Intrinsic Grassmann Averages for Online Linear, Robust and Nonlinear Subspace Learning

Rudrasis Chakraborty, Liu Yang, Søren Hauberg and Baba C. Vemuri*, *Fellow, IEEE*

Abstract—Principal Component Analysis (PCA) and Kernel Principal Component Analysis (KPCA) are fundamental methods in machine learning for dimensionality reduction. The former is a technique for finding this approximation in finite dimensions and the latter is often in an infinite dimensional Reproducing Kernel Hilbert-space (RKHS). In this paper, we present a geometric framework for computing the principal linear subspaces in both (finite and infinite) situations as well as for the robust PCA case, that amounts to computing the intrinsic average on the space of all subspaces: the Grassmann manifold. Points on this manifold are defined as the subspaces spanned by K-tuples of observations. The intrinsic Grassmann average of these subspaces are shown to coincide with the principal components of the observations when they are drawn from a Gaussian distribution. We show similar results in the RKHS case and provide an efficient algorithm for computing the projection onto the this average subspace. The result is a method akin to KPCA which is substantially faster. Further, we present a novel online version of the KPCA using our geometric framework. Competitive performance of all our algorithms are demonstrated on a variety of real and synthetic data sets.

Index Terms—Online Subspace Learning, Robust, PCA, Kernel PCA, Grassmann Manifold, Fréchet Mean, Fréchet Median.

1 Introduction

Principal component analysis (PCA), a key work-horse of machine learning, can be derived in many ways: Pearson [1] proposed to find the subspace that minimizes the projection error of the observed data; Hotelling [2] instead sought the subspace in which the projected data has maximal variance; and Tipping & Bishop [3] considered a probabilistic formulation where the covariance of normally distributed data is predominantly given by a low-rank matrix. All these derivations led to the same algorithm. There are many generalizations of PCA including but not limited to sparse PCA [4], generalized PCA [5] and we refer the interested reader to a recent survey article [6]. We will not be addressing any of these in our current work. Recently, Hauberg et al. [7] noted that the average of all one-dimensional subspaces spanned by normally distributed data coincides with the leading principal component. They computed the average over the Grassmann manifold of one-dimensional subspaces (cf. Sec. 2). This average was computed very efficiently, but unfortunately their formulation does not generalize to higher-dimensional subspaces.

In this paper, we provide a formulation for estimating the average K-dimensional subspace spanned by the observed data, and present a very simple online algorithm for computing this average. This algorithm is parameter free, and highly reliable, which is in contrast to current state-of-the-art online techniques, that are sensitive to hyper-parameter tuning. When the data is normally distributed, we show that our estimated average subspace coincides with that spanned by the leading K principal components. Moreover, since our algorithm is online, it has a linear complexity in terms of the number of samples. Furthermore, we propose an online robust subspace averaging algorithm which can be used to get the leading K robust principal components. Analogous to its non-robust counterpart, it has a linear time complexity in terms of the number of samples. In this article, we extend an early conference paper [8] and perform online non-linear subspace learning which is an online version of the kernel PCA. In comparison to our preliminary work in [8], this paper contains a more detailed analysis in addition to the generalization akin to kernel PCA [9].

- *R. Chakraborty and B. C. Vemuri are with University of Florida, Gainesville, FL, USA. * indicates corresponding author*
  E-mail: rudrasischa@gmail.com, vemuri@ufl.edu
- *S. Hauberg is at the Technical University of Denmark, Kgs. Lyngby, Denmark.*
  E-mail: sohau@dtu.dk
- *L. Yang is at the University of California, Berkeley.*
  E-mail: liu-yang@berkeley.edu

1.1 Related Work

In this paper we consider a simple linear dimensionality reduction algorithm that works in an online setting, i.e. only uses each data point once. There are several existing approaches in the literature that tackle the online PCA and the online Robust PCA problems and we discuss some of these approaches here before discussing some related works on kernel PCA:

Online PCA and online robust PCA: *Oja's rule* [10] is a classic online estimator for the leading principal components of a dataset. Given a basis $V_{t-1} \in \mathbf{R}^{D\times K}$ this is updated recursively via $V_t = V_{t-1} + \gamma_t X_t (X_t^T V_{t-1})$ upon receiving the observation X_t. Here γ_t is the step-size (learning rate) parameter that must be set manually; small values yields slow-but-sure convergence, while larger values may lead to fast-but-unstable convergence. Several variants of Oja's method have been proposed in literature and analyzed theoretically. Most recently, Allen-Zhu and Li [11] presented global convergence analysis of Oja's algorithm. Their algorithm convergence is however dependent on the gap between k^{th} and $(k+1)^{th}$ eigen values (which in practice is unknown without already knowing the eigen spectrum) and a gap free

result which further depends on another parameter called 'virtual gap'. In contrast, our proposed online algorithm for computing the principal components is parameter free.

EM PCA [12] is usually derived for probabilistic PCA, and is easily be adapted to the online setting [13]. Here, the E- and M-steps are given by:

$$\textbf{(E-step)}\ Y_t = (V_{t-1}^T V_{t-1})^{-1}(V_{t-1}^T X_t) \quad (1)$$

$$\textbf{(M-step)}\ \tilde{V}_t = (X_t Y_t^T)(Y_t Y_t^T)^{-1}. \quad (2)$$

The basis is updated recursively via the recursion, $V_t = (1-\gamma_t)V_{t-1} + \gamma_t \tilde{V}_t$, where γ_t is a step-size.

GROUSE and GRASTA [14], [15] are online PCA and matrix completion algorithms. GRASTA can be applied to estimate principal subspaces incrementally on subsampled data. Both of these methods are online and use rank-one updating of the principal subspace at each iteration. GRASTA is an online robust subspace tracking algorithm and can be applied to subsampled data and specifically matrix completion problems. He et al. [15] proposed an ℓ_1-norm based fidelity term that measures the error between the subspace estimate and the outlier corrupted observations. The robustness of GRASTA is attributed to this ℓ_1-norm based cost. Their formulation of the subspace estimation involves the minimization of a non-convex function in an augmented Lagrangian framework. This optimization is carried out in an alternating fashion using the well-known ADMM [16] for estimating a set of parameters involving the weights, the sparse outlier vector and the dual vector in the augmented Lagrangian framework. For fixed estimated values of these parameters, they employed an incremental gradient descent to solve for the low dimensional subspace. Note that the solution obtained is not the optimum of the combined non-convex function of GRASTA.

Recursive covariance estimation [17] is straight-forward, and the principal components can be extracted via standard eigen decompositions. Boutsidis et al. [17] considered efficient variants of this idea, and provided elegant performance bounds. The approach does not however scale to high-dimensional data as the covariance cannot practically be stored in memory for situations involving very large data sets as those considered in our work.

Candes et al. [18] formulated Robust PCA (RPCA) as separating a matrix into a low rank (L) and a sparse matrix (S), i.e., data matrix $X \approx L + S$. They proposed Principal Component Pursuit (PCP) method to robustly find the principal subspace by decomposing into L and S. They showed that both L and S can be computed by optimizing an objective function which is a linear combination of nuclear norm on L and ℓ_1-norm on S. Recently, Lois et al. [19] proposed an online RPCA algorithm to solve two interrelated problems, matrix completion and online robust subspace estimation. Candes et al. [18] had some assumptions including a good estimate of the initial subspace and that the basis of the subspace is dense. Though the authors have shown correctness of their algorithm under these assumptions, they are often not practical. In [20], authors proposed an alternative approach to solve RPCA by developing an augmented Lagrangian based approach, IALM.

A practical application to do robust PCA is where one needs to deal with missing data. Though incremental SVD (ISVD) [21] can be used to develop online PCA, the extension to missing data is not straightforward. In literature, there are several methods to do online robust PCA for missing data cases including MDISVD [22], Brand's [23] and PIMC [24]. All these approaches start with learning the projection matrix to the coordinates which are present for a data sample. These methods differ in the way they compute singular value matrix. If S_n is the n^{th} singular value matrix, then these methods try to find λS where $\lambda = 1$ for a natural extension of ISVD for missing data, denoted by MDISVD [22]. Brand [23] used $\lambda \in (0, 1)$ while in PIMC [24], the authors chose λ based on the norm of the data observed so far. PETRELS [25] is an extension of PAST [26] on missing data. PAST used subspace tracking idea with an approximation of projection operator. Several other researchers have proposed subspace tracking algorithm for missing data including REPROCS [27].

In another recent work, Ha and Barber [28] proposed an online RPCA algorithm when $X = (L + S)C$ where C is a data compression matrix. They proposed an algorithm to extract L and S when the data X are compress-sensed. This problem is quite interesting in its own right but not something pursued in our work presented here. Feng et al. [29] solved RPCA using a stochastic optimization approach. They showed that if each observation is bounded, then their solution converges to the batch mode RPCA solution, i.e., their sequence of robust subspaces converges to the "true" subspace. Hence, they claimed that as the "true subspace" (subspace recovered by RPCA) is robust, so is their online estimate. Though their algorithm is online, the optimization steps (Algorithm 1 in [29]) are computationally expensive for high-dimensional data. In an earlier paper, Feng et al. [30] proposed a deterministic approach to solve RPCA (dubbed DHR-PCA) for high-dimensional data. They also showed that they can achieve maximal robustness, i.e., a breakdown point of 50%. They proposed a robust computation of the variance matrix and then performed PCA on this matrix to get robust PCs. This algorithm is suitable for very high dimensional data. As most of our real applications in this paper are in very high dimensions, we find DHR-PCA to be well suited to carry out comparisons with. For further literature study on this rich topic, we refer the reader to [31], [32].

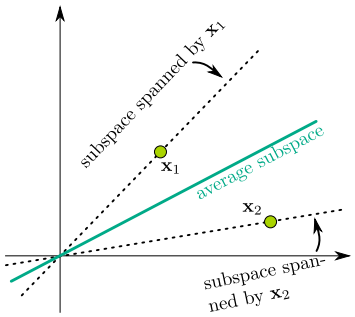


Fig. 1. The average of two subspaces.

Online kernel PCA: In this paper, we propose an online subspace averaging algorithm that we also extend to an algorithm akin to kernel PCA [9], i.e., to compute non-linear subspaces. We show that with the popular kernel

trick, we can extend our subspace averaging algorithm to compute a non-linear average subspace. But, because of the infinite dimensionality of the *reproducing kernel Hilbert space (RKHS)*, it is not computationally feasible to make this an online algorithm. We however resolve this problem by a finite approximation of the kernel using the method proposed in [33] leading to an online non-linear subspace averaging algorithm. The past work in this context includes extension of Oja's rule to perform kernel PCA [9]. Honeine [34] proposed an online kernel PCA algorithm. They pointed out that as the principal vector is a linear combination of the kernel functions associated with the available training data, it becomes a bottleneck in making the kernel PCA online. They overcame this problem by controlling the order of the model. The algorithm starts with a set of pre-selected kernel functions. Upon the arrival of a new observation, the algorithm decides whether to include or discard the observation from the set of kernel functions. Thus, by restricting the number of kernel functions, they made their kernel PCA algorithm an online algorithm. Ghashami et al. [35] proposed a streaming kernel PCA algorithm (SKPCA) which uses the finite approximation of kernel and stores a small set of basis in an online fashion. Some other online KPCA algorithms include [36], [37].

Motivation and contribution: Our work is motivated by the work presented by Hauberg et al. [7], who showed that for a data set drawn from a zero-mean multivariate Gaussian distribution, the average subspace spanned by the data coincides with the leading principal component. This idea is sketched in Fig. 1. Given, $\{\mathbf{x}_i\}_{i=1}^N \subset \mathbf{R}^D$, the 1-dimensional subspace spanned by each $\mathbf{x}_i$ is a point on the Grassmann manifold (Sec. 2). Hauberg et al. then computed the average of these subspaces on the Grassmannian using an "extrinsic" metric, i.e. the Euclidean distance. Besides the theoretical insight, this formulation gave rise to highly efficient algorithms. Unfortunately, the extrinsic approach is limited to one-dimensional subspaces, and Hauberg et al. resorted to deflation methods to estimate higher dimensional subspaces. We overcome this limitation with an intrinsic metric, extend the theoretical analysis of Hauberg et al., and provide an efficient online algorithm for subspace estimation. We further propose an online non-linear and robust subspace averaging algorithm akin to online KPCA and RPCA respectively. We also present a proof that in the limit, our proposed online robust intrinsic averaging method returns the leading robust principal components.

2 An Online Linear Subspace Learning Algorithm

In this section, we present an efficient online linear subspace learning algorithm for finding the principal components of a data set. We first briefly discuss the geometry of the Riemannian manifold of K-dimensional linear subspaces in $\mathbf{R}^D$. Then, we present an online algorithm using the geometry of these subspaces to get the first K principal components of the D-dimensional data vectors.

2.1 The Geometry of Subspaces

The Grassmann manifold (or the Grassmannian) is defined as the set of all K-dimensional linear subspaces in $\mathbf{R}^D$ and is denoted by $\mathsf{Gr}(K, D)$, where $K \in \mathbf{Z}^+$, $D \in \mathbf{Z}^+$, $D \geq K$. A special case of the Grassmannian is when $K = 1$, i.e., the space of one-dimensional subspaces of $\mathbf{R}^D$, which is known as the *real projective space* (denoted by $\mathbf{R}P^D$). A point $\mathcal{X} \in \mathsf{Gr}(K, D)$ can be specified by a basis, X, i.e., a set of K linearly independent vectors in $\mathbf{R}^D$ (the columns of X) that spans $\mathcal{X}$. We have $\mathcal{X} = \mathrm{Col}(X)$ if X is a basis of $\mathcal{X}$, where $\mathrm{Col}(\cdot)$ is the column span operator. Given $\mathcal{X}, \mathcal{Y} \in \mathsf{Gr}(K, D)$, with their respective orthonormal basis X and Y, the unique geodesic $\Gamma_{\mathcal{X}}^{\mathcal{Y}} : [0, 1] \to \mathsf{Gr}(K, D)$ between $\mathcal{X}$ and $\mathcal{Y}$ is given by:

$$\Gamma_{\mathcal{X}}^{\mathcal{Y}}(t) = \mathrm{Col}\left(X\hat{V}\cos(\Theta t) + \hat{U}\sin(\Theta t)\right) \tag{3}$$

with $\Gamma_{\mathcal{X}}^{\mathcal{Y}}(0) = \mathcal{X}$ and $\Gamma_{\mathcal{X}}^{\mathcal{Y}}(1) = \mathcal{Y}$, where, $\hat{U}\hat{\Sigma}\hat{V}^T = (I - XX^T)Y(X^TY)^{-1}$ is the "thin" singular value decomposition (SVD), (i.e., $\hat{U}$ is $D \times K$ and $\hat{V}$ is $K \times K$ column orthonormal matrix, and $\hat{\Sigma}$ is $K \times K$ diagonal matrix), and $\Theta = \arctan\hat{\Sigma}$. The length of the geodesic constitutes the geodesic distance on $\mathsf{Gr}(K, D)$, $d : \mathsf{Gr}(K, D) \times \mathsf{Gr}(K, D) \to \mathbf{R}^+ \cup \{0\}$ which is as follows: Given $\mathcal{X}, \mathcal{Y}$ with respective orthonormal basis X and Y,

$$d(\mathcal{X}, \mathcal{Y}) \triangleq \sqrt{\sum_{i=1}^{K} \arccos(\sigma_i)^2}, \tag{4}$$

where $\bar{U}\Sigma\bar{V}^T = X^TY$ is the SVD of X^TY, and, $[\sigma_1, \ldots, \sigma_K] = \mathrm{diag}(\Sigma)$. Here $\arccos(\sigma_i)$ is known as the i^{th} *principal angle* between subspace $\mathcal{X}$ and $\mathcal{Y}$. Hence, the geodesic distance is defined as the ℓ_2-norm of principal angles.

2.2 The Intrinsic Grassmann Average (IGA)

We now consider *intrinsic averages*[1] (IGA) on the Grassmannian. To examine the existence and uniqueness of IGA, we need to define an open ball on the Grassmannian.

Definition 1 (Open ball)**.** *An open ball $\mathcal{B}(\mathcal{X}, r) \subset \mathsf{Gr}(K, D)$ of radius $r > 0$ centered at $\mathcal{X} \in \mathsf{Gr}(K, D)$ is defined as*

$$\mathcal{B}(\mathcal{X}, r) = \{\mathcal{Y} \in \mathsf{Gr}(K, D) | d(\mathcal{X}, \mathcal{Y}) < r\}. \tag{5}$$

Let κ be the supremum of the sectional curvature in the ball. Then, we call this ball "regular" [40] if $2r\sqrt{\kappa} < \pi$. Using the results in [41], we know that, for $\mathbf{R}P^D$ with $D \geq 2$, $\kappa = 1$, while for general $\mathsf{Gr}(K, D)$ with $\min(K, D) \geq 2$, $0 \leq \kappa \leq 2$. So, on $\mathsf{Gr}(K, D)$ the radius of a "regular geodesic ball" is $< \pi/2\sqrt{2}$, for $\min(K, D) \geq 2$ and on $\mathbf{R}P^D$, $D \geq 2$, the radius is $< \pi/2$.

Let $\mathcal{X}_1, \ldots, \mathcal{X}_N$ be independent samples on $\mathsf{Gr}(K, D)$ drawn from a distribution $P(\mathcal{X})$, then we can define an *intrinsic average* (FM), [38], [39], $\mathcal{M}^*$ as:

$$\mathcal{M}^* = \underset{\mathcal{M} \in \mathsf{Gr}(K,D)}{\operatorname{argmin}} \sum_{i=1}^{N} d^2\Big(\mathcal{M}, \mathcal{X}_i\Big). \tag{6}$$

As mentioned before, on $\mathsf{Gr}(K, D)$, IGA exists and is unique if the support of $P(\mathcal{X})$ is within a "regular geodesic ball" of radius $< \pi/2\sqrt{2}$ [42]. Note that for $\mathbf{R}P^D$, we can choose this bound to be $\pi/2$. For the rest of the paper, we have assumed that data points on $\mathsf{Gr}(K, D)$ are within a

1. These are also known as Fréchet means [38], [39].

"regular geodesic ball" of radius $< \pi/2\sqrt{2}$ unless otherwise specified. With this assumption, the IGA is unique. *Note that this assumption is only needed for proving the theorem presented below.*

Observe, when $n \to \infty$, FM in Eq. 6 turns out to be Fréchet expectation (FE), [38], [39]:

$$\mathcal{M}^* = \underset{\mathcal{M} \in \mathsf{Gr}(K,D)}{\operatorname{argmin}} \; E\left[d^2\Big(\mathcal{M}, \mathcal{X}\Big)\right]. \tag{7}$$

Note that, here the expectation is with respect to the underlying density from which the samples $\{\mathcal{X}\}$ are drawn.

The IGA may be computed using a Riemannian steepest descent, but this is computationally expensive and requires selecting a suitable step-size [43]. Recently Chakraborty et al. [44] proposed a simple and efficient recursive/inductive Fréchet mean estimator given by:

$$\begin{aligned} \mathcal{M}_1 &= \mathcal{X}_1, \\ \mathcal{M}_{k+1} &= \Gamma_{\mathcal{M}_k}^{\mathcal{X}_{k+1}}\Big(\frac{1}{k+1}\Big), \quad \forall k \geq 1. \end{aligned} \tag{8}$$

This approach only needs a single pass over the data set to estimate the IGA. Consequently, Eq. 8 has linear complexity in the number of observations. Furthermore, it is truly an online algorithm since each iteration only needs one new observation.

Eq. 8 merely performs repeated geodesic interpolation, which is analogous to standard recursive estimators of Euclidean averages: Consider observations $\mathbf{x}_k \in \mathbf{R}^D, k = 1, \ldots, N$. Then the Euclidean average can be computed recursively by moving an appropriate distance away from the k^{th} estimator $\mathbf{m}_k$ towards $\mathbf{x}_{k+1}$ on the straight line joining $\mathbf{x}_{k+1}$ and $\mathbf{m}_k$. The inductive algorithm (8) for computing the IGA works in the same way and is entirely based on traversing geodesics in $\mathsf{Gr}(K, D)$ and without requiring any optimization.

Theorem 1. *(**Weak Consistency [44]**) Let $\mathcal{X}_1, \ldots, \mathcal{X}_N$ be samples on $\mathsf{Gr}(K, D)$ drawn from a distribution $P(\mathcal{X})$. Then, $\mathcal{M}_N$ (Eq. 8) converges to the IGA of $\{\mathcal{X}_i\}_{i=1}^N$ (in terms of intrinsic distance defined in Eq. 4) in probability as $N \to \infty$.*

2.3 Principal Components as Grassmann Averages

Following Hauberg et al. [7] we cast the linear dimensionality reduction problem as an averaging problem on the Grassmannian. We consider an intrinsic Grassmann average (IGA), which allows us to reckon with $K > 1$ dimensional subspaces. We then propose an *online linear subspace learning* and show that for the zero-mean Gaussian data, the expected IGA on $\mathsf{Gr}(K, D)$, i.e., expected K-dimensional linear subspace, coincides with the first K principal components.

Given $\{\mathbf{x}_i\}_{i=1}^N$, the algorithm to compute the IGA that produces the leading K-dimensional principal subspace is sketched in Algorithm 1.

Let $\{\mathcal{X}_i\}$ be the set of K-dimensional subspaces as constructed by IGA in Algorithm 1. Moreover, assume that the maximum principal angle between $\mathcal{X}_i$ and $\mathcal{X}_j$ is $< \pi/2\sqrt{2}$, $\forall i \neq j$. The above condition is assumed to ensure that the IGA exists and is unique on $\mathsf{Gr}(K, D)$. This condition can be ensured if the angle between $\mathbf{x}_l$ and $\mathbf{x}_k$ is smaller than $\pi/2\sqrt{2}, \forall \mathbf{x}_l, \mathbf{x}_k$ belonging to different blocks. For $\mathbf{x}_l, \mathbf{x}_k$ in a same block, the angle must be below $\pi/2$. *Note that, this assumption is needed to prove Theorem 2. In practice, even if IGA is not unique, we find a local minimizer of Eq. 6 [38], which serves as the principal subspace.*

Algorithm 1: The IGA algorithm to compute PCs

Input: $\{\mathbf{x}_i\}_{i=1}^N \subset \mathbf{R}^D$, $K > 0$
Output: $\{\mathbf{v}_1, \ldots, \mathbf{v}_K\} \subset \mathbf{R}^D$

1. Partition the data $\{\mathbf{x}_j\}_{j=1}^N$ into blocks of size $D \times K$;
2. Let the i^{th} block be denoted by, $X_i = [\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iK}]$;
3. Orthogonalize each block and let the orthogonalized block be denoted by X_i ;
4. Let the subspace spanned by each X_i be denoted by $\mathcal{X}_i \in \mathsf{Gr}(K, D)$;
5. Compute IGA, $\mathcal{M}^*$, of $\{\mathcal{X}_i\}$;
6. Return the K columns of an orthogonal basis of $\mathcal{M}^*$; these span the principal K-subspace.

Theorem 2. *(**Relation between IGA and PCA**) Let $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, with a positive definite matrix Σ. The eigen vectors of Σ span the Fréchet expectation as defined in Eq. 7.*

Proof. Let X be the corresponding orthonormal basis of $\mathcal{X}$. Let $X = [\mathbf{x}_1, \cdots, \mathbf{x}_K]$, where each $\mathbf{x}_i$ are samples drawn from $\mathcal{N}(\mathbf{0}, \Sigma)$. Let $M = [M_1, \ldots, M_K]$ be an orthonormal basis of an arbitrary $\mathcal{M} \in \mathsf{Gr}(K, D)$. The squared distance between $\mathcal{X}$ and $\mathcal{M}$ is defined as $d^2(\mathcal{X}, \mathcal{M}) = \sum_{j=1}^K \left(\arccos\left(S_{jj}\right)\right)^2$, where, $\bar{U}S\bar{V}^T = M^T X$ be the SVD, and $S_{jj} \geq 0$. Substituting into the expression for FE from equation 7 we obtain:

$$\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmin}} \, E\left[\sum_{j=1}^K \left(\arccos\left(S_{jj}\right)\right)^2\right]. \tag{9}$$

Now recall that the Taylor expansion of $\arccos(x)$ is given by $\arccos(x) = \frac{\pi}{2} - \sum_{n=0}^{\infty} \frac{(2n)!}{2^{2n}(n!)^2} \frac{x^{2n+1}}{2n+1}$ Substituting this expansion into the equation 9 for FE, we get:

$$\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmax}} \, E\left[\sum_{j=1}^K \pi \; f\left(S_{jj}\right) - \sum_{j=1}^K \left(f\left(S_{jj}\right)\right)^2\right]. \tag{10}$$

where, $f(S_{jj}) = \sum_{n=0}^{\infty} \frac{(2n)!}{2^{2n}(n!)^2} \frac{(S_{jj})^{2n+1}}{2n+1}$. Let $s_n = -\left(\frac{(2n)!}{(2n+1)2^{2n}(n!)^2}\right)^2$ if n is even and $s_n = \pi \, \frac{(2n)!}{(2n+1)2^{2n}(n!)^2}$ if n is odd. Then we can rewrite the objective function to be optimized on the RHS of equation 10 denoted by F as:

$$\begin{aligned} F &= \sum_{n=1}^{\infty} s_n E\left[\sum_{j=1}^K \Big(\left(S_{jj}\right)^2\Big)^{n/2}\right] \\ &= \sum_{n=1}^{\infty} s_n E\left(\operatorname{trace}\left[\Big(M^T X X^T M\Big)^{n/2}\right]\right) \end{aligned} \tag{11}$$

In the last equality above, we use $\sum_{j=1}^K (S_{jj})^2 = \operatorname{trace}\left(M^T X X^T M\right)$. Note that since M is a column orthonormal matrix and hence lies on a Stiefel manifold, $\mathsf{St}(K, D)$, $M^T M = I$. Hence, for all $n > 0$ and for all $\Lambda \in \mathbf{R}^{K \times K}$, $\operatorname{trace}\left[\left(\Lambda\left(M^T M - I\right)\right)^{n/2}\right] = 0$. Since $(M^T X X^T M)$ is a

symmetric positive semi-definite matrix, for $n \geq 2$, maximization of trace $\left[(M^T XX^T M)^{n/2}\right]$ with the constraint $M^T M = I$ is achieved by optimizing the following objective:

$$F = \sum_{n=1}^{\infty} s_n E\left(\text{trace}\left[\left(M^T XX^T M - \Lambda(M^T M - I)\right)^{n/2}\right]\right),$$

for some diagonal matrix $\Lambda \in \mathbf{R}^{K\times K}$. To optimize F, we take its derivative (on the manifold) with respect to M, $\nabla_M F := \left(\frac{\partial F}{\partial M} - M\frac{\partial F}{\partial M}^T M\right) \in T_M \mathsf{St}(K, D)$, and equate it to 0. Now, observe that $\nabla_M F = 0$ if and only if $\frac{\partial F}{\partial M} = 0$. Here,

$$\frac{\partial F}{\partial M} = \sum_{n=1}^{\infty} s_n E\left[n(XX^T M - M\Lambda) \left(M^T XX^T M - \Lambda(M^T M - I)\right)^{n/2-1}\right] \quad (12)$$

In the above equation, the interchange of expectation and derivative uses the well known dominated convergence theorem. Rewriting the above equation 12 using $Z = \left(M^T XX^T M - \Lambda(M^T M - I)\right)$, taking the norm on both sides and then applying the Cauchy-Schwartz's inequality, we get

$$\begin{aligned}
\|\frac{\partial F}{\partial M}\| &= \|\sum_{n=1}^{\infty} s_n E\left[n\left(XX^T M - M\Lambda\right)(Z)^{n/2-1}\right]\| \\
&= \|\sum_{n=1}^{\infty} n s_n E\left[(Z)^{n/2-1}\left(XX^T M - M\Lambda\right)\right]\| \\
&\leq \sum_{n=1}^{\infty} n|s_n| E\left[\|(Z)^{n/2-1}\|^2\right]^{\frac{1}{2}} E\left[\|\left(XX^T M - M\Lambda\right)\|^2\right]^{\frac{1}{2}}
\end{aligned}$$

Now, if M is formed from the top K eigen vectors of $\Sigma = 1/kE[XX^T]$, then, $\|\frac{\partial F}{\partial M}\| = 0$, which implies, $\frac{\partial F}{\partial M} = 0$, which in turn makes $\nabla_M F = 0$. Thus, M is a solution of F in Eq. 11. This completes the proof. ■

Now, using Theorem 1 and Theorem 2, we replace the line 5 of the IGA Algorithm 1 by Eq. 8 to get an *online subspace learning* algorithm that we call, *Recursive IGA (RIGA)*, to compute leading K principal components, $K \geq 1$.

The key advantages of our proposed RIGA algorithm to compute PCs are as follows:

1) In contrast to work in [7], "IGA" will return the first K PCs, $K \geq 1$.
2) Using the recursive computation of the FM in IGA leads to RIGA, an online PC computation algorithm. Moreover, Theorem 1 ensures the convergence of "RIGA" to "IGA". Hence, our proposed RIGA is an online PCA algorithm.
3) Unlike previous online PCA algorithms, RIGA is parameter free.

3 A Kernel Extension

In this section, we extend RIGA to perform the principal component analysis in a Reproducing Kernel Hilbert Space (RKHS). We extend RIGA to obtain an efficient nonlinear subspace estimator in RKHS akin to kernel PCA [9] and dub our algorithm *Kernel RIGA (KRIGA)*. The key issue to be addressed in KRIGA is that, in order to perform IGA in the RKHS, we will need to cope with an infinite-dimensional Grassmannian. Fortunately, we observe that the distance between two subspaces in RKHS is same as the distances between span of the coefficient matrices with respect to an orthogonal basis. Hence, instead of performing IGA on the subspaces in RKHS, we will perform IGA of the span of the coefficients, which are finite dimensional. The IGA is then computable using the kernel trick. *A key advantage of KRIGA is that it does not require an eigen decomposition of the Gram matrix*. Furthermore, we extend this formulation to propose an *online* KRIGA algorithm by approximating the kernel function.

3.1 Deriving the Kernel Recursive Intrinsic Grassmann Average (KRIGA)

Let $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i \in \mathbf{R}^D$, for all i. We seek K principal components, $K \leq D$. Let $\mathcal{K}(\cdot,\cdot)$ be the kernel associated with RKHS H and let $\phi(X) = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_N)]$, where $\phi : X \to H$. Let $\widetilde{\phi}(X) = [\widetilde{\phi}(\mathbf{x}_1), \ldots, \widetilde{\phi}(\mathbf{x}_N)]$ be the orthogonalization of $\phi(X)$. Since each $\phi(\mathbf{x}_i) \in \text{Col}(\widetilde{\phi}(X))$ we have $\phi(\mathbf{x}_i) = \widetilde{\phi}(X)\langle\widetilde{\phi}(X), \phi(\mathbf{x}_i)\rangle_H$, where,

$$\begin{aligned}
&\langle\widetilde{\phi}(X), \phi(\mathbf{x}_i)\rangle_H \\
&\quad = [\langle\widetilde{\phi}(\mathbf{x}_1), \phi(\mathbf{x}_i)\rangle_H, \ldots, \langle\widetilde{\phi}(\mathbf{x}_N), \phi(\mathbf{x}_i)\rangle_H]^t.
\end{aligned} \quad (13)$$

Here, $\langle\cdot,\cdot\rangle_H$ is the inner product in the RKHS.

Let $Y_l = \left[\phi(\mathbf{x}_{K(l-1)+1}), \ldots, \phi(\mathbf{x}_{Kl})\right] = \widetilde{\phi}(X)\langle\widetilde{\phi}(X), Y_l\rangle_H$, where $C_l = \langle\widetilde{\phi}(X), Y_l\rangle_H$. Here, $(C_l)_{ij} = \langle\widetilde{\phi}(\mathbf{x}_i), \phi(\mathbf{x}_{K(l-1)+j})\rangle_H$. Note that, C_l is a matrix of dimension $N \times K$.

We observe that $d(\text{Col}(Y_i), \text{Col}(Y_j)) = d(\text{Col}(C_i), \text{Col}(C_j))$ as proved in the following lemma.

Lemma 1. *Using the above notations,* $d(Col(Y_i), Col(Y_j)) = d(Col(C_i), Col(C_j))$.

Proof. $Y_i = \tilde{\phi}(X)\, C_i$ and $Y_j = \tilde{\phi}(X)\, C_j$. Now, using (4), we can see that, $d(\text{Col}(Y_i), \text{Col}(Y_j)) = d(\text{Col}(C_i), \text{Col}(C_j))$ *iff* the SVD of $Y_i^T Y_j$ is same as that of $C_i^T C_j$. Now, as $\tilde{\phi}(X)$ is column orthogonal, hence the result follows. ■

So, instead of IGA on $\{\mathcal{Y}_l\}$, we can perform IGA on $\{\mathcal{C}_l\}$, where $\mathcal{Y}_l = \text{Col}(Y_l)$ and $\mathcal{C}_l = \text{Col}(C_l)$, $\forall l$.

The orthogonalization $\widetilde{\phi}(X)$ is achieved using the Gram-Schimdt orthogonalization process as follows:

$$\widetilde{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \sum_{j=1}^{i-1}\langle\phi(\mathbf{x}_i), \widetilde{\phi}(\mathbf{x}_j)\rangle_H \widetilde{\phi}(\mathbf{x}_j) \quad (14)$$

$$\widetilde{\phi}(\mathbf{x}_i) = \widetilde{\phi}(\mathbf{x}_i)/\|\widetilde{\phi}(\mathbf{x}_i)\|. \quad (15)$$

The elements of C_l, i.e., $\langle\widetilde{\phi}(\mathbf{x}_i), \phi(\mathbf{x}_{K(l-1)+j})\rangle_H$ can be computed using the kernel $\mathcal{K}(\cdot,\cdot)$ as given in the Lemma 2 below.

Let the basis matrix of the IGA of $\{\mathcal{C}_l\}$ be M. Then the basis matrix of the IGA of $\{\mathcal{Y}_l\}$ is denoted by $\widetilde{U} = \widetilde{\phi}(X)M$. The columns of $\widetilde{U}$ will give the PCs in RKHS. Note that this tallies with the *Representer theorem* [45], which tells us that the PC in RKHS is a linear combination of the mapped data vectors, $\phi(X)$.

The following corollary holds by virtue of Theorem 2.

Corollary 1. *The expected IGA of $\{\mathcal{Y}_l\}$ is the same as the PC of $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ in the Hilbert space H.*

The projection of $\mathbf{x}_i$ onto $\widetilde{U}$ is given by $\mathbf{Proj}(\mathbf{x}_i) = \langle \phi(\mathbf{x}_i), \widetilde{\phi}(X)\rangle_H M$, where,

$$\begin{aligned}&\langle \phi(\mathbf{x}_i), \widetilde{\phi}(X)\rangle_H \\ &\quad = [\langle \phi(\mathbf{x}_i), \widetilde{\phi}(\mathbf{x}_1)\rangle_H, \ldots, \langle \phi(\mathbf{x}_i), \widetilde{\phi}(\mathbf{x}_N)\rangle_H].\end{aligned} \tag{16}$$

The following Lemma, gives the analytic form of $\langle \widetilde{\phi}(\mathbf{x}_m), \phi(\mathbf{x}_i)\rangle_H$.

Lemma 2. $\langle \widetilde{\phi}(\mathbf{x}_m), \phi(\mathbf{x}_i)\rangle_H =$

$$\begin{cases} \frac{\mathcal{K}(\mathbf{x}_m, \mathbf{x}_i) - \sum_{j=1}^{m-1} \langle \phi(\mathbf{x}_m), \widetilde{\phi}(\mathbf{x}_j)\rangle_H \langle \widetilde{\phi}(\mathbf{x}_j), \phi(\mathbf{x}_i)\rangle_H}{\sqrt{\mathcal{K}(\mathbf{x}_m, \mathbf{x}_m) - \sum_{j=1}^{m-1} \langle \phi(\mathbf{x}_m), \widetilde{\phi}(\mathbf{x}_j)\rangle_H}}, & i \geq m \\ 0, & \text{otherwise.} \end{cases}$$

Proof. $\widetilde{\phi}(\mathbf{x}_m) = \phi(\mathbf{x}_m) - \sum_{j=1}^{m-1} \langle \phi(\mathbf{x}_m), \widetilde{\phi}(\mathbf{x}_j)\rangle_H \widetilde{\phi}(\mathbf{x}_j)$ and $\widetilde{\phi}(\mathbf{x}_m) = \widetilde{\phi}(\mathbf{x}_m)/\|\widetilde{\phi}(\mathbf{x}_m)\|$ where $\|\widetilde{\phi}(\mathbf{x}_m)\| = \sqrt{\mathcal{K}(\mathbf{x}_m, \mathbf{x}_m) - \sum_{j=1}^{m-1} \langle \phi(\mathbf{x}_m), \widetilde{\phi}(\mathbf{x}_j)\rangle_H}$. Now, since $\{\widetilde{\phi}(\mathbf{x}_i)\}_{i=1}^N$ serves as a basis to represent each $\phi(\mathbf{x}_i)$, clearly, $\langle \widetilde{\phi}(\mathbf{x}_m), \phi(\mathbf{x}_i)\rangle_H = 0$ when, $i < m$, $m = 1, \cdots, N$.

So, consider $m \in \{1, \cdots, N\}$, $i \geq m$, then, $\langle \widetilde{\phi}(\mathbf{x}_m), \phi(\mathbf{x}_i)\rangle_H = \frac{1}{\|\widetilde{\phi}(\mathbf{x}_m)\|} (\mathcal{K}(\mathbf{x}_m, \mathbf{x}_i) - \sum_{j=1}^{m-1} \langle \phi(\mathbf{x}_m), \widetilde{\phi}(\mathbf{x}_j)\rangle_H \langle \widetilde{\phi}(\mathbf{x}_j), \phi(\mathbf{x}_i)\rangle_H)$. ■

Note that, thus far, we have implicitly assumed that $\sum_i \phi(\mathbf{x}_i) = 0$, i.e., the mapped data in RKHS is centered. For non-centered data, we have to first center the data. Given the non-centered data, $\{\phi(\mathbf{x}_i)\}_{i=1}^N$, let the centered data be denoted by $\{\bar{\phi}(\mathbf{x}_i)\}_{i=1}^N$, where $\bar{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{N}\sum_{j=1}^N \phi(\mathbf{x}_j)$. Then, using Lemma 2, the coefficient matrix C_l is computed using

$$\begin{aligned}\langle \widetilde{\phi}(\mathbf{x}_m), \bar{\phi}(\mathbf{x}_i)\rangle_H &= \langle \widetilde{\phi}(\mathbf{x}_m), \phi(\mathbf{x}_i)\rangle_H - \\ &\quad \frac{1}{N}\sum_{j=1}^N \langle \widetilde{\phi}(\mathbf{x}_m), \phi(\mathbf{x}_j)\rangle_H.\end{aligned} \tag{17}$$

Thus, the coefficient matrices $\{C_l\}$ can be computed using only the Gram matrix $\mathcal{K}$ as can be seen from Eq. 17. In terms of computational complexity, KRIGA takes $\mathcal{O}(N^3 - N^2)$ computations while KPCA takes $\mathcal{O}(N^3)$ computations.

Now, observe that the above KRIGA algorithm (analog to KPCA) is not online because of the following reasons: **(i)** centering step of the data; **(ii)** choice of basis, i.e., $\{\phi(\mathbf{x}_i)\}_{i=1}^N$. Consistent with the online KPCA algorithm, we assume data to be centered. Then, in order to make the above algorithm online, we need to find a predefined basis in RKHS. We will use the idea proposed by Rahimi et al. [33], to approximate the shift-invariant kernel $\mathcal{K}$. They observed that infinite kernel expansions can be well-approximated using randomly drawn features. For shift-invariant $\mathcal{K}$, this relates to Bochner's lemma [46] as stated below.

Lemma 3. *$\mathcal{K}$ is positive definite* iff *$\mathcal{K}$ is the Fourier transform of a non-negative measure, $\mu(\mathbf{w})$.*

This in turn implies the existence of a probability density $p(\mathbf{w}) := \mu(\mathbf{w})/C$, where C is the normalizing constant. Hence,

$$\begin{aligned}\mathcal{K}(\mathbf{x}, \mathbf{y}) &= C \int \exp\left(-j\mathbf{w}^t(\mathbf{x} - \mathbf{y})\right) p(\mathbf{w})\, d\mathbf{w} \\ &= C E_{\mathbf{w}}\left[\cos\left(\mathbf{w}^t(\mathbf{x} - \mathbf{y})\right)\right].\end{aligned}$$

The above expectation can be approximated using Monte Carlo methods, more specifically, we will draw M i.i.d. $\mathbf{R}^D$ vectors from $p(\mathbf{w})$ and form matrix W of size $M \times D$. Then, we can approximate $\mathcal{K}(\mathbf{x}, \mathbf{y})$ by, $\mathcal{K}(\mathbf{x}, \mathbf{y}) = \psi(W\mathbf{x})^t \psi(W\mathbf{y})$, where $\psi(W\mathbf{x}) = \sqrt{C/M}(\cos(W\mathbf{x}), \sin(W\mathbf{x}))^t$. Now, depending on the choice of $\mathcal{K}$, $p(\mathbf{w})$ will change. For example, for the Gaussian RBF kernel, i.e., $\mathcal{K}(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}\right)$. $\mathbf{w}$ should be sampled from $N\left(\mathbf{0}, \mathrm{diag}\left(\sigma^2\right)^{-1}\right)$.

Rahimi et al. [33] provided a bound on the error in the approximation of the kernel. Now, because of this approximation, in our KRIGA algorithm, we can replace ϕ by ψ. As ψ is finite dimensional, we will choose the canonical basis in $\mathbf{R}^{2M}$, i.e., replacing ϕ in the above derivation by $\{\mathbf{e}_i\}_{i=1}^{2M}$. This gives us an online KRIGA algorithm using the recursive IGA.

4 A Robust Online Linear Subspace Learning Algorithm

In this section, we will propose an online robust PCA algorithm using intrinsic Grassmann averages. Let $\{\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_N\} \subset \mathsf{Gr}(K, D), K < D$ be inside a regular geodesic ball of radius $< \pi/2\sqrt{2}$ s.t., the Fréchet Median (FMe) [47] exists and is unique. Let $X_1, X_2, \cdots, X_N$ be the corresponding orthonormal basis, i.e., X_i spans $\mathcal{X}_i$, for all i. The FMe can be computed via the following minimization:

$$\mathcal{M}^* = \arg\min_{\mathcal{M}} \sum_{i=1}^N d(\mathcal{X}_i, \mathcal{M}). \tag{18}$$

With a slight abuse of notation, we use the notation $\mathcal{M}^*$ (M) to denote both the FM and the FMe (and their orthonormal basis). The FMe is robust as was shown by Fletcher et al. [47], hence we call our estimator *Robust IGA* (RoIGA). In the following theorem, we will prove that RoIGA leads to the robust PCA in the limit as the number of the data samples goes to infinity. An algorithm to compute RoIGA is obtained by simply replacing Step 5 of Algorithm 1 by computation of RoIGA via minimization of Eq. 18 instead of Eq. 6. This minimization can be achieved using the Riemannian steepest descent, but instead, here we use the stochastic gradient descent of batch size 5 to compute RoIGA. As at each iteration, we need to store only 5 samples, the algorithm is online. The update step for each iteration of the online algorithm to compute RoIGA (we refer to our online RoIGA algorithm as Recursive RoIGA (RRIGA)) is as follows:

$$\begin{aligned}\mathcal{M}_1 &= \mathcal{X}_1, \\ \mathcal{M}_{k+1} &= \mathrm{Exp}_{\mathcal{M}_k}\left(\frac{\mathrm{Exp}^{-1}_{\mathcal{M}_k}(\mathcal{X}_{k+1})}{(k+1)d(\mathcal{M}_k, \mathcal{X}_{k+1})}\right).\end{aligned} \tag{19}$$

where, $k \geq 1$, Exp and Exp^{-1} are Riemannian Exponential and inverse Exponential functions as defined below.

Definition 2 (Exponential map)**.** *Let $\mathcal{X} \in \mathcal{Gr}(K,D)$. Let $\mathcal{B}(\mathbf{0},r) \subset T_{\mathcal{X}}\mathcal{Gr}(K,D)$ be an open ball centered at the origin in the tangent space at $\mathcal{X}$, where r is the injectivity radius [38] of $\mathcal{Gr}(K,D)$. Then, the Riemannian Exponential map is a diffeomorphism $Exp_{\mathcal{X}} : \mathcal{B}(\mathbf{0},r) \rightarrow \mathcal{Gr}(K,D)$.*

Definition 3 (Inverse Exponential map)**.** *Since, inside $\mathcal{B}(\mathbf{0},r)$, Exp is a diffeomorphism, hence the inverse Exponential map is defined and is a map $Exp_{\mathcal{X}}^{-1} : \mathcal{U} \rightarrow \mathcal{B}(\mathbf{0},r)$, where $\mathcal{U} = Exp_{\mathcal{X}}(\mathcal{B}(\mathbf{0},r)) := \{Exp_{\mathcal{X}}(U) \,|U \in \mathcal{B}(\mathbf{0},r)\}$.*

We refer the readers to [48] for the consistency proof of the estimator. Notice that, Eq. 19 can be rewritten as,

$$\begin{aligned} \mathcal{M}_1 &= \mathcal{X}_1, \\ \mathcal{M}_{k+1} &= \Gamma_{\mathcal{M}_k}^{\mathcal{X}_{k+1}}(\gamma_k). \end{aligned} \tag{20}$$

Where, $\gamma_k = \frac{1}{(k+1)d(\mathcal{M}_k,\mathcal{X}_{k+1})}$. This is because, $\Gamma_{\mathcal{X}}^{\mathcal{Y}}(t)$ can be written as $\mathrm{Exp}_{\mathcal{X}}\left(t\mathrm{Exp}_{\mathcal{X}}^{-1}(\mathcal{Y})\right)$, where, $\mathrm{Exp}_{\mathcal{X}}^{-1}(\mathcal{Y})$ is the velocity vector when moving from $\mathcal{X}$ to $\mathcal{Y}$ and $\mathrm{Exp}_{\mathcal{X}}(U)$ is a point on $\mathsf{Gr}(K,D)$ which can be reached by going from $\mathcal{X}$ along the shortest geodesic given by the velocity vector U. Let M_k and X_{k+1} be orthonormal basis of $\mathcal{M}_k$ and $\mathcal{X}_{k+1}$ respectively. Then, the shortest geodesic can be expressed as:

$$\Gamma_{\mathcal{M}_k}^{\mathcal{X}_{k+1}}(t) = \mathrm{Col}\left(M_k V\cos(tS) + U\sin(tS)\right), \tag{21}$$

where, $U\Sigma V^T = X_{k+1}\left(M_k^T X_{k+1}\right)^{-1} - M_k$ is the thin singular value decomposition and $S = \mathrm{atan}(\Sigma)$.

Theorem 3. ***(Robustness of RoIGA)*** *Assuming the above hypotheses and notations, as $N \rightarrow \infty$, the columns of M are robust to outliers, where M is the orthonormal basis of $\mathcal{M}^*$ as defined in Eq. 18.*

Proof. Let, $X_i = [\mathbf{x}_{i1}, \cdots, \mathbf{x}_{iK}]$ and $\mathbf{x}_{ij}$ be i.i.d. samples drawn from $N(\mathbf{0},\Sigma)$, with a positive definite matrix Σ. Let, $M = [M_1, \cdots, M_K]$ be an orthonormal basis of $\mathcal{M}$. Recall that the distance between $\mathcal{X}_i$ and $\mathcal{M}$ is defined as $d(\mathcal{X}_i,\mathcal{M}) = \sqrt{\sum_{j=1}^{K}(\arccos((S_i)_{jj}))^2}$, where $\bar{U}_i S_i V_i^T = M^T X_i$ is the SVD, and $(S_i)_{jj} \geq 0$.

Observe that Eq. 18 is equivalent to maximizing $\sum_{i=1}^{N}\sqrt{\sum_{j=1}^{K}(\arcsin((S_i)_{jj}))^2}$. Here, we will only focus on the first term of the Taylor expansion (which bounds the other terms) of $\sum_{i=1}^{N}\sqrt{\sum_{j=1}^{K}(\arcsin((S_i)_{jj}))^2}$, i.e., $\sqrt{\sum_{n=1}^{\infty} t_n \sum_{j=1}^{K}((S_i)_{jj})^{2n}}$, where $\{t_n\}$ denote the coefficients in the Taylor expansion. The first term in the expansion is $\widetilde{V} = \sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}$. Note that, $\sum_{j=1}^{K}((S_i)_{jj})^2 \sim \Gamma\left(\frac{1}{2}\sum_{j=1}^{K}\sigma_{M_j}^2, 2\right)$, as, $\sum_{j=1}^{K}((S_i)_{jj})^2 = \mathrm{trace}(M^T X_i X_i^T M)$, and $M^T X_i \sim \mathcal{N}(\mathbf{0},\Sigma_M)$. Hence, $\widetilde{V}$ follows $N_g\left(\frac{1}{2}\sum_{j=1}^{K}\sigma_{M_j}^2, \sum_{j=1}^{K}\sigma_{M_j}^2\right)$, where N_g is the Nakagami distribution [49]. Now, as $N \rightarrow \infty$, the RHS of Eq. 18 becomes $E\left[\sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}\right]$. $E\left[\sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}\right] = \sqrt{2}\Gamma(\sum_{j=1}^{K}\sigma_{\bar{U}_{ij}M_j}^2 + 0.5)/\Gamma(\sum_{j=1}^{K}\sigma_{\bar{U}_{ij}M_j}^2)$, where Γ is the well-known gamma function. Thus, $E\left[\sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}\right] = \rho(\mathfrak{m}) \triangleq \sqrt{2}\Gamma(\mathfrak{m}+0.5)/\Gamma(\mathfrak{m})$, where $\mathfrak{m} = \sum_{j=1}^{K}\sigma_{M_j}^2$. Hence, the *influence function* [50] of ρ is proportional to $\psi(\mathfrak{m}) \triangleq \frac{\partial E[\sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}]}{\partial \mathfrak{m}}$ and if we can show that $\lim_{\mathfrak{m}\rightarrow\infty}\psi(\mathfrak{m}) = 0$, then we can claim that our objective function in Eq. 18 is robust [50].

This can be justified by noting that in the presence of outliers, i.e., when $\mathfrak{m} \rightarrow \infty$ as variance becomes larger, we want no change in the objective function, i.e., the gradient with respect to $\mathfrak{m}$ should be zero. This is due to the fact that, the other terms in the Taylor expansion are upper bounded by $\sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}$, hence, as $\frac{\partial E[\sqrt{\sum_{j=1}^{K}((S_i)_{jj})^2}]}{\partial \mathfrak{m}}$ goes to 0, so do the gradients of the other terms.

Now, $\psi(\mathfrak{m}) = \Gamma(\mathfrak{m})\Gamma(\mathfrak{m}+0.5)\frac{\phi(\mathfrak{m}+0.5)-\phi(\mathfrak{m})}{\Gamma(\mathfrak{m})^2}$, where ϕ is the polygamma function [51] of order 0. After some simple calculations, we get,

$$\begin{aligned} \lim_{\mathfrak{m}\rightarrow\infty}(\phi(\mathfrak{m}+0.5) - \phi(\mathfrak{m})) &= \lim_{\mathfrak{m}\rightarrow\infty}\log(1+1/(2\mathfrak{m})) \\ &+ \lim_{\mathfrak{m}\rightarrow\infty}\sum_{k=1}^{\infty}\left(B_k\left(\frac{1}{k\mathfrak{m}^k} - \frac{1}{k(\mathfrak{m}+0.5)^k}\right)\right) \\ &= \lim_{\mathfrak{m}\rightarrow\infty}\log(1+1/(2\mathfrak{m})) + 0 = 0. \end{aligned}$$

Here, $\{B_k\}$ are the Bernoulli numbers of the second kind [52]. Hence, $\lim_{\mathfrak{m}\rightarrow\infty}\psi(\mathfrak{m}) = 0$. ■

We would like to point out that the outlier corrupted data can be modeled using a mixture of independent random variables, Y_1, Y_2, where $Y_1 \sim N(\mathbf{0},\Sigma_1)$ (to model non-outlier data samples) and $Y_2 \sim N(\boldsymbol{\mu},\Sigma_2)$ (to model outliers), i.e., $(\forall i)$, $\mathbf{x}_i = w_1 Y_1 + (1-w_1)Y_2$, $w_1 > 0$ is generally large, so that the probability of drawing outliers is low. Then, as the mixture components are independent, $(\forall i)$, $\mathbf{x}_i \sim N((1-w_1)\boldsymbol{\mu}, w_1^2\Sigma_1 + (1-w_1)^2\Sigma_2)$. A basic assumption in any online PCA algorithm is that data is centered. So, in case the data are not centered (similar to the model of $\mathbf{x}_i$), the first step of PCA would be to center the data. But then the algorithm cannot be made online, hence our above assumption that $\mathbf{x}_i \sim N(\mathbf{0},\Sigma)$ is a common assumption in an online scenario. But, in a general case, after centering the data as the first step of PCA, the above theorem is valid.

5 Experimental Results

We evaluate the performance of the proposed recursive estimators on both real and synthetic data. Our overall findings are that the RIGA estimator is more accurate than other online linear subspace estimators since it is *parameter free*. The Kernel RIGA (KRIGA) is found to yield results that are almost identical to Kernel PCA (KPCA) but at a significant reduction in run time. Below we consider RIGA and KRIGA separately.

5.1 Online Linear Subspace Estimation

Baselines: We compare with Oja's rule and and the online version of EM PCA (Sec. 1.1). For Oja's rule we follow common guidelines and consider step-sizes $\gamma_t = {}^{\alpha}/_{D\sqrt{t}}$ with α-values between 0.005 and 0.2. For EM PCA we follow the recommendations in Cappé [13] and use step-sizes $\gamma_t = {}^{1}/_{t^{\alpha}}$ with α-values between 0.6 and 0.9 along with Polyak-Ruppert averaging.

(Synthetic) Gaussian Data: Theorem 2 state that the RIGA estimates coincide in expectation with the leading principal subspace when the data is drawn from a zero-mean Gaussian distribution. We empirically verify this for

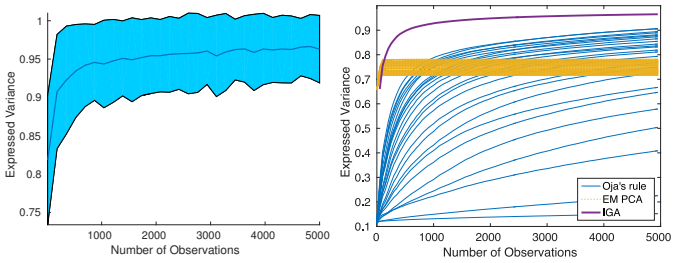


Fig. 2. Expressed variance as a function of number of observations. *Left:* The mean and one standard deviation of the RIGA estimator computed over 150 trials. In each trial data are generated in $\mathbf{R}^{50}$ and we estimate a $K = 2$ dimensional subspace. *Right:* The performance of different estimators for varying step-sizes. Here data are generated in $\mathbf{R}^{250}$ and we estimate a $K = 20$ dimensional subspace.

an increasing number of observations drawn from randomly generated zero-mean and $0.5I$ variance Gaussian. We measure the *expressed variance* which is the ratio of the variance captured by the estimated subspace to the variance captured by the true principal subspace:

$$\text{Expressed Variance} = \sum_{k=1}^{K} \frac{\sum_{n=1}^{N} \mathbf{x}_n^T \mathbf{v}_k^{(\text{est})}}{\sum_{n=1}^{N} \mathbf{x}_n^T \mathbf{v}_k^{(\text{true})}} \in [0, 1]. \quad (22)$$

An expressed variance of 1 implies that the estimated subspace captures as much variance as the principal subspace. The right panel of Fig. 2 shows the mean ($\pm$ one standard deviation) expressed variance of RIGA over 150 trials. It is evident that for the Gaussian data, the RIGA estimator does indeed converge to the true principal subspace.

A key aspect of any online estimator is that it should be stable and converge fast to a good estimate. Here, we compare RIGA to the above-mentioned baselines. Both Oja's rule and EM PCA require a step-size to be specified, so we consider a larger selection of such step-sizes. The left panel of Fig. 2 shows the expressed variance as a function of number of observations for different estimators and step-sizes. In Fig. 3, we have comparative performance analysis of EM PCA, GROUSE, Oja's rule and RIGA. EM PCA was found to be quite stable with respect to the choice of step-size, though it does not seem to converge to a good estimate. Oja's rule, on the other hand, seems to converge to a good estimate, but its practical performance is critically dependent on the step-size (as evident from Fig. 2). GROUSE is seen to oscillate for small data size however, with a large number of samples, it yields a good estimate. On the other hand, RIGA is parameter free and is observed to have good convergence properties. This behavior of RIGA is consistent for $D \geq 100$ and $K \geq 10$ as observed empirically and depicted in Fig. 4.

In the right panel of Fig. 5, we perform a stability analysis of GROUSE and RIGA. Here, for a fixed value of N, we generate a data matrix and perform 200 independent runs on the data matrix and report the mean ($\pm$ one standard deviation) expressed variance. On the left and middle panels, we show the performance of GROUSE with varying D and K for learning rates 0.0001 and 0.01 respectively. We can see that with a larger learning rate, GROUSE can achieve better expressed variance but with less stability. As can be seen from the figure, RIGA is very stable in comparison to GROUSE.

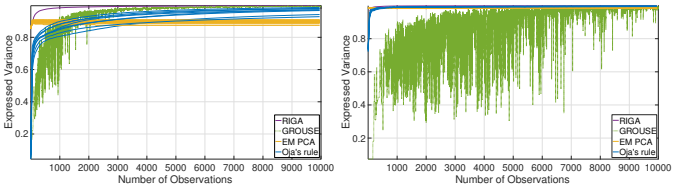


Fig. 3. Expressed variance as a function of number of observations. The performance of different estimators. *Left:* Data are generated in $\mathbf{R}^{250}$ and we set $K = 20$. *Right:* Data are generated in $\mathbf{R}^{100}$ and we set $K = 10$. We observe that our estimator is better than its competitors for other values of $D > 100$ and $K \geq 10$.

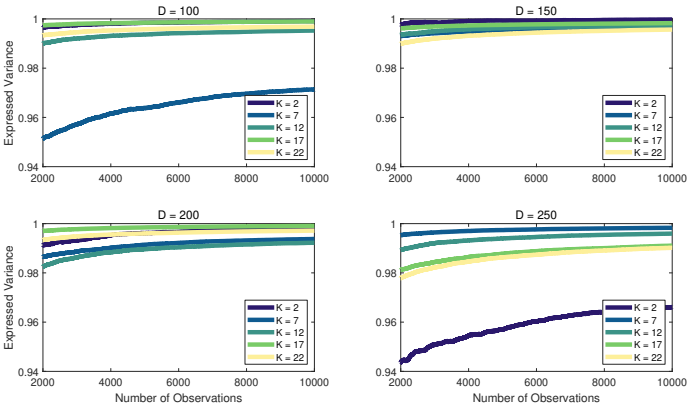


Fig. 4. Performance of RIGA with varying D and K. We can see that for moderately large D and $K \geq 10$, the performance of RIGA is very good.

In the rest of the paper, we will use *average reconstruction error (ARE)* [9] to measure the "goodness" of the estimated subspace, which is defined as follows:

$$\text{Average Reconstruction Error} = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \widetilde{\mathbf{x}}_n\|^2, \quad (23)$$

where, $\widetilde{\mathbf{x}}$ is the reconstructed sample using the estimated principal subspace spanned by $\{\mathbf{v}_k\}_{k=1}^{K}$.

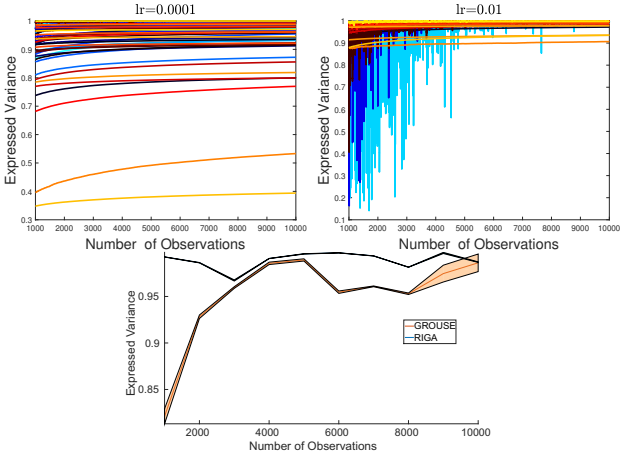


Fig. 5. *Left and Middle:* Performance of GROUSE with varying D and K and learning rates of 0.0001 and 0.01 respectively. D ranges between 50 and 500 in increments of 50 and K ranges between 2 and 22 in increments of 5. The plots are color coded from hot to cold, with warmest being $D = 50, K = 2$ and coolest being $D = 500, K = 22$. We can see that though for relatively larger learning rate the performance of GROUSE is better, it is not stable. *Right:* Stability analysis comparison of GROUSE and RIGA (for a fixed N, we randomly generate a data matrix, X, from a Gaussian distribution on $\mathbf{R}^{250}$. We estimate $K = 20$ dimensional subspace and report the mean and standard deviation over 200 runs on X with learning rate 0.0001. This plot is a close-up look at the leftmost plot with $D = 250$ and $K = 20$.)

Human Body Shape: Online algorithms are generally well-suited for solving large-scale problems as they, by construction, should have linear time-complexity in the number of observations. As an example we consider a large collection of three-dimensional scans of human body shape [53]. This dataset contains $N = 21862$ meshes which each consist of 6890 vertices in $\mathbf{R}^3$. Each mesh is, thus, viewed as a $D = 6890 \times 3 = 20670$ vector. We estimate a $K = 10$ dimensional principal subspace using Oja's rule, EM PCA and RIGA respectively. The average reconstruction error over all meshes are 16.8 mm for Oja's rule, 1.9 mm for EM PCA, and 1.0 mm for RIGA. *Note that both Oja's rule and EM PCA explicitly minimize the reconstruction error, while RIGA does not but yet outperforms the baseline methods.* We speculate that this is due to RIGA's excellent convergence properties and it being a parameter free algorithm is not bogged down by the hard problem of step-size tuning confronted in the baseline algorithms used here.

Santa Claus Conquers the Martians: We now consider an even larger scale experiment and consider all frames of the motion picture *Santa Claus Conquers the Martians (1964)*[2]. This consist of $N = 145,550$ RGB frames of size 320×240, corresponding to an image dimension of $D = 230,400$. We estimate a $K = 10$ dimensional subspace using Oja's rule, EM PCA and RIGA respectively. Again, we measure the accuracy of the different estimators via the reconstruction error. Pixel intensities are scaled to be between 0 and 1. Oja's rule gives an average reconstruction error of 0.054, EM PCA gives 0.025, while RIGA gives 0.023. Here RIGA and EM PCA gives roughly equally good results, with a slight advantage to RIGA. Oja's rule does not fare as well. As with the shape data, it is interesting to note that RIGA outperforms the other baseline methods on the error measure that they optimize even though RIGA optimizes a different measure.

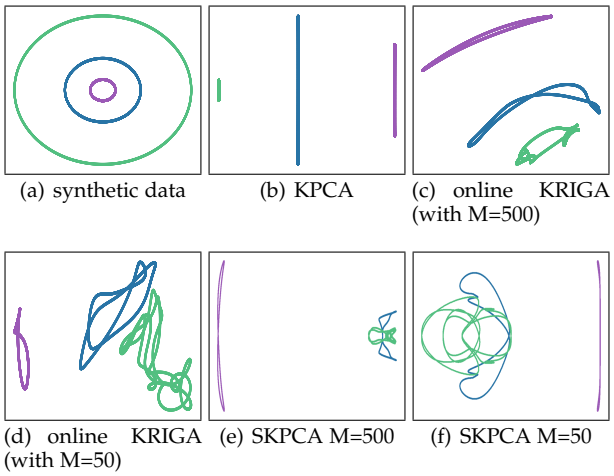


Fig. 6. Results from KRIGA and SKPCA on synthetic data.

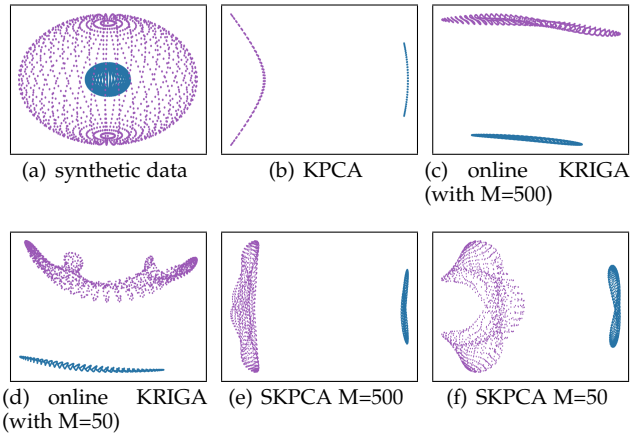


Fig. 7. Results from KRIGA and SKPCA on 3D synthetic data.

5.2 Nonlinear Subspace Estimation

We now analyze comparative performance of the proposed online KRIGA with KPCA as the baseline. In our experiments, we use a Gaussian kernel with $\sigma = 1$. The performance is compared in terms of the time required and average reconstruction error (ARE). In Fig. 6, we present a synthetic experiment, where the data generated is in the form of three concentric circles. We can see that both KPCA and online KRIGA yield similar cluster separation. As expected, we can see that with very few dimensions for approximation (i.e., with small M), the performance of online KRIGA is poor. Similar observation can be made from Fig. 7, where with $M = 500$, we get almost as good result as KPCA. We also compared KRIGA's performance with SKPCA [35] and though in Fig. 6, the results using SKPCA are worse than ours, the results in Fig. 7 are however comparable.

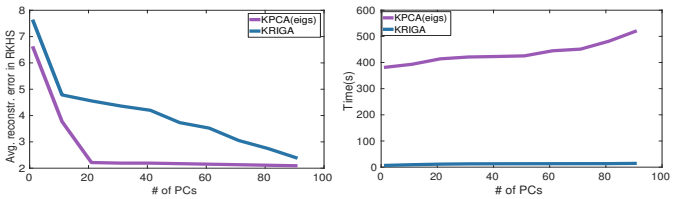
Now, we assess the performance of KRIGA and KPCA in terms of ARE and computation time, based on randomly generated synthetic data. Here, we compare our online KRIGA with KPCA. In order to make a fair comparison, we have used the MATLAB 'eigs' function of KPCA which is significantly faster than KPCA. The results for ARE and computation time are shown in Fig. 8. We can see that our online KRIGA is faster than KPCA with 'eigs' without sacrificing much ARE. For this experiment, we chose $K = 5$. Though, the ARE of KPCA is better than that of KRIGA, we can see from Fig. 8 that with increasing number of PCs, performance of KRIGA is similar to KPCA.

Finally, we test the KPCA and the online KRIGA algorithms on the entire movie, *Santa Claus Conquers the Martians (1964)* samples at 10 FPS, and present the time comparison in Fig. 8. We observe a cubic time growth for KPCA while for the online KRIGA, the time is almost a constant. This demonstrates the scalability of our proposed method.

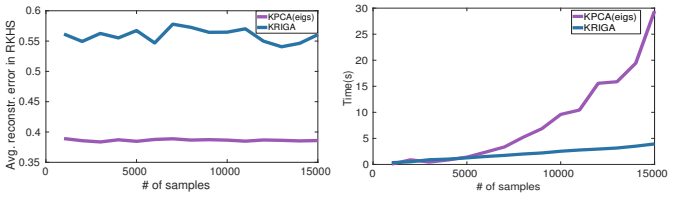
5.3 Robust Subspace Estimation

We now present a comparative experimental evaluation of robust extension (RRIGA). Here we use several baseline methods and measure the performance using the *reconstruction error (RE)*. We use UCSD anomaly detection database [54] and the Extended YaleB database [55]. Before presenting these experiments, we present a synthetic experiment to show comparisons of several robust and non-robust algorithms in a simulated setting. In this simulated setting, we demonstrate the necessity of robust PCA algorithm in the case of an increased amount of noise present in the data.

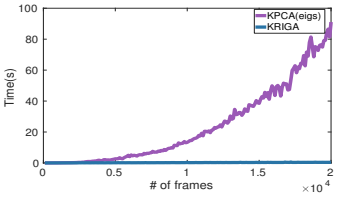
2. https://archive.org/details/SantaClausConquerstheMartians1964



(a) *(Left)* ARE and *(Right)* time with number of PCs



(b) *(Left)* ARE and *(Right)* time with number of samples



(c) Time with number of frames

Fig. 8. Comparison of KPCA and KRIGA in terms of ARE and running time.

5.3.1 Synthetic experiment

We follow the exact same setup for the synthetic experiment as in [56]. We choose $D = 200$ and $K = 10$ and select the ground truth subspace as $\mathcal{U}^* = \text{Col}\,(U^*)$, where U^* is composed of i.i.d. samples from standard normal distribution. The coefficients are drawn from a standard normal distribution. The samples of the data are generated by adding a Gaussian noise with zero mean and σ standard deviation. σ is set to lie in the range $\left[10^{-2}, 10^{-5}, 0\right]$. Given an orthonormal basis, $\widetilde{U}$ of the estimated subspace, we compute the projected error using, $\|U^* - \left(\widetilde{U}\widetilde{U}^T\right) U^*\|$. We compare our results with an extensive sets of algorithms including MDISVD [22], Brand's [23], PIMC [24], PETRELS [25], IALM [20], GROUSE and Oja's algorithm. We should mention that for IALM requires the number of observations to be $\geq D$, which is the reason for using a constant value for the first $D - 1$ observations. The performance is depicted in Fig. 9, which clearly indicates that RRIGA outperforms others in the comparisons.

5.3.2 Real experiment

Now, we present experiments on real datasets. In these set of experiments we compare performance of RRIGA with GRASTA, DHR-PCA, IALM and REPROCS. As mentioned earlier, we present results of experiments performed on **(a)** the UCSD anomaly detection database, **(b)** the Extended YaleB database and the **(c)** Wallflower database.

UCSD anomaly detection database: This data contains images of pedestrian movement on walkways captured by a stationary mounted camera. The crowd density on the walkway varies from sparse to very crowded. The anomaly includes bikers, skaters, carts, people in wheelchair etc. This database is divided in two sets: "Peds1" (people are walking towards the camera) and "Peds2" (people are walking parallel to the camera plane). We will only consider "Peds1" in this experiment. In "Peds1" there are 36 training and 34 testing videos where each video contains 200 frames of dimension 158×238 ($D = 37604$). The test frames do not have any anomalous activities. Some sample frames (with and without outliers) are shown in Fig. 10. We first extract K principal components on the training data (including anomalies) and then compute reconstruction error on the test frames (without anomalies) using the computed principal components. It is expected that if the PC computation technique is robust, the reconstruction error will be good since PCs should not be affected by the anomalies in training samples. In Figs. 11,12, we compare the performance of RRIGA with GRASTA, DHR-PCA, IALM, REPROCS in terms of RE and the (computation) time required. In terms of time it is evident that RRIGA is very fast compared to the competitors. RRIGA also outperforms the state-of-the-art in terms of the RE.

Yale ExtendedB database: This data contains 2414 face images of 38 subjects. Each image was cropped to a 32×32 image ($D = 1024$). Due to varying lighting conditions, some of the face images are shaded/ dark and can be treated as outliers (this experimental setup is similar to the one in [57]). In Fig. 10 some sample face images (outliers and non-outliers) are shown. One can see that due to poor lighting condition, though the face in the middle of the top row is a face image, it appears completely dark and as an outlier. For testing, we use 142 non-outlier face images of 38 subjects and the rest are used to extract the PCs. We report RE (with varying K) and computation time required for RRIGA, GRASTA, DHR-PCA, IALM, REPROCS methods in Figs. 13, 12 respectively. As evident, RRIGA is faster than the competitors while outperforming all of the state-of-the-art methods except REPROCS in terms of reconstruction error.

6 Conclusions

In this paper, we present a geometric framework to compute principal linear subspaces in finite and infinite dimensional reproducing kernel Hilbert spaces (RKHS). We compute an intrinsic Grassmann average as a proxy for the principal linear subspace and show that if the samples are drawn from a Gaussian distribution, the intrinsic Grassmann average coincides with the principal subspace in expectation. We further show that the approach extends to the RKHS setting. A robust version of the online PCA is also presented along with several experiments demonstrating its performance in comparison to the state-of-the-art. The approach has several advantages. Unlike the work by Hauberg et al. in [7], our estimator returns the first $K \geq 1$ components. The proposed algorithm is inherently online, which also makes it scalable to large datasets. We have demonstrated this by performing principal component analysis of an entire Hollywood movie. Unlike most other online algorithms there are no step-sizes or other parameters to tune; a very useful property in practical settings. We extend the approach to RKHS and thereby provided an algorithm that serves the same purpose as kernel PCA. A benefit of our formulation is that, unlike KPCA, our estimator does not require an eigen decomposition of the

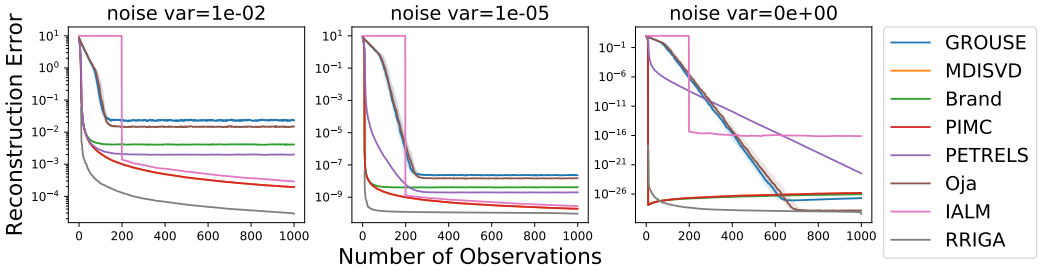


Fig. 9. Comparative analysis for different noise levels. For all the competing methods, we use the suggested parameters in [56]

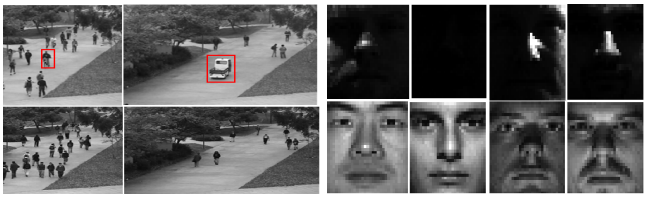


Fig. 10. Top and bottom row contain outliers (identified in a rectangular box) and non-outliers frames of UCSD and YaleExtendedB data respectively.

Gram matrix. Empirically, we observe that our algorithm is significantly faster than KPCA while giving similar results.

ACKNOWLEDGEMENTS

We thank Chun-Hao Yang for several helpful suggestions. This research was in part funded in part by the NSF grants IIS-1525431 and IIS-1724174 to BCV. SH was supported by a research grant (15334) from VILLUM FONDEN. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement nº 757360).

REFERENCES

[1] K. Pearson, "On lines and planes of closest fit to system of points in space," *Philosophical Magazine*, vol. 2, no. 11, pp. 559–572, 1901.
[2] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.
[3] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.
[4] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of computational and graphical statistics*, vol. 15, no. 2, pp. 265–286, 2006.
[5] R. Vidal, Y. Ma, and S. Sastry, "Generalized principal component analysis (gpca)," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 12, pp. 1945–1959, 2005.
[6] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Phil. Trans. R. Soc. A*, vol. 374, no. 2065, p. 20150202, 2016.
[7] S. Hauberg, A. Feragen, R. Enficiaud, and M. J. Black, "Scalable robust principal component analysis using grassmann averages," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.
[8] R. Chakraborty, S. Hauberg, and B. C. Vemuri, "Intrinsic grassmann averages for online linear and robust subspace learning," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
[9] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Artificial Neural Networks*. Springer, 1997, pp. 583–588.
[10] E. Oja, "Simplified neuron model as a principal component analyzer," *Journal of mathematical biology*, vol. 15, no. 3, pp. 267–273, 1982.
[11] Z. Allen-Zhu and Y. Li, "First efficient convergence for streaming k-PCA: a global, gap-free, and near-optimal rate," in *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, Berkely, CA, october 2017.
[12] S. Roweis, "EM algorithms for pca and spca," *Advances in neural information processing systems*, pp. 626–632, 1998.
[13] O. Cappé, "Online expectation-maximisation," *Mixtures: Estimation and Applications*, pp. 31–53, 2011.
[14] L. Balzano, R. Nowak, and B. Recht, "Online identification and tracking of subspaces from highly incomplete information," in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. IEEE, 2010, pp. 704–711.
[15] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *CVPR*, 2012, pp. 1568–1575.
[16] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
[17] C. Boutsidis, D. Garber, Z. Karnin, and E. Liberty, "Online principal components analysis," in *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2015, pp. 887–901.
[18] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, p. 11, 2011.
[19] B. Lois and N. Vaswani, "Online robust pca and online matrix completion," *arXiv preprint arXiv:1503.03525*, 2015.
[20] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.
[21] J. R. Bunch and C. P. Nielsen, "Updating the singular value decomposition," *Numerische Mathematik*, vol. 31, no. 2, pp. 111–129, 1978.
[22] R. Kennedy, L. Balzano, S. J. Wright, and C. J. Taylor, "Online algorithms for factorization-based structure from motion," *Computer Vision and Image Understanding*, vol. 150, pp. 139–152, 2016.
[23] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," in *European Conference on Computer Vision*. Springer, 2002, pp. 707–720.
[24] R. Kennedy, C. J. Taylor, and L. Balzano, "Online completion of ill-conditioned low-rank matrices," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2014, pp. 507–511.
[25] Y. Chi, Y. C. Eldar, and R. Calderbank, "PETRELS: Parallel subspace estimation and tracking by recursive least squares from partial observations," *IEEE Transactions on Signal Processing*, vol. 61, no. 23, pp. 5947–5959, 2013.
[26] B. Yang, "Projection approximation subspace tracking," *IEEE Transactions on Signal processing*, vol. 43, no. 1, pp. 95–107, 1995.
[27] H. Guo, C. Qiu, and N. Vaswani, "An online algorithm for separating sparse and low-dimensional signal sequences from their sum," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4284–4297, 2014.
[28] W. Ha and R. F. Barber, "Robust pca with compressed data," in *Advances in Neural Information Processing Systems*, 2015, pp. 1936–1944.
[29] J. Feng, H. Xu, and S. Yan, "Online robust pca via stochastic optimization," in *Advances in Neural Information Processing Systems*, 2013, pp. 404–412.
[30] ——, "Robust pca in high-dimension: A deterministic approach," *arXiv preprint arXiv:1206.4628*, 2012.

Fig. 11. Comparison of reconstructions of a random test sample. *(Left-to-Right)* Original sample, RRIGA, GRASTA, DHR-PCA, IALM, REPROCS outputs respectively. For GRASTA, default parameters suggested in [15] are used. An average reconstruction error of 9.51, 18.02, 21.46, 30.67, 21.01 is obtained for RRIGA, GRASTA, DHR-PCA, IALM, REPROCS respectively.

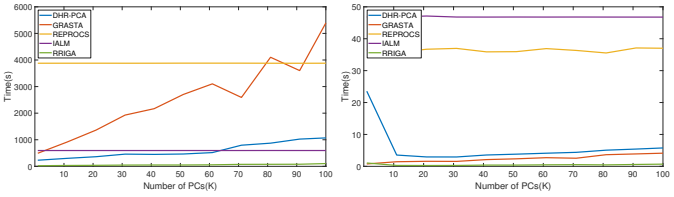


Fig. 12. Computation time required for "Peds1" (*Left*) and "YaleB" (*Right*) respectively.

Fig. 13. Comparison of reconstruction of random test samples. *(Left-to-Right)* Original sample, RRIGA, GRASTA, DHR-PCA, IALM and REPROCS outputs respectively. Default parameter settings suggested in [15] are used for GRASTA. An average reconstruction error of 2.91, 4.85, 3.80, 3.03, 1.65 is obtained for RRIGA, GRASTA, DHR-PCA, IALM and REPROCS respectively.

[31] X. Xu, "Online robust principal component analysis for background subtraction: A system evaluation on toyota car data," Master's thesis, University of Illinois at Urbana-Champaign, 2014.

[32] C. Qiu, N. Vaswani, B. Lois, and L. Hogben, "Recursive robust pca or recursive sparse recovery in large but structured noise," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 5007–5039, 2014.

[33] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in neural information processing systems*, 2008, pp. 1177–1184.

[34] P. Honeine, "Online kernel principal component analysis: A reduced-order model," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 9, pp. 1814–1826, 2012.

[35] M. Ghashami, D. J. Perry, and J. Phillips, "Streaming kernel principal component analysis," in *Artificial Intelligence and Statistics*, 2016, pp. 1365–1374.

[36] K. I. Kim, M. O. Franz, and B. Scholkopf, "Iterative kernel principal component analysis for image modeling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 9, pp. 1351–1366, 2005.

[37] S. Günter, N. N. Schraudolph, and S. Vishwanathan, "Fast iterative kernel principal component analysis," *Journal of Machine Learning Research*, vol. 8, no. Aug, pp. 1893–1918, 2007.

[38] H. Karcher, "Riemannian center of mass and mollifier smoothing," *Communications on pure and applied mathematics*, vol. 30, no. 5, pp. 509–541, 1977.

[39] M. Fréchet, "Les éléments aléatoires de nature quelconque dans un espace distancié," in *Annales de l'institut Henri Poincaré*, vol. 10. Presses universitaires de France, 1948, pp. 215–310.

[40] W. S. Kendall, "Probability, convexity, and harmonic maps with small image i: uniqueness and fine existence," *Proceedings of the London Mathematical Society*, vol. 3, no. 2, pp. 371–406, 1990.

[41] Y.-C. Wong, "Sectional curvatures of grassmann manifolds," *Proceedings of the National Academy of Sciences*, vol. 60, no. 1, pp. 75–79, 1968.

[42] B. Afsari, "Riemannian Lp center of mass: existence, uniqueness, and convexity," *Proceedings of the American Mathematical Society*, vol. 139, no. 2, pp. 655–673, 2011.

[43] X. Pennec, "Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements," *Journal of Mathematical Imaging and Vision*, vol. 25, no. 1, pp. 127–154, 2006.

[44] R. Chakraborty and B. C. Vemuri, "Recursive frechet mean computation on the grassmannian and its applications to computer vision," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4229–4237.

[45] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Computational learning theory*. Springer, 2001, pp. 416–426.

[46] W. Rudin, *Fourier analysis on groups*. Courier Dover Publications, 2017.

[47] P. T. Fletcher, S. Venkatasubramanian, and S. Joshi, "The geometric median on riemannian manifolds with application to robust atlas estimation," *NeuroImage*, vol. 45, no. 1, pp. S143–S152, 2009.

[48] S. Bonnabel, "Stochastic gradient descent on riemannian manifolds," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2217–2229, 2013.

[49] M. Nakagami, "The m-distribution-a general formula of intensity distribution of rapid fading," *Statistical Method of Radio Propagation*, 1960.

[50] P. J. Huber, *Robust statistics*. Springer, 2011.

[51] M. Abramowitz, I. A. Stegun *et al.*, "Handbook of mathematical functions," *Applied mathematics series*, vol. 55, p. 62, 1966.

[52] S. Roman, *The umbral calculus*. Springer, 2005.

[53] G. Pons-Moll, J. Romero, N. Mahmood, and M. J. Black, "Dyna: A model of dynamic human shape in motion," *ACM Transactions on Graphics*, vol. 34, no. 4, pp. 120:1–120:14, Aug. 2015.

[54] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes." in *CVPR*, vol. 249, 2010, p. 250.

[55] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 643–660, 2001.

[56] L. Balzano, Y. Chi, and Y. M. Lu, "Streaming pca and subspace tracking: The missing data case," *Proceedings of the IEEE*, vol. 106, no. 8, pp. 1293–1310, 2018.

[57] W. Jiang, F. Nie, and H. Huang, "Robust dictionary learning with capped l 1-norm," in *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, 2015, pp. 3590–3596.

Rudrasis Chakraborty received his Ph.D. in computer science from the Univ. of Florida in 2018. He is currently a post doctoral researcher at UC Berkeley. His research interest lies in the intersection of Geometry, Machine Learning and Computer Vision.

Liu Yang is an undergraduate student in Xi'an Jiaotong University. She is currently an exchange student in UC Berkeley. Her research interest lies in Machine Learning and Computer Vision.

Søren Hauberg received his Ph.D. in computer science from the Univ. of Copenhagen in 2012. He has been a visiting scholar at UC Berkeley (2010), and a post doc at the Perceiving Systems department at the Max Planck Institute for Intelligent Systems in Tübingen, Germany (2012–2014). He is currently an associate professor at the Section for Cognitive Systems, at the technical Univ. of Denmark. His research is at the interplay of geometry and statistics.

Baba C. Vemuri is the Wilson and Marie Collins professor of Engineering at the Department of Computer Information Science and Engineering and the Department of Statistics, University of Florida. His research interests lie in Geometric Statistics, Machine Learning, Computer Vision and Medical Imaging. He is an associte editor of the IJCV and MedIA journals. He received the IEEE Computer Society's Technical Achievement Award (2017) and is a Fellow of the ACM (2009) and the IEEE (2001).