Geodesic Analysis on the Gaussian RKHS Hypersphere

Nicolas Courty[1,3], Thomas Burger[2], and Pierre-François Marteau[1]

[1] IRISA, Université de Bretagne Sud, Vannes, France
[2] iRTSV (FR3425) / BGE (U1038), CNRS/CEA/UJF/INSERM, Grenoble, France
[3] Institute of Automation, Chinese Academy of Science, Beijing, China

Abstract. Using kernels to embed non linear data into high dimensional spaces where linear analysis is possible has become utterly classical. In the case of the Gaussian kernel however, data are distributed on a hypersphere in the corresponding Reproducing Kernel Hilbert Space (RKHS). Inspired by previous works in non-linear statistics, this article investigates the use of dedicated tools to take into account this particular geometry. Within this geometrical interpretation of the kernel theory, Riemannian distances are preferred over Euclidean distances. It is shown that this amounts to consider a new kernel and its corresponding RKHS. Experiments on real publicly available datasets show the possible benefits of the method on clustering tasks, notably through the definition of a new variant of kernel k-means on the hypersphere. Classification problems are also considered in a classwise setting. In both cases, the results show improvements over standard techniques.

1 Introduction

Most of the well known methods using the kernel trick [1,2] postulate that since the data are embedded in a Kernel Reproducing Hilbert Space (RKHS) with high dimensionality, non-linear data description is likely to become linear. As such, most of the classical linear methods can be applied with benefits. However, in the RKHS associated to numerous kernels (including the Gaussian kernel, on which this work is focused), all vectors have a unitary norm: the dataset lies on a hypersphere [3]. Hence, should this particular geometry be explicitly exploited by using non linear statistical tools in the RKHS? This work is a step in this direction. We notably show on two different applications (classification and clustering) that this idea can yield enhanced results over some real world datasets. The key idea is to consider a geodesic distance on the hypersphere rather than the Euclidean one to perform the data analysis. The geodesic distance corresponds to the total length of the shortest path over the hypersphere between two points, and it can be computed readily using trigonometric operators (Figure 1). Interestingly enough, this leads us to the definition of a new kernel: It appears that the geodesic distances in the original RKHS are equivalent to the Euclidean distances in a new RKHS. Thus, when data are embedded in this latter, it is indeed really justified to use linear methods. Our construction can

P. Flach et al. (Eds.): ECML PKDD 2012, Part I, LNCS 7523, pp. 299–313, 2012.

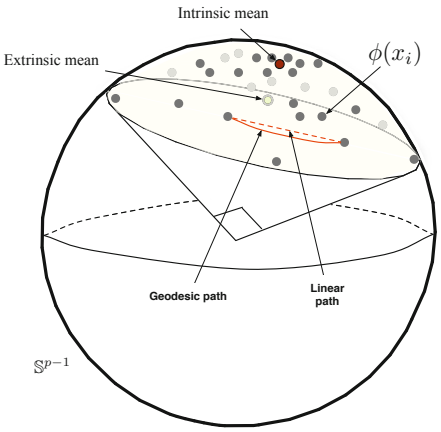

Fig. 1. Whatever the distribution of X, $\phi(X)$ lies within sphere quadrant. We propose to consider geodesic distance between elements of $\phi(X)$ rather than the Euclidean one. The Karcher (intrinsic) mean of $\phi(X)$ is represented as a red point, whereas the extrinsic mean is depicted in green. Note the latter is inside the hypersphere, whereas the Karcher mean lies on it.

be related to the work of Lafferty and Lebanon [4], who define a family of kernels based on diffusion operators over a Riemannian manifold. In our case, the geometric structure of the manifold is directly used to give a closed-form kernel expression instead of using a Fischer information metric.

The article is organized as follows: In Section 2, we set notations, and we provide background materials on geodesic distances and Riemannian manifolds. In Section 3 we adapt the classical tools of geodesic analysis to the Gaussian RKHS: To overcome the main drawback of kernelized space (the coordinates of the vectors are unknown), we find a transformation of the Gram matrix induced by the Gaussian kernel which takes into account geodesic distances. Next, in Section 4, we derive from the Gaussian kernel and from its modified Gram matrix a new data-dependent kernel. At this point, we remark that this derivation does not stand only for the Gaussian kernel, but for numerous other Radial Basis Function (RBF) kernels, leading to a whole family of data-dependent kernels. These latter are proved to be interesting on real datasets in Section 5: First, a clustering task is achieved by considering a k-means algorithm with geodesic distances on the Gaussian hypersphere. Second, we compare our new kernel to the Gaussian one in a classification task.

2 Geodesic Analysis on the Hypersphere

This section introduces the basis of a geodesic analysis on the hypersphere in the RKHS induced by the Gaussian Kernel. After stating the problem, basic facts about Riemannian geometry are presented and the notion of geodesic analysis is introduced.

2.1 Problem Statement

Let $X = \{x_1, \ldots, x_p\}_{(x_i \in \mathbb{R}^n)}$ be a set of p separated training samples described with n variables, and living in a space isomorphic to $\mathbb{R}^n$ and referred to as the *input space*. It is endowed with the Euclidean inner product denoted $\langle ., . \rangle_{\mathbb{R}^n}$ in the following. Let $k(.,.)$ be a symmetric form measuring the similarity among pairs of X, also called *kernel*. Let $\mathcal{H}$ be the associated RKHS, or *feature space*, also equipped with a dedicated inner product noted $\langle ., . \rangle_{\mathcal{H}}$, such that for any pair $(x_i, x_j) \in X^2$, we have:

$$\langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}} = k(x_i, x_j) \tag{1}$$

where $\phi(.)$ is an implicit mapping from $\mathbb{R}^n$ onto $\mathcal{H}$. We use the shorthand notation $\phi(X)$ for the set $\{\phi(x_1), \ldots, \phi(x_p)\}_{(\phi(x_i) \in \mathcal{H})}$. $\mathbf{K}$ is the Gram matrix of $\phi(X)$, and as such $\mathbf{K}_{ij} = k(x_i, x_j)$. We use the generic notation x for any vector of $\mathbb{R}^n$. Similarly, any vector of $\mathcal{H}$ is noted $\phi(x)$ (if its pre-image is assumed to be x) or simply y (if there is no assumption on its pre-image).

A kernel of particular interest in this work is the Gaussian kernel, defined as:

$$k(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \tag{2}$$

with the variance parameter $\sigma^2 \in \mathbb{R}_+^*$. Remark that: (1) the norm of any $\phi(x_i) \in \mathcal{H}$ is the unity, *i.e.* $\langle \phi(x_i), \phi(x_i) \rangle_{\mathcal{H}} = 1$, (2) the Gaussian RKHS is of infinite dimension. As a consequence, whatever X, $\phi(X)$ spans a subspace of dimension exactly p, and as such $\phi(X)$ lies on the unit hypersphere $\mathbb{S}^{p-1} \subset \mathcal{H}$. Moreover, as the inner product of two unit vectors corresponds to the cosine of their angle, and as $\forall (x_i, x_j),\ k(x_i, x_j) \in [0, 1]$, whatever X, $\phi(X)$ lies in a restriction $\mathcal{R}$ of $\mathbb{S}^{p-1}$ which is embedded in a sphere quadrant (its maximum angle is smaller than or equal to $\pi/2$, such as illustrated on Figure 1). Naturally, as $k(x_i, x_j)$ varies according to the value of the σ parameter, the surface of $\mathcal{R}$ varies accordingly: When σ increases, $k(x_i, x_j)$ increases, (*i.e.* the cosine between x_i and x_j increases), and thus the surface of $\mathcal{R}$ decreases. Conversely, when $\sigma \to 0$, $\mathcal{R}$ tends to a sphere quadrant.

2.2 Analysis on Riemannian Manifolds

A Riemannian manifold $\mathcal{M}$ in a vector space $\mathcal{V}$ with inner product $\langle ., . \rangle_{\mathcal{V}}$ is a real differentiable manifold such that the tangent space $\mathcal{T}_{x^*}$ associated to each vector x^* is endowed with an inner product $\langle ., . \rangle_{\mathcal{T}_{x^*}}$. In this work, $\langle ., . \rangle_{\mathcal{T}_{x^*}}$ reduces to $\langle ., . \rangle_{\mathcal{V}}$ on $\mathcal{T}_{x^*}$, so for simplicity we assimilate $\langle ., . \rangle_{\mathcal{T}_{x^*}}$ to $\langle ., . \rangle_{\mathcal{V}}$.

Classically data analysis is performed in $\mathcal{V} = \mathbb{R}^n$ and not in $\mathcal{M}$, as in the former it is rather natural to formalize the intuitive geometric notions (distance, mean, variance, direction, etc.) which are necessary to characterize the dataset. On the other hand, the statistical analysis of a dataset within $\mathcal{M}$ requires the non-trivial generalization of these notions to the setting of Riemmanian geometry. One of the first statistical analysis tool designed for Riemannian manifold

is the Principal Geodesic Analysis (or PGA), the goal of which is to find a set of directions, called *geodesic directions* or *principal geodesics*, that best encode the statistical variability of the data. PGA was first introduced by Fletcher et al. [5], and received since then numerous addenda [6,7], which are beyond the scope of this work. Here, we only focus on the tools of Riemannian geometry which are involved in the definition of PGA. The crucial observation of Fletcher is that a first order approximation of the distances among the samples of the dataset can be obtained if one projects the dataset in $\mathcal{T}_\mu$, the tangent space at μ, the Karcher mean of the dataset. We recall that the *Karcher mean* [8] $\mu \in \mathcal{M}$ differs from the traditional mean $\overline{x} \in \mathcal{V}$ (also called the *extrinsic mean*): It is the point of $\mathcal{M}$ which minimizes the sum of squared geodesic distances to every input data. As such, it constitutes an *intrinsic mean* (see Figure 1 for an illustration). We have:

$$\mu = \arg\min_{x \in \mathcal{M}} \sum_{i=1}^{p} d_{geod}(x_i, x)^2. \tag{3}$$

This approximation of the geodesic distances in $\mathcal{M}$ by the Euclidean distances in $\mathcal{T}_\mu$ seems particularly appealing, and it has been shown [9] that for a sphere the induced error is rather low. However, as this manifold lies in $\mathcal{V} = \mathcal{H}$ (instead of $\mathbb{R}^n$), the tractability of this approximation addresses several questions: First, how to define geodesic distances on the manifold embedding $\phi(X)$, and compute the associated Karcher mean μ of $\phi(X)$? Second, how to characterize $\mathcal{T}_\mu$ and project $\phi(X)$ onto $\mathcal{T}_\mu$? These two questions are addressed in two dedicated subsections of the next section.

3 Data Analysis over the Hypersphere in the Gaussian RKHS

Let us consider the unit hypersphere $\mathbb{S}^{p-1} \in \mathcal{H}$, the surface of which is the Riemannian manifold which embeds $\phi(X)$.

3.1 Geodesic Distance and Karcher Mean

The Riemannian distance (or the geodesic distance) between $\phi(x_i)$ and $\phi(x_j)$ on $\mathbb{S}^{p-1}$ corresponds to the length of the portion of the great circle embedding $\phi(x_i)$ and $\phi(x_j)$. It is simply given by:

$$d_{geod}(\phi(x_i), \phi(x_j)) = \arccos(\langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}}). \tag{4}$$

Then Equation (3) reads:

$$\mu = \arg\min_{y \in \mathcal{H}} \sum_{i=1}^{p} \arccos(\langle \phi(x_i), y \rangle_{\mathcal{H}})^2. \tag{5}$$

The Karcher mean of X exists and is uniquely defined as long as X belongs to a Riemannian ball of radius $\pi/4$ [8,10] which is the case since two points can be

at maximum distant from $\pi/2$. Usually, non-linear optimization methods can be used to compute this mean. However, finding the coordinates for μ is impossible, since we do not have access to the coordinates of $\phi(X)$. Instead, we turn on the search of the pre-image $\tilde{x} \in \mathbb{R}^n$ of $\mu \in \mathcal{H}$ (such that $\mu = \phi(\tilde{x})$). It is the solution of the following (non-linear) minimization problem:

$$\tilde{x} = \arg\min_{x \in \mathbb{R}^n} \sum_{i=1}^{p} \arccos(\langle \phi(x_i), \phi(x) \rangle_{\mathcal{H}})^2, \tag{6}$$

$$= \arg\min_{x \in \mathbb{R}^n} \sum_{i=1}^{p} \arccos(k(x_i, x))^2. \tag{7}$$

To operate this minimization, let us consider

$$f : \mathbb{R}^n \to \mathbb{R}$$
$$x \mapsto \sum_{i=1}^{p} \arccos(k(x_i, x))^2$$

and compute its gradient:

$$\nabla f(x) = \sum_{i=1}^{p} \frac{\partial}{\partial x} \arccos(k(x_i, x))^2, \tag{8}$$

$$= \frac{2}{\sigma^2} \sum_{i=1}^{p} \frac{\arccos(k(x_i, x)) k(x_i, x)}{\sqrt{1 - k(x_i,\ x)^2}} (x_i - x).$$

Setting this derivative to zero leads to a fixed point algorithm similar to the seminal work on pre-image computation proposed by Mika et al. [11]. This algorithm amounts to refining in several iterations a solution $\tilde{x}^t$ such that:

$$\tilde{x}^{t+1} = \frac{\sum_i \alpha_t(i) x_i}{\sum_i \alpha_t(i)}, \tag{9}$$

with

$$\alpha_t(i) = \frac{\arccos(k(x_i, x)) k(x_i, \tilde{x}^t)}{\sqrt{1 - k(x_i, \tilde{x}^t)^2}}$$

However, as stated in [11], this approach is prone to find local minima and its output is strongly dependent on the choice of the initial guess. Therefore, we propose a simple greedy algorithm (Alg. 1), which simply consists in repeating p times the previous optimization by setting the initial guess as the different inputs x_i (this latter is then omitted in the sum of equation 9). The estimation of the Karcher's mean pre-image is achieved using Algorithm 1 with an $\mathcal{O}(k.n^2)$ complexity, where k is the number of iteration and n the number of samples. In practice k is small, namely less than 10 for the tested datasets when an RBF kernel is used. However, a possible drawback of this approach is that it only provides an approximation for the Karcher mean, since the true

Algorithm 1. Pre-image of the Karcher mean on the sphere in the RKHS

$\epsilon \leftarrow$ small value, $\tilde{x} \leftarrow \text{mean}(X)$
for $i = 1$ **to** p **do**
  $x_i^{t=0} \leftarrow x_i$
  repeat
    update $\tilde{x}_i^{t+1}$ using equation 9 with $\tilde{x}_i^t$
  until $||\tilde{x}_i^{t+1} - \tilde{x}_i^t||^2 < \epsilon$
  if $f(\tilde{x}_i^{t+1}) < f(\tilde{x})$ **then**
    $\tilde{x} \leftarrow \tilde{x}_i^{t+1}$
  end if
end for
Output $\tilde{x}$

one may not have an exact pre-image in the input space. Thus, it may be interesting to consider other approaches to find the pre-image of the Karcher mean, e.g. distance based [12] or local isomorphism [13]. Nevertheless, their direct application is impossible since the Karcher mean is only defined through a minimization procedure without a closed-form solution. Fig. 2 illustrates the result of Alg. 1 to compute the pre-image of the Karcher mean on two toy datasets (points randomly sampled over a square and a spiral in 2 dimensions).

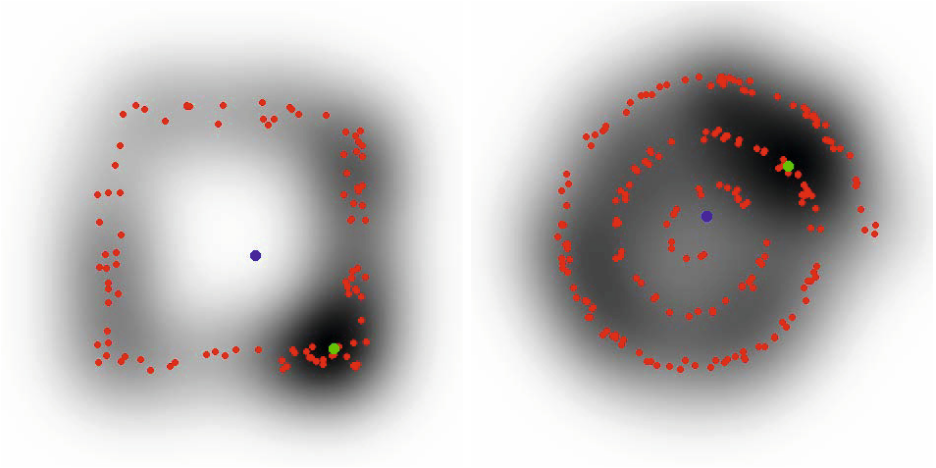

Fig. 2. Illustration of Karcher mean on two datasets: The dataset is represented by red points. The blue point is the data mean in input space, The green point is the pre-image of the Karcher mean after mapping onto the RKHS (the grayscale represents the function f values as described in Equation 8).

3.2 Projection on the Tangent Space

In the particular case of hypersperical manifolds, the mapping of any point onto a tangent space (this mapping is usually referred to as the *logarithmic map*), and the reverse mapping (the *exponential map*) are easy to define: The logarithmic map at location μ which projects any point $\phi(x_i) \in \mathcal{R} \subset \mathbb{S}^{p-1}$ onto $\mathcal{T}_\mu$ has the following form:

$$\begin{aligned} \mathrm{Log}_\mu : \mathcal{R} \setminus \mu &\to \mathcal{T}_\mu \\ y &\mapsto \frac{\theta}{\sin(\theta)}(y - \cos(\theta) \cdot \mu) \end{aligned} \tag{10}$$

where θ is the angle between μ and y *i.e.* $\theta = \arccos(\langle \mu, y \rangle_{\mathcal{H}})$. When $\theta = 0$, it is natural to consider that $y = \mu$. Conversely, the exponential map[1], which projects a vector y of $\mathcal{T}_\mu$ onto $\mathbb{S}^{p-1}$, is defined as:

$$\begin{aligned} \mathrm{Exp}_\mu : \mathcal{T}_\mu &\to \mathbb{S}^{p-1} \\ y &\mapsto \frac{\sin(\theta)}{\theta} \cdot y + \cos(\theta) \cdot \mu \end{aligned} \tag{11}$$

where θ is given by $\theta = \arccos\left(\frac{\langle y, \mu \rangle}{||y||}\right) = ||y||$.

When using the kernel notation, and for $\phi(x_i) \neq \mu$ Equation 10 reads:

$$\mathrm{Log}_{\phi(\tilde{x})}(\phi(x_i)) = \frac{\arccos(k(x_i, \tilde{x}))}{\sqrt{1 - k(x_i, \tilde{x})^2}}(\phi(x_i) - k(x_i, \tilde{x})\phi(\tilde{x})). \tag{12}$$

So far, the exact computation of this projection cannot be conducted, as ϕ remains unknown. However, it is possible to derive the Gram matrix of $\mathrm{Log}_{\phi(\tilde{x})}(\phi(X))$:

$$\begin{aligned} \mathbf{K}^{\tilde{x}}_{ij} &= \langle \mathrm{Log}_{\phi(\tilde{x})}(\phi(x_i)), \mathrm{Log}_{\phi(\tilde{x})}(\phi(x_j)) \rangle_{\mathcal{H}}, \\ &= \frac{\arccos(k(x_i, \tilde{x}))\arccos(k(x_j, \tilde{x}))}{\sqrt{1 - k(x_i, \tilde{x})^2}\sqrt{1 - k(x_j, \tilde{x})^2}} \cdot \\ &\quad (\phi(x_i) - k(x_i, \tilde{x})\phi(\tilde{x}))^T(\phi(x_j) - k(x_j, \tilde{x})\phi(\tilde{x})). \end{aligned} \tag{13}$$

Noting that:

$$\begin{aligned} &(\phi(x_i) - k(x_i, \tilde{x})\phi(\tilde{x}))^T(\phi(x_j) - k(x_j, \tilde{x})\phi(\tilde{x})) \\ &= \phi(x_i)^T\phi(x_j) - \phi(\tilde{x})^T\phi(x_j)k(x_i, \tilde{x}) - \phi(x_i)^T\phi(\tilde{x})k(x_j, \tilde{x}) + k(x_i, \tilde{x})k(x_j, \tilde{x}) \\ &= k(x_i, x_j) - 2k(x_i, \tilde{x})k(x_j, \tilde{x}) + k(x_i, \tilde{x})k(x_j, \tilde{x}) \\ &= k(x_i, x_j) - k(x_i, \tilde{x})k(x_j, \tilde{x}), \end{aligned} \tag{14}$$

we finally have a simple form for the entries of $\mathbf{K}^{\tilde{x}}$:

$$\mathbf{K}^{\tilde{x}}_{ij} = \frac{\arccos(k(x_i, \tilde{x}))\arccos(k(x_j, \tilde{x}))}{\sqrt{1 - k(x_i, \tilde{x})^2}\sqrt{1 - k(x_j, \tilde{x})^2}} \cdot (k(x_i, x_j) - k(x_i, \tilde{x})k(x_j, \tilde{x})). \tag{15}$$

Finally, it is possible to consider the geodesic distances in $\mathcal{H}$, by simply replacing the Gram matrix $\mathbf{K}$ associated to the kernel $k(.,.)$ by another Gram matrix $\mathbf{K}^{\tilde{x}}$.

[1] It is important to note that points on $\mathcal{R}$ are presented as vectors from the center of the hypersphere, while points on $\mathcal{T}_\mu$ are presented as vectors from μ.

4 A New Kernel Accounting for Geodesics in Hyperspherical RKHS

4.1 The Gaussian Case

However, it is possible to interpret $\mathbf{K}^{\tilde{x}}$ directly as the Gram matrix derived from a new kernel $k^{\tilde{x}}$ such that:

$$k^{\tilde{x}}(x_i, x_j) = \frac{\arccos(k(x_i, \tilde{x}))\arccos(k(x_j, \tilde{x}))}{\sqrt{1-k(x_i,\tilde{x})^2}\sqrt{1-k(x_j,\tilde{x})^2}} \cdot(k(x_i, x_j) - k(x_i, \tilde{x})k(x_j, \tilde{x})).$$

with the assumption that if $x_i = \tilde{x}$ (resp. x_j), then $k^{\tilde{x}}(x_i, x_j) = \arccos k(x_i, x_j)$. In such a perspective, we first need to establish the following result:

Proposition 1. *$k^{\tilde{x}}$ is a kernel.*

Proof:
First, let us prove that

$$\begin{aligned} k_1 : \mathbb{R}^n \times \mathbb{R}^n &\to \mathbb{R} \\ x_i, x_j &\mapsto k(x_i, x_j) - k(x_i, \tilde{x})k(x_j, \tilde{x}) \end{aligned}$$

is a kernel. To do so, let us simply consider

$$\begin{aligned} \Phi : \mathbb{R}^n &\to \mathcal{H} \\ x &\mapsto \phi(x) - k(x, \tilde{x})\phi(\tilde{x}) \end{aligned}$$

and remark that, obviously,

$$\begin{aligned} k_2 : \mathbb{R}^n \times \mathbb{R}^n &\to \mathbb{R} \\ x_i, x_j &\mapsto \Phi(x_i)^T \Phi(x_j) \end{aligned}$$

is a kernel, as k_2 corresponds to the Euclidean inner product in an another RKHS, onto which Φ maps. As it appears in Equation 14 that $k_1(x_i, x_j) = k_2(x_i, x_j)$, k_1 is also a kernel.

Second, remark that $k^{\tilde{x}}(x_i, x_j)$ can be re-written as the following conformal transformation $g(x_i)k_1(x_i, x_j)g(x_j)$ with:

$$\begin{aligned} g : \mathbb{R}^n &\to \mathbb{R} \\ x \neq \tilde{x} &\mapsto \frac{\arccos(k(x, \tilde{x}))}{\sqrt{1-k(x,\tilde{x})^2}} \\ \tilde{x} &\mapsto 1 \end{aligned}$$

which shows [3,14] that $k^{\tilde{x}}$ is a kernel and concludes the proof. □

Finally, we can consider a new vector space $\mathcal{H}^{\tilde{x}}$ with Euclidean inner product noted $\langle ., . \rangle_{\mathcal{H}^{\tilde{x}}}$ such that for any pair $x_i, x_j \in X^2$, we have:

$$\langle \phi^{\tilde{x}}(x_i), \phi^{\tilde{x}}(x_j) \rangle_{\mathcal{H}^{\tilde{x}}} = k^{\tilde{x}}(x_i, x_j)$$

with $\phi^{\tilde{x}}$ being the mapping from $\mathbb{R}^n$ onto $\mathcal{H}^{\tilde{x}}$. Then, the non-Euclidean distance in $\mathbb{R}^n$ derived from $k^{\tilde{x}}$ can be interpreted in two different ways: First, as the geodesic distances among $\phi(X)$ on the Gaussian RKHS, and according to a particular reference point $\tilde{x}$; Second, as the Euclidean distances among $\phi^{\tilde{x}}(X)$ on a new parametric RKHS $\mathcal{H}^{\tilde{x}}$ (with a data-dependent parameter $\tilde{x}$ corresponding to the pre-image of the Karcher mean of $\phi(X)$).

4.2 The General Case

Now, let us remark that these results stand not only for the Gaussian RKHS, on which our work is based, but also for any kernel which maps the dataset onto a hypersphere, and such as the angle between any pair of vectors is smaller than or equal to $\pi/2$.

This is notably the case for any "normalized" RBF kernel. Let us recall that a RBF kernel is of the form $k(x_i, x_j) = h(d(x_i, x_j))$ where d is a metric on $\mathbb{R}^n$ and where h is a function from $\mathbb{R}$ onto $\mathbb{R}^+$. By "normalized", we mean that

- $h(0) = 1$ (so that the vectors are of unit length in the RKHS, leading to a hyperspherical manifold)
- $h(x) \in [0, 1]\ \forall x \in \mathbb{R}$ (so that the dataset remains in a sphere quadrant)

5 Experiments

In this section, we assess the interest of geodesic analysis on hypersperical manifolds thanks to several experiments. First, we evaluate during a clustering task, the well-grounded of the use of geodesic distances and of the pre-image of Karcher mean. Then, during a second task involving supervised classification, we evaluate their interest through the kernel trick, as we compare our new kernel $k^{\tilde{x}}$, presented in Section 4.1, to the classical Gaussian kernel k. In both tests, we use a series of UCI datasets [15].

5.1 Hyperspherical Kernel Clustering

In this experiment, we propose to modify the kernel k-means procedure: First, the centroids of the clusters are computed as the pre-image of Karcher mean of each class, instead of the extrinsic mean. Second, the Euclidean distances are replaced by geodesic distances on the hypersphere of the Gaussian RKHS. Let us note that the distances are always computed between a centroid m_i and a sample x_j (*i.e.* between a pre-image in $\mathbb{R}^n$ and a vector in $\mathbb{R}^n$). Hence, even if this algorithm (see Algorithm 2) corresponds to a k-means in $\mathcal{H}^{\tilde{x}}$, there is no need to use the kernel trick with our new kernel: The geodesic distance simply reads $d_{geod}(x_j, m_i) = \arccos(k(x_j, m_i))$.

We compare this algorithm, that we call **hyperspherical clustering** to the classical k-means algorithm, its kernelized version [16] and to the spectral clustering algorithm described in [17]. From a qualitative point of view, let us remark

Algorithm 2. Hyperspherical clustering

Input: dataset X, number of clusters k
Output: k clusters
for $i = 1$ **to** k **do**
  Randomly initialize the m_i centroid of cluster i
end for
repeat
  for $j = 1$ **to** p **do**
    for $i = 1$ **to** k **do**
      Compute $d_{geod}(x_j, m_i) = \arccos(k(x_j, m_i))$
    end for
    $c_j = \arg\min_i d_{geod}(x_j, m_i)$
  end for
  Assign to m_i the Karcher mean of the set
    $\{x_\ell \in X / \ c_\ell = i\}$
until no more changes in the partitioning

that while the k-means does not requires the tuning of any parameter, the three other algorithms require the setting of σ to the proper value. Moreover, the traditional k-means and our version in $\mathcal{H}^{\tilde{x}}$ exhibit comparable complexities. They both are very light from a computational point of view with respect to the spectral clustering algorithm, as this latter requires the computation of the spectrum of a $p \times p$ matrix, which is $\mathcal{O}(p^3)$-complex.

The results are presented in Figure 3 for various values of σ, and in Table 1 where the best accuracy rates over the σ's are given. For all evaluations, the experiments are repeated 20 times to limit the effect of the random initialization, and only the mean accuracy and variance over the repetitions is displayed. Apart from Ionosphere and Glass, on which we are slightly less efficient than respectively the spectral clustering and the kernel k-means algorithms, our algorithm appears to be the most accurate in peak performance. As suggested by Figure 3, the performances of Hyperspectral clustering is also rather stable over a range of sigma values. Of course these preliminary results call for a broader comparison with more data and other clustering approaches.

5.2 Classification

The main restriction of the new kernel $k^{\tilde{x}}$ is that it relies on a data-dependent parameter, $\tilde{x}$. As the pre-image of the Karcher mean, $\tilde{x}$ can be understood as a representative of the dataset X. Thus, if X is separated into several classes, there is little chances that $\tilde{x}$ fits as a good representative of all the classes (it may fall between several classes). Hence, we think $k^{\tilde{x}}$ is more adapted to generative (class-wise) algorithms, in which a dedicated Karcher mean is computed for each class.

As a consequence, we do not use $k^{\tilde{x}}$ with the state-of-the-art SVM [1], as it is a discriminative algorithm for which the computation of the Karcher mean is likely to be unadapted. Instead, we consider the PerTurbo algorithm [18], which

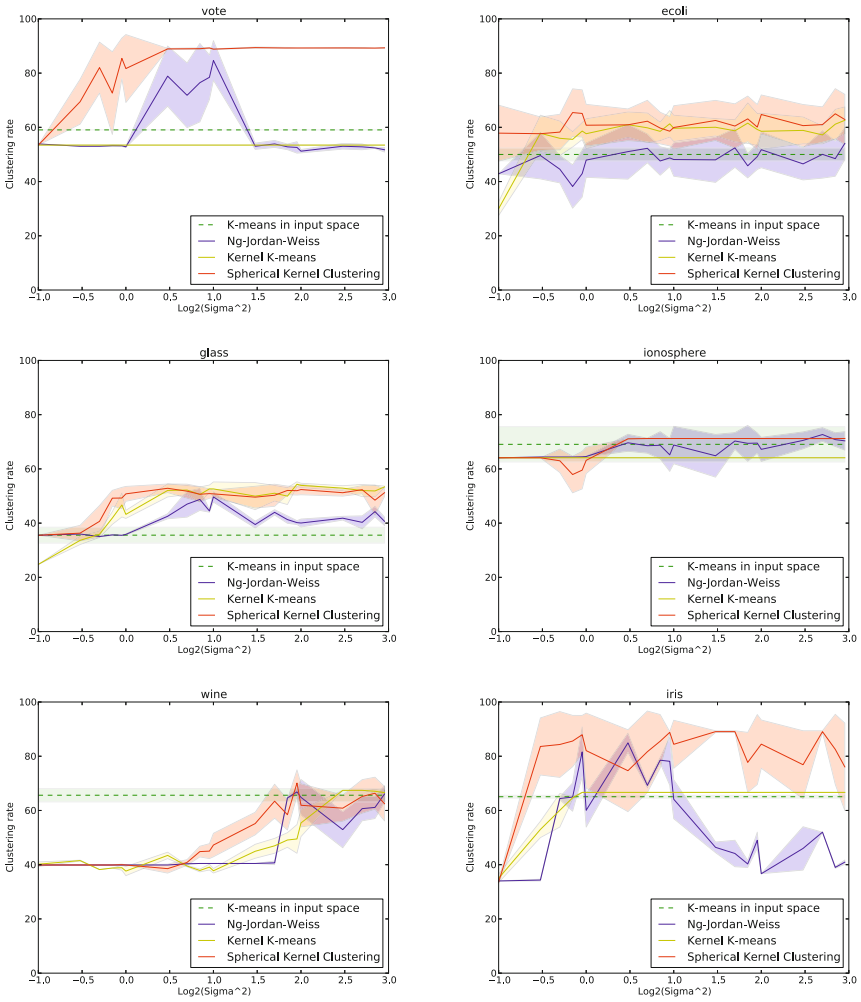

Fig. 3. Accuracy rates on the clustering task for different values of σ (on a logarithmic scale), and for the following algorithms: k-means, spectral clustering, kernel k-means and hypserspherical clustering

appears to provide similar performances while being generative. Nevertheless, let us note that our kernel would be adapted to one-class SVM for multi-class classification problems, such as in [19].

PerTurbo is a classification algorithm inspired from recent advances in computer graphics: Each class is characterized as a manifold in the input space (in a manner similar to the cloud of points giving birth to the 3D surface of a virtual object) thanks to an approximation of the Laplace-Beltrami operator [20,21]. As this approximation happens to be the Gaussian kernel, its perturbation measure (when a test sample is added to the manifold) can be re-interpreted in the

Table 1. Performances for the best σ. The last column indicates if our method outperforms the others.

Datasets	k-means	kernel k-means	spectral clustering	Hyperspherical clustering	Best ?
Iris	65.1 (0.5)	66.7 (0.0)	84.9 (3.6)	**89.1**(**0.3**)	√
Ecoli	50.0 (2.4)	62.6 (4.9)	54.1 (5.9)	**65.5**(**8.7**)	√
Ionosphere	69.1 (6.5)	64.1 (1.4)	**72.7**(**2.5**)	71.2 (0.0)	•
glass	35.6 (2.9)	**54.3**(**0.9**)	49.7 (1.9)	52.9 (1.6)	•
vote	59.1 (0.0)	53.4 (0.0)	84.7 (7.3)	**89.4**(**0.21**)	√
wine	65.6 (2.5)	67.4 (0.0)	66.8 (1.4)	**70.2**(**4.8**)	√

kernel machine learning setting. Moreover, the perturbation measure appears to correspond to the reconstruction error in Kernel-PCA. Hence, in a nutshell, PerTurbo can be interpreted as the following algorithm: (1) For each class, learn a set of eigenfunctions thanks to Kernel-PCA; (2) project any test sample onto the subspace associated to each class; (3) classify the test sample into the class for which the distance between the projection into the corresponding subspace and the sample is the smallest. Thus, PerTurbo can be related to some particular cases of subspace classifiers [22] in kernelized space.

This interpretation of PerTurbo in the kernel machines setting allows to use other kernels than the Gaussian one, whereas this latter is the only one which is proved to approximate the Laplace-Beltrami operator. Hence, we compare the result of PerTurbo with (1) the Gaussian kernel, (2) our new kernel. In addition, in order to remain comparable with more classical results of the state of the art, we also compare the results with C-SVM (using the `R` package kernlab [23]) using Gaussian kernel only. For the three algorithms, the experimental conditions are identical: the training set is made of 50% of the dataset randomly picked up, the process is repeated 30 times and the mean accuracy and its standard deviation are considered. The optimal value for the σ parameter is found with a logarithmic grid-search. For SVM, we did not use a grid-search for the C parameter, in order to make sure that the three algorithms have the same number of degrees of freedom which are fixed through a grid-search. Hence, following [3], we automatically tune C such that it is 10 times the number of training samples involved. In a multiclass setting, we consider the average number of training samples per class, *i.e.*: C = (Total number of training samples $\times$ 5)/Number of labels. Although this rule is very efficient, it only provides nearly optimal results, which may explain why on some datasets, there are little differences with the state-of-the-art accuracy. On the other hand, if one wants to fully optimize the value of C so that the regularization of the separating hyperplane is completely controlled, it is also possible to introduce an additional regularization parameter for PerTurbo and to tune it similarly to C [18]. The results are given in Table 2. The performances of PerTurbo are slightly lower than the SVM. However, this is advantageously balanced on more than half of the datasets by the use of a kernel accounting for geodesic distances. As a consequence, it appears that the

Table 2. Description of performances for classification. Mean classification accuracy rates for Perturbo with Gaussian Kernel with Euclidean (second column) and with geodesic distances (third column). The last column indicates whether the geodesic distance leads to some improvements over the Euclidean distance. Results with SVM are also given for references. The value between parenthesis is the standard deviation.

Datasets	SVM	PerTurbo (Euclidean)	PerTurbo (geodesic)	Better
Ionosphere	94.0 (1.1)	92.5 (0.9)	94.2 (1.2)	$\surd$
Blood	76.2 (1.8)	76.9 (1.2)	77.5 (1.8)	$\surd$
Parkinsons	91.0 (2.5)	95.3 (3.3)	94.9 (1.0)	$\approx$
Iris	96.4 (1.6)	96.3 (1.4)	96.8 (1.0)	$\approx$
Haberman	74.6 (2.4)	75.2 (4.2)	73.7 (0.7)	$\bullet$
Glasses	66.0 (3.9)	62.1 (0.9)	68.8 (3.0)	$\surd$
Wines	97.2 (1.5)	84.7 (2.2)	96.8 (1.0)	$\surd$
Diabetes	76.9 (1.3)	75.0 (1.8)	73.1 (1.9)	$\bullet$
Australian	86.5 (1.2)	86.2 (1.0)	84.5 (1.6)	$\bullet$
German	75.5 (1.5)	70.5 (1.8)	72.0 (1.5)	$\surd$

use of such a kernel on generative classifiers may enhance the results up to the performances of SVM. On a more qualitative point of view, it is interesting to recall that when σ increases, the portion of the sphere embedding $\phi(X)$ reduces, so that both (1) the error due to the approximation of the geodesic distance on the tangent space, and (2) the difference between the geodesic distance and the Euclidean one, decrease. As a consequence, the difference of performances between the two kernels should vanish for very high values of σ, and the well-grounded of the use of geodesic distance (in spite of the approximation in the tangent space) appears when it induces better performances than the Gaussian Kernel for small values of σ. These two phenomena are displayed in Figure 4 (for the Blood dataset).

6 Conclusion and Discussion

This work is motivated by a new idea: Some kernels have the interesting property to map the data onto a portion of the unit hypersphere, and on such a Riemannian manifold, non-linear data description techniques may be more adapted than linear ones. Thus, we first show how to adapt tools from Riemannian geometry (geodesic distances, Karcher mean) to RKHS, and we establish on clustering experiments that this path of study is worthwhile, notably through a new adaptation of the k-means algorithm on the hypersphere. Moreover, we prove that considering first order approximation of geodesic distances in the tangent space of the manifold is equivalent to use another kernel derived from the original one: when using this new kernel which directly embeds the geometry of the hypersphere, it is natural to consider linear separability method. This is also assessed by experiments on classification tasks.

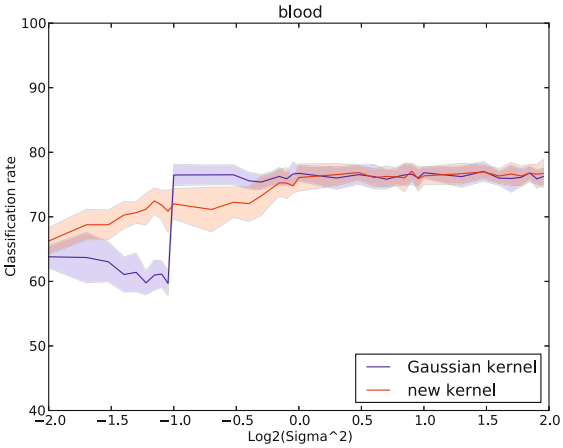

Fig. 4. Classification accuracy as a function of σ obtained on the Blood dataset. For large values of σ, accuracies of the two experimented kernels converge.

Although the proposed kernel that relies on the Karcher's mean is indeed data dependent, recent applications [19,24] seem to demonstrate that data dependent kernels may outperform data independent ones, provided that sufficient training data are available. This opens promising perspectives in multi-kernels learning, including the boosting of one-class SVM classifiers to address multi-class problems. This constitutes the most probable follow-up of this work.

Acknowledgments. The authors would like to thank the reviewers for their useful comments and remarks that helped in improving this paper. This work was supported by a Chinese Academy of Sciences visiting professorship for senior international scientists grant.

References

1. Cortes, C., Vapnik, V.: Support vector machine. Machine Learning 20(3), 273–297 (1995)
2. Schölkopf, B., Smola, A., Müller, K.R.: Kernel Principal Component Analysis. In: Gerstner, W., Hasler, M., Germond, A., Nicoud, J.-D. (eds.) ICANN 1997. LNCS, vol. 1327, pp. 583–588. Springer, Heidelberg (1997)
3. Schölkopf, B., Smola, A.J.: Learning with kernels: Support vector machines, regularization, optimization, and beyond. The MIT Press (2002)
4. Lafferty, J., Lebanon, G.: Diffusion kernels on statistical manifolds. Journal of Machine Learning Research 6, 129–163 (2005)
5. Fletcher, T., Lu, C., Pizer, S., Joshi, S.: Principal geodesic analysis for the study of nonlinear statistics of shape. IEEE Trans. Med. Imaging 23(8), 995–1005 (2004)
6. Said, S., Courty, N., LeBihan, N., Sangwine, S.J.: Exact principal geodesic analysis for data on so(3). In: Proceedings of EUSIPCO 2007, Poznan, Poland (2007)

7. Sommer, S., Lauze, F., Nielsen, M.: The differential of the exponential map, jacobi fields and exact principal geodesic analysis. CoRR, abs/1008.1902 (2010)
8. Karcher, H.: Riemannian center of mass and mollifier smoothing. Communications on Pure and Applied Mathematics 30(5), 509–541 (1977)
9. Sommer, S., Lauze, F., Hauberg, S., Nielsen, M.: Manifold Valued Statistics, Exact Principal Geodesic Analysis and the Effect of Linear Approximations. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part VI. LNCS, vol. 6316, pp. 43–56. Springer, Heidelberg (2010)
10. Kendall, W.S.: Convexity and the hemisphere. Journal of the London Mathematical Society 2(3), 567 (1991)
11. Mika, S., Schölkopf, B., Smola, A.J., Müller, K.R., Scholz, M., Rätsch, G.: Kernel pca and de-noising in feature spaces. In: Advances in Neural Information Processing Systems, pp. 536–542. MIT Press (1999)
12. Kwok, J., Tsang, I.: The pre-image problem in kernel methods. IEEE Trans. on Neural Networks 15(6), 1517–1525 (2004)
13. Huang, D., Tian, Y., De la Torre, F.: Local isomorphism to solve the pre-image problem in kernel methods. In: CVPR 2011, pp. 2761–2768 (2011)
14. Amari, S.I., Wu, S.: Improving support vector machine classifiers by modifying kernel functions. Neural Networks 12(6), 783–789 (1999)
15. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
16. Dhillon, I., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: KDD, pp. 551–556 (2004)
17. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. Advances in Neural Information Processing Systems 2, 849–856 (2002)
18. Courty, N., Burger, T., Laurent, J.: PerTurbo: A New Classification Algorithm Based on the Spectrum Perturbations of the Laplace-Beltrami Operator. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part I. LNCS, vol. 6911, pp. 359–374. Springer, Heidelberg (2011)
19. Chi-Yuan, Y., Zhi-Ying, L., Shie-Jue, L.: Boosting one-class support vector machines for multi-class classification. Applied Artificial Intelligence 23(4), 297–315 (2009)
20. Coifman, R.R., Lafon, S.: Diffusion maps. Applied and Computational Harmonic Analysis 21(1), 5–30 (2006)
21. Öztireli, C., Alexa, M., Gross, M.: Spectral sampling of manifolds. ACM Transaction on Graphics, Siggraph Asia (December 2010)
22. Cevikalp, H., Larlus, D., Neamtu, M., Triggs, B., Jurie, F.: Manifold based local classifiers: Linear and nonlinear approaches. Journal of Signal Processing Systems 61(1), 61–73 (2010)
23. Karatzoglou, A., Smola, A., Hornik, K., Zeileis, A.: kernlab-an s4 package for kernel methods in r (2004)
24. Gong, Y., Lazebnik, S.: Comparing data-dependent and data-independent embeddings for classification and ranking of internet images. In: CVPR, pp. 2633–2640. IEEE (2011)