Discrete Elastic Inner Vector Spaces with Application in Time Series and Sequence Mining

Pierre-François Marteau, Nicolas Bonnel, Gilbas Ménier

Discrete Elastic Inner Vector Spaces with Application to Time Series and Sequence Mining

Pierre-Francois Marteau, *Member, IEEE,* Nicolas Bonnel, and Gildas Ménier,

Abstract—This paper proposes a framework dedicated to the construction of what we call *discrete elastic inner product* allowing one to embed sets of non-uniformly sampled multivariate *time* series or sequences of varying lengths into inner product space structures. This framework is based on a recursive definition that covers the case of multiple embedded *time elastic* dimensions. We prove that such inner products exist in our general framework and show how a simple instance of this inner product class operates on some prospective applications, while generalizing the Euclidean inner product. Classification experimentations on time series and symbolic sequences datasets demonstrate the benefits that we can expect by embedding time series or sequences into elastic inner spaces rather than into classical Euclidean spaces. These experiments show good accuracy when compared to the euclidean distance or even dynamic programming algorithms while maintaining a linear algorithmic complexity at exploitation stage, although a quadratic indexing phase beforehand is required.

Index Terms—Vector space, Discrete time series, Sequence mining, Non-uniform sampling, Elastic inner product, Time warping.

1 Introduction

Time series analysis in metric spaces has attracted much attention over numerous decades and in various domains such as biology, statistics, sociology, networking, signal processing, etc, essentially due to the ubiquitous nature of time series, whether they are symbolic or numeric. Among other characterizing tools, time warp distances (see [20], [13], and more recently [7], [9] among other references) have shown some interesting robustness compared to the Euclidean metric especially when similarity searching in time series data bases is an issue. Unfortunately, this kind of elastic distance does not enable direct construction of definite kernels which are useful when addressing regression, classification or clustering of time series. A fortiori, they do not make it possible to directly construct inner products involving some *time elasticity*, which are basically able to cope with some stretching or some compression along specific dimension. Recently, [10] have shown that it is quite easy to propose inner product with time elasticity capability at least for some restricted time series spaces, basically spaces containing uniformly sampled time series, all of which have the same lengths (in such cases, time series can be embedded easily in Euclidean spaces).

- *P.-F. Marteau is with Université de Bretagne Sud, IRISA (UMR 6074), Campus de Tohannic, 56000 Vannes, France E-mail: pierre-francois.marteau AT univ-ubs.fr*
- *N. Bonnel is with Université de Bretagne Sud, IRISA (UMR 6074), Campus de Tohannic, 56000 Vannes, France E-mail: nicolas.bonnel AT univ-ubs.fr*
- *G. Ménier is with Université de Bretagne Sud, IRISA (UMR 6074), Campus de Tohannic, 56000 Vannes, France E-mail: gildas.menier AT univ-ubs.fr*

The aim of this paper is to derive an extension from this preliminary work for the construction of time elastic inner products, to achieve the construction of an elastic inner product structure for a *quasi-unrestricted* set of sequential data or time series, i.e. sets for which the data is not necessarily uniformly sampled and may have any lengths. Section two of the paper gives the main notations used throughout this paper and presents a recursive construction for inner-like products. It then gives the conditions and the proof of existence of elastic inner products (and elastic vector spaces) defined on a *quasi-unrestricted* set of time series while explaining what we mean by *quasi-unrestricted*. The third section succinctly presents some preliminary applications, mainly to highlight some of the features of elastic inner product vector spaces such as orthogonality. The fourth section presents two effective experimentations, the first one relating to time series classification and the second one addressing symbolic sequences classification.

2 Elastic Inner Product Vector Spaces

Starting from the recursive structure of the Dynamic Time Warping (DTW) equation in which the non linear Max operator is replaced by a Sum operator, we propose a quite general and parameterized recursive equation that we call *elastic product*. Our goal in this section is to derive the conditions for which such *elastic product* is an *inner product* that maintains a form of *time elasticity*.

2.1 Sequence and sequence element

Definition 2.1. Given a finite sequence A we denote by $A(i)$ the i^{th} element (symbol or sample) of sequence A. We will consider that $A(i) \in S \times T$ where $(S, \oplus_S, \otimes_S)$ is a vector space that embeds the multidimensional

space variables (e.g. $S \subset \mathbb{R}^d$, with $d \in \mathbb{N}^+$) and $T \subset \mathbb{R}$ embeds the *timestamps* variable, so that we can write $A(i) = (a(i), t_{a(i)})$ where $a(i) \in S$ and $t_{a(i)} \in T$, with the condition that $t_{a(i)} > t_{a(j)}$ whenever $i > j$ (timestamps strictly increase in the sequence of samples). A_i^j with $i \leq j$ is the subsequence consisting of the i_{th} through the j_{th} element (inclusive) of A. So $A_i^j = A(i)A(i+1)...A(j)$. Λ denotes the null element. By convention A_i^j with $i > j$ is the null time series, e.g. Ω.

2.2 Sequence set

Definition 2.2. The set of all finite discrete time series is thus embedded in a spacetime characterized by a single discrete *temporal* dimension, that encodes the timestamps, and any number of *spatial* dimensions that encode the value of the time series at a given timestamp. We note $\mathbb{U} = \{A_1^p | p \in \mathbb{N}\}$ the set of all finite discrete time series. A_1^p is a time series with discrete index varying between 1 and p. We note Ω the empty sequence (with null length) and by convention $A_1^0 = \Omega$ so that Ω is a member of set $\mathbb{U}$. $|A|$ denotes the length of the sequence A. Let $\mathbb{U}_p = \{A \in \mathbb{U} \mid |A| \leq p\}$ be the set of sequences whose length is shorter or equal to p. Finally let $\mathbb{U}^*$ be the set of discrete time series defined on $(S - \{0_S\}) \times T$, i.e. the set of time series that do not contain the null *spatial* value. We denote by 0_S the null value in S.

2.3 Scalar multiplication on $\mathbb{U}^*$

Definition 2.3. For all $A \in \mathbb{U}^*$ and all $\lambda \in \mathbb{R}$, $C = \lambda \otimes A \in \mathbb{U}^*$ is such that for all $i \in \mathbb{N}$ such that $0 \leq i \leq |A|$, $C(i) = (\lambda.a(i), t_{a(i)})$ and thus $|C| = |A|$.

2.4 Addition on $\mathbb{U}^*$

Definition 2.4. For all $(A, B) \in (\mathbb{U}^*)^2$, the addition of A and B, noted $C = A \oplus B \in \mathbb{U}^*$, is defined in a constructive manner as follows: let i, j and k be in $\mathbb{N}$.

$k = i = j = 1$,
As long as $1 \leq i \leq |A|$ and $1 \leq j \leq |B|$,
  if $t_{a(i)} < t_{b(j)}$, $C(k) = (a(i), t_{a(i)})$ and $i \leftarrow i + 1, k \leftarrow k + 1$
  else if $t_{a(i)} > t_{b(j)}$, $C(k) = (b(j), t_{b(j)})$ and $j \leftarrow j + 1, k \leftarrow k + 1$
  else if $a(i) + b(j) \neq 0$, $C(k) = (a(i) + b(j), t_{a(i)})$ and $i \leftarrow i + 1, j \leftarrow j + 1, k \leftarrow k + 1$
  else $i \leftarrow i + 1, j \leftarrow j + 1$

Three comments need to be made at this level to clarify the semantic of the operator $\oplus$:

i) Note that the $\oplus$ addition of two time series of equal lengths and uniformly sampled coincides with the classical addition in vector spaces. Fig. 1 gives an example of the addition of two time series that are not uniformly sampled and that have different lengths, except that zero values are discarded to ensure that the sum of two time series will remain a member of $\mathbb{U}^*$.
ii) Implicitly (in light of the last case described in Def. 2.4), any sequence element of the sort $(0_S, t)$, where 0_S is the null value in S and $t \in T$ must be assimilated to the null sequence element Λ. For instance, the addition of $A = (1,1)(1,2)$ with $B = (-1,1)(1,2)$ is $C = A \oplus B = (2,2)$: the addition of the two first sequence elements is $(0,1)$ that is assimilated to Λ and as such suppressed in C.
iii) The $\oplus$ operator, when restricted to the set $\mathbb{U}^*$ is reversible in that if $C = A \oplus B$ then $A = C \oplus ((-1) \otimes B)$ or $B = C \oplus ((-1) \otimes A)$. This is not the case if we consider the entire set $\mathbb{U}$.

2.5 Elastic product (ep)

Definition 2.5. A function $< .,. >: \mathbb{U}^* \times \mathbb{U}^* \rightarrow \mathbb{R}$ is called an Elastic Product if, there exists a function $f : S^2 \rightarrow \mathbb{R}$, a strictly positive function $g : T^2 \rightarrow \mathbb{R}^+$ and three constants α, β and ξ in $\mathbb{R}$ such that, for any pair of sequences A_1^p, B_1^q, the following recursive equation holds:

$$< A_1^p, B_1^q >_{ep} = \sum \begin{cases} \alpha \cdot < A_1^{p-1}, B_1^q >_{ep} \\ \beta \cdot < A_1^{p-1}, B_1^{q-1} >_{ep} + f(a(p), b(q)) \cdot g(t_{a(p)}, t_{b(q)}) \\ \alpha \cdot < A_1^p, B_1^{q-1} >_{ep} \end{cases} \quad (1)$$

This recursive definition requires defining an initialization. To that end we set, $\forall A \in \mathbb{U}^*$, $< A, \Omega >_{ep} = < \Omega, A >_{ep} = < \Omega, \Omega >_{ep} = \xi$, where ξ is a real constant (typically we set $\xi = 0$), and Ω is the null sequence, with the convention that $A_i^j = \Omega$ whenever $i > j$.

This paper addresses the most interesting question of the existence of elastic inner products on the set $\mathbb{U}^*$, i.e. without any restriction on the lengths of the considered time series nor the way they are sampled. If the choice of functions f and g, although constrained, is potentially large, we show hereinafter that the choice for constants α, β and ξ is unique.

2.6 Existence of elastic inner products(eip) defined on $\mathbb{U}^*$

Theorem 2.1. *$< .,. >_{ep}$ is an inner product on $(\mathbb{U}^*, \oplus, \otimes)$ iff:*

i) $\xi = 0$.
ii) $g : (T \times T) \rightarrow \mathbb{R}$ *is symmetric and strictly positive,*
iii) f *is an inner product on* $(S, \oplus_S, \otimes_S)$*, if we extend the domain of* f *on* S *while setting* $f(0_S, 0_S) = 0$.
iv) $\alpha = 1$ *and* $\beta = -1$,

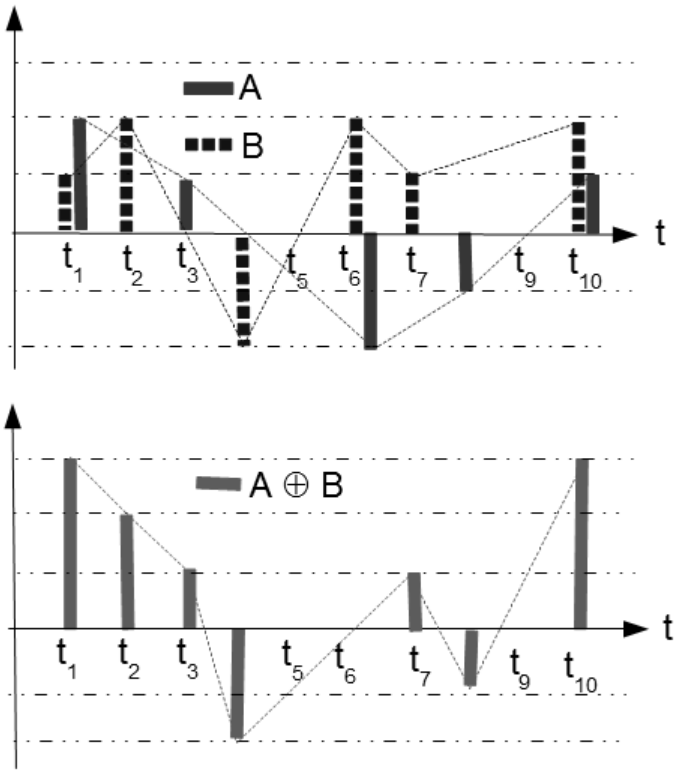

Fig. 1. The $\oplus$ binary operator when applied to two discrete time series of variable lengths and not uniformly sampled. Co-occurring events have been slightly separated at the top of the figure for readability purposes.

2.6.1 proof of theorem 2.1

Proof of the direct implication
Let us suppose first that $< .,. >_{ep}$ is an inner product defined on $\mathbb{U}^*$. Then since $< .,. >_{ep}$ is positive-definite necessarily $< \Omega, \Omega >_{ep} = \xi = 0$. Furthermore, for any $A_1 = (a, t_1)$ and $A_2 = (a, t_2) \in \mathbb{U}^*$ with $a \neq 0_S$, $< A_1, A_2 >_{ep} = g(t_1, t_2).f(a, a) \neq 0$. Since $< .;. >_{ep}$ is symmetric, we get that $g(t_1, t_2) = g(t_2, t_1)$ for any $(t_1, t_2) \in T^2$ which establishes that g is symmetric. Since g is strictly positive by definition of $< .,. >_{ep}$, i) and ii) are satisfied.

For any $A = (a, t_a) \in \mathbb{U}^*$, $< A, A >_{ep} = f(a, a)g(t_a, t_a) > 0$. Since g is strictly positive, then we get that $f(a, a) > 0$. If we set $f(0_S, 0_S) = 0$, we establish that f is positive-definite on S.

Since $\xi = 0$, for any A, B, and $C \in \mathbb{U}^*$ such that $A = (a, t)$, $B(b, t)$ and $C = (c, t_c)$, we have:
$< A \oplus B, C >_{ep} = f(a \oplus_S b, c).g(t, t_c)$.
As $< A \oplus B, C >_{ep} = < A, C >_{ep} + < B, C >_{ep}$
$= f(a, c).g(t, t_c) + f(b, c).g(t, t_c) = (f(a, c) + f(b, c)).g(t, t_c)$,
As g is strictly positive, we get that $f(a \oplus_S b, c) = (f(a, c) + f(b, c))$.
Furthermore, $< \lambda \otimes A, C >_{ep} = f(\lambda \otimes_S a, c).g(t, t_c)$.
As $< \lambda \otimes A, C >_{ep} = \lambda. < A, C >_{ep} = \lambda.f(a, c).g(t, t_c)$ and g is strictly positive, we get that $f(\lambda \otimes_S a, c) = \lambda.f(a, c)$.
This shows that f is linear, symmetric and positive-definite. Hence it is an inner product on $(S, \oplus_S, \otimes_S)$ and iii) is satisfied.

Let us show that necessarily $\alpha = 1$ and $\beta = -1$. To that end, let us consider any A_1^p, B_1^q and C_1^r in $\mathbb{U}^*$, such that $p > 1, q > 1, r > 1$ and such that $t_{a(p)} < t_{b(q)}$, i.e. if $X_1^s = A_1^p \oplus B_1^q$, then $X_1^{s-1} = A_1^p \oplus B_1^{q-1}$.
Since by hypothesis $< .,. >_{ep}$ is an inner product $(\mathbb{U}^*, \oplus, \otimes)$, it is linear and thus we can write:
$< A_1^p \oplus B_1^q, C_1^r >_{ep} = < A_1^p, C_1^r >_{ep} + < B_1^q, C_1^r >_{ep}$.

Decomposing $< A_1^p \oplus B_1^q, C_1^r >_{ep}$, we obtain:
$< A_1^p \oplus B_1^q, C_1^r >_{ep} = \alpha. < A_1^p \oplus B_1^{q-1}, C_1^r >_{ep} +$
$\beta. < A_1^p \oplus B_1^{q-1}, C_1^{r-1} >_{ep} + f(b(q), c(r)).g(t_{b(q)}, t_{c(r)}) +$
$\alpha. < A_1^p \oplus B_1^q, C_1^{r-1} >_{ep}$
As $< .,. >_{ep}$ is linear we get:
$< A_1^p \oplus B_1^q, C_1^r >_{ep} = \alpha. < A_1^p, C_1^r >_{ep} + \alpha. < B_1^{q-1}, C_1^r >_{ep}$
$+$
$\beta. < A_1^p, C_1^{r-1} >_{ep} + \beta. < B_1^{q-1}, C_1^{r-1} >_{ep}$
$+ f(b(q), c(r)).g(t_{b(q)}, t_{c(r)}) +$
$\alpha. < A_1^p, C_1^{r-1} >_{ep} + \alpha. < B_1^q, C_1^{r-1} >_{ep}$
Hence,
$< A_1^p \oplus B_1^q, C_1^r >_{ep} = \alpha. < A_1^p, C_1^r >_{ep} + \beta. < A_1^p, C_1^{r-1} >_{ep}$
$+$
$\alpha. < A_1^p, C_1^{r-1} >_{ep} + < B_1^q, C_1^r >_{ep}$

If we decompose $< A_1^p, C_1^r >_{ep}$, we get:

$< A_1^p \oplus B_1^q, C_1^r >_{ep} = (\alpha^2+\beta+\alpha) < A_1^p, C_1^{r-1} >_{ep} + \alpha.\beta. < A_1^{p-1}, C_1^{r-1} >_{ep} + \alpha.f(a(p),c(r)).g(t_{a(p)},t_{c(r)}) + \alpha^2. < A_1^{p-1}, C_1^r >_{ep} + < B_1^q, C_1^r >_{ep}$

Thus we have to identify $< A_1^p, C_1^r >_{ep} = \alpha. < A_1^p, C_1^{r-1} >_{ep} + \beta. < A_1^{p-1}, C_1^{r-1} >_{ep} + f(a(p),c(r)).g(t_{a(p)},t_{c(r)}) + \alpha. < A_1^{p-1}, C_1^r >_{ep}$
with $(\alpha^2+\beta+\alpha) < A_1^p, C_1^{r-1} >_{ep} + \alpha.\beta. < A_1^{p-1}, C_1^{r-1} >_{ep} + \alpha.f(a(p),c(r)).g(t_{a(p)},t_{c(r)}) + \alpha^2. < A_1^{p-1}, C_1^r >_{ep}$.

The unique solution is $\alpha = 1$ and $\beta = -1$. That is if $< .,. >_{ep}$ is an existing inner product, then necessarily $\alpha = 1$ and $\beta = -1$, establishing iv).

Proof of the converse implication
Let us suppose that i), ii), iii) and iv) are satisfied and show that $< .,. >_{ep}$ is an inner product on $\mathbb{U}^*$.

First, by construction, since f and g are symmetric, so is $< .,. >_{ep}$.

It is easy to show by induction that $< .,. >_{ep}$ is non-decreasing with the length of its arguments, namely, $\forall A_1^p$ and B_1^q in $\mathbb{U}^*$,
$< A_1^p, B_1^q >_{ep} - < A_1^p, B_1^{q-1} >_{ep} \geq 0$. Let $n = p + q$. The proposition is true at rank $n = 0$. It is also true if $A_1^p = \Omega$, whatever B_1^q is, or $B_1^q = \Omega$, whatever $< A_1^p$ is. Suppose it is true at a rank $n \geq 0$, and consider $A_1^p \neq \Omega$ and $B_1^q \neq \Omega$ such that $p + q = n$.
By decomposing $< A_1^p, B_1^q >_{ep}$ we get:
$< A_1^p, B_1^q >_{ep} - < A_1^p, B_1^{q-1} >_{ep} = - < A_1^{p-1}, B_1^{q-1} >_{ep} + f(a(p),b(q)).g(t_{a(p)},t_{b(q)}) + < A_1^{p-1}, B_1^q >_{ep}$
Since $f(a(p),b(q)).g(t_{a(p)},t_{b(q)}) > 0$ and the proposition is true by inductive hypothesis at rank n, we get that $< A_1^p, B_1^q >_{ep} - < A_1^p, B_1^{q-1} >_{ep}) > 0$. By induction the proposition is proved.

Let us show by induction on the length of the time series the positive definiteness of $< .,. >_{ep}$.
At rank 0 we have $< \Omega, \Omega >_{ep} = \xi = 0$. At rank 1, let us consider any time series of length 1, A_1^1. $< A_1^1, A_1^1 >_{ep} = f(a(1),a(1)).g(t_{a(1)},t_{a(1)}) > 0$ by hypothesis on f and g. Let us suppose that the proposition is true at rank $n > 1$ and let consider any time series of length $n+1$, A_1^{n+1}. Then, since $\alpha = 1$ and $\beta = -1$, we get
$< A_1^{n+1}, A_1^{n+1} >_{ep} = 2. < A_1^{n+1}, A_1^n >_{ep} - < A_1^n, A_1^n >_{ep} + f(a(n+1),a(n+1)).g(t_{a(n+1)},t_{a(n+1)})$.
Since $< A_1^{n+1}, A_1^n >_{ep} - < A_1^n, A_1^n >_{ep} \geq 0$, and $f(a(n+1),a(n+1)).g(t_{a(n+1)},t_{a(n+1)}) > 0$, $< A_1^{n+1}, A_1^{n+1} >_{ep} > 0$, showing that the proposition is true at rank $n+1$. By induction, the proposition is proved, which establishes the positive-definiteness of $< .,. >_{ep}$ since $< A_1^p, A_1^p >_{ep} = 0$ only if $A_1^p = \Omega$.

Let us consider any $\lambda \in \mathbb{R}$, and any A_1^p, B_1^q in $\mathbb{U}^*$ and show by induction on $n = p + q$ that $< \lambda \otimes A_1^p, B_1^q >_{ep} = \lambda. < A_1^p, B_1^q >_{ep}$:
The proposition is true at rank $n = 0$. Let us suppose that the proposition is true at rank $n \geq 0$, i.e. for all $r \leq n$, and consider any pair A_1^p, B_1^q of time series such that $p + q = n + 1$.
We have: $< \lambda \otimes A_1^p, B_1^q >_{ep} = \alpha. < \lambda \otimes A_1^p, B_1^{q-1} >_{ep} + \beta. < \lambda \otimes A_1^{p-1}, B_1^{q-1} >_{ep} + f(\lambda \otimes_S a(p), b(q)).g(t_{a(p)},t_{b(q)}) + \alpha. < \lambda \otimes A_1^{p-1}, B_1^q >_{ep}$
Since f is linear on $(S, \oplus_S, \otimes_S)$, and since the proposition is true by hypothesis at rank n, we get that $< \lambda \otimes A_1^p, B_1^q >_{ep} = \lambda.\alpha < A_1^p, B_1^{q-1} >_{ep} + \lambda.\beta. < A_1^{p-1}, B_1^{q-1} >_{ep} + \lambda.f(a(p),b(q)).g(t_{a(p)},t_{b(q)}) + \lambda.\alpha. < A_1^{p-1}, B_1^q >_{ep} = \lambda. < A_1^p, B_1^q >_{ep}$.
By induction, the proposition is true for any n, and we have proved this proposition.

Furthermore, for any A_1^p, B_1^q and C_1^r in $\mathbb{U}^*$, let us show by induction on $n = p + q + r$ that $< A_1^p \oplus B_1^q, C_1^r >_{ep} = < A_1^p, C_1^r >_{ep} + < B_1^q, C_1^r >_{ep}$. Let X_1^s be equal to $A_1^p \oplus B_1^q$. The proposition is obviously true at rank $n = 0$. Let us suppose that it is true up to rank $n \geq 0$, and consider any A_1^p, B_1^q and C_1^r such that $p + q + r = n + 1$.

Three cases need then to be considered:

1) if $X_1^{s-1} = A_1^{p-1} \oplus B_1^{q-1}$, then $t_{a(p)} = t_{b(q)} = t$ and
$< A_1^p \oplus B_1^q, C_1^r >_{ep} = \alpha. < A_1^p \oplus B_1^q, C_1^{r-1} >_{ep} + \beta. < A_1^{p-1} \oplus B_1^{q-1}, C_1^{r-1} >_{ep} + f((a(p)+b(q)),c(r)).g(t,t_{c(r)}) + \alpha. < A_1^{p-1} \oplus B_1^{q-1}, C_1^r >_{ep}$.
Since f is linear on $(S, \oplus_S, \otimes_S)$, and the proposition true at rank n, we get the result.
2) if $X_1^{s-1} = A_1^p \oplus B_1^{q-1}$, then $t_{a(p)} < t_{b(q)} = t$ and
$< A_1^p \oplus B_1^q, C_1^r >_{ep} = \alpha. < A_1^p \oplus B_1^q, C_1^{r-1} >_{ep} + \beta. < A_1^p \oplus B_1^{q-1}, C_1^{r-1} >_{ep} + f(b(q),c(r)).g(t,t_{c(r)}) + \alpha. < A_1^p \oplus B_1^{q-1}, C_1^r >_{ep}$. Having $\alpha = 1$ and $\beta = -1$ with the proposition supposed to be true at rank n we get the result.
3) if $X_1^{s-1} = A_1^{p-1} \oplus B_1^{q-1}$, we proceed similarly to case 2).

Thus the proposition is true at rank $n+1$, and by induction the proposition is true for all n. This establishes the linearity of $< .,. >_{ep}$.
This ends the proof of the converse implication and theorem 2.1 is therefore established □
.

The existence of functions f and g entering into the definition of $< .,. >_{ep}$ and satisfying the conditions allowing for the construction of an inner product on $(\mathbb{U}^*, \oplus, \otimes)$ is ensured by the following proposition:

Proposition 2.2. *The functions $f : S^2 \to \mathbb{R}$ defined as $f(a,b) = < a, b >_S$ where $< .,. >_S$ is an inner product on $(S, \oplus_S, \otimes_S)$ and $g : T^2 \to \mathbb{R}$ defined as $f(t_a,t_b) = e^{-\nu \cdot d(t_a,t_b)}$, where d is a distance defined on T^2 and $\nu \in \mathbb{R}^+$, satisfy the conditions required to construct an elastic inner product on $(\mathbb{U}^*, \oplus, \otimes)$.*

The proof of Prop.2.2 is obvious. This proposition establishes the existence of ep inner products, that we will denote eip (Time Elastic Inner Product). An eip as

thus the following structure:

$$< A_1^p, B_1^q >_{eip} = \sum \begin{cases} < A_1^{p-1}, B_1^q >_{eip} \\ - < A_1^{p-1}, B_1^{q-1} >_{eip} + \\ \quad g(t_{a(p)}, t_{b(q)}) \cdot < (a(p), b(q) >_{eip(S)} \\ < A_1^p, B_1^{q-1} >_{eip} \end{cases} \quad (2)$$

With the initialization: $\forall A \in \mathbb{U}^*$, $< A, \Omega >_{eip} = < \Omega, A >_{eip} = < \Omega, \Omega >_{eip} = \xi$, where ξ is a real constant (typically we set $\xi = 0$), and Ω is the null sequence, with the convention that ${A_i}^j = \Omega$ whenever $i > j$.

Note that $< ., . >_S$ can be chosen to be a eip as well, in the case where a second *time elastic* dimension is required. This leads naturally to recursive definitions for ep and eip.

Proposition 2.3. *For any $n \in \mathbb{N}$, and any discrete subset $T = \{t_1, t_2, \cdots, t_n\} \subset \mathbb{R}$, let $\mathbb{U}_{n,\mathbb{R},T}$ be the set of all time series defined on $\mathbb{R} \times T$ whose lengths are n (the time series in $\mathbb{U}_{n,\mathbb{R},T}$ are considered to be uniformly sampled). Then, the eip on $\mathbb{U}_{n,\mathbb{R}}$ constructed from the functions f and g defined in Prop. 2.2 tends towards the Euclidean inner product when $\nu \to \infty$ if S is an Euclidean space and $< a, b >_S$ is the Euclidean inner product defined on S.*

The proof of Prop.2.3 is straightforward and is omitted. This proposition shows that eip generalizes the classical Euclidean inner product.

2.7 Algorithmic complexity

The general complexity associated to the calculation of the eip of two time series of lengths p and q as specified by Eq.2 is $O(p \cdot q)$. Basically this is the same complexity as that required for the calculation of the complete dynamic programming solutions such as the Dynamic Time Warping (DTW) [20] [13] algorithm evaluated on these two time series. Nevertheless, as discussed in section 3.3, Prop. 3.2 allows for efficient implementations of retrieval process (in linear time complexity) once a straightforward indexing phase has been implemented. This result is demonstrated in practice the experimentation section (sec. 4).

3 Some preliminary applications

We present in the following sections some applications to highlight the properties of Elastic Inner Product Vector Spaces ($EIPVS$).

3.1 Distance in $EIPVS$

The following proposition provides $\mathbb{U}^*$ with a norm and a distance, both induced by a eip.

Proposition 3.1. *For all $A_1^p \in \mathbb{U}^*$, and any $< ., . >_{eip}$ defined on $(\mathbb{U}^*, \oplus, \otimes)$ $\sqrt{< A_1^p, A_1^p >_{eip}}$ is a norm on $\mathbb{U}^*$.*

For all pair $(A_1^p, B_1^q) \in (\mathbb{U}^*)^2$, and any eip defined on $(\mathbb{U}^*, \oplus, \otimes)$,
$\delta_{eip}(A_1^p, B_1^q) = \sqrt{< A_1^p \oplus (-1. \otimes B_1^q), A_1^p \oplus (-1. \otimes B_1^q) >_{eip}}$
$= \sqrt{< A_1^p, A_1^p >_{eip} + < B_1^q, B_1^q >_{eip} - 2 \cdot < A_1^p, B_1^q >_{eip}}$
defines a distance metric on $\mathbb{U}^*$.

The proof of Prop. 3.1 is straightforward and is omitted.

3.2 Orthogonalization in $EIPVS$

To exemplify the effect of elasticity in $EIPVS$, we give below the result of the Gram-Schmidt orthogonalization algorithm for two families of independent univariate time series. The first family is composed of uniformly sampled time series having increasing lengths. The second family (a sine-cosine basis) is composed of uniformly sampled time series, all of which have the same length.

The tests which are described in the next sections were performed on a set $\mathbb{U}^*$ of discrete time series whose elements are defined on $(\mathbb{R} - \{0\} \times [0; 1])^2$ using the following eip:

$$< A_1^p, B_1^q >_{eip} = \sum \begin{cases} < A_1^p, B_1^{q-1} >_{eip} \\ - < A_1^{p-1}, B_1^{q-1} >_{eip} + \\ \quad a(p)b(q) \cdot e^{-\nu . |t_{a(p)} - t_{b(q)}|^2} \\ < A_1^{p-1}, B_1^q >_{eip} \end{cases} \quad (3)$$

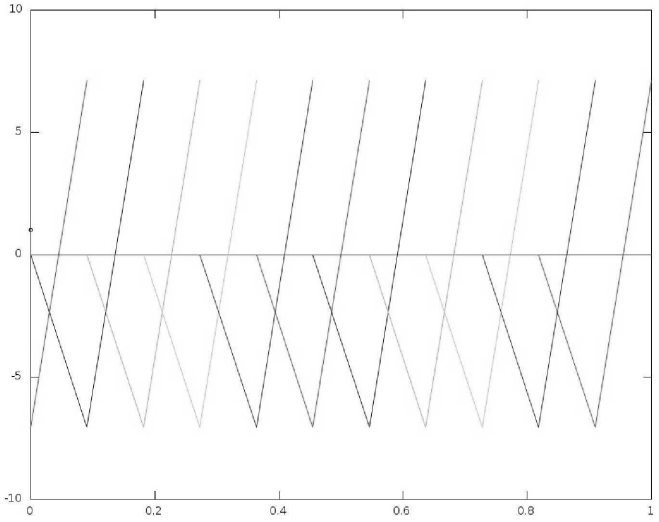

Fig. 2. Result of the orthogonalization of the family of length time series defined in Eq.4 using $\nu = .01$: except for the first *spike* located at *time* 0, each original *spike* is replaced by two *spikes*, one negative the other positive.

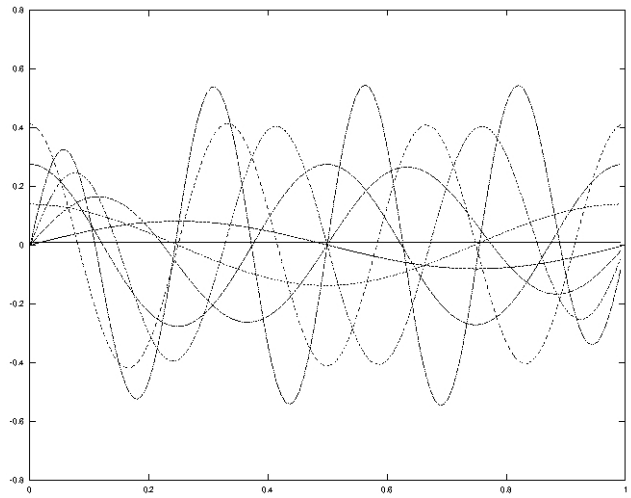

Fig. 3. Orthogonalization of the *sine-cosine* basis using $\nu = .01$: the waves are slightly deformed jointly in amplitude and in frequency. For readability of the figure, we have presented the 8 first components

3.2.1 Orthogonalization of an independent family of time series with increasing lengths

The family of time series we are considering is composed of 11 time series uniformly sampled, whose lengths are 11 samples:

$$\begin{array}{l}(1,0)\\(\epsilon,0)(1,1/10)\\(\epsilon,0)(\epsilon,0)(1,1/10)\\\cdots\\(\epsilon,0)(\epsilon,1/10)(\epsilon,2/10)\cdots(1,1)\end{array} \quad (4)$$

Since, the zero value cannot be used for the space dimension, we replaced it by ϵ, which is the smallest non zero positive real for our test machine (i.e. 2^{-1074}). The result of the Gram-Schmidt orthogonalization process using $\nu = .01$ on this basis is given in Fig.2.

3.2.2 Orthogonalization of a sine-cosine basis

An orthonormal family of discrete sine-cosine functions is not anymore orthogonal in a $EIPVS$. The result of the Gram-Schmidt orthogonalization process using $\nu = .01$ when applied on a discrete sine-cosine basis is given in Fig.3, in which only the 8 first components are displayed. The lengths of the waves are 128 samples.

3.3 Indexing for fast retrieval in time series data bases

Prop.2.3 shows how the elastic inner product generalizes the Euclidean inner product when time series are embedded into a finite dimensional vector space (one dimension per timestamps). We consider in this subsection such embeddings with the convention that if a time series has no value for a given timestamps a zero value is added on the dimension corresponding to the missing timestamps. Each time series is thus represented by a finite dimensional vector, let say in $\mathbb{R}^n$.

Proposition 3.2. *Given a symmetric and strictly positive function g, let consider the so-called $n \times n$ symmetric elastic matrix in $\mathbb{R}^{n^2}$ defined as:*

$$E = \begin{bmatrix} g(t_1,t_1) & g(t_1,t_2) & g(t_1,t_3) & \cdots & g(t_1,t_n) \\ g(t_2,t_1) & g(t_2,t_2) & g(t_2,t_3) & \cdots & g(t_2,t_n) \\ g(t_3,t_1) & g(t_3,t_2) & g(t_3,t_3) & \cdots & g(t_3,t_n) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ g(t_n,t_1) & g(t_n,t_2) & g(t_n,t_3) & \cdots & g(t_n,t_n) \end{bmatrix}$$

and two time series A_1^n and B_1^n represented by two vectors in $\mathbb{R}^n$. Then the elastic inner product defined recursively as:

$$\begin{array}{l} < A_1^n, B_1^n >_{eip} = \\ \sum \left\{ \begin{array}{l} < A_1^{n-1}, B_1^n >_{eip} \\ - < A_1^{n-1}, B_1^{n-1} >_{eip} + \\ \qquad a(n)b(n) \cdot g(t_{a(n)}, t_{b(n)}) \\ < A_1^n, B_1^{n-1} >_{eip} \end{array} \right. \end{array} \quad (5)$$

identifies to the following matrix products $[A_1^n]^T \, E \, [B_1^n]$

This result is quite interesting in the scope of time series information retrieval especially when addressing very large databases. Let $D = \{B(i)_1^n\}_{i=1\cdots m}$ be a time series database of size m, and consider the indexing phase that consists in constructing $D_E = \{E\,[B(i)_1^n]\}_{i=1\cdots m}$. Then the retrieval of all the time series in D *elastically* similar to a given request A_1^n will require the computation of $< A_1^n, B(i)_1^n >_{eip}$, for $i \in \{1, \cdots, m\}$, which reduces to evaluating:

$$< A_1^n, B(i)_1^n >_{eip} = [A_1^n]^T \, [EB(i)_1^n] \quad (6)$$

for $i \in \{1, \cdots, m\}$ which is done in linear complexity. Basically this construction breaks the quadratic complexity of the eip at retrieval stage and meets the same linear complexity as the classical Euclidean inner product.

Moreover one can notice that the eip defined by Eq.5 has it *continuous time* equivalent that expresses as: $< a, b >= \int\int a(t)b(\tau)g(t,\tau)\,\mathrm{d}t\mathrm{d}\tau$

3.4 Kernel methods in $EIPVS$

A wide range of literature exists on kernel theory, among which [3], [15] and [16] present some large syntheses of major results. We give hereinafter basic definitions and immediate results regarding kernel construction based on eip.

Definition 3.1. A kernel on a non empty set U refers to a complex (or real) valued symmetric function $\varphi(x, y) : U \times U \to \mathbb{C}$ (or $\mathbb{R}$).

Definition 3.2. Let U be a non empty set. A function $\varphi : U \times U \to \mathbb{C}$ is called a positive (resp. negative) definite kernel if and only if it is Hermitian (i.e. $\varphi(x, y) = \overline{\varphi(y, x)}$ where the *overline* stands for

the conjugate number) for all x and y in U and $\sum_{i,j=1}^{n} c_i\bar{c_j}\varphi(x_i,x_j) \geq 0$ (resp. $\sum_{i,j=1}^{n} c_i\bar{c_j}\varphi(x_i,x_j) \leq 0$), for all n in $\mathbb{N}$, $(x_1,x_2,...,x_n) \in U^n$ and $(c_1,c_2,...,c_n) \in \mathbb{C}^n$.

Definition 3.3. Let U be a non empty set. A function $\varphi : U \times U \rightarrow \mathbb{C}$ is called a conditionally positive (resp. conditionally negative) definite kernel if and only if it is Hermitian (i.e. $\varphi(x,y) = \overline{\varphi(y,x)}$ for all x and y in U) and $\sum_{i,j=1}^{n} c_i\bar{c_j}\varphi(x_i,x_j) \geq 0$ (resp. $\sum_{i,j=1}^{n} c_i\bar{c_j}\varphi(x_i,x_j) \leq 0$), for all $n \geq 2$ in $\mathbb{N}$, $(x_1,x_2,...,x_n) \in U^n$ and $(c_1,c_2,...,c_n) \in \mathbb{C}^n$ with $\sum_{i=1}^{n} c_i = 0$.

In the last two above definitions, it is easy to show that it is sufficient to consider mutually different elements in U, i.e. collections of distinct elements $x_1,x_2,...,x_n$.

Definition 3.4. A positive (resp. negative) definite kernel defined on a finite set U is also called a positive (resp. negative) semidefinite matrix. Similarly, a positive (resp. negative) conditionally definite kernel defined on a finite set is also called a positive (resp. negative) conditionally semidefinite matrix.

3.4.1 Definiteness of eip based kernel

Proposition 3.3. *A eip is a positive definite kernel.*

The proof of Prop. 3.3 is straightforward and is omitted. The consequence is that numerous definite (positive or negative) kernels can be derived from a eip; among other immediate derivations it can be stated that:

- $(< .,. >_{eip})^p$ is positive definite for all $p \in \mathbb{N}$ (polynomial kernel).
- δ_{eip} defined by Prop.3.1 is negative definite.
- $e^{-\nu\cdot\delta_{eip}}$ is positive definite for all $\nu > 0$.
- $e^{-\nu\cdot(\delta_{eip})^p}$ is positive definite for all $\nu > 0$ and any $0 < p \leq 2$.
- $e^{<A,B>_{eip}}$ is positive definite.

Some experimentations on Support vector Machine involving *elastic* kernels are reported in Sec.4.

3.5 Elastic Cosine similarity in $EIPVS$, with application to symbolic (e.g. textual) information retrieval

Similarly to the definition of the cosine of two vectors in Euclidean space, we define the elastic cosine of two sequences by using any ep that satisfies the conditions of theorem 2.1.

Definition 3.5. Given two sequences, A and B, the *elastic cosine* similarity of these two sequences is given using a time elastic inner product $< X,Y >_e$ and the induced norm $\|X\|_e = \sqrt{< X,X >_e}$ as
similarity $= eCOS(\theta) = \frac{<A\cdot B>_e}{\|A\|_e\|B\|_e}$

In the case of textual or sequential data information retrieval, namely text matching or sequence matching, the timestamps variable coincides with the index of words into the text or sequence, and the spatial dimensions encode the words or symbol into a given dictionary or alphabet. For instance, in text mining, each word can be represented using a vector whose dimension is the size of the set of concepts (or senses) that covers the conceptual domain associated to the dictionary, and each coordinate value, that is selected into $[0;1]$, encodes the degree of presence of the concept or senses into the considered word. In any case, the elastic cosine similarity measure takes value into $[0;1]$, 0 indicating the lowest possible similarity value between two sequential data and 1 the greatest possible similarity value between two sequential data. The elastic cosine similarity takes into account the order of occurrence of the words or symbols into the sequential data which could be an advantage compared to the Euclidean cosine measure that does not cope at all with the words or symbols ordering.

Let us consider the following elastic inner product dedicated to text matching. In the following definition, A_1^p and B_1^q are sequences of words that represent textual content.

Definition 3.6.

$$< A_1^p, B_1^q >_{eip_{tm}} = \sum \left\{ \begin{array}{l} < A_1^{p-1}, B_1^q >_{eip_{tm}} \\ - < A_1^{p-1}, B_1^{q-1} >_{eip_{tm}} + \\ \qquad\qquad e^{-\nu.|t_{a(p)}-t_{b(q)}|^2}\delta(a(p),b(q)) \\ < A_1^p, B_1^{q-1} >_{eip_{tm}} \end{array} \right. \quad (7)$$

where $a(p)$ and $b(q)$ are vectors whose coordinates identify words with weightings, $\delta(a,b) =< a,b >$ is the Euclidan inner product, and ν a *time stiffness* parameter.

Proposition 3.4. *For $\nu = 0$ and δ redefined as $\delta(a,b) = 1$ if $a = b$, 0 otherwise, the elastic inner product defined in Eq.7 coincides with the euclidean inner product between two vectors whose coordinates correspond to term frequencies observed into the A_1^p and B_1^q text sequences. If, we change the definition of δ by the $\delta(a,b) = (IDF(a))^2$ if $a = b$, 0 otherwise, where $IDF(a)$ is the inverse document frequency of term a into the considered collection, then for $\nu = 0$, $< A_1^p, B_1^q >_{eip_{tm}}$ coincides with the euclidean inner product between two vectors whose coordinates correspond to the TF-IDF (term frequency times the inverse document frequency) of terms occurring into the A_1^p and B_1^q text sequences.*

The proof of proposition 3.4 is straightforward an is omitted.

Thus, the elastic cosine measure derived from the elastic inner product defined by Eq.7 generalizes somehow the cosine measure implemented in the vector space model [14] and commonly used in the text information retrieval community.

To exemplify the behavior of the elastic cosine similarity on sequential data, we consider the four sequences depicted in Eq.8. The variations of the elastic cosine as

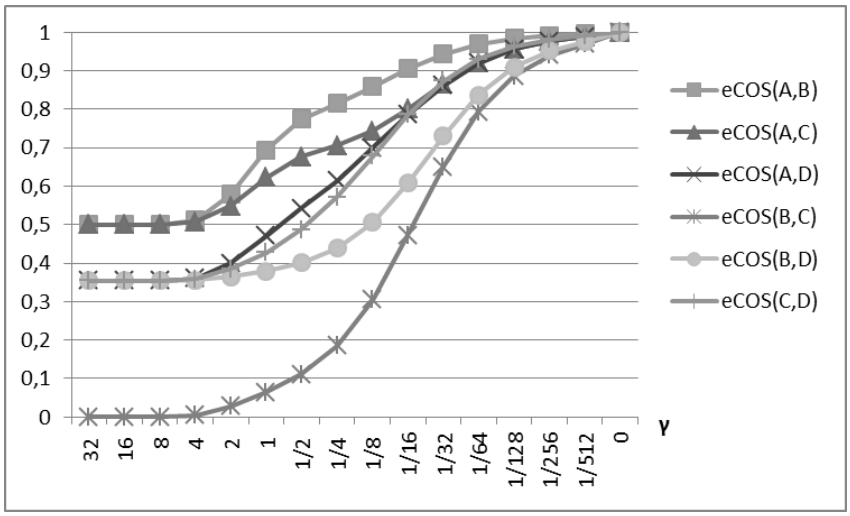

Fig. 4. Elastic cosine as a function of ν evaluated on pairwise sequences selected in $\{A = [abababab], B = [aaaabbbb], C[bbbbaaaa], D = [bbaa]\}$. The timestamps correspond to the index of occurrence of the symbol into the sequence ($i = 1, 2, \cdots,$).

a function of ν evaluated on pairwise sequences are reported in Fig.4

$$\begin{array}{l} A = [abababab] \\ B = [aaaabbbb] \\ C = [bbbbaaaa] \\ D = [bbaa] \end{array} \quad (8)$$

In Fig.4 the extreme left part of the curves, characterized with high ν values, corresponds, when applied on sequences having the same length and represented by vectors, to the cosine similarity constructed with the Euclidean inner product. The right part of the curves, characterized with very low ν values, corresponds to the cosine similarity evaluated using the *tf(-idf)* vector space model. The center part of the figure, characterized with medium ν values shows that *elasticity* allows to better discriminating between the sequences.

4 EXPERIMENTATIONS

4.1 Time series classification

We empirically evaluate the effectiveness of the distance induced by an elastic inner product comparatively to the Euclidean and the Dynamic Time Warping distances using some classification tasks on a set of time series coming from quite different application fields. The classification task we have considered consists of assigning one of the possible categories to an unknown time series for the 20 data sets available at the UCR repository [8]. As time is not explicitly given for these datasets, we used the index value of the samples as the timestamps for the whole experiment.

For each dataset, a training subset (TRAIN) is defined as well as an independent testing subset (TEST). We use the training sets to train two kinds of classifiers:

- the first one is a first near neighbor (1-NN) classifier: first we select a training data set containing time series for which the correct category is known. To assign a category to an unknown time series selected from a testing data set (different from the train set), we select its nearest neighbor (in the sense of a distance or similarity measure) within the training data set, then, assign the associated category to its nearest neighbor. For that experiment, a leave one out procedure is performed on the training dataset to optimize the meta parameter ν of the considered elastic distance.
- the second one is a SVM classifier [4], [19] configured with a Gaussian RBF kernel whose parameters are $C > 0$, a trade-off between regularization and constraint violation and σ that determines the width of the Gaussian function. To determine the C and σ hyper parameter values, we adopt a 5-folded cross-validation method on each training subset. According to this procedure, given a predefined training set TRAIN and a test set TEST, we adapt the meta parameters based on the training set TRAIN: we first divide TRAIN into 5 stratified subsets $TRAIN_1, TRAIN_2, \cdots, TRAIN_5$; then for each subset $TRAIN_i$ we use it as a new test set, and regard $(TRAIN - TRAIN_i)$ as a new training set; Based on the average error rate obtained on the five classification tasks, the optimal values of meta parameters are selected as the ones leading to the minimal average error rate. For the elastic kernel, the meta parameter ν is also optimized using this 5-folded cross-validation method performed on the training datasets.

The classification error rates are then estimated on the TEST datasets on the basis of the parameter values

optimized on the TRAIN datasets. We have used the LIBSVM library [6] to implement the SVM classifiers.

We tested the time elastic inner product $< A, B >_{eip}$ (Eq.3). Precisely, we used the time-warp distance induced by $< A, B >_{eip}$, basically $\delta_{eip}(A,B) = (< A-B, A-B >_{eip})^{1/2} = (< A, A >_{eip} + < B, B >_{eip} -2. < A, B >_{eip})^{1/2}$.

To speed up the computation at exploitation stage, we have used the construction proposed in Sec.3.3 Prop.3.2 with the following *elastic* Matrix E, where $\nu > 0$:

$$E = \begin{bmatrix} 1.0 & e^{-\nu|t_1-t_2|^2} & e^{-\nu|t_1-t_3|^2} & \dots & e^{-\nu|t_1-t_n|^2} \\ e^{-\nu|t_2-t_1|^2} & 1.0 & e^{-\nu|t_2-t_3|^2} & \dots & e^{-\nu|t_2-t_n|^2} \\ e^{-\nu|t_3-t_1|^2} & e^{-\nu|t_3-t_2|^2} & 1.0 & \dots & e^{-\nu|t_3-t_n|^2} \\ \dots & \dots & \dots & \dots & \dots \\ e^{-\nu|t_n-t_1|^2} & e^{-\nu|t_n-t_2|^2} & e^{-\nu|t_n-t_3|^2} & \dots & 1.0 \end{bmatrix}$$

The databases $D_E = \{EB(i)_1^n\}_{i=1\cdots m}$ are thus constructed *off-line* from the TRAIN datasets, once the optimization procedure of the ν parameter has been completed. D_E is then exploited *on-line* by the 1-NN and SVM classifiers.

4.1.1 Meta parameters

δ_{eip} is characterized by the meta parameter ν (the *stiffness* parameter) that is optimized for each dataset on the train data by minimizing the classification error rate of a first near neighbor classifier. For this kernel, ν is selected in $\{100, 10, 1, .1, .01, ..., 1e-5, 0\}$.

To explore the potential benefits of an elastic inner product against the Euclidean inner product, we also tested the Euclidean Distance δ_{ed} which, as already stated, is the limit when $\nu \to \infty$ of δ_{eip}.

The kernels exploited by the SVM classifiers are the Gaussian kernels $K_{eip}(A,B) = e^{-\delta_{eip}(A,B)^2/(2\cdot\sigma^2)}$ and $K_{ed}(A,B) = e^{-\delta_{ed}(A,B)^2/(2\cdot\sigma^2)}$. The meta parameter C is selected from the discrete set $\{2^{-8}, 2^{-7}, ..., 1, 2, ..., 2^{10}\}$, and σ^2 from $\{2^{-5}, 2^{-4}, ..., 1, 2, ..., 2^{10}\}$.

Table 1 gives for each data set and each tested kernels (K_{ed} and K_{eip}) the corresponding optimized values of the meta parameters.

According to the classification results, this experiment shows that the distance induced by the elastic inner product δ_{eip} is significantly more effective for the considered tasks comparatively to the Euclidean distance. It exhibits, on average, the lowest error rates for most of the tested datasets and for both the 1-NN and SVM classifiers, as shown in Table 2 and Figures 5 and 6. The stiffness parameter ν in δ_{eip} seems to play a significant role in these classification tasks, and this for a quite large majority of data sets.

Only one dataset, *yoga*, is better classified by the 1-NN δ_{ed} classifier on the test data, although the error rate on the train data is lower for the 1-NN δ_{eip} classifier. For the SVM classifiers, only two datasets, *Face (all)* and *Coffee*, are significantly better classified on the test data by SVM K_{ed} classifiers. Nevertheless, for these two datasets, K_{eip} reaches a better (or best) score on the train data. We are facing here the trade-off between learning and generalization capabilities. The meta parameter ν is selected such as to minimized the classification error on the train data. If this strategy is on average a winning strategy, some datasets show that it does not always lead to a good trade-off, this is the case for *Face (all)* and *yoga* datasets.

However, δ_{dtw} based classifiers outperform δ_{eip} based classifiers on a majority of datasets. The average ranking of the classifiers shows clearly that δ_{eip} ranks in between δ_{ed} and δ_{dtw}. δ_{eip} can be considered as a compromise between the linear Euclidean distance and the non-linear δ_{dtw}. It maintains a low computational cost and nevertheless compensates some of the limitations of the Euclidean distance that it is very sensitive to distortions in the time axis. It should be noted that δ_{eip} can cope with sample substitution, deletion and insertion as well as δ_{dtw}. In addition, δ_{eip} can deal with sample permutations while δ_{dtw} cannot.

4.2 Sequence classification

We report here a protein classification experiment carried out using the Protein Classification Benchmark Collection (PCBC) [18] [1]. This benchmark contains structural and functional annotations of proteins. The two datasets that we have exploited, SCOP95 and CATH95, are available at http://hydra.icgeb.trieste.it/benchmark.

The entries of the SCOP95 dataset are characterized by sequences with variable lengths and relatively little sequence similarity (less than 95% sequence identity) between the protein families.

The CATH95 dataset contains near-identical protein families of variable lengths in which the proteins have a high sequence similarity (more than 95% sequence identity).

Basically, the considered classification tasks involve protein domain sequence and structure comparisons at various levels of the structural hierarchies. We have considered the following 14 PCB subsets:

- PCB00001 $SCOP95\_Superfamily\_Family$
- PCB00002 $SCOP95\_Superfamily\_5fold$
- PCB00003 $SCOP95\_Fold\_Superfamily$
- PCB00004 $SCOP95\_Fold\_5fold$
- PCB00005 $SCOP95\_Class\_Fold$
- PCB00006 $SCOP95\_Class\_5fold$
- PCB00007 $CATH95\_Homology\_Similarity$
- PCB00008 $CATH95\_Homology\_5fold$
- PCB00009 $CATH95\_Topology\_Homology$
- PCB00010 $CATH95\_Topology\_5fold$
- PCB00011 $CATH95_Architecture\_Topology$

TABLE 1
Dataset sizes and meta parameters used in conjunction with K_{ed}, K_{eip} and K_{dtw} kernels

DATASET	length|#class|#train|#test	$K_{ed} : C, \sigma$	$K_{Deip} : \nu, C, \sigma$	$K_{DTW} : C, \sigma$
Synthetic control	60|6|300|300	1.0;0.125	.1;.25;.0625	8.0;4.0
Gun-Point	150|2|50|150	256;.5	0.01;256;.5	16.0;0.0312
CBF	128|3|30|900	8;1.0	.001;4.0;.0312	1.0;1.0
Face (all)	131|14|560|1690	4;0.5	.1;8.0;.5	2.0;0.25
OSU Leaf	427|6|200|242	2;0.125	.1;4;0.125	4.0;0.062
Swedish Leaf	128|15|500|625	128;0.125	10;8;.0625	4.0;0.031
50 Words	270|50|450|455	32;0.5	0.01;32;0.5	4.0;0.062
Trace	275|4|100|100	8;0.0156	.001;256;.0625	4;0.25
Two Patterns	128 |4|1000|4000	4.0;0.25	.01;1;.0312	0.25,0.125
Wafer	152|2|1000|6174	4.0;0.5	0.01;32;.5	1.0;0.016
face (four)	350|4|24|88	8.0;2.0	.01;16;2	16;0.5
Ligthing2	637|2|60|61	2.0;0.125	.001;128;2	2.0;0.031
Ligthing7	319|7|70|73	32.0;256.0	.1;16;.5	4;0.25
ECG200	96|2|100|100	8.0;0.25	0, 1024, .0625	2;0.62
Adiac	176|37|390|391	1024.0;0.125	10;256.0;.0312	16;0.0039
Yoga	426|2|300|3000	64.0;0.125	1;32;.0625	4;0.008
Fish	463|7|175|175	64.0;1.0	.01;256;1	8;0.016
Coffee	286|2|28|28	128.0;4.0	.01;1024;4	8;0.062
OliveOil	570|4|30|30	2.0;0.125	.01;64;2	2;0.125
Beef	470|5|30|30	128.0;4.0	0;64;.5	16;0.016

TABLE 2
Comparative study using the UCR datasets: classification error rates (in %) obtained using the first near neighbor (1-NN) classification rule and a SVM classifier for the K_{ed}, K_{eip} and K_{eip} kernels. Two scores are given S1|S2: the first one, S1, is evaluated on the training data, while the second one, S2, is evaluated on the test data. For the each classification methods (1-NN and SVM) the rank of each classifier is given in parenthesis ((1): best classifier, (2): 2nd best classifier, (3): 3rd best classifier

DATASET	1-NN δ_{ed}	**1-NN** δ_{eip}	**1-NN** δ_{dtw}	**SVM** K_{ed}	**SVM** K_{eip}	**SVM** K_{dtw}
Synthetic control	9(3)|12(3)	.67(1)|1(2)	1.0(2)|0.67(1)	3(3)|2.33(3)	.33(2)|.67(1)	0(1)|1(2)
Gun-Point	4.0(1)|8.67(1)	4.0(1)|8.67(1)	18.36(3)|9.3(3)	4.0(3)|6.0(3)	2.0(2)|2.67(2)	0(1)|1.33(1)
CBF	16.67(3)|14.78(3)	3.33(2)|4.22(2)	0(1)|0.33(1)	3.33(2)|10.89(3)	3.33(1)|5(1)	3.33(1)|5.44(2)
Face (all)	11.25(3)|28.64(3)	7.5(2)|26.33(2)	6.8(1)|19.23(1)	9.82(3)|16.45(3)	6.25(2)|24.91(2)	.54(1)|16.98(1)
OSU Leaf	37.0(2)|48.35(2)	37(2)|48.35(2)	33.17(1)|40.9(1)	35(3)|44.21(2)	34.5(2)|44.21(2)	20(1)|23.55(1)
Swedish Leaf	26.6(3)|21.12(3)	24.4(1)|20.96(2)	24.65(2)|20.8(1)	15(2)|8.64(2)	15(2)|8.64(2)	7(1)|5.6(1)
50 Words	34.47(3)|36.32(3)	32.2(1)|32.73(2)	33.18(2)|31(1)	33.56(3)|30.99(3)	31.78(2)|29.67(2)	15.21(1)|17.58(1)
Trace	18(3)|24(2)	16(2)|24(2)	0(1)|0(1)	9(3)|19(3)	1(2)|7(2)	0(1)|2(1)
Two Patterns	8.5(3)|9.32(3)	4.3(2) |3.62(2)	0(1)|0(1)	8.6(3)|7.45(3)	5.5(2)|3.52(2)	0(1)|0(1)
Wafer	0.7(2)|0.45(2)	.5(1)|.42(1)	1.4(3)|2.01(3)	.7(3)|.7(3)	.2(2)|.68(2)	0(1)|0.39(1)
face (four)	37.5(3)|21.59(3)	33.33(2)|19.31(2)	26.09(1)|17.05(1)	20.84(3)|19.31(3)	16.67(2)|13.63(2)	8.33(1)|5.68(1)
Ligthing2	25.0(3)|24.59(3)	20(2)|16.39(2)	13.56(1)|13.1(1)	21.77(3)|31.14(3)	20(2)|26.22(2)	8.33(1)|19.67(1)
Ligthing7	35.71(3)|42.47(3)	30.0(1)|32.87(2)	33.33(2)|27.4(1)	37.14(3)|36.98(3)	34.29(2)|35.61(2)	17.14(1)|16.43(1)
ECG200	14.0(2)|12.0(2)	1.0(1)|2.0(1)	23.23(3)|23(3)	8.0(3)|9.0(2)	3.0(1)|7.0(1)	7(2)|13(3)
Adiac	41.28(3)|38.87(1)	39.59(1)|38.87(1)	40.62(2)|39.64(3)	26.67(3)|24.04(1)	25.13(2)|24.04(1)	24.61(1)|25.32(3)
Yoga	22.67(3)|16.9(2)	21.67(2)|22.26(3)	16.37(1)|16.4(1)	17.66(3)|14.43(2)	15.33(2)|14.4(2)	11(1)|11.2(1)
Fish	24.0(1)|21.71(2)	24.0(1)|21.71(2)	26.44(3)|16.57(1)	14.86(3)|13.14(3)	13.14(2)|12.57(2)	6.86(1)|4.57(1)
Coffee	21.43(2)|25.0(2)	21.43(2)|25.0(2)	14.81(1)|17.86(1)	0(1)|0(1)	0(1)|7.14(2)	10.71(3)|17.86(3)
OliveOil	13.33(1)|13.33(1)	13.33(1)|13.33(1)	13.79(3)|13.33(1)	10.0(1)|10.0(1)	10.0(1)|10.0(1)	13.33(3)|16.67(3)
Beef	46.67(1)|46.67(1)	46.67(1)|46.67(1)	55.17(3)|50(3)	37.67(2)|30(1)	37.67(2)|30(1)	32.14(1)|42.85(3)
Average Rank	(2.4)|(2.25)	(1.45)|(1.75)	(1.85)|(1.5)	(2.65)|(2.4)	(1.8)|(1.7)	(1.25)|(1.6)

- PCB00012 $CATH95\_Architecture\_5fold$
- PCB00013 $CATH95\_Class\_Architecture$
- PCB00014 $CATH95\_Class\_5fold$

We evaluate the elastic cosine similarity based on the eip defined for symbolic sequences (Def.3.6, Eq.7) comparatively to five other similarity measures commonly used in Bioinformatics:

- BLAST [2]: the Basic Local Alignment Search Tool is a very popular family of fast heuristic search methods used for finding similar regions between two or more nucleotides or amino acids.
- SW [17]: The SmithWaterman algorithm is used for performing local sequence alignment, for determining similar regions between two nucleotide or protein sequences. Instead of looking at the sequence globally as NW does, the SmithWaterman algorithm compares subsequences of all possible lengths.
- NW [11]: The Needleman Wunsch algorithm performs a maximal global alignment of two strings. It is commonly used in bioinformatics to align protein sequences or nucleotides.
- LA kernel [12]: The Local Alignment kernel is used

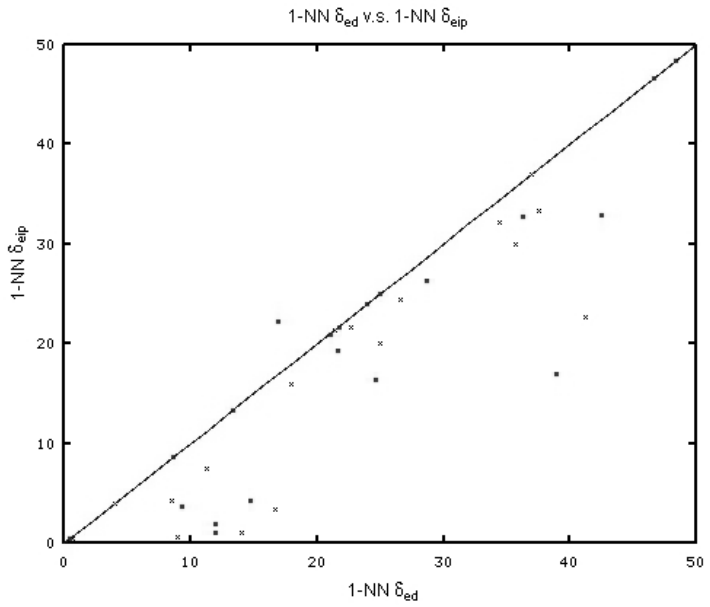

Fig. 5. Comparison of error rates (in %) between two 1-NN classifiers based on the Euclidean Distance (1-NN δ_{ed}), and the distance induced by a time-warp inner product (1-NN δ_{eip}). The straight line has a slope of 1.0 and dots correspond, for the pair of classifiers, to the error rates on the train (star) or test (circle) data sets. A dot below (resp. above) the straight line indicates that distance δ_{eip} has a lower (resp. higher) error rate than distance δ_{ed}. Black 'x' indicate error rates evaluated on the training data, while the red squares indicate error rates evaluated on the test data.

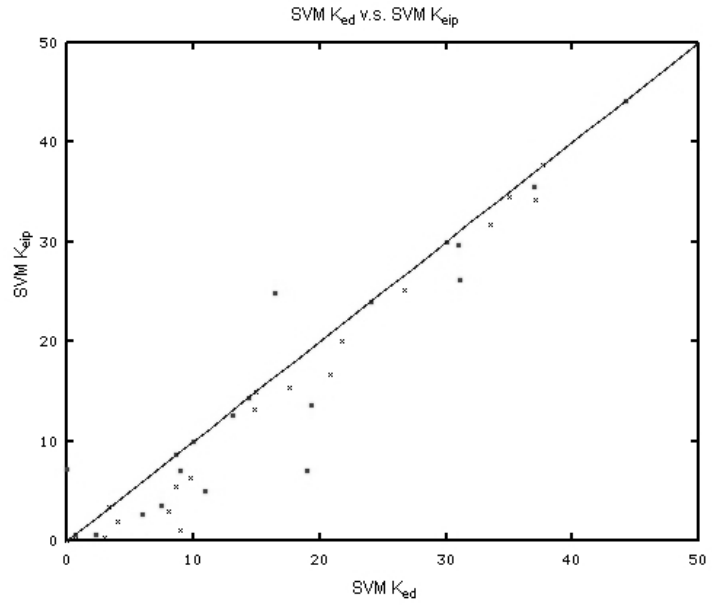

Fig. 6. Comparison of error rates (in %) between two SVM classifiers, the first one based on the Euclidean Distance Gaussian kernel (SVM K_{ed}), and the second one based on a Gaussian kernel induced by a time-warp inner product (SVM K_{eip}). The straight line has a slope of 1.0 and dots correspond, for the pair of classifiers, to the error rates on the train (star) or test (circle) data sets. A dot below (resp. above) the straight line indicates that SVM K_{eip} has a lower (resp. higher) error rate than distance SVM K_{ed}. Black 'x' indicate error rates evaluated on the training data, while the red squares indicate error rates evaluated on the test data.

to detect local alignment between strings by convolving simple basic kernels. Its construction mimic the local alignment scoring schemes proposed in the SW algorithm.

- PRIDE [5]: The PRIDE score is estimated as the PRobability of IDEntity between two protein 3D

TABLE 3
Comparative study using the PCBC protein datasets available at http://hydra.icgeb.trieste.it/benchmark. The assessment measure is the average AUC values based on the ROC curves obtained by the first near neighbor classification rules when BLAST, SW, NW, LA, PRIDE and eCOS(ν=1,.1,.05, .01, .001) similarity measures are used. Best average AUC values are in boldface characters.

1NN	BLAST	SW	NW	LA	PRIDE	$(\nu = 1)$	$(\nu = .1)$	$(\nu = .05)$	$(\nu = .01)$	$(\nu = .001)$	$(Best)$
SCOP95_supfam_fam_1	70.01	83.52	83.95	78.78	**89.48**	77.70	81.99	82.94	83.63	80.74	83.63
SCOP95_supfam_fam_kfold_2	94.87	**97.74**	97.73	96.71	97.41	91.02	94.84	95.02	94.15	91.44	95.02
SCOP95_fold_supfam_3	56.87	64.82	67.47	58.38	**84.72**	75.20	79.27	79.09	75.71	69.89	79.27
SCOP95_fold_supfam_kfold_4	90.30	93.77	94.54	91.38	**95.82**	88.93	93.40	93.32	90.67	85.79	93.4
SCOP95_class_fold_5	60.18	61.04	62.90	56.90	**78.27**	59.25	65.65	67.13	67.23	64.45	67.23
SCOP95_class_fold_kfold_6	92.23	92.14	92.85	88.28	**93.24**	80.43	88.38	88.76	85.32	79.81	88.76
CATH95_H_S_7	97.90	99.20	**99.29**	98.81	95.47	95.42	97.03	96.84	95.78	91.81	97.03
CATH95_H_S_kfold_8	96.63	98.30	**98.32**	97.80	95.02	95.10	97.28	97.22	96.36	93.67	97.28
CATH95_T_H_9	57.33	64.75	66.97	61.44	76.71	73.25	78.57	**79.12**	77.38	70.20	79.12
CATH95_T_H_kfold_10	89.76	92.60	**93.14**	90.66	90.96	88.98	93.02	92.78	90.28	84.25	93.02
CATH95_A_T_11	54.52	56.82	57.97	54.12	**67.37**	57.78	61.63	62.09	61.33	59.70	62.09
CATH95_A_T_kfold_12	90.21	91.28	**91.77**	88.91	84.89	82.44	88.28	88.14	84.96	80.09	88.28
CATH95_C_A_13	64.81	**71.28**	69.87	69.17	55.72	37.74	46.33	49.79	62.61	63.63	63.63
CATH95_C_A_kfold_14	90.80	89.74	**90.51**	86.21	80.63	76.75	83.29	83.94	80.57	75.71	83.94
Average	79.03	82.64	83.38	79.83	**84.69**	77.14	82.07	82.58	81.86	77.94	84.00

structures. The calculation of similarity between two proteins, is based on the comparison of histograms of the pairwise distances between $C - \alpha$ residues whose distribution is highly characteristic of protein folds.

The average AUC (area under the ROC Curve) measure is evaluated for 1-NN classifiers exploiting respectively BLAST, SW, NW, LA, PRIDE and $eCOS(\nu)$ as alignment methods. One can notice that these datasets are quite well suited for global alignment since, as shown in table 3, the NW algorithm performs better than the SW algorithm. The eip structure that considers global alignment is thus well adapted to the task. We show on these experiments that for $\nu = .05$ the $eCOS$ classifier in average performs significantly better than BLAST heuristics [2] and LA [12], the local alignment kernel for string. Furthermore, it performs almost as well as the SW and NW algorithms. The PRIDE method [5] gets the best results, but it uses the tertiary structure of the proteins while all the other methods exploit the primary structure. Here again, a ranking based on $eCOS$ similarity has a complexity that could be maintained linear at exploitation stage (i.e. when testing numerous sequences against massive datasets). These very positive results offer perspective in fast filtering of biological symbolic sequences.

4.3 Experimental complexity

To evaluate in practice the computational cost of δ_{eip}, we compare it with two other distances, namely δ_{ed} (the Euclidean distance) which has a linear complexity, and δ_{dtw}, the Dynamic Time Warping distance which has a quadratic complexity. In addition we evaluate the computational cost of the indexed version of δ_{eip} that we refer to as δ_{i-eip}. The experiment consists in producing random datasets of 100 time series of increasing lengths (10, 100, 1000 and 10000 samples) and computing the 100x100 distance matrices. Figure 7 gives the elapsed time in second, according to a logarithmic scale, for the four distances as a function of the length of the time series. When compared to δ_{dtw}, δ_{eip} is evaluated very efficiently using the matrix computation given in Eq.6, although, without any off line indexing of the time series, δ_{eip} cannot compete with δ_{ed} when the length of the time series increases. The δ_{i-eip} curve has clearly the same slope than the δ_{ed} curve, showing that the off-line indexed version of δ_{eip} is characterized with a linear complexity, that includes a nevertheless linear overhead when compared to δ_{ed}, mainly due to the loading of the index.

5 Conclusion

This paper has proposed what we call a family of *elastic inner products* able to cope with non-uniformly sampled time series of various lengths, as far as they do not contain the *zero* or $null$ symbol value. These constructions allow one to embed any such time series in a single inner space, that some how generalizes the notion of Euclidean inner space. The recursive structure of the proposed construction offers the possibility to manage several *elastic* dimensions. Some applicative benefits can be expected in time series or sequence analysis when time *elasticity* is an issue, for instance in the field of numeric or symbolic sequence data mining.

If the algorithmic complexity required to evaluate an elastic inner product is in general quadratic, its computation can be much more efficiently performed than dynamic programming algorithms. In addition we have shown that for some information retrieval applications for which embeddings of time series or symbolic sequences into a finite dimensional Euclidean space is possible, one can break this quadratic complexity down to a linear complexity at exploitation time, although a quadratic computational cost should still be paid once during a preprocessing step at indexing phase.

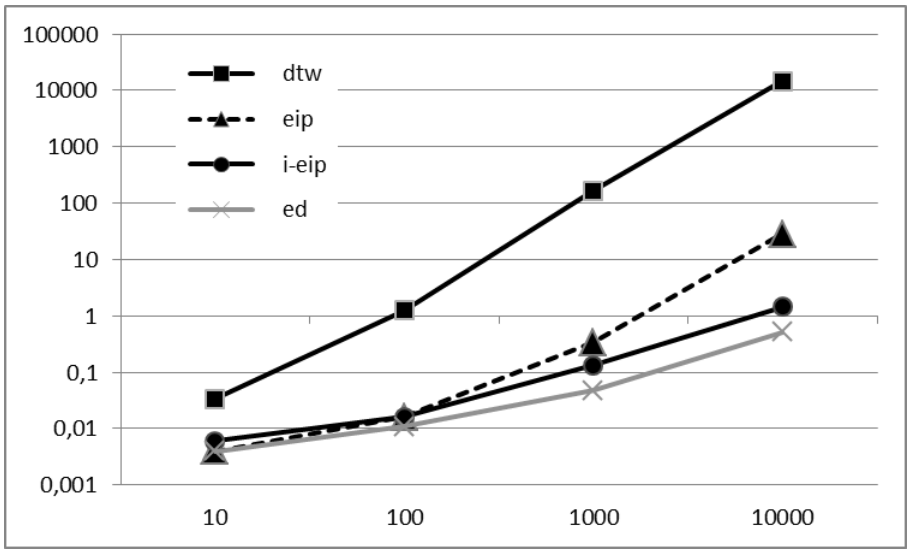

Fig. 7. Elapsed time in second as a function of the time series length for 10000 distance computations and for the four experimented distances: δ_{ed} (cross symbol, plain grey line), δ_{eip} (triangle symbol plain line), δ_{i-eip} (round symbol, dashed lines) and δ_{dtw}(square symbol, plain line).

The preliminary experiments we have carried out on some time series and symbolic sequence classification tasks show that embedding time series into elastic inner product space may brought significant accuracy improvement when compared to Euclidean inner product space embeddings as they compensate, at least partially, the limitations of Euclidean distance which is very sensitive to distortions in the time axis. Although our experiments show that Dynamic Programming matching algorithms outperforms on a majority of dataset distances that are derived from an elastic inner product, on some datasets such distances lead to similar accuracies .

The experiment carried out on symbolic sequences alignment involves sequences of various lengths. It shows also some very interesting perspectives in the scope of fast filtering of massive data, since the accuracy obtained by a 1-NN symbolic *elastic cosine* classifier with a potentially linear complexity at exploitation time is somehow comparable to the one obtained using dynamic programming algorithms (NW and SW) whose complexity are quadratic when the alignment search space is not restricted.

Finally, the general recursive structure of the elastic inner product opens perspectives in the processing of more complex data such as tree data mining for which considering several elastic dimensions may be relevant and efficient. The possibility to decompose complex structures onto sets of elastic basis vectors opens perspectives in various areas of application such as data compression, multi-dimensional scaling or matching pursuits.

REFERENCES

[1] The protein classification benchmark collection, 2006. http://hydra.icgeb.trieste.it/benchmark.

[2] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, October 1990.

[3] Christian Berg, Jens Peter Reus Christensen, and Paul Ressel. *Harmonic Analysis on Semigroups: Theory of Positive Definite and Related Functions*, volume 100 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, April 1984.

[4] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *COLT*, pages 144–152, 1992.

[5] O. Carugo and S. Pongor. Protein fold similarity estimated by a probabilistic approach based on c(alpha)-c(alpha) distance comparison. *Journal of Molecular Biology*, 315(4):887–898, 2002.

[6] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.

[7] L. Chen and R. Ng. On the marriage of lp-norm and edit distance. In *Proceedings of the 30th International Conference on Very Large Data Bases*, pages 792–801, 2004.

[8] E. J. Keogh, X. Xi, L. Wei, and C.A. Ratanamahatana. The ucr time series classification-clustering datasets, 2006. http://wwwcs.ucr.edu/ eamonn/time_series_data/.

[9] P. F. Marteau. Time warp edit distance with stiffness adjustment for time series matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):306–318, 2009.

[10] Pierre-François Marteau and Sylvie Gibet. Constructing positive elastic kernels with application to time series classification. *CoRR*, abs/1005.5141, 2010.

[11] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[12] H. Saigo, J.P. Vert, N. Ueda, and T. Akutsu. Protein homology detection using string alignment kernels. *Bioinformatics*, 20:1682–1689, 2004.

[13] H. Sakoe and S. Chiba. A dynamic programming approach to continuous speech recognition. In *Proceedings of the 7th International Congress of Acoustic*, pages 65–68, 1971.

[14] Gerard Salton and Michael McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, 1984.

[15] Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[16] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

[17] T. Smith and Waterman M. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[18] Paolo Sonego, Mircea Pacurar, Somdutta Dhir, Attila Kertész-Farkas, András Kocsor, Zoltán Gáspári, Jack A. M. Leunissen, and Sándor Pongor. A protein classification benchmark collection for machine learning. *Nucleic Acids Research*, 35(Database-Issue):232–236, 2007.
[19] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
[20] V. M. Velichko and N. G. Zagoruyko. Automatic recognition of 200 words. *International Journal of Man-Machine Studies*, 2:223–234, 1970.