpha_{k.}^{\Omega_k} = \sigma_1 \boldsymbol{v}_1^t$
11:      end for
12:  until Convergence (stopping rule)
```

where DA in Eq. (3) is expressed as the sum of K rank-1 matrices, each one giving the sparse representation of X involving one atom. $\boldsymbol{\alpha}_{j.} = (\alpha_{j1}, ..., \alpha_{jN})$ denotes the jth row of A, it provides the contributions of the atom $\boldsymbol{d}_j$ to reconstruct the N samples. The matrix $E_k \in \mathbb{R}^{p \times N}$ in Eq. (4) gives the error of reconstruction for the N samples based on all the atoms except $\boldsymbol{d}_k$ (line 6). To restrain the error of reconstruction to the samples that involve $\boldsymbol{d}_k$, the columns of E_k related to $\boldsymbol{x}_i$ with $\alpha_{ki} = 0$ are removed. Let $E_k^{\Omega_k} \in \mathbb{R}^{p \times |\Omega_k|}$ be such restricted matrix, with Ω_k the set of samples involving $\boldsymbol{d}_k$ (lines 7-8). To update the kth atom, an SVD is used to estimate the closest rank-1 matrix that approximates $E_k^{\Omega_k}$ and minimises the reconstruction error given in Eq. (2). The first left-singular vector $\boldsymbol{u}_1$ (first column of U), first singular value σ_1 and the first right-singular vector $\boldsymbol{v}_1$ (first column of V) are used to update the atom $\boldsymbol{d}_k$ as well as the coefficients $\boldsymbol{\alpha}_{k.}^{\Omega_k}$ of the samples belonging to Ω_k (lines 9-10). Once the K atoms updated, the step 1 and 2 are iterated until convergence (*i.e.*, stabilisation of the error of the reconstruction). In the conducted experiments the process converges in less than 20 iterations. Algorithm 2 summarises the main steps of the kSVD approach. Note that, in the second step of kSVD, the update of the dictionary is done jointly with an update of its related coefficients, resulting in accelerated convergence.

3. Time warp invariant sparse coding

In this Section, we give the formalisation of time series sparse coding under time warp invariances. To resolve this problem, we present a new operator COSTW that ensures cosine maximisation between time series under time warp and give the recurrence relation that ensures its computation in quadratic complexity. Finally, thanks to COSTW and to the induced sibling atoms, we present a time warp invariant OMP (TWI-OMP), as a solution for the time series sparse coding under time warp problem.

Let $\boldsymbol{x} = (x_1, ..., x_q)^t$ be an input time series and $D = \{\boldsymbol{d}_j\}_{j=1}^K$ the dictionary defined as a set of K time series atoms $\boldsymbol{d}_j \in \mathbb{R}^{p_j}$. Note that both $\boldsymbol{x}$ and $\boldsymbol{d_j}$ are time series of different lengths that may involve varying delays. The dictionary may be initialised from a training set or from a family of basis functions. The sparse coding problem under time warp invariances can be formalised as:

$$\underset{\alpha,\Delta}{min} \|\boldsymbol{x} - \sum_{j=1}^{K} \Delta_j\, \boldsymbol{d}_j\, \alpha_j\|_2^2 \tag{5}$$

$$\text{s.t. :} \quad \|\boldsymbol{\alpha}\|_0 \leq \tau$$

$$\Delta_j \in \{0,1\}^{q \times p_j}$$

$$\Delta_j \mathbf{1}_{p_j} = \mathbf{1}_q$$

where $\Delta = \{\Delta_j\}_{j=1}^{K}$ with Δ_j a binary matrix that encodes the alignment between $\boldsymbol{x}$ and $\boldsymbol{d}_j$. Thus, the problem defined in Eq. (5) remains to estimate the coefficients $\boldsymbol{\alpha}$ to sparse code $\boldsymbol{x}$ as a linear combination of the warped atoms $\Delta_j\, \boldsymbol{d}_j$. The last constraint is a row normalisation of the estimated Δ_j that ensures for $\boldsymbol{x}$ equally weighted time stamps. To resolve this problem, we propose an extended variant of OMP that can be mainly summarised in the following steps:

1. For each $\boldsymbol{d}_j$, estimate Δ_j by dynamic programming to maximise the cosine between $\boldsymbol{x}$ and $\boldsymbol{d}_j$.
2. Use the projector Δ_j to align $\boldsymbol{d}_j$ to $\boldsymbol{x}$. Let $\boldsymbol{d}_j^s = \Delta_j\, \boldsymbol{d}_j \in \mathbb{R}^q$ be the obtained aligned atom of the same length as $\boldsymbol{x}$, denoted in the following as $\boldsymbol{d}_j$'s *sibling* atom.
3. Estimate the sparse code $\boldsymbol{\alpha}$ based on the sibling atoms.

For that and to estimate the projectors $\Delta_j\ \forall\ j \in \{1, ...K\}$, first we propose a new operator COSTW to estimate the cosine between two time series under time warp. To the best of our knowledge, this is the first time that the cosine operator is generalised to time series under time warp. Then, we present a time warp invariant OMP (TWI-OMP), that extends the standard OMP approach, to sparse code time series under non linear time warping transformations.

*3.1. Cosine maximisation time warp (*COSTW*)*

The problem of estimating the cosine between two time series that involve varying delays amounts to learning the time series alignment that maximises their cosine. In the following, we recall the standard definition of time series alignment, then formalise the cosine maximisation under time warp problem. Finally, we propose a recurrence relation that allows to perform the computation of the alignment in quadratic complexity.

Let $\mathbf{x} = (x_1, ..., x_{T_x})$, $\mathbf{y} = (y_1, ..., y_{T_y})$ be two time series of length T_x and T_y. An alignment $\boldsymbol{\pi}$ of length $|\boldsymbol{\pi}| = m$ between $\mathbf{x}$ and $\mathbf{y}$ 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 $\{1, .., T_x\}$ and $\{1, .., T_y\}$ respectively obey to the following boundary and monotonicity conditions:

$$1 = \pi_1(1) \leq \pi_1(2) \leq ... \leq \pi_1(m) = T_x$$
$$1 = \pi_2(1) \leq \pi_2(2) \leq ... \leq \pi_2(m) = T_y$$

and $\forall\, l \in \{1, ..., m\}$,

$$\pi_1(l+1) \leq \pi_1(l) + 1 \text{ and } \pi_2(l+1) \leq \pi_2(l) + 1$$
$$(\pi_1(l+1) - \pi_1(l)) \; + \; (\pi_2(l+1) - \pi_2(l)) \geq 1$$

Intuitively, an alignment $\boldsymbol{\pi}$ between $\mathbf{x}$ and $\mathbf{y}$ describes a way to associate each element of $\mathbf{x}$ to one or more elements of $\mathbf{y}$ and vice versa. Such an alignment can be conveniently represented by a path in the $T_x \times T_y$ grid, where the above monotonicity conditions ensure that the path is neither going back nor jumping. We will denote $\mathcal{A}$ as the set of all alignments between two time series. The cosine maximisation under time warp can be formalised as:

$$\begin{aligned}
\textsc{costw}(\boldsymbol{x}, \boldsymbol{y}) &= s(\boldsymbol{\pi}^*) \qquad (6)\\
\boldsymbol{\pi}^* &= \arg\max_{\boldsymbol{\pi} \in \mathcal{A}} s(\boldsymbol{\pi})\\
s(\boldsymbol{\pi}) &= \cos(\boldsymbol{x}_{\pi_1}, \boldsymbol{y}_{\pi_2}) = \frac{< \boldsymbol{x}_{\pi_1}, \boldsymbol{y}_{\pi_2} >}{||\boldsymbol{x}_{\pi_1}||_2 \; ||\boldsymbol{y}_{\pi_2}||_2}\\
&= \frac{\sum_{i=1}^{|\boldsymbol{\pi}|} x_{\pi_1(i)}\, y_{\pi_2(i)}}{\sqrt{\sum_{i=1}^{|\boldsymbol{\pi}|} x^2_{\pi_1(i)}} \; \sqrt{\sum_{i=1}^{|\boldsymbol{\pi}|} y^2_{\pi_2(i)}}}
\end{aligned}$$

where s is the cost function of an alignment $\boldsymbol{\pi}$ and $\cos(\boldsymbol{x}_{\pi_1}, \boldsymbol{y}_{\pi_2})$ is the standard cosine between the aligned time series $\boldsymbol{x}_{\pi_1} = (x_{\pi_1(1)}, ..., x_{\pi_1(m)})$ and $\boldsymbol{y}_{\pi_2} = (y_{\pi_2(1)}, ..., y_{\pi_2(m)})$. The solution of the Eq. (6) is obtained by dynamic programming, where the main trick is to define a useful recurrence relation for the cosine estimation, as detailed here after. Let $\boldsymbol{x}_{N+1} = (x_1, ..., x_{N+1})$, $\boldsymbol{y}_{N+1} = (y_1, ..., y_{N+1})$ be two time series of length $N+1$, assumed without delays for the sake of clarity. Let $\boldsymbol{x}_N$, $\boldsymbol{y}_N$ be the sub-time series composed of the N first elements of $\boldsymbol{x}_{N+1}$, $\boldsymbol{y}_{N+1}$, respectively. To estimate recursively $\cos(\boldsymbol{x}_{N+1}, \boldsymbol{y}_{N+1})$, the following incremental relation can be established between $\cos(\boldsymbol{x}_{N+1}, \boldsymbol{y}_{N+1})$ and $\cos(\boldsymbol{x}_N, \boldsymbol{y}_N)$:

$$\cos(\boldsymbol{x}_N, \boldsymbol{y}_N) = \frac{< \boldsymbol{x}_N, \boldsymbol{y}_N >}{\sqrt{||\boldsymbol{x}_N||_2^2} \; \sqrt{||\boldsymbol{y}_N||_2^2}} \qquad (7)$$

$$\cos(\boldsymbol{x}_{N+1}, \boldsymbol{y}_{N+1}) = \frac{< \boldsymbol{x}_N, \boldsymbol{y}_N > \; + x_{N+1} y_{N+1}}{\sqrt{||\boldsymbol{x}_N||_2^2 + x_{N+1}^2} \; \sqrt{||\boldsymbol{y}_N||_2^2 + y_{N+1}^2}} \qquad (8)$$

where the estimation of $\cos(\boldsymbol{x}_{N+1}, \boldsymbol{y}_{N+1})$ in Eq. (8) is deduced from the triplet of scalars $(< \boldsymbol{x}_N, \boldsymbol{y}_N >, ||\boldsymbol{x}_N||_2^2, ||\boldsymbol{y}_N||_2^2)$ related to $\cos(\boldsymbol{x}_N, \boldsymbol{y}_N)$ and the couple of values (x_{N+1}, y_{N+1}).

For time series including delays and based on the incremental property given in Eq. (8), we introduce the computation and recurrence relation that allows to estimate the alignment $\boldsymbol{\pi}^*$ that maximise $\textsc{costw}(\boldsymbol{x}, \boldsymbol{y})$ in Eq. (6).

Computation and recurrence relation. Let us define $M \in \mathbb{R}^{T_x \times T_y}$ the matrix mapping $\boldsymbol{x}$ and $\boldsymbol{y}$ of general term $M_{i,j} = (< \boldsymbol{x}_{i,j}, \boldsymbol{y}_{i,j} >, ||\boldsymbol{x}_{i,j}||_2^2, ||\boldsymbol{y}_{i,j}||_2^2)$, where $\boldsymbol{x}_{i,j}, \boldsymbol{y}_{i,j}$ are the sub-time series aligned to maximise the cosine at $M_{i,j}$ cell. Let $f(M_{i,j}) = \frac{<\boldsymbol{x}_{i,j}, \boldsymbol{y}_{i,j}>}{\sqrt{||\boldsymbol{x}_{i,j}||_2^2} \; \sqrt{||\boldsymbol{y}_{i,j}||_2^2}}$ be the cost function that associates to each M_{ij} the related cosine value. Based on the incremental property established in (8), computing recursively for $(i, j) \in \{1, ..., T_x\} \times \{1, ..., T_y\}$ the terms $M_{i,j}$:

$$\begin{aligned}
M_{1,1} &= (x_1 y_1, x_1^2, y_1^2)\\
\forall i \geq 2, j = 1, M_{i,1} &= (< \boldsymbol{x}_{i-1,1}, \boldsymbol{y}_{i-1,1} > + x_i y_1, ||\boldsymbol{x}_{i-1,1}||_2^2 + x_i^2, ||\boldsymbol{y}_{i-1,1}||_2^2 + y_1^2)\\
\forall j \geq 2, i = 1, M_{1,j} &= (< \boldsymbol{x}_{1,j-1}, \boldsymbol{y}_{1,j-1} > + x_1 y_{j-1}, ||\boldsymbol{x}_{1,j-1}||_2^2 + x_1^2, ||\boldsymbol{y}_{1,j-1}||_2^2 + y_j^2)\\
\forall i \geq 2, j \geq 2, M_{i,j} &= \arg\max(\\
& f(< \boldsymbol{x}_{i,j-1}, \boldsymbol{y}_{i,j-1} > + x_i y_j, ||\boldsymbol{x}_{i,j-1}||_2^2 + x_i^2, ||\boldsymbol{y}_{i,j-1}||_2^2 + y_j^2),\\
& f(< \boldsymbol{x}_{i-1,j}, \boldsymbol{y}_{i-1,j} > + x_i y_j, ||\boldsymbol{x}_{i-1,j}||_2^2 + x_i^2, ||\boldsymbol{y}_{i-1,j}||_2^2 + y_j^2),\\
& f(< \boldsymbol{x}_{i-1,j-1}, \boldsymbol{y}_{i-1,j-1} > + x_i y_j, ||\boldsymbol{x}_{i-1,j-1}||_2^2 + x_i^2, ||\boldsymbol{y}_{i-1,j-1}||_2^2 + y_j^2))
\end{aligned}$$

we obtain $\textsc{costw}(\boldsymbol{x}, \boldsymbol{y}) = f(M_{T_x, T_y})$ with a quadratic complexity of $O(T_x T_y)$. Note that, the three first equations give the first row and column update rules, the fourth equation gives the recurrence formula that retains among the triplets $M_{i-1,j}$, $M_{i,j-1}$ and $M_{i-1,j-1}$ the one that maximises the cosine at $M_{i,j}$.

3.2. *Time warp invariant* OMP *(*TWI-OMP*)*

Based on the defined COSTW, let us present the time warp invariant OMP (TWI-OMP) that extends OMP to deal with time series input samples and atoms under varying delays. The proposed TWI-OMP follows the three steps given in Section 3. First, perform a COSTW between $\boldsymbol{x}$ and each $\boldsymbol{d}_j$. Let $\{\Delta_j\}_{j=1}^K$ be the induced alignment matrices. Select the atom $\boldsymbol{d}_j$ that maximises $\textsc{costw}(\boldsymbol{x}, \boldsymbol{d}_j)$ (line 3-4 in Algorithm 3). Let $\boldsymbol{d}_j^s = \Delta_j \, \boldsymbol{d}_j$ be the $\boldsymbol{d}_j$'s sibling atom, update the dictionary $S_\Omega = [\boldsymbol{d}_j^s]_{j \in \Omega}$ of the yet selected atoms $\boldsymbol{d}_j$ (line 5). The updated S_Ω is then used to estimate the coefficients as in the standard OMP (line 6-7).

Algorithm 3 TWI-OMP($\boldsymbol{x}$, D, τ)

```
    Input: $\boldsymbol{x}$, $D = \{\boldsymbol{d}_j\}_{j=1}^K$, $\tau$
    Output: $\alpha$, $\Delta$
    $\boldsymbol{r} = \boldsymbol{x}, \Omega = \{\phi\}$
    while $|\Omega| \leq \tau$ do
        For all $j \notin \Omega$, perform COSTW($\boldsymbol{r}, \boldsymbol{d}_j$) and set $\Delta_j$
        Select the atom $\boldsymbol{d}_j$ ($j \notin \Omega$) that maximises COSTW($\boldsymbol{r}, \boldsymbol{d}_j$)
        Update the set of selected atoms $\Omega = \Omega \cup \{j\}$ and $S_\Omega = [\boldsymbol{d}_j^s]_{j \in \Omega}$
        Update the coefficients: $\boldsymbol{\alpha}_{\boldsymbol{\Omega}} = (S_\Omega^T S_\Omega)^{-1}(S_\Omega^T \boldsymbol{x})$
        Estimate the residual: $\boldsymbol{r} = \boldsymbol{x} - S_\Omega \boldsymbol{\alpha}_{\boldsymbol{\Omega}}$
    end while
```

4. Time warp invariant *k*SVD (TWI-*k*SVD)

Let $X = \{\boldsymbol{x}_i\}_{i=1}^N$ be a set of N input samples $\boldsymbol{x}_i \in \mathbb{R}^{q_i}$ and $D = \{\boldsymbol{d}_j\}_{j=1}^K$ the dictionary composed of K atoms $\boldsymbol{d}_j \in \mathbb{R}^{p_j}$. Both input samples and atoms define time series of different lengths that involve varying delays. The time warp invariant dictionary learning problem can be formalised as:

$$\begin{aligned} & \underset{A,D,\Delta}{min} \sum_{i=1}^{N} ||\boldsymbol{x}_i - \sum_{j=1}^{K} \Delta_{ij}\, \boldsymbol{d}_j\, \alpha_{ji}||_2^2 \qquad (9) \\ \text{s.t. :} \quad & ||\boldsymbol{\alpha}_i||_0 \leq \tau \text{ and } ||\boldsymbol{d}_j||_2 = 1 \\ & \Delta_{ij} \in \{0,1\}^{q_i \times p_j} \\ & \Delta_{ij}\, \boldsymbol{1}_{p_j} = \boldsymbol{1}_{q_i} \end{aligned}$$

with $A = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_N]$ is the sparse codes matrix, $\boldsymbol{\alpha}_i = (\alpha_{1i}, ..., \alpha_{Ki})^t$ and Δ_{ij} a binary matrix that encodes the alignment between $\boldsymbol{x}_i$ and $\boldsymbol{d}_j$. In the same spirit as in a standard *k*SVD, the problem in Eq. (9) can be solved by iterating two steps. In the first step, the problem in Eq. (9) is decomposed into N distinct time warp invariant sparse coding problems, that can be solved each by using TWI-OMP (Algorithm 3) to estimate A and Δ based on a fixed D. In the second step, based on the learned A and Δ, the objective is to update one atom and its related coefficients at a time. At each iteration, once an atom $\boldsymbol{d}_k$ is removed, it induces the removal of all its sibling atoms $\boldsymbol{d}_k^{s_i}$. Thus, the residuals $\boldsymbol{e}_i \in \mathbb{R}^{q_i}$ of $\boldsymbol{x}_i$ are estimated w.r.t the sibling atoms $\boldsymbol{d}_k^{s_i}$. Note that, whereas in standard *k*SVD all the residuals are determined based on the same referential vector $\boldsymbol{d}_k$, here each residual $\boldsymbol{e}_i$ is related to a distinct referential vector $\boldsymbol{d}_k^{s_i}$, making the build of the residual matrix E_k (Eq. 4) unfeasible and SVD inapplicable. To address this issue, we propose a solution that consists of two main steps:

1. use a projection and rotation transformations $\varphi(\boldsymbol{e_i})$ to represent all the residuals w.r.t a common referential vector, the residual matrix E_k can then be estimated and SVD applied to estimate the first left-singular vector $\boldsymbol{u}_1$ and update $\boldsymbol{d}_k$ accordingly,
2. use back transformations $\gamma_i(\boldsymbol{u}_1)$ of $\boldsymbol{u}_1$ to update each sibling atom $\boldsymbol{d}_k^{s_i}$ as well as its related coefficients.

Let $\boldsymbol{d}_k$ and $\{\boldsymbol{d}_k^{s_i}\}_{i=1}^N$ be the kth atom and its sibling atoms to be removed. The residual $\boldsymbol{e}_i$ (*i.e.,* the reconstruction error) of $\boldsymbol{x}_i$ once $\boldsymbol{d}_k^{s_i}$ removed is then:

$$\boldsymbol{e}_i = \boldsymbol{x}_i - \sum_{j \neq k} \boldsymbol{d}_j^{s_i} \alpha_{ji} \qquad (10)$$

Let us define φ ($\varphi(\boldsymbol{e}_i) \in \mathbb{R}^{p_k}$) as the transformation that represents the residuals $\boldsymbol{e}_i$ w.r.t. the common referential vector $\boldsymbol{d}_k$. First, it consists to align $\boldsymbol{e}_i$ and $\boldsymbol{d}_k^{s_i}$ to $\boldsymbol{d}_k$ thanks to the projector Δ_{ik}. Let $\Delta_{ik}^T \boldsymbol{e}_i$ and $\Delta_{ik}^T \boldsymbol{d}_k^{s_i}$ be the obtained aligned vectors of the same lengths as $\boldsymbol{d}_k$. Subsequently, $\Delta_{ik}^T \boldsymbol{e}_i$ is rotated such that the angle between $\varphi(\boldsymbol{e}_i)$ and $\boldsymbol{d}_k$ is the same as the one between $\boldsymbol{e}_i$ and $\boldsymbol{d}_k^{s_i}$. The effect of the transformation φ is described in Figure 1 and formalised as:

$$\varphi(\boldsymbol{e}_i) \quad = Rotation(\Delta_{ik}^T \boldsymbol{e}_i, \Delta_{ik}^T \boldsymbol{d}_k^{s_i}, \boldsymbol{d}_k) \qquad (11)$$

where $\boldsymbol{a}_r = Rotation(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ operates a rotation of the vector $\boldsymbol{a}$ to $\boldsymbol{a}_r$ such that $\theta_{\boldsymbol{a},\boldsymbol{b}} = \theta_{\boldsymbol{a}_r,\boldsymbol{c}}$ [1] (Arfken and Weber, 1999), the detail is given in Algorithm 4.

[1] $\theta_{\boldsymbol{a},\boldsymbol{b}}$ denotes the angle between the vectors $\boldsymbol{a}$ and $\boldsymbol{b}$

Algorithm 4 *Rotation*$(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$

```
Input: $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$
Output: $\boldsymbol{a}_r$
Set $\theta = \theta_{b,c}$, $\boldsymbol{u} = \frac{\boldsymbol{b}}{||\boldsymbol{b}||}$, $\boldsymbol{v} = \frac{\boldsymbol{c} - (\boldsymbol{u}^t\boldsymbol{c})\boldsymbol{u}}{||\boldsymbol{c} - (\boldsymbol{u}^t\boldsymbol{c})\boldsymbol{u}||}$
Compute the rotation matrix $R$ as follows:
    $R = I - \boldsymbol{u}\boldsymbol{u}^t - \boldsymbol{v}\boldsymbol{v}^t + [\boldsymbol{u}; \boldsymbol{v}]R_\theta[\boldsymbol{u}; \boldsymbol{v}]^t$
where $R_\theta \in \mathbb{R}^{2\times 2} = [(cos(\theta), sin(\theta))^t; (-sin(\theta), cos(\theta))^t]$
Compute $\boldsymbol{a}_r = R\,\boldsymbol{a}$.
```

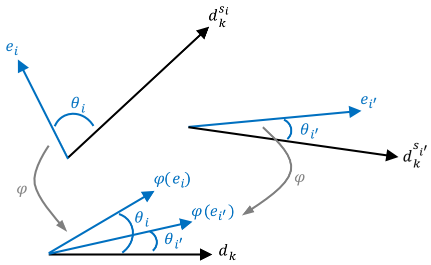

Fig. 1. φ**: representation of the residuals** $\boldsymbol{e}_i$**,** $\boldsymbol{e}_{i'}$ **w.r.t the common referential** $\boldsymbol{d}_k$

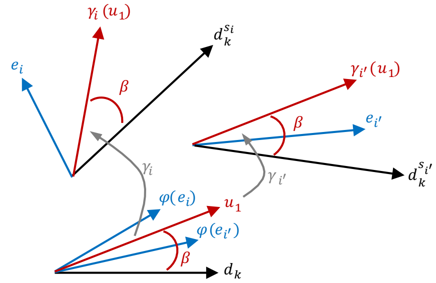

Fig. 2. γ_i**: representation of** u_1 **eigen vector w.r.t the sibling referential** $\boldsymbol{d}_k^{s_i}$

Let us denote $E_k[\varphi(\boldsymbol{e}_i)]_{i\in\omega_k} \in \mathbb{R}^{p_k\times|\omega_k|}$ the matrix of the residuals $\varphi(\boldsymbol{e}_i)$ with $\omega_k = \{i\,|\alpha_{ki} \neq 0, i = 1, ...N\}$ is the set of $\boldsymbol{x}_i$ indices that involve $\boldsymbol{d}_k^{s_i}$. An SVD is then applied on E_k to determine the first left-singular vector $\boldsymbol{u}_1$. The kth atom is updated as $\boldsymbol{d}_k = \boldsymbol{u}_1$. The sibling $\boldsymbol{d}_k^{s_i}$ for $i \in \omega_k$ as well as their coefficients are obtained based on the back transformation γ_i of $\boldsymbol{u}_1$ as:

$$\gamma_i(\boldsymbol{u}_1) = \Delta_{ik}\, Rotation(\boldsymbol{u}_1, \boldsymbol{d}_k, \Delta_{ik}^T\boldsymbol{d}_k^{s_i}) \tag{12}$$

$$\boldsymbol{d}_k^{s_i} = \gamma_i(\boldsymbol{u}_1) \tag{13}$$

$$\alpha_{ki} = \frac{<\boldsymbol{e}_i, \gamma_i(\boldsymbol{u}_1)>}{||\gamma_i(\boldsymbol{u}_1)||} \tag{14}$$

where γ_i allows to represent $\boldsymbol{u}_1$ w.r.t. the sibling referential $\boldsymbol{d}_k^{s_i}$. It consists first to rotate $\boldsymbol{u}_1$ such that to have the angle between $\gamma_i(\boldsymbol{u}_1)$ and $\boldsymbol{d}_k^{s_i}$ the same as between $\boldsymbol{u}_1$ and $\boldsymbol{d}_k$, then aligns the rotated $\boldsymbol{u}_1$ to $\boldsymbol{d}_k^{s_i}$ based on the projector Δ_{ik}. Figure 2 illustrates the effect of γ_i. The algorithm given in Algorithm 5 summarises the main steps described above of the time warp invariant kSVD (TWI-kSVD).

5. Experiments

The proposed approach TWI-kSVD is compared to five major dictionary learning methods on challenging datasets, composed of time series of different lengths that involve variables delays. The considered alternative dictionary learning methods deploy different strategies to address varying lengths and delays issues in time series data. First, we consider the Shift Invariant Sparse Coding (SISC) (Grosse et al., 2012), a convolved dictionary learning method, that learns basis functions as well as the time offsets and shifts to sparse code input time series. Then we consider kSVD (Aharon et al., 2006), LASSO-DL$_1$ (Engan et al., 1999) and LASSO-DL$_2$ (Yang et al., 2010) where time series are zero padded to render them of the same length, while the delay aspect is simply ignored. For LASSO-DL$_1$ and LASSO-DL$_2$, the sparse coding stage uses l_1 instead of l_0-norm, whereas in the dictionary learning step, all the atoms are learned at once by using MOD in LASSO-DL$_1$ and updated one atom at a time by using Lagrangian solver in LASSO-DL$_2$. Finally, we use the kernel kSVD (κ-kSVD) (Chen et al., 2015), where both varying length and delays are circumvent by using a Gaussian DTW kernel. In the following, first we give the description of the datasets, then detail the validation protocol conducted, before giving and discussing the results obtained.

5.1. Data description

To evaluate the performances of the proposed time warp invariant kSVD (TWI-kSVD), we have considered four public handwritten character datasets that involve naturally multivariate time series of different lengths while including varying delays. The three first datasets DIGITS, LOWER, and UPPER give the description of 2-D air-handwritten motion gesture of digits, upper and lower case letters performed on a WII device by several writers (Chen et al., 2012). The fourth dataset CHAR-TRAJ gives the 2-dimensional handwritten

Algorithm 5 TWI-kSVD(X, D, τ)

```
Input: $X = \{\boldsymbol{x}_i\}_{i=1}^N$ ($\boldsymbol{x}_i \in \mathbb{R}^{q_i}$), $D = \{\boldsymbol{d}_j\}_{j=1}^K$ ($\boldsymbol{d}_j \in \mathbb{R}^{p_j}$)
Output: $A$, $\Delta$, $D$
repeat
    for $i = 1, ..., N$ do
        $(\alpha_i, \{\Delta_{ij}\}_{j=1}^K)$ =TWI-OMP($\boldsymbol{x}_i, D, \tau$)
    end for
    for $k = 1, ..., K$ do
        Set $\omega_k = \{i \,|\, \alpha_{ki} \neq 0,\ i = 1, ..., N\}$ (the set of samples involving $\boldsymbol{d}_k$)
        Estimate $\varphi(\boldsymbol{e}_i)$ for $i \in \omega_k$ by using Eqs. 10 and 11
        Apply an SVD on $E_k[\varphi(\boldsymbol{e}_i)]_{i \in \boldsymbol{w}_k}$ to estimate $\boldsymbol{u}_1$
        Update $\boldsymbol{d}_j = \boldsymbol{u}_1$ and $\alpha_{ki}$, $\boldsymbol{d}_k^{s_i}$ for $i \in \omega_k$ by using Eq. 13 and Eq. 14
    end for
until Convergence (stopping rule)
```

Table 1. Data description

Dataset	Nb. class	Train size	Test size	Length range
DIGITS	10	100	300	29~218
LOWER	26	260	430	27~163
UPPER	26	260	520	27~412
CHAR-TRAJ	20	200	200	109~205
BME	3	30	150	128
UMD	3	36	144	150

character trajectory performed on a Wacom tablet by a same user (A. Frank, 2010). Furthermore, we consider two challenging synthetic datasets BME and UMD (Soheily-Khah et al., 2016), where time series share local temporal features within the classes while being of distinctive global behaviour. The data characteristics are given in Table 1. Figure 3 shows some instances to illustrate the characteristics of the considered time series. For instance, time series may have variable lengths and delays within a same class (”e”), they may have different global behaviours as in ”UP” class while sharing only local events (”small bell”) that may arise at different time stamps, or even have time series of different classes ”0” and ”6” that may share similar global behaviours.

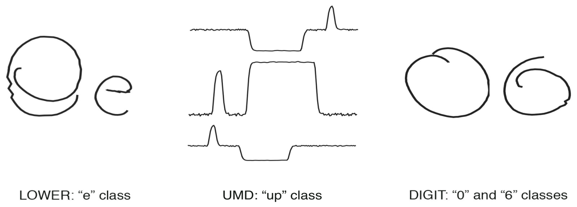

Fig. 3. Time series characteristics within and between classes

5.2. *Validation protocol*

To compare the accuracy of the considered dictionary learning methods we evaluate their potential to classify time series under time warp invariances. For that, the Sparse Coding for Classification (SRC) schedule is used (Wright et al., 2009). For a given dictionary learning method, SRC process consists first to learn one dictionary per class, then to form one global dictionary by concatenating the dictionaries learned for all the classes. The global dictionary is then used to sparse code test samples. Finally, based on both the estimated sparse coefficients and the dictionaries learned for the classes, the test samples are assigned to the class whose dictionary yields to the minimum reconstruction error. For each method, the related parameters (Table 2) are learned by a grid search on a validation set for handwritten DIGITS and CHAR-TRAJ datasets. For the small datasets BME and UMD, a 1-fold cross-validation is deployed. The best configuration of parameters is then used to estimate the accuracy on the test set. This process is reiterated 10-times and the obtained average error-rates are provided in Table 3. In addition, we have developed the algorithms of kSVD and LASSO-DL$_1$ and use the codes provided for LASSO-DL$_2$[2], SISC[3] and κ-kSVD[4]. For all the methods, a dictionary of size

[2] https://goo.gl/B5sKMc
[3] https://goo.gl/HirU4P
[4] https://goo.gl/j6nrzz

Table 2. Table of parameters

	Para.	Range Val.	Description
LASSO-DL$_1$	λ	$[0.1, 1]$ lag of 0.1	Regularisation
LASSO-DL$_2$	λ	$[0.1, 1]$ lag of 0.1	Regularisation
SISC	β	$[0.05, 5]$ lag of 0.05	Regularisation
	p	$\{50, 70, 90\}$	patch size
	q	$\{20, 40, 60\}$	atom size
TWI-*k*SVD	*sc*	$[0, 100]$ lag of 10	Sakoe-Chiba band

Table 3. Classification error rates

	τ	LASSO-DL$_1$	LASSO-DL$_2$	*k*SVD	SISC	κ-*k*SVD	TWI-*k*SVD
DIGIT	2	0.63	0.31	0.12	0.75	*0.02*	**0.01**
	5	0.55	0.22	0.15	0.77	*0.03*	**0.01**
	10	0.50	0.04	0.16	0.87	0.03	**0.00**
LOWER	2	0.64	0.43	0.29	0.79	**0.02**	0.07
	5	0.58	0.30	0.35	0.91	**0.02**	0.08
	10	0.56	0.26	0.32	0.93	**0.02**	0.07
UPPER	2	0.69	0.26	0.26	0.86	**0.06**	*0.09*
	5	0.61	0.27	0.31	0.90	**0.06**	*0.09*
	10	0.58	0.29	0.32	0.96	**0.06**	0.10
CHAR-TRAJ	2	0.34	0.05	**0.02**	0.78	0.09	*0.03*
	5	0.18	0.05	0.09	0.85	0.09	**0.03**
	10	0.14	0.05	0.07	0.93	0.09	**0.03**
UMD	2	0.77	0.42	0.57	0.45	0.04	**0.01**
	5	0.68	0.37	0.62	0.50	*0.02*	**0.01**
	10	0.52	0.41	0.65	0.52	*0.02*	**0.01**
BME	2	0.80	0.24	0.59	0.42	0.03	**0.00**
	5	0.68	0.24	0.40	0.46	0.03	**0.00**
	10	0.56	0.38	0.46	0.50	0.03	**0.00**
Nb. Best		0	0	1	0	6	**11**
Avg. Rank		5.25	3.08	3.83	5.47	*1.97*	**1.39**

$K = 10\times$ the number of classes is initialised randomly from the training set expect for SISC that is initialised in the provided code as Haar basis functions. Table 3 gives, for several sparsity levels, the error-rates obtained by using each dictionary learning method to classify the time series datasets. The best result is indicated in bold and the italic values reference the performances that are not significantly different from the best value (t-test at 5% risk). In addition, for each dataset, the performances obtained by the methods are first ranked, then the average ranking of each method is reported at the end of the Table (Avg. Rank). The best ranking (*i.e.*, the lowest one) is indicated in bold, and the italic values show the average ranking non significantly different from the best (Wilcoxon matched-pairs ranks test at 5% risk).

5.3. Results and discussion

From Table 3, we can see that the good performances are obtained by TWI-*k*SVD followed closely by κ-*k*SVD. In particular, the best results are reached by TWI-*k*SVD at lower sparsity coefficient ($\tau = 2$) with a total number of best values (Nb. Best) of 11 and an average ranking (Avg.Rank) of 1.39. Reasonable results are obtained for LASSO-DL$_2$ and *k*SVD, with good performances for CHAR-TRAJ dataset and slightly better results for LASSO-DL$_2$ than *k*SVD at larger sparsity ($\tau = 10$). The weak results are obtained for LASSO-DL$_1$ and SISC. First, we can note that the methods LASSO-DL$_1$, LASSO-DL$_2$ and *k*SVD that use zero-padding, to circumvent the problem of variable lengths and delays, lead to lower performances than TWI-*k*SVD and κ-*k*SVD that use appropriate approaches to deal with time series under time warp. For SISC, the problem of time warp is addressed by segmenting each input time series into small parts of the same length, then a dictionary is learned to sparse represent the small parts. Although the learned dictionary show a very good ability to reconstruct time series of each class, it fails under SRC classification. The main reason is due to the learned dictionary, that is composed of basis atoms that are not discriminative nor specific to each class and instead are commonly shared by the dictionaries of all the classes. Thus, SISC method seems appropriate to reconstruct time series under time warp not for their classification. For κ-*k*SVD, the variable lengths and delays are addressed by using a time warp kernel. It leads, similarly to TWI-*k*SVD, to good classification results. As a kernel-based machine, input samples are mapped into a higher dimensional space (Hilbert feature space), where the dictionary is learned and samples sparse coded as a non linear combination of the basis atoms. Sparse coding based on non linear combination functions allows a more precise representation than when relying on linear combination functions, as used by the other studied methods. However, as the sparse codes and the learned dictionary are processed into the feature space, there is no explicit description of the sparse representations nor of the learned atoms into the input space, which constitutes a major limitation for κ-*k*SVD. κ-*k*SVD seems to be a powerful and efficient method for time series classification under time warp, but not as a dictionary learning method with a main purpose to extract the basic primitives that discriminate the classes,

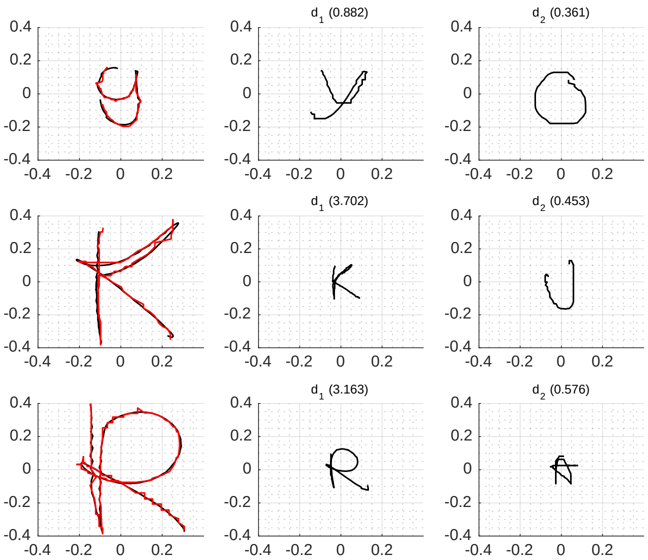

Fig. 4. TWI-*k*SVD**: The first column gives the input time series (in black) and their reconstruction (in red). The second and third columns show the two first atoms used for the reconstruction.**

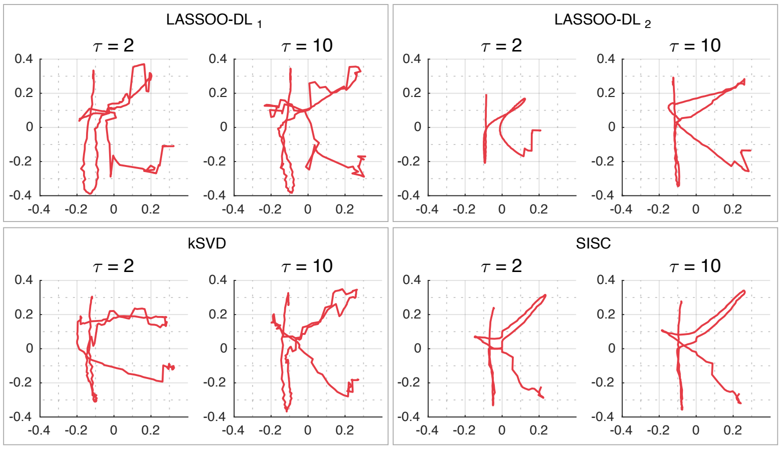

Fig. 5. The reconstruction of the letter "K" at (τ = 2) and (τ = 10).

to be used for any further learning tasks into the input space. Before concluding the discussion, let us underline that although TWI-*k*SVD relies on a linear combination of the basic atoms for sparse coding, it leads to the best performances on almost all datasets, while providing an explicit description of both the sparse codes and the extracted primitives into the input space. For instance, Figure 4 shows for several time series, their reconstruction (in red) based on the two first learned atoms. In particular, we can see that the first learned atom reveals the latent character that characterises the class, whereas the second atom contributes to reconstruct the residual to best fit the input sample. In addition, we can see that when TWI-*k*SVD requires only two atoms ($\tau = 2$) to reconstruct precisely the letter "K" (Figure 4), the alternative methods need 10 atoms ($\tau = 10$) to reach a reasonable reconstruction (Figure 5). Note that, there is no visualisation for κ-*k*SVD as there is no explicit descriptions into the input space. Lastly, notice that LASSO-DL$_1$, LASSO-DL$_2$ and *k*SVD are the fastest methods, as κ-*k*SVD and TWI-*k*SVD require additional treatment to face with time warping constraints. In particular, the complexity per class is of $O(TN\tau|D|)$ for *k*SVD and of $O(T^2N\tau|D|)$ for κ-*k*SVD and TWI-*k*SVD, where T is the maximum length observed and $|D|$ the dictionary size.

6. Conclusion

This paper proposes a time warp invariant *k*SVD where both input samples and atoms define time series of different lengths that involve varying delays. For that, we formalise and develop an efficient solution for time warp invariant orthogonal matching pursuit based on a new cosine maximisation time warp operator and the induced sibling atoms. Subsequently, rotation transformations between each atom and its sibling atoms allow to perform a singular value decomposition to jointly approximate the coefficients and update the dictionary. The proposed method is compared through several challenging data to major dictionary learning methods. The conducted experiments and the obtained results assess the pertinence and the efficiency of the proposed method for sparse coding and dictionary learning on time series under time warp.

References

A. Frank, A.A., 2010. UCI machine learning repository URL: http://archive.ics.uci.edu/ml/. [Online access].

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.

Arfken, G.B., Weber, H.J., 1999. Mathematical methods for physicists. AAPT.

Ataee, M., Zayyani, H., Babaie-Zadeh, M., Jutten, C., 2010. Parametric dictionary learning using steepest descent, in: Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on, IEEE. pp. 1978–1981.

Barchiesi, D., Plumbley, M.D., 2011. Dictionary learning of convolved signals, in: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, IEEE. pp. 5812–5815.

Barthélemy, Q., Larue, A., Mayoue, A., Mercier, D., Mars, J.I., 2012. Shift & 2d rotation invariant sparse coding for multivariate signals. Signal Processing, IEEE Transactions on 60, 1597–1611.

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.

Chen, Z., Zuo, W., Hu, Q., Lin, L., 2015. Kernel sparse representation for time series classification. Information Sciences 292, 15–26.

Do, M.N., Vetterli, M., 2005. The contourlet transform: an efficient directional multiresolution image representation. IEEE Transactions on Image Processing 14, 2091–2106.

Donoho, D.L., 2006. Compressed sensing. IEEE Transactions on information theory 52, 1289–1306.

Elad, M., Aharon, M., 2006. Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image processing 15, 3736–3745.

Engan, K., Aase, S.O., Husoy, J.H., 1999. Method of optimal directions for frame design, in: Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on, IEEE. pp. 2443–2446.

Grosse, R., Raina, R., Kwong, H., Ng, A.Y., 2012. Shift-invariance sparse coding for audio classification. arXiv preprint arXiv:1206.5241 .

Guha, T., Ward, R.K., 2012. Learning sparse representations for human action recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 34, 1576–1588.

Jost, P., Vandergheynst, P., Lesage, S., Gribonval, R., 2006. MoTIF: An efficient algorithm for learning translation invariant dictionaries, in: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on, IEEE. pp. V–V.

Lewicki, M.S., Sejnowski, T.J., 1999. Coding time-varying signals using sparse, shift-invariant representations. Advances in Neural Information Processing Systems , 730–736.

Mailhé, B., Lesage, S., Gribonval, R., Bimbot, F., Vandergheynst, P., 2008. Shift-invariant dictionary learning for sparse representations: extending k-svd, in: Signal Processing Conference, 2008 16th European, IEEE. pp. 1–5.

Mallat, S., 1999. A wavelet tour of signal processing. Academic press.

Mallat, S.G., Zhang, Z., 1993. Matching pursuits with time-frequency dictionaries. IEEE Transactions on Signal Processing 41, 3397–3415.

Ophir, B., Lustig, M., Elad, M., 2011. Multi-scale dictionary learning using wavelets. IEEE Journal of Selected Topics in Signal Processing 5, 1014–1024.

Pati, Y.C., Rezaiifar, R., Krishnaprasad, P.S., 1993. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, in: Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on, IEEE. pp. 40–44.

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.

Starck, J.L., Candès, E.J., Donoho, D.L., 2002. The curvelet transform for image denoising. IEEE Transactions on Image Processing 11, 670–684.

Tibshirani, R.J., et al., 2013. The LASSO problem and uniqueness. Electronic Journal of Statistics 7, 1456–1490.

Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y., 2009. Robust face recognition via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31, 210–227.

Yang, M., Zhang, L., 2010. Gabor feature based sparse representation for face recognition with gabor occlusion dictionary, in: European conference on computer vision, Springer. pp. 448–461.

Yang, M., Zhang, L., Yang, J., Zhang, D., 2010. Metaface learning for sparse representation based face recognition, in: Image Processing (ICIP), 2010 17th IEEE International Conference on, IEEE. pp. 1601–1604.

Yang, S., Min, W., Zhao, L., Wang, Z., 2013. Image noise reduction via geometric multiscale ridgelet support vector transform and dictionary learning. IEEE Transactions on Image Processing 22, 4161–4169.