An efficient Exact-PGA algorithm for constant curvature manifolds

Rudrasis Chakraborty[1], Dohyung Seo[2], and Baba C. Vemuri[1]
[1]Department of CISE, University of Florida, FL 32611, USA
[2]U-Systems, A GE Healthcare Company, CA, USA
[1]{rudrasis, vemuri}@cise.ufl.edu   [2]{dhseo.118}@gmail.com

Abstract

Manifold-valued datasets are widely encountered in many computer vision tasks. A non-linear analog of the PCA algorithm, called the Principal Geodesic Analysis (PGA) algorithm suited for data lying on Riemannian manifolds was reported in literature a decade ago. Since the objective function in the PGA algorithm is highly non-linear and hard to solve efficiently in general, researchers have proposed a linear approximation. Though this linear approximation is easy to compute, it lacks accuracy especially when the data exhibits a large variance. Recently, an alternative called the exact PGA was proposed which tries to solve the optimization without any linearization. For general Riemannian manifolds, though it yields a better accuracy than the original (linearized) PGA, for data that exhibit large variance, the optimization is not computationally efficient. In this paper, we propose an efficient exact PGA algorithm for constant curvature Riemannian manifolds (CCM-EPGA). The CCM-EPGA algorithm differs significantly from existing PGA algorithms in two aspects, (i) the distance between a given manifold-valued data point and the principal submanifold is computed analytically and thus no optimization is required as in the existing methods. (ii) Unlike the existing PGA algorithms, the descent into codimension-1 submanifolds does not require any optimization but is accomplished through the use of the Rimeannian inverse Exponential map and the parallel transport operations. We present theoretical and experimental results for constant curvature Riemannian manifolds depicting favorable performance of the CCM-EPGA algorithm compared to existing PGA algorithms. We also present data reconstruction from the principal components which has not been reported in literature in this setting.

1. Introduction

Principal Component Analysis (PCA) is a widely used dimensionality reduction technique in Science and Engineering. PCA however requires the input data to lie in a vector space. With the advent of new technologies and wide spread use of sophisticated feature extraction methods, manifold-valued data have become ubiquitous in many fields including but not limited to, Computer Vision, Medical Imaging and Machine Learning. A nonlinear version of PCA, called the Principal Geodesic Analysis (PGA), for data lying on Riemannian manifolds was introduced in [8].

Since the objective function of PGA is highly non-linear and hard to solve in general, researchers proposed a linearized version of the PGA [8]. Though this linearized PGA, hereafter referred to as PGA, is computationally efficient, it lacks accuracy for data with large spread/variance. In order to solve the objective function exactly, Sommer et al., [25] proposed to solve the original objective function (not the approximation) and called it *exact PGA* . While *exact PGA* attempts to solve this complex nonlinear optimization problem, it is however computationally inefficient. Though it is not possible to efficiently and accurately solve this optimization problem for a general manifold, however, for manifolds with constant sectional curvature, we formulate an efficient and exact PGA algorithm, dubbed CCM-EPGA. It is well known in geometry, by virtue of the Killing-Hopf theorem [4], that any non-zero constant curvature manifold is isomorphic to either the hypersphere ($\mathbf{S}^N$) or the hyperbolic space ($\mathbf{H}^N$), hence in this work, we present the CCM-EPGA formulation for ($\mathbf{S}^N$) and ($\mathbf{H}^N$). Our formulation has several applications to Computer Vision and Statistics including directional data [21] and color spaces [19]. Several other applications of hyperbolic geometry are, shape analysis [30], Electrical Impedance Tomography, Geoscience Imaging [28], Brain Morphometry [29], Catadiaoptric Vision [3] etc.

In order to depict the effectiveness of our proposed

1

CCM-EPGA algorithm, we use the *average projection error* as defined in [25]. We also report the computational time comparison of the CCM-EPGA with the PGA [8] and the *exact PGA* [25] algorithms respectively. Several variants of the PGA exist in literature and we briefly mention a few here. In [23], authors computed the principal geodesics (without approximation) only for a special Lie group, $SO(3)$. Geodesic PCA (GPCA) [14, 13] solves a different optimization function namely, optimizing the projection error along the geodesics. Authors in [13], minimize the projection error instead of maximizing variance in geodesic subspaces (defined later in the paper). GPCA does not use a linear approximation, but it is restricted to manifolds where a closed form expression for the geodesics exists. More recently, a probabilistic version of PGA called PPGA was presented in [31], which is a nonlinear version of PPCA [27]. None of these methods attempt to compute the solution to the exact PGA problem defined in [25]. Another recent work in [11], reports a non-linear generalization of PGA, namely the principal geodesic curves, and argues about its usefulness over PGA.

The rest of the paper is organized as follows. In Section 2, we present the formulation of PGA. We also discuss the details of the linearized version of PGA [8] and *exact PGA* [25]. Our formulation of CCM-EPGA is presented in Section 2. Experimental results for the CCM-EPGA algorithm along with comparisons to *exact PGA* and PGA are presented in Section 3. In addition to synthetic data experiments, we present the comparative performance of CCM-EPGA on two real data applications. In Section 4, we present the formulation for the reconstruction of data from principal directions and components in this nonlinear setting. Finally, in section 5, we draw conclusions.

2. Principal Geodesic Analysis

Principal Component Analysis (PCA) [17] is a well known and widely used statistical method for dimensionality reduction. Given a vector valued dataset, it returns a sequence of linear subspaces that maximize the variance of the projected data. The kth subspace is spanned by the principal vectors $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k\}$ which are mutually orthogonal. PCA is well suited for vector-valued data sets but not for manifold-valued inputs. A decade ago, the nonlinear version called the Principal Geodesic Analysis (PGA) was developed to cope with manifold-valued inputs [8]. In this section, first, we briefly describe this PGA algorithm, then, we show the key modification performed in [25] to arrive at what they termed as the exact PGA algorithm. We then motivate and present our approach which leads

to an efficient and novel algorithm for exact PGA on constant curvature manifolds (CCM-EPGA).

Let M be a Riemannian manifold. Let us suppose we are given a dataset, $X = \{x_1, \cdots, x_n\}$, where $x_j \in M$. Let us assume that the finite sample Fréchet mean [9] of the data set exists and be denoted by μ. Let V_k be the space spanned by mutually orthogonal vectors (principal directions) $\{\mathbf{v}_1, \cdots, \mathbf{v}_k\}$, $\mathbf{v}_j \in T_\mu M, \forall j$. Let S_k be the kth geodesic subspace of $T_\mu M$, i.e., $S_k = Exp_\mu(V_k)$, where Exp is the Riemannian exponential map (see [4] for definition). Then, the principal directions, $\mathbf{v}_i$ are defined recursively by

$$\mathbf{v}_i = \arg\max_{\|\mathbf{v}\|=1, \mathbf{v} \in V_{i-1}^\perp} \frac{1}{n} \sum_{j=1}^n d^2(\mu, \Pi_{S_i}(x_j)) \quad (1)$$

$$S_i = Exp_\mu(span V_{i-1}, \mathbf{v}_i) \quad (2)$$

where $d(x, y)$ is the geodesic distance between $x \in M$ and $y \in M$, $\Pi_S(x)$ is the point in S closest to $x \in M$. The PGA algorithm on M is summarized in Alg. 1.

Algorithm 1 The PGA algorithm on manifold M

1: Given a data set $X = \{x_1, \cdots, x_n\} \in M$, and $1 \leq L \leq dim(M)$
2: Compute the FM, μ, of X [1]
3: Set $k \leftarrow 1$
4: Set $\{\bar{x}_1^0, \cdots, \bar{x}_n^0\} \leftarrow \{x_1, \cdots, x_n\}$
5: **while** $k \leq L$ **do**
6:     Solve $\mathbf{v}_k = \arg\max_{\|\mathbf{v}\|=1, \mathbf{v} \in T_\mu M, \mathbf{v} \in V_{k-1}^\perp} \frac{1}{n} \sum_{j=1}^n d^2(\mu, \Pi_{S_k}(\bar{x}_j^{k-1}))$
    as in Eq. (1).
7:     Project $\{\bar{x}_1^{k-1}, \cdots, \bar{x}_n^{k-1}\}$ to a k co-dimension one submanifold Z of M, which is orthogonal to the current geodesic subspace.
8:     Set the projected points to $\{\bar{x}_1^k, \cdots, \bar{x}_n^k\}$
9:     $k \leftarrow k + 1$
10: **end while**

2.1. PGA and exact PGA

In Alg. 1 (lines $6 - 7$), as the projection operator Π is hard to compute, hence a common alternative is to locally linearize the manifold. This approach [8] maps all data points on to the tangent space at μ, and as the tangent plane is a vector space, one can use the PCA to compute the principal directions. This simple scheme is an approximation to the PGA and naturally raises the following question: *Is it possible to do PGA (solve Eq. (1)) without any linearization?* The answer is yes. But, computation of the projection operator, $\Pi_S(x)$, i.e., the closest point to x in S is computationally expensive. In [25], Sommer et al. give an alterna-

tive formulation for the PGA by minimizing the average squared reconstruction error, i.e., $d^2(x_j, \Pi_{S_i}(x_j))$ instead of $d^2(\mu, \Pi_{S_i}(x_j))$ in eqns. (1). They use an optimization scheme to compute this projection. Further, they termed their algorithm, *exact PGA*, as it does not require any linearization. *However, their optimization scheme is in general computationally expensive and for a data set with large variance, convergence is not guaranteed.* Hence, for large variance data, their *exact PGA* is still an approximation as it might not converge. This motivated us to formulate an accurate and computationally efficient *exact PGA*, at least in cases where it is feasible to do so.

2.2. Efficient and accurate exact PGA

In this paper, we present an analytic expression for the projected point and design an effective way to project data points on to the co-dimension k submanifold (as in 1, line 7). An analytic expression is in general not possible to derive for arbitrary Riemannian manifolds. However, for constant curvature Riemannian manifolds, i.e., $\mathbf{S}^N$ (positive constant curvature) and $\mathbf{H}^N$ (negative constant curvature), we derive an analytic expression for the projected point and devise an efficient algorithm to project data points on to a co-dimension k submanifold. Both these manifolds are quite commonly encountered in Computer Vision [10, 20, 3, 29, 30] as well as in many other fields of Science and Engineering. The former more so than the latter. Even though, there are applications that can be naturally posed in hyperbolic spaces (e.g., color spaces in Vision [19], catadiaoptric images [3] etc.), their full potential has not yet been exploited in Computer Vision research as much as in the former case.

We first present some background material for the N-dimensional spherical and hyperbolic manifolds and then derive an analytical expression for the projected point.

2.2.1 Basic Riemannian Geometry of $\mathbf{S}^N$

- **Geodesic distance**: The geodesic distance between $\psi, \bar{\psi} \in \mathbf{S}^N$ is given by, $d(\psi, \bar{\psi}) = \arccos(\psi^t \bar{\psi})$.

- **Exponential Map**: Given a vector $\mathbf{v} \in T_\psi \mathbf{S}^N$, the Riemannian Exponential map on $\mathbf{S}^N$ is defined as $Exp_\psi(\mathbf{v}) = \cos(|\mathbf{v}|)\psi + \sin(|\mathbf{v}|)\mathbf{v}/|\mathbf{v}|$. The Exponential map gives the point which is located on the great circle along the direction defined by the tangent vector $\mathbf{v}$ at a distance $|\mathbf{v}|$ from ψ.

- **Inverse Exponential Map**: The tangent vector $\mathbf{v} \in T_\psi \mathbf{S}^N$ directed from ψ to $\bar{\psi}$ is given

by, $Exp_\psi^{-1}(\bar{\psi}) = \frac{\theta}{\sin(\theta)}(\bar{\psi} - \psi\cos(\theta))$ where, $\theta = d(\psi, \bar{\psi})$.

2.2.2 Basic Riemannian Geometry of $\mathbf{H}^N$

The hyperbolic N-dimensional manifold can be embedded in $\mathbf{R}^{N+1}$ using any of three different models. In this paper, we use the hyperboloid model [15]. In this model, $\mathbf{H}^N$ is defined as $\mathbf{H}^N = \{\mathbf{x} = (x_1, \cdots, x_{N+1})^t \in \mathbf{R}^{N+1} | <x, x>_H = -1, x_1 > 0\}$, where the inner product on $\mathbf{H}^N$, denoted by $<x, y>_H$ is defined as $<x, y>_H = -x_1 * y_1 + \sum_{i=2}^{N+1}(x_i * y_i)$.

- **Geodesic distance**: The geodesic distance between $\psi, \bar{\psi} \in \mathbf{H}^N$ is given by, $d(\psi, \bar{\psi}) = \cosh^{-1}(-<\psi, \bar{\psi}>_H)$.

- **Exponential Map**: Given a vector $\mathbf{v} \in T_\psi \mathbf{H}^N$, the Riemannian Exponential map on $\mathbf{H}^N$ is defined as, $Exp_\psi(\mathbf{v}) = \cosh(|\mathbf{v}|)\psi + \sinh(|\mathbf{v}|)\mathbf{v}/|\mathbf{v}|$.

- **Inverse Exponential Map**: The tangent vector $\mathbf{v} \in T_\psi \mathbf{H}^N$ directed from ψ to $\bar{\psi}$ is given by $Exp_\psi^{-1}(\bar{\psi}) = \frac{\theta}{\sinh(\theta)}(\bar{\psi} - \psi\cosh(\theta))$ where, $\theta = d(\psi, \bar{\psi})$.

For the rest of this paper, we consider the underlying manifold, M, as a constant curvature Riemannian manifold, i.e., M is diffeomorphic to either $\mathbf{S}^N$ or $\mathbf{H}^N$, where $N = dim(M)$ [4]. Let $\psi, \bar{\psi} \in M$, $\mathbf{v} \in T_{\bar{\psi}}M$. Further, let $y(\mathbf{v}, \psi)$ be defined as the projection of ψ on the geodesic submanifold defined by $\bar{\psi}$ and $\mathbf{v}$. Now, we will derive a closed form expression for $y(\mathbf{v}, \psi)$ in the case of $\mathbf{S}^N$ and $\mathbf{H}^N$.
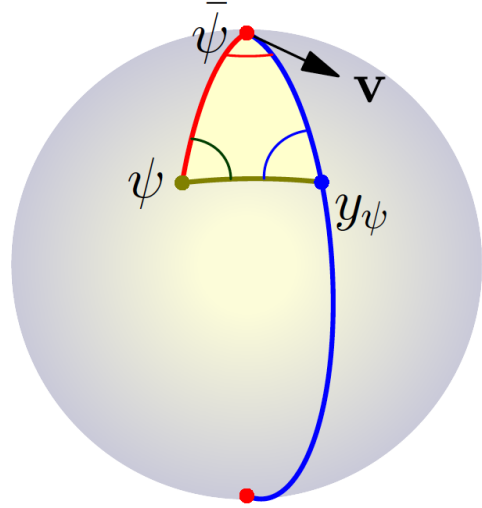

Figure 1: Projection of a data point on to a geodesic submanifold of the sphere.

2.3. Analytic expression for $y(\mathbf{v}, \psi)$ on $\mathbf{S}^N$

Theorem 1. *Let $\psi \in \mathbf{S}^N$ and $\mathbf{v} \in T_{\bar{\psi}}\mathbf{S}^N$. Then the projection of ψ on the geodesic submanifold defined by $\bar{\psi}$ and $\mathbf{v}$, i.e., $y(\mathbf{v}, \psi)$ is given by:*

$$y(\mathbf{v}, \psi) = \cos(\arctan\left(\frac{(<\mathbf{v}, \psi>)/(<\psi, \bar{\psi}>)}{|\mathbf{v}|}\right))\bar{\psi}$$

$$+ \sin(\arctan\left(\frac{(<\mathbf{v}, \psi>)/(<\psi, \bar{\psi}>)}{|\mathbf{v}|}\right))\mathbf{v}/|\mathbf{v}| \tag{3}$$

Proof. Consider the spherical triangle shown in Fig. 1, $\triangle \bar{\psi}\psi y_\psi$, where $y_\psi = y(\mathbf{v}, \psi)$. Let, $a = d(\bar{\psi}, y_\psi)$, $b = d(y_\psi, \psi)$ and $c = d(\psi, \bar{\psi})$. Also, let $A = \angle \bar{\psi}\psi y_\psi$, $B = \angle \psi \bar{\psi} y_\psi$, $C = \angle \bar{\psi} y_\psi \psi$. Clearly, since y_ψ is the projected point , $C = \pi/2$. So,

$$\cos B = \frac{<\frac{c}{\sin c}(\psi - \bar{\psi}\cos c), \mathbf{v}>}{c|\mathbf{v}|}$$

$$= \frac{\frac{<\mathbf{v}, \psi>}{\sin c} - \cot c <\mathbf{v}, \bar{\psi}>}{|\mathbf{v}|} \tag{4}$$

Here, $< ., . >$ denotes the Euclidean inner product, where both ψ and $\mathbf{v}$ are viewed as points in $\mathbf{R}^{N+1}$, i.e., the ambient space. Note that, $<\mathbf{v}, \bar{\psi}> = 0$, as $\mathbf{v} \in T_{\bar{\psi}}\mathbf{S}^N$. From spherical trigonometry, we know that $\tan a = \cos B \tan c$.

$$\therefore \cos B \tan c = \frac{\frac{<\mathbf{v}, \psi>}{\cos c}}{|\mathbf{v}|}$$

$$= \frac{(<\mathbf{v}, \psi>)/(<\psi, \bar{\psi}>)}{|\mathbf{v}|}$$

$$\therefore a = \arctan\left(\frac{(<\mathbf{v}, \psi>)/(<\psi, \bar{\psi}>)}{|\mathbf{v}|}\right) \tag{5}$$

Hence, using the Exponential map, we can show that y_ψ is given by,

$$y_\psi = \cos(a)\bar{\psi} + \sin(a)\mathbf{v}/|\mathbf{v}| \tag{6}$$

∎

Analogously, we can derive the formula for $y(\mathbf{v}, \psi)$ on $\mathbf{H}^N$, $\mathbf{v} \in T_{\bar{\psi}}\mathbf{H}^N$.

Theorem 2. *Let $\psi \in \mathbf{H}^N$ and $\mathbf{v} \in T_{\bar{\psi}}\mathbf{H}^N$. Then the projection of ψ on the geodesic submanifold defined by $\bar{\psi}$ and $\mathbf{v}$, i.e., $y(\mathbf{v}, \psi)$ is given by:*

$$y_\psi = \cosh(a)\bar{\psi} + \sinh(a)\mathbf{v}/|\mathbf{v}| \tag{7}$$

where,

$$a = \tanh^{-1}\left(\frac{(<\mathbf{v}, \psi>_H)/(-<\psi, \bar{\psi}>_H)}{|\mathbf{v}|}\right)$$

.

Proof. As before, consider the hyperbolic triangle shown in, $\triangle \bar{\psi}\psi y_\psi$, where $y_\psi = y(\mathbf{v}, \psi)$. Let, $a = d(\bar{\psi}, y_\psi)$, $b = d(y_\psi, \psi)$ and $c = d(\psi, \bar{\psi})$. Also, let $A = \angle \bar{\psi}\psi y_\psi$, $B = \angle \psi \bar{\psi} y_\psi$, $C = \angle \bar{\psi} y_\psi \psi$. Clearly, since y_ψ is the projected point , $C = \pi/2$. Then, B is the angle between $Log_{\bar{\psi}}(\psi)$ and $\mathbf{v}$. Hence,

$$\cosh B = \frac{\frac{<\mathbf{v}, \psi>_H}{\sinh c} - \coth c <\mathbf{v}, \bar{\psi}>_H}{|\mathbf{v}|} \tag{8}$$

Then, from hyperbolic trigonometry, as $\tanh a = \cosh B \tanh c$, we get

$$a = \tanh^{-1}\left(\frac{(<\mathbf{v}, \psi>_H)/(-<\psi, \bar{\psi}>_H)}{|\mathbf{v}|}\right) \tag{9}$$

Note that, since the arc length between $\bar{\psi}$ and y_ψ is a, hence, using the Exponential map, we can show that $y_\psi = \cosh(a)\bar{\psi} + \sinh(a)\mathbf{v}/|\mathbf{v}|$.

∎

Given the closed form expression for the projected point, now we are in a position to develop an efficient projection algorithm (for line 7 in Alg. 1), which is presented in Alg. 2. Note that, using Alg. 2, data points on the current submanifold are projected to a submanifold of dimension one less, which is needed by the PGA algorithm in line 6. *Also note that, in order to ensure existence and uniqueness of FM on $\mathbf{S}^N$, we have restricted the data points to be within a geodesic ball of convexity radius $< \pi/2$ [1]. On $\mathbf{H}^N$, FM exists and is unique everywhere.*

Note that in order to descend to the codimension-1 submanifolds, we use step-1 and step-2 instead of the optimization method used in the exact PGA algorithm of [25].

3. Experimental Results

In this section, we present experiments demonstrating the performance of CCM-EPGA compared to PGA [8] and *exact PGA* [25]. We used the *average projection error*, defined in [25], as a measure of performance in our experiments. The *average projection error* is defined as follows. Let $\{x_i\}_{i=1}^n$ be data points on a manifold M. Let μ be the mean of the data points. Let, $\mathbf{v}$

Algorithm 2 Algorithm for projecting the data points to a co-dimension one submanifold

1: *Input*: a data point $x_i \in \mathbf{S}^N(\mathbf{H}^N)$, a geodesic submanifold defined at μ and $\mathbf{v} \in T_\mu \mathbf{S}^N(T_\mu \mathbf{H}^N)$, and $y(\mathbf{v}, x_i)$ which is the projection of ψ on to the geodesic submanifold.

2: *Output*: $\bar{x}_i$ which is the projection of the data point x_i to a subspace, $\mathbf{S}^{N-1}(\mathbf{H}^{N-1})$, that is orthogonal to the current geodesic submanifold.

3: Step 1. Evaluate the tangent vector, $\mathbf{v}_i \in T_{y(\mathbf{v},x_i)}\mathbf{S}^N(T_{y(\mathbf{v},x_i)}\mathbf{H}^N)$ directed towards x_i using the inverse Exponential map. It is clear that $\mathbf{v}_i$ is orthogonal to $\mathbf{v}$.

4: Step 2. Parallel transport $\mathbf{v}_i$ to μ. Let $\mathbf{v}_i^\mu$ denote the parallel transported vector. The geodesic submanifold defined by μ and $\mathbf{v}_i^\mu$ is orthogonal to geodesic submanifolds obtained from the previous steps in Alg. 1.

5: Step 3. Set $\bar{x}_i \leftarrow y(\mathbf{v}_i^\mu, x_i)$

be the first principal direction and $S = Exp_\mu(\mathbf{v})$. Then the error (E) is defined as follows:

$$E = \frac{1}{n} \sum_{i=1}^n d^2(x_i, \Pi_S(x_i)) \qquad (10)$$

where $d(.,.)$ is the geodesic distance function on M. We also present the computation time for each of the three algorithms. All the experimental results reported here were obtained on a desktop with a single 3.33 GHz Intel-i7 CPU and 24 GB RAM.

3.1. Comparative performance of CCM-EPGA on Synthetic data

In this section, we present the comparative performance of CCM-EPGA on several synthetic datasets. For each of the synthetic data, we have reported the *average projection error* and computation time for all three PGA algorithms in Table 1. All the four datasets are on S^2 and the Fréchet mean is at the "north pole". For all the datasets, samples are in the northern hemisphere to ensure that the Fréchet mean is unique. Data_1 and Data_2 are generated by taking samples along a geodesic with a slight perturbation. The last two datasets are constructed by drawing random samples on the northern hemisphere. In addition, data points from Data_1 are depicted in Fig. 2. The first principal direction is also shown (black for CCM-EPGA, blue for PGA and red for *exact PGA*). Further, we also report the data variance for these synthetic datasets. By examining the results, it's evident that for data with low variance, the significance of CCM-EPGA in terms of projection error is marginal, while
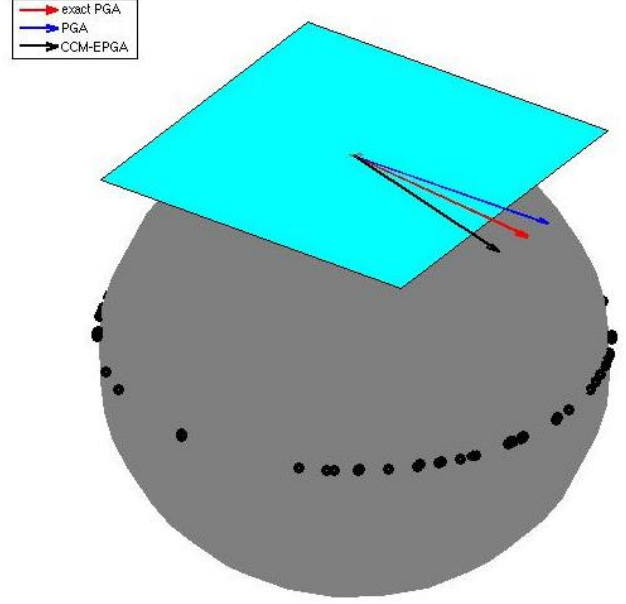
Figure 2: Synthetic data (Data_1) on S^2

for high variance data, CCM-EPGA yields significantly better accuracy. Also, CCM-EPGA is computationally very fast in comparison to *exact PGA*. The results in Table 1 indicate that CCM-EPGA outperforms *exact PGA* in terms of efficiency and accuracy. Although, the required time for PGA is less than that of CCM-EPGA, in terms of accuracy, CCM-EPGA dominates PGA.

3.2. Comparative performance on point-set data ($\mathbf{S}^N$ example)

In this section, we depict the performance of the proposed CCM-EPGA algorithm on 2D point-set data. The database is called GatorBait-100 dataset [22]. This dataset consists of 100 images of shapes of different fish. From each of these images of size 20×200, we first extract boundary points, then we apply the Schrödinger distance transform [7] to map each of these point sets on a hypersphere (S^{3999}). Hence, this data consists of 100 point-sets each of which lie on S^{3999}. As before, we have used the *average projection error* [25], to measure the performance of algorithms in the comparisons. Additionally, we report the computation time for each of these PGA algorithms. We used the code available online for *exact PGA* [24]. This online implementation is not scalable to large (even moderate) number of data points, and further requires the computation of the Hessian matrix in the optimization step, which is computationally expensive. Hence, for this real data application on the high dimensional hyper-

Data	Var.	CCM-EPGA		PGA		*exact PGA*	
		avg. proj. err.	Time(s)	*avg. proj. err.*	Time(s)	*avg. proj. err.*	Time(s)
Data_1	2.16	**1.13e − 04**	0.70	0.174	**0.46**	2.54e-02	14853
Data_2	0.95	**5.87e − 02**	0.27	0.59	**0.12**	0.59	84.38
Data_3	7.1e-03	**2.33e − 03**	0.19	0.55	**0.05**	0.55	16.87
Data_4	5.9e-02	**0.27**	0.33	0.37	**0.14**	0.37	71.84

Table 1: Comparison results on synthetic datasets

Method	*avg. proj. error*	Time(s)
CCM-EPGA	**2.83e − 10**	0.40
PGA	9.68e − 02	**0.28**

Table 2: Comparison results on *Gator-Bait* database

sphere, we could not report the results for the *exact PGA* algorithm. Though one can use a Sparse matrix version of the exact PGA code, along with efficient parallelization to make the *exact PGA* algorithm suitable for moderately large data, we would like to point out that since our algorithm does not need such modifications, it clearly gives CCM-EPGA an advantage over *exact PGA* from a computational efficiency perspective. In terms of accuracy, it can be clearly seen that CCM-EPGA outperforms *exact PGA* from the results on synthetic datasets. Both *average projection error* and computational time on GatorBait-100 dataset are reported in Table 2. This result demonstrates accuracy of CCM-EPGA over the PGA algorithm with a marginal sacrifice in efficiency but significant gains in accuracy.

3.3. PGA on population of Gaussian distributions ($\mathbf{H}^N$ example)

In this section, we propose a novel scheme to compute principal geodesic submanifolds for the manifold of Gaussian densities. Here, we use concepts from information geometry presented in [2], specifically, the Fisher information matrix [18] to define a metric on this manifold [6]. Consider a normal density $f(.|\boldsymbol{\theta})$ in an $n-$dimensional space, with parameters represented by $\boldsymbol{\theta}$. Then the ij^{th} entry of the $n \times n$ Fisher matrix, denoted by g_{ij}, is defined as follows:

$$g_{ij}(\boldsymbol{\theta}) = \int_{\mathbf{R}} f(x|\boldsymbol{\theta}) \frac{\partial ln f(x|\boldsymbol{\theta})}{\partial \theta_i} \frac{\partial ln f(x|\boldsymbol{\theta})}{\partial \theta_j} dx \quad (11)$$

For example, for a univariate normal density $f(.|\mu, \sigma)$, the fisher information matrix is

$$(g_{ij}(\mu, \sigma)) = \begin{pmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{2}{\sigma^2} \end{pmatrix} \quad (12)$$

So, the metric is defined as follows:

$$< u, v >= u^t G v \quad (13)$$

where $G = (g_{ij})$ is the Fisher information matrix. Now, consider the parameter space for the univariate normal distributions. The parameter space is $H_F = (\mu, \sigma) \in \mathbf{R}^2 | \sigma > 0$, i.e., positive half space, which is the Hyperbolic space, modeled by the Poincaré half-plane, denoted by $\mathbf{P}^2$. We can define a bijection $F_1 : H_F \to \mathbf{P}^2$ as $F(\mu, \sigma) = (\frac{\mu}{\sqrt{2}}, \sigma)$. Hence, the univariate normal distributions can be parameterized by the 2−dimensional hyperbolic space. Moreover, there exists a diffeomorphsim between $\mathbf{P}^2$ and $\mathbf{H}^2$ (the mapping is analogous to stereographic projection for $\mathbf{S}^N$), thus, we can readily use the formulation in Section 2 to compute principal geodesic submanifold on the manifold of univariate normal distributions.

Motivated by the above formulation, we ask the following question: *Does there exist a similar relation for multivariate normal distributions?* The answer is no in general. But if the multivariate distributions have diagonal covariance matrix, (i.e., independent uncorrelated variables in the multivariate case), the above relation between $\mathbf{P}^2$ and $\mathbf{H}^2$ can be generalized. Consider an $N-$dimensional normal distribution parameterized by $(\boldsymbol{\mu}, \Sigma)$ where $\boldsymbol{\mu} = (\mu_1, \cdots, \mu_N)^t$ and Σ is a diagonal positive definite matrix (i.e., $\Sigma_{ij} = \sigma_i$, if $i = j$, else $\Sigma_{ij} = 0$). Then, analogous to the univariate normal distribution case, we can define a bijection $F_N : H_F^N \to \mathbf{P}^{2N}$ as follows:

$$F_N(\boldsymbol{\mu}, \Sigma) = (\frac{\mu_1}{\sqrt{2}}, \sigma_1, \cdots, \frac{\mu_N}{\sqrt{2}}, \sigma_N) \quad (14)$$

Hence, we can use our formulation in Section 2 since there is a diffeomorphism between $\mathbf{P}^{2N}$ and $\mathbf{H}^{2N}$. But, for general non-diagonal $N-$dimensional covariance matrix space, $SPD(N)$, the above formulation does not hold. This motivated us to go one step further to search for a parameterization of $SPD(N)$ where we can use the above formulation. In [16], authors have used the Iwasawa coordinates to parameterize $SPD(N)$. Using the Iwasawa coordinates [26],

we can get a one-to-one mapping between $SPD(N)$ and the product manifold of $PD(N)$ and $U(N-1)$, where $PD(N)$ is manifold of $N-$dimensional diagonal positive definite matrix and $U(N-1)$ is the space of $(N-1)-$dimensional upper triangular matrices, which is isomorphic to $R^{N(N+1)/2}$. We have used the formulation in [26], as discussed below.

Let $Y = V_N \in SPD(N)$, then we can use Iwasawa decomposition to represent V_N as a tuple $(V_{N-1}, x_{N-1}, w_{N-1})$. And repeating the following partial Iwasawa decomposition:

$$V_N = \begin{pmatrix} I & x_{N-1} \\ 0 & 1 \end{pmatrix}^T \begin{pmatrix} V_{N-1} & 0 \\ 0 & w_{N-1} \end{pmatrix} \qquad (15)$$

where $w_{N-1} > 0$ and $x_{N-1} \in \mathbf{R}^{N-1}$. We get the following vectorized expression: $V_N \mapsto (((w_0, x_1^t, w_1), x_2^t, w_2), \cdots, x_{N-1}^t, w_{N-1})$. Note that as each of w_i is > 0, we can construct a positive definite diagonal matrix with i^{th} diagonal entry being w_i. And as each x_i is in $\mathbf{R}^i$, we will arrange them column-wise to form a upper triangular matrix. Thus, for $PD(N)$, we can use our formulation for the hyperboloid model of the hyperbolic space given in Section 2, and the standard PCA can be applied for $R^{N(N+1)/2}$.

We now use the above formulation to compute the principal geodesic submanifolds for a covariance descriptor representation of Brodatz texture dataset [5]. Similar to our previous experiment on point-set data, in this experiment, we report the *average projection error* and the computation time. We adopt a similar procedure as in [12] to derive the covariance descriptors for the texture images in the Brodatz database. Each 256×256 texture image is first partitioned into 64 non-overlapping 8×8 blocks. Then, for each block, the covariance matrix (FF^T) is summed over the blocks. Here, the covariance matrix is computed from the feature vector $F = \left(I, |\frac{\partial I}{\partial x}|, |\frac{\partial I}{\partial y}|, |\frac{\partial^2 I}{\partial x^2}|, |\frac{\partial^2 I}{\partial y^2}|\right)^t$. We make the covariance matrix positive definite by adding a small positive diagonal matrix. Then, each image is represented as a normal distribution with zero mean and this computed covariance matrix. Then, we used the above formulation to map each normal distribution on to $\mathbf{H}^{10}$. The comparative results of CCM-EPGA with PGA and *exact PGA* are presented in Table 3. The results clearly demonstrate the efficiency and accuracy of CCM-EPGA over the PGA and the exact PGA algorithms.

Method	avg. proj. error	Time(s)
CCM-EPGA	**7.73e − 03**	0.09
PGA	0.14	**0.05**
exact PGA	0.09	732

Table 3: Comparison results on *Brodatz* database

Figure 3: Approximation of x_j from the first k principal components.

4. Data Reconstruction from principal directions and coefficients

In this section, we present a recursive scheme to approximate an original data point with principal directions and coefficients. We present a reconstruction method for data on $\mathbf{S}^N$, the reconstruction for data points on $\mathbf{H}^N$ can be done in an analogous manner. Let $x_j \in \mathbf{S}^N$ be the jth data point and $\mathbf{v}_k$ the kth principal vector. $\bar{x}_j^k$ is the kth principal component of x_j. Note that on $\mathbf{S}^N$, k^{th} principal component of a data point is $y(\mathbf{v}_k, x_j)$. Let x_j^k be the approximated x_j from the first k principal components. Let $\mathbf{w}_j^{k-1}$ be $Log_\mu x_j^{k-1}/\|Log_\mu x_j^{k-1}\|$. Let $\bar{\mathbf{w}}_j^{k-1}$ be the parallel transported vector $\mathbf{w}_j^{k-1}$ from μ to $\bar{x}_j^k$. Let, $\bar{\mathbf{v}}_k$ be the parallel transported version of $\mathbf{v}_k$ to x_j^{k-1}. We refer readers to Fig. 3 for a geometric interpretation.

Now, we will formulate a recursive scheme to reconstruct x_j. Let us reconstruct the data using the first $(k-1)$ principal components. Then, the kth approximated point x_j^k is the intersection of two geodesics defined by x_j^{k-1}, $\bar{\mathbf{v}}_k$ and $\bar{x}_j^k$, $\bar{\mathbf{w}}_j^{k-1}$. Let these two great

circles be denoted by

$$G_1(t) = \cos(t)x_j^{k-1} + \sin(t)\bar{\mathbf{v}}_k \qquad (16)$$

$$G_2(u) = \cos(u)\bar{x}_j^k + \sin(u)\bar{\mathbf{w}}_j^{k-1} \qquad (17)$$

At $t = \alpha_1$ and $u = \alpha_2$, let $G_1(\alpha_1) = G_2(\alpha_2) = x_j^k$. Since, $\bar{\mathbf{v}}_k$ and $\bar{\mathbf{w}}_j^{k-1}$ are mutually orthogonal, we get,

$$\tan(\alpha_1)\tan(\alpha_2) = <x_j^{k-1}, \bar{\mathbf{w}}_j^{k-1}><\bar{x}_j^k, \bar{\mathbf{v}}_k> \quad (18)$$

Note that, as our goal is to solve for α_1 or α_2 to get x_j^k, we need two equations. The second equation can be derived as follows:

$$d(\mu, G_1(\alpha_1)) = d(\mu, G_2(\alpha_2))$$

This leads to,

$$\cos(\alpha_2) = \frac{\cos(\alpha_1)<\mu, x_j^{k-1}>}{<\mu, \bar{x}_j^k>} \qquad (19)$$

Then, by solving Eqs. (18) and (19) we get,

$$a\cos^4(\alpha_1) + b\cos^2(\alpha_1) + d = 0 \qquad (20)$$

where,

$$a = <\mu, x_j^{k-1}>^2 <x_j^{k-1}, \bar{\mathbf{w}}_j^{k-1}>^2 <\bar{x}_j^k, \bar{\mathbf{v}}_k>^2$$
$$- <\mu, x_j^{k-1}>^2$$

$$b = <\mu, \bar{x}_j^k>^2 + <\mu, x_j^{k-1}>^2$$

and

$$d = - <\mu, \bar{x}_j^k>^2$$

By solving the equation (20), we get

$$\alpha_1 = \arccos\left(\sqrt{\frac{-b + \sqrt{(b^2 - 4ad)}sgn(a)}{2a}}\right) \qquad (21)$$

where, $sgn(.)$ is the signum function. Hence, $x_j^k = G_1(\alpha_1)$. This completes the reconstruction algorithm. Our future efforts will be focused on using this reconstruction algorithm in a variety of applications mentioned earlier.

5. Conclusions

In this paper, we presented an efficient and accurate exact-PGA algorithm for (non-zero) constant curvature manifolds, namely the hypersphere S^n and the hyperbolic space H^n. We presented an analytic expression for the projection of a data point on a geodesic submanifold, which is required in the PGA algorithm and in general involves solving a difficult optimization

problem. Using these analytic expressions, we achieved a much more accurate and efficient solution for PGA on constant curvature manifolds, that are frequently encountered in Computer Vision, Medical Imaging and Machine Learning tasks. We presented comparison results on synthetic and real data sets demonstrating favorable performance of our algorithm in comparison to the state-of-the-art.

Acknowledgement

References

[1] B. Afsari. Riemannian L^{p} center of mass: Existence, uniqueness, and convexity. *Proceedings of the American Mathematical Society*, 139(2):655–673, 2011. 2, 4

[2] S.-i. Amari and H. Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Soc., 2007. 6

[3] I. Bogdanova, X. Bresson, J.-P. Thiran, and P. Vandergheynst. Scale space analysis and active contours for omnidirectional images. *Image Processing, IEEE Transactions on*, 16(7):1888–1901, 2007. 1, 3

[4] W. M. Boothby. *An introduction to differentiable manifolds and Riemannian geometry*, volume 120. Academic press, 1986. 1, 2, 3

[5] P. Brodatz. *Textures: a photographic album for artists and designers*. Dover Pubns, 1966. 7

[6] S. I. Costa, S. A. Santos, and J. E. Strapasson. Fisher information distance: a geometrical reading. *Discrete Applied Mathematics*, 2014. 6

[7] Y. Deng, A. Rangarajan, S. Eisenschenk, and B. C. Vemuri. A riemannian framework for matching point clouds represented by the schrödinger distance transform. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3756–3761. IEEE, 2014. 5

[8] P. T. Fletcher, C. Lu, S. M. Pizer, and S. Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *Medical Imaging, IEEE Transactions on*, 23(8):995–1005, 2004. 1, 2, 4

[9] M. Fréchet. Les éléments aléatoires de nature quelconque dans un espace distancié. In *Annales de l'institut Henri Poincaré*, volume 10, pages 215–310. Presses universitaires de France, 1948. 2

[10] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *International journal of computer vision*, 103(3):267–305, 2013. 3

[11] S. Hauberg. Principal curves on riemannian manifolds. 2015. 2

[12] J. Ho, Y. Xie, and B. Vemuri. On a nonlinear generalization of sparse coding and dictionary learning. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1480–1488, 2013. 7

[13] S. Huckemann, T. Hotz, and A. Munk. Intrinsic shape analysis: Geodesic pca for riemannian manifolds modulo isometric lie group actions. *Statistica Sinica*, 2010. 2

[14] S. Huckemann and H. Ziezold. Principal component analysis for riemannian manifolds, with an application to triangular shape spaces. *Advances in Applied Probability*, pages 299–319, 2006. 2

[15] B. Iversen. *Hyperbolic geometry*, volume 25. Cambridge University Press, 1992. 3

[16] B. Jian and B. C. Vemuri. Metric learning using iwasawa decomposition. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–6. IEEE, 2007. 6

[17] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2002. 2

[18] E. L. Lehmann and G. Casella. *Theory of point estimation*, volume 31. Springer Science & Business Media, 1998. 6

[19] R. Lenz, P. L. Carmona, and P. Meer. The hyperbolic geometry of illumination-induced chromaticity changes. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–6. IEEE, 2007. 1, 3

[20] R. Lenz and G. Granlund. If i had a fisheye i would not need so (1, n), or is hyperbolic geometry useful in image processing. In *Proc. of the SSAB Symposium, Uppsala, Sweden*, pages 49–52, 1998. 3

[21] K. Mardia and I. Dryden. Shape distributions for landmark data. *Advances in Applied Probability*, pages 742–755, 1989. 1

[22] A. Rangarajan. https://www.cise.ufl.edu/~anand/publications.html. 5

[23] S. Said, N. Courty, N. Le Bihan, and S. J. Sangwine. Exact principal geodesic analysis for data on so (3). In *15th European Signal Processing Conference (EUSIPCO-2007)*, pages 1700–1705. EURASIP, 2007. 2

[24] S. Sommer. https://github.com/nefan/smanifold, 2014. 5

[25] S. Sommer, F. Lauze, S. Hauberg, and M. Nielsen. Manifold valued statistics, exact principal geodesic analysis and the effect of linear approximations. In *Computer Vision–ECCV 2010*, pages 43–56. Springer, 2010. 1, 2, 4, 5

[26] A. Terras. *Harmonic analysis on symmetric spaces and applications*. Springer, 1985. 6, 7

[27] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999. 2

[28] G. Uhlmann. *Inverse Problems and Applications: Inside Out II*, volume 60. Cambridge University Press, 2013. 1

[29] Y. Wang, W. Dai, X. Gu, T. F. Chan, S.-T. Yau, A. W. Toga, and P. M. Thompson. Teichmüller shape space theory and its application to brain morphometry. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2009*, pages 133–140. Springer, 2009. 1, 3

[30] W. Zeng, D. Samaras, and X. D. Gu. Ricci flow for 3d shape analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(4):662–677, 2010. 1, 3

[31] M. Zhang and P. T. Fletcher. Probabilistic principal geodesic analysis. In *Advances in Neural Information Processing Systems*, pages 1178–1186, 2013. 2