

Dynamic Subspace Estimation with Grassmannian Geodesics

Cameron J. Blocker* Haroon Raja† Jeffrey A. Fessler‡ Laura Balzano§

March 28, 2023

Abstract

Dynamic subspace estimation, or subspace tracking, is a fundamental problem in statistical signal processing and machine learning. This paper considers a geodesic model for time-varying subspaces. The natural cost function for this model is non-convex. We propose a novel optimization algorithm for this cost that estimates the model parameters from data with Grassmannian constraints. We show that with this algorithm, the cost is monotonically non-increasing. We demonstrate the performance of this model and our algorithm on synthetic data, video data, and dynamic fMRI data.

1 Introduction

Modeling data using linear subspaces is a powerful analytical tool that enables practitioners to more efficiently and reliably solve high-level tasks like inference and decision making, classification, and anomaly detection, among others. In some applications of interest, the data generation process is time-varying or dynamic in nature, which motivates the use of a dynamic linear subspace for data modeling. Some example applications where dynamic subspace models are prevalent include array signal processing (Yang, 1995; Fuhrmann, 1997; Srivastava and Klassen, 2004; Lake and Keenan, 1998), communication systems (Haghighatshoar and Caire, 2018), video processing (Vaswani et al., 2018), and dynamic magnetic resonance imaging (MRI) (Otazo et al., 2015). The goal in these applications is to learn a time-varying subspace from the observed data.

Most previous theoretical work for modeling a dynamic subspace relies on very strong assumptions of the dynamics – either assuming very simple dynamics like sudden changes with an otherwise static subspace, or assuming a specific known dynamical model. A much broader empirical literature for *subspace tracking* considers a wide range of algorithms with different strengths and weaknesses with regards to signal-to-noise ratios, speed of dynamics, and computational complexity. For the vast majority of these algorithms, accuracy guarantees in the presence of dynamics are still an open question.

This paper starts with a flexible and natural dynamic subspace model: the piecewise geodesic model. A piecewise geodesic can approximate any curve on the Grassmannian, i.e., any continuously varying subspace. This model generalizes both the previously studied time-varying subspace models and piecewise linear approximations that are pervasive in the theory and practice of statistical signal processing. This model has only been very briefly discussed in existing literature, probably in part due to the difficulty of parameter estimation and algorithmic guarantees in this setting. In this paper, we start by learning a single geodesic. The central contribution of this paper, therefore, is an algorithm for learning the parameters of this model in a batch setting that is guaranteed to descend an appropriate cost function (corresponding to a log-likelihood for Gaussian noise) at every step. We also demonstrate the performance of the proposed algorithm empirically on both synthetic and real datasets.

*Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor. (cblocker@umich.edu).

†Eli Lilly, Indianapolis, IN (raja_haroon@lilly.com).

‡Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor. (fessler@umich.edu).

§Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor. (girasole@umich.edu.)

1.1 Problem Formulation and Geodesic Model

We start with the following broad generative model for data arising from a time-varying subspace. At each time point i we observe ℓ vectors from a time-varying subspace. Let $\mathbf{X}_i \in \mathbb{R}^{d \times \ell}$ for $i = 1, 2, \dots, T$ be data generated from a low-rank model with noise:

$$\mathbf{X}_i = \mathbf{U}_i \mathbf{G}_i + \mathbf{N}_i \quad (1)$$

where $\mathbf{U}_i \in \mathbb{R}^{d \times k}$ is a matrix with orthonormal columns representing a point on the Grassmannian $\mathcal{G}(k, d)$, the space of all rank- k subspaces in $\mathbb{R}^d$; $\mathbf{G}_i \in \mathbb{R}^{k \times \ell}$ holds weight or loading vectors; and $\mathbf{N}_i \in \mathbb{R}^{d \times \ell}$ is an independent additive noise matrix. We observe $\mathbf{X}_i$ and our goal is to estimate a sequence of subspaces $\{\mathbf{U}_i\}_{i=1}^T$ that generates the observed samples. Though we use “time-varying” to describe this generative model, it can vary over any index, and the algorithms we consider are batch in the sense that they use all the data $\mathbf{X}_i$, $i = 1, \dots, T$ for estimating $\mathbf{U}_i$, $i = 1, \dots, T$.

If $\mathbf{U}_i = \bar{\mathbf{U}}$ is static for all $i = 1, 2, \dots, T$, and if $\mathbf{G}_i$ has zero-mean columns, then we could concatenate all $\mathbf{X}_i$ together and apply the SVD, which is well-known to recover a good approximation of $\bar{\mathbf{U}}$ as long as the number of samples ℓT is large enough to overcome the noise. However, if $\mathbf{U}_i$ is varying for every i , one can immediately see that if $\ell < k$, estimating $\mathbf{U}_i$ is impossible without further assumptions. Even when $\ell \geq k$, in many applications it is natural to impose a relationship between the $\mathbf{U}_i$ subspace matrices over time, to guarantee regularity properties or known application constraints. Various constraints have been studied in the literature, such as a slowly rotating subspace, a subspace that is mostly static except for intermittent sudden changes, or a subspace that changes one dimension at a time (Narayanamurthy and Vaswani, 2018). Those models are all subsumed by the **piecewise geodesic** model for dynamic subspaces, illustrated in Figure 1. In this work, we focus on efficiently learning a single geodesic.

Model for a Single Geodesic Let $2k \leq d$. We model each $\mathbf{U}_i$ as an orthonormal basis whose span has been sampled from a single continuous Grassmannian geodesic $\mathbf{U}(t) : [0, 1] \rightarrow \mathcal{V}^{d \times k}$ parameterized as follows:

$$\mathbf{U}_i = \mathbf{U}(t_i) = \mathbf{H} \cos(\boldsymbol{\Theta} t_i) + \mathbf{Y} \sin(\boldsymbol{\Theta} t_i) \quad (2)$$

where $\mathcal{V}^{d \times k}$ is the set of $d \times k$ matrices with orthonormal columns (the Stiefel manifold), $\mathbf{H} \in \mathcal{V}^{d \times k}$ is an orthonormal basis for a point on the Grassmannian, $\mathbf{Y}$ is a matrix with orthonormal columns whose span is in the tangent space of the Grassmannian at $\text{span}(\mathbf{H})$, *i.e.*, $\mathbf{Y} \in \{\mathbf{Y} | \mathbf{H}^\top \mathbf{Y} = \mathbf{0}, \mathbf{Y} \in \mathcal{V}^{d \times k}\}$, and $\boldsymbol{\Theta} \in \mathbb{R}^{k \times k}$ is a diagonal matrix where θ_j is the j th principal angle between the two endpoints of the geodesic, and sine/cosine denote the matrix versions. These constraints ensure each $\mathbf{U}_i$ has orthonormal columns. The scalars $t_i \in [0, 1]$ represent the location of each $\mathbf{U}_i$ along the geodesic, *e.g.*, if the geodesic is sampled over time, then these are time-points scaled (or normalized) to the interval. For more information, see (Absil et al., 2004, Section 3.8) and Edelman et al. (1998).

Because we are interested only in the span of $\mathbf{U}_i$, this parameterization of a Grassmannian geodesic is not unique. Permuting the columns of $\mathbf{H}$, $\mathbf{Y}$ and the diagonal elements of $\boldsymbol{\Theta}$ would result in a $\mathbf{U}_i$ with the same span. Additionally, there is a sign ambiguity between columns of $\mathbf{Y}$ and diagonal elements of $\boldsymbol{\Theta}$. In practice, our loss is invariant to these ambiguities, so they are not a problem. Any specific parameterization can easily be transformed into another.

$\mathbf{H}$, $\mathbf{Y}$ and $\boldsymbol{\Theta}$ are all learnable parameters of $\mathbf{U}(t)$. Conceptually, we can think of $\mathbf{H}$ as a starting point on the Grassmannian, $\mathbf{Y}$ as a normalized direction along which the model walks, and the diagonal elements of $\boldsymbol{\Theta} t_i$ as the distances in each dimension we should walk from $\mathbf{H}$ on the surface of the manifold to get to $\mathbf{U}_i$.

Single Geodesic vs. Piecewise Geodesic In this work, we focus on learning a single geodesic from data with given time points t_i . This focus essentially makes two key simplifying assumptions: (1) the locations of the knots, or change-points, in a piecewise approximation are given, and (2) between two knots in the piecewise approximation, either the time-points t_i are given, or observed matrices $\mathbf{X}_i$ are equidistant along a geodesic curve. With these assumptions, our high-level approach is to take each set of data matrices between change-points and learn a single geodesic. Section B.2.1 discusses how to learn multiple geodesics with knots known. Relaxing these assumptions is potential future work.

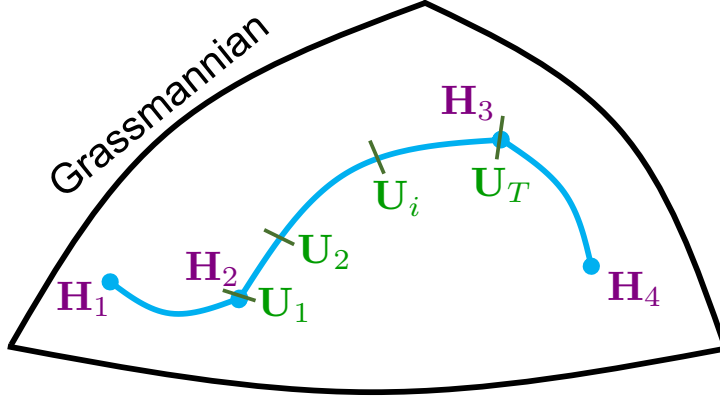


Figure 1: Illustration of the piecewise geodesic model. $\mathbf{H}_1, \dots, \mathbf{H}_4$ are points on the Grassmannian. When estimating a single geodesic, e.g., the one from $\mathbf{H}_2$ to $\mathbf{H}_3$, then $\mathbf{H}$ is an orthonormal basis for $\mathbf{H}_2$ and $\mathbf{Y}$ is an orthonormal basis for $(\mathbf{I} - \mathbf{H}\mathbf{H}^\top)\mathbf{H}_3$.

1.2 Related work

Classical literature on subspace tracking uses online approaches to estimate the time-varying subspaces (Yang, 1995; Chi et al., 2013; Allen-Zhu and Li, 2017; Balzano et al., 2018; Haghighatshoar and Caire, 2018; Narayanamurthy and Vaswani, 2018; Vaswani et al., 2018; Comon and Golub, 1990). Early theoretical results were limited to asymptotic convergence guarantees with static underlying subspaces. Among the more recent works, the PETRELS algorithm (Chi et al., 2013) portrays a recursive least squares approach and provides convergence theory that assumes that the subspace changes at a particular instant and then stays constant for sufficient time so that the change can be tracked (also called the piecewise constant model). Narayanamurthy and Vaswani (2018) relax the assumption of constant subspace to a very slowly varying subspace between the change points. For a review of these methods, see Vaswani et al. (2018).

Dynamic subspace estimation has also been studied for the more general Grassmannian geodesic model that is the focus of this paper (Lake and Keenan, 1998; Fuhrmann, 1997; Srivastava and Klassen, 2004; Hong et al., 2016). Unlike the subspace tracking problem, these contributions focus on batch data settings with access to the whole dataset for estimation, which is the approach we take herein. For example, Lake and Keenan (1998) formulate the subspace tracking for any given epoch in the generative geodesic model in the form of a likelihood function to maximize. The likelihood is nonconcave, but the authors provide an annealing approach to solve it, which will have very high computational burden for many modern high-dimensional and large-data applications. Fuhrmann (1997) and Srivastava and Klassen (2004) have also studied the generative geodesic model. The solution provided by Srivastava and Klassen (2004) is not applicable to large-scale settings since it relies on a sampling based strategy like Markov chain Monte-Carlo (MCMC) and hence is computationally intensive. On the other hand, the solution provided by Fuhrmann (1997) is computationally inexpensive, but it only handles one-dimensional subspaces ($k = 1$). In summary, major weaknesses of the state-of-the-art methods include high computational costs, lack of theoretical guarantees, and/or need to tune hyperparameters. This paper approaches the problem using modern nonconvex optimization tools, alleviating these issues by devising an algorithm that is parameter free other than the subspace dimension, descends the loss function monotonically, and uses thin SVDs to solve the problem.

2 Method

Given data and associated time points $\{\mathbf{X}_i, t_i\}_{i=1}^T$, we fit the proposed geodesic model for $\mathbf{U}(t)$ by minimizing the following loss function

$$\mathcal{L}(\mathbf{U}) = \mathcal{L}(\mathbf{H}, \mathbf{Y}, \boldsymbol{\Theta}) \quad (3)$$

$$= \min_{\{\mathbf{G}_i\}_{i=1}^T} \sum_{i=1}^T \|\mathbf{X}_i - \mathbf{U}(t_i) \mathbf{G}_i\|_F^2 \quad (4)$$

$$= - \sum_{i=1}^T \|\mathbf{X}_i^\top \mathbf{U}(t_i)\|_F^2 + c, \quad (5)$$

where in the last equality we have substituted the optimal $\mathbf{G}_i = (\mathbf{U}(t_i))^\top \mathbf{X}_i$ and simplified and c is a constant (Golub and Pereyra, 2003). At a high level, we minimize our loss function with respect to $\mathbf{H}$, $\mathbf{Y}$, and $\boldsymbol{\Theta}$ via block coordinate descent. Our first block updates $(\mathbf{H}, \mathbf{Y})$ jointly via an SVD of a $d \times 2k$ matrix. Next, we update $\boldsymbol{\Theta}$ via a first-order iterative minimization. Specifically, we design an efficient majorize-minimize (MM) iteration for updating $\boldsymbol{\Theta}$.

2.1 $(\mathbf{H}, \mathbf{Y})$ Update

Let $\mathbf{Q} \triangleq [\mathbf{H} \ \mathbf{Y}]$ and $\mathbf{Z}_i \triangleq [\cos(\boldsymbol{\Theta} t_i); \sin(\boldsymbol{\Theta} t_i)]$. Then we rewrite our model (2) as $\mathbf{U}_i = \mathbf{Q} \mathbf{Z}_i$. By constraining $\mathbf{Q} \in \mathcal{V}^{d \times 2k}$, we also satisfy the constraints of $\mathbf{H}$ and $\mathbf{Y}$ individually.

Following Breloy et al. (2021), we form a linear majorizer for our loss (5) at $\mathbf{Q}^{(n)}$ and minimize it with a Stiefel constraint simply by projecting its negative gradient onto the Stiefel manifold. The update is then given by

$$\mathbf{Q}^{(n+1)} = \arg \min_{\mathbf{Q} \in \mathcal{V}^{d \times 2k}} \left\| \mathbf{Q} - \sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Q}^{(n)} \mathbf{Z}_i \mathbf{Z}_i^\top \right\|_F^2 = \mathbf{W} \mathbf{V}^\top, \quad (6)$$

where $\mathbf{W} \boldsymbol{\Sigma} \mathbf{V}^\top$ is the SVD of $\sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Q}^{(n)} \mathbf{Z}_i \mathbf{Z}_i^\top$. Further, using the simplification from (5) we have $\hat{\mathbf{G}}_i^\top = \mathbf{X}_i^\top \mathbf{U}_i^{(n)} = \mathbf{X}_i^\top \mathbf{Q}^{(n)} \mathbf{Z}_i$. Using this equality we can express the summation term in (6) in terms $\mathbf{X}_i$, $\hat{\mathbf{G}}_i$, and $\boldsymbol{\Theta}$ as follows

$$\sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Q}^{(n)} \mathbf{Z}_i \mathbf{Z}_i^\top = \sum_{i=1}^T \begin{bmatrix} \mathbf{X}_i \hat{\mathbf{G}}_i^\top \cos(\boldsymbol{\Theta} t_i) & \mathbf{X}_i \hat{\mathbf{G}}_i^\top \sin(\boldsymbol{\Theta} t_i) \end{bmatrix},$$

where we let $\hat{\mathbf{G}}_i^\top = \mathbf{X}_i^\top \mathbf{U}_i^{(n)} = \mathbf{X}_i^\top \mathbf{Q}^{(n)} \mathbf{Z}_i$. Although here we eliminated $\{\mathbf{G}_i\}_{i=1}^T$ from our loss (5), if we had not, this same update could be derived as a block coordinate update on $\mathbf{Q}$ and $\{\mathbf{G}_i\}_{i=1}^T$. See Appendix A.1 for more details.

2.2 $\boldsymbol{\Theta}$ Update

The loss (5) for fixed $\mathbf{H}$ and $\mathbf{Y}$ is a smooth function of $\boldsymbol{\Theta}$ and can be effectively optimized via an iterative quadratic majorize-minimize scheme. Here we provide an overview of the method. Appendix A.2 provides a more complete derivation of the majorizer. First, we simplify the loss (5)

$$\begin{aligned} \hat{\boldsymbol{\Theta}} &= \arg \min_{\boldsymbol{\Theta}} - \sum_{i=1}^T \|\mathbf{X}_i^\top \mathbf{U}_i\|_F^2 \\ &= \arg \min_{\boldsymbol{\Theta}} - \sum_{i=1}^T \|\mathbf{X}_i^\top (\mathbf{H} \cos(\boldsymbol{\Theta} t_i) + \mathbf{Y} \sin(\boldsymbol{\Theta} t_i))\|_F^2 \\ &= \arg \min_{\boldsymbol{\Theta}} - \sum_{i=1}^T \sum_{j=1}^k r_{i,j} \cos(2\theta_j t_i - \phi_{i,j}) + b_{i,j}, \end{aligned}$$

where defining $\arctan2(y, x)$ as the angle of the point (x, y) in the 2D plane counter-clockwise from the positive x -axis, the associated constants $r_{i,j}, \phi_{i,j}, b_{i,j}$ are defined as

$$\phi_{i,j} = \arctan2\left(\beta_{i,j}, \frac{\alpha_{i,j} - \gamma_{i,j}}{2}\right) \quad (7)$$

$$r_{i,j} = \sqrt{\left(\frac{\alpha_{i,j} - \gamma_{i,j}}{2}\right)^2 + \beta_{i,j}^2} \quad (8)$$

$$b_{i,j} = \frac{\alpha_{i,j} + \gamma_{i,j}}{2} \quad (9)$$

$$\alpha_{i,j} = [\mathbf{H}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{H}]_{j,j} \quad (10)$$

$$\beta_{i,j} = \text{real}\left\{[\mathbf{Y}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{H}]_{j,j}\right\} \quad (11)$$

$$\gamma_{i,j} = [\mathbf{Y}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Y}]_{j,j}. \quad (12)$$

This loss is separable for each diagonal element of Θ , so we find each $\hat{\theta}_j$ via a (1D) minimization. Let $f_{i,j}(\theta_j) \triangleq -r_{i,j} \cos(2\theta_j t_i - \phi_{i,j}) + b_{i,j}$. Then (cf. Funai et al. (2008)) the following $q_{i,j}$ defines a quadratic majorizer for $f_{i,j}$ at θ_j

$$\begin{aligned} q_{i,j}(\theta_j; \bar{\theta}_j) &= f_{i,j}(\bar{\theta}_j) + \dot{f}_{i,j}(\bar{\theta}_j)(\theta_j - \bar{\theta}_j) + \frac{1}{2} w_{f_{i,j}}(\bar{\theta}_j) (\theta_j - \bar{\theta}_j)^2 \\ &\geq f_{i,j}(\theta_j) \end{aligned} \quad (13)$$

where the derivative $\dot{f}_{i,j}$ and curvature function $w_{f_{i,j}}$ are given by

$$\begin{aligned} \dot{f}_{i,j}(\theta_j) &= 2r_{i,j} t_i \sin(2\theta_j t_i - \phi_{i,j}) \\ w_{f_{i,j}}(\theta_j) &= \frac{\dot{f}_{i,j}(\theta_j)}{\text{mod}\left((\theta_j - \frac{\phi_{i,j}}{2t_i}) + \frac{\pi}{2t_i}, \frac{2\pi}{2t_i}\right) - \frac{\pi}{2t_i}}. \end{aligned}$$

Appendix A.2.2 gives a detailed construction of $w_{f_{i,j}}$.

Our majorize-minimize iterations for each diagonal element of Θ are then given by

$$\theta_j^{(n+1)} = \arg \min_{\theta_j} \sum_{i=1}^T q_{i,j}(\theta_j; \theta_j^{(n)}) \quad (14)$$

$$= \theta_j^{(n)} - \frac{\sum_{i=1}^T \dot{f}_{i,j}(\theta_j^{(n)})}{\sum_{i=1}^T w_{f_{i,j}}(\theta_j^{(n)})}. \quad (15)$$

Conceptually, each MM update can be interpreted as a gradient descent step with a variable step size $s^{(n)} = 1/(\sum_{i=1}^T w_{f_{i,j}}(\theta_j^{(n)}))$ that is guaranteed to not increase the loss even without any line search. Indeed, as both updates just outlined are known for their monotonicity properties, we have the following monotonicity result for our overall algorithm. Since the algorithm monotonically decreases the loss, which is bounded, this implies that the loss always converges. Empirically we see convergence to the planted subspace end points in the vast majority of experiments, but not in all. A more thorough investigation of the loss landscape and algorithmic convergence properties is of great interest for future work.

Theorem 1 *Algorithm 1 produces iterates $\mathbf{U}^{(n)}(t) = \mathbf{H}^{(n)} \cos(\Theta^{(n)} t) + \mathbf{Y}^{(n)} \sin(\Theta^{(n)} t)$ that are monotonically non-increasing in loss (5), i.e., $\mathcal{L}(\mathbf{U}^{(n+1)}) \leq \mathcal{L}(\mathbf{U}^{(n)})$.*

Proof: It suffices to show that each block coordinate update does not increase the loss. Both the $[\mathbf{H} \ \mathbf{Y}]$ block and Θ block updates are instances of MM methods that guarantee this property. See, for example, Sun et al. (2017) Section II.C for a general treatment and Breloy et al. (2021) Section III.B for MM convergence with the nonconvex Stiefel constraint.

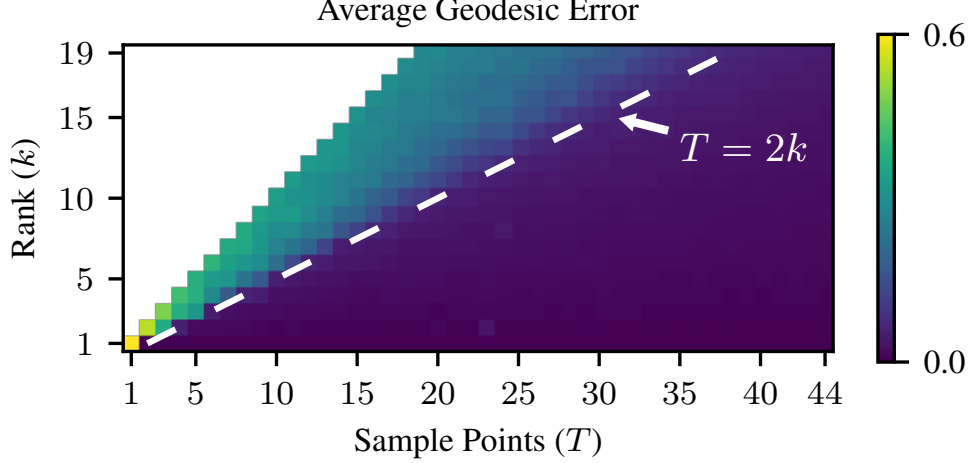


Figure 2: The average geodesic error over 15 trials for varying rank k and number of sample points T . One vector was sampled at each of T points ($\ell = 1$). The ambient dimension was $d = 40$, and we added zero-mean white Gaussian noise with standard deviation $\sigma = 10^{-5}$. We see a phase transition at $T = 2k$; with at least this many samples, we recover the true subspace with low error.

Complexity The time complexity for each iteration is $O(Tdk\ell)$, assuming $T\ell \geq k$. This is the complexity for both the $\mathbf{Q}$ and Θ update.

For the $\mathbf{Q}$ update we need to form the matrix $\sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Q}^{(n)} \mathbf{Z}_i \mathbf{Z}_i^\top$ and take its SVD (see (6) and the surrounding text). This requires $O(Tdk\ell + dk^2)$ operations, where $O(Tdk\ell)$ forms the sum of the product of matrices and $O(dk^2)$ computes the SVD of this $d \times 2k$ matrix.

For the Θ update, the key computational steps form the matrix products $\mathbf{H}^\top \mathbf{X}_i$ and $\mathbf{Y}^\top \mathbf{X}_i$ that are used in the calculation of α, β, γ from (10), (11), (12), which require $O(dk\ell)$ operations for each $i = 1, \dots, T$. The remaining computations involve scalar operations on k or Tk variables, and thus the overall computational complexity is $O(Tdk\ell)$.

3 Experiments

To show the effectiveness of the proposed method, we present results on both synthetic and real data. On the synthetic data, we show the effect of the different data parameters, such as d, k, ℓ, T and the additive noise standard deviation σ . With the intuition we build on the synthetic data, we present results on real measured data and show how we determine the underlying rank. All experiments were performed on a 2021 Macbook Pro laptop computer and implemented in Python.

3.1 Synthetic Data

In the case of synthetic data, we are able to compare our estimated geodesic $\hat{\mathbf{U}}(t)$ against the true geodesic from which the data was generated. Our error metric is the square root of the average squared subspace error between corresponding points along the geodesic

$$\text{Subspace Error} = \frac{1}{\sqrt{2k}} \|\hat{\mathbf{U}}\hat{\mathbf{U}}^\top - \mathbf{U}\mathbf{U}^\top\|_{\text{F}} \quad (16)$$

$$\text{Geodesic Error} = \sqrt{\int_0^1 \frac{1}{2k} \|\hat{\mathbf{U}}(t)\hat{\mathbf{U}}(t)^\top - \mathbf{U}(t)\mathbf{U}(t)^\top\|_{\text{F}}^2 dt}. \quad (17)$$

In practice, we approximate the integral by sampling the geodesic at a large number of time points. The subspace error (16) takes a minimum value of 0 when $\text{span}(\hat{\mathbf{U}}) = \text{span}(\mathbf{U})$ and a maximum value of 1 when $\text{span}(\hat{\mathbf{U}}) \perp \text{span}(\mathbf{U})$. As such, the geodesic error (17) is similarly bounded. For all of our synthetic

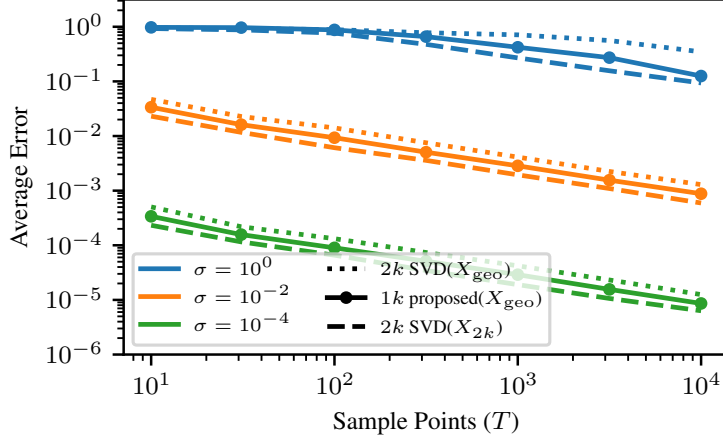


Figure 3: The recovered geodesic error (solid lines) as a function of sample size and additive noise standard deviation, averaged over 10 problems. $\mathbf{X}_{\text{geo}}$ represents the batched geodesic data that has data distributed like a geodesic in a rank- $2k$ subspace. $\mathbf{X}_{2k}$ denotes data that is distributed isotropically in a rank- $2k$ subspace. For data generated from a geodesic, the proposed method recovers the geodesic error with a lower error than an SVD can estimate its span.

experiments, we generated data from our planted model (1). Θ was constrained to generate distance-minimizing geodesics, *i.e.*, $\|\Theta\|_2 < \pi/2$. We drew $\mathbf{G}_i$ from a standard normal distribution. The noise $\mathbf{N}_i$ is additive white Gaussian noise (AWGN) with standard deviation σ . Unless otherwise noted, we initialized the proposed method with a random geodesic for all experiments. For the experiments in Figure 2 and Figure 3, we formed a coarse estimate of the starting and ending rank- k subspaces, and computed a geodesic between these subspaces with (Absil et al., 2004, (19)) for initialization.

Figure 2 shows the average geodesic error when $\ell = 1$, *i.e.*, we receive one vector per k -dimensional subspace, as a function of both the true underlying rank (k) and the number of sample points (T). This plot shows that a phase transition occurs at $T = 2k$, where if we have at least this many samples, the proposed method can recover the true geodesic with low error. At least $2k$ samples are necessary to compute the rank- $2k$ SVD, and this figure shows that $2k$ samples are also sufficient for computing the geodesic endpoints.

In Figure 3, we further investigate the effect of the number of samples on the average geodesic error. Because a rank- k geodesic spans a space as large as $2k$, we have also shown for reference the subspace error of recovering a rank- $2k$ subspace with an SVD under two different distributions of loading vectors. The dashed lines show the subspace recovery error for data generated isotropically in a rank- $2k$ subspace with additive white Gaussian noise. The dotted lines show the subspace recovery error for data distributed on a geodesic in the rank- $2k$ subspace. Both SVD-based methods are only recovering a single rank- $2k$ subspace and not recovering a geodesic. Empirically, we can see that the sample complexity of the proposed geodesic model and method tracks well with rank- $2k$ SVD and outperforms SVD on geodesic data.

When the number of samples per time point (ℓ) is less than k the proposed method is still able to recover the true geodesic given that T is large enough. When $\ell \geq k$, one could estimate the subspace at each time point by applying a rank- k SVD at each time point. But, as shown in Figure 4, even when $\ell \geq k$, the proposed method recovers the subspaces at each time point with lower error for data generated from a geodesic.

Like a low-rank SVD approximation, our method requires choosing the rank k before fitting. Figure 5 shows the loss as a function of assumed rank for data generated from a rank-1 geodesic (left) and rank-2 geodesic (right). Because a rank- k geodesic spans a rank- $2k$ subspace, rank- k and rank- $2k$ SVD results are shown for comparison. Rank- $2k$ SVD will always have a lower loss by definition. Similarly, the proposed model will always have a lower loss than the rank- k SVD, since a rank- k subspace is a special case of a rank- k geodesic. Thus, we can lower-bound and upper-bound the loss of the proposed model on any data by

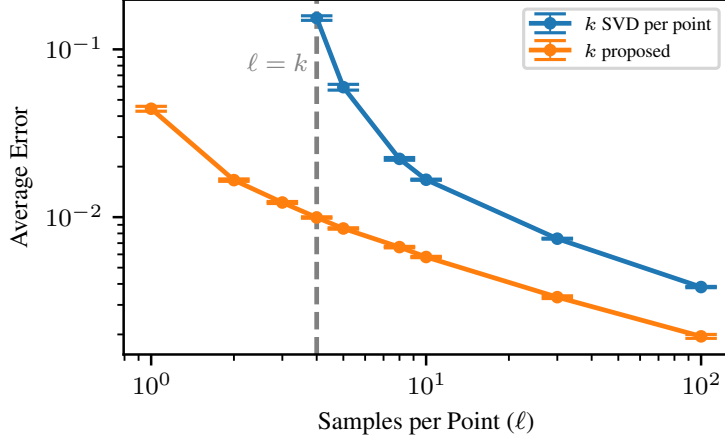


Figure 4: Average geodesic error over 100 trials, with standard error bars, for varying number of samples (ℓ) collected at each time point for a fixed number of time points ($T = 11$) on a planted rank-4 geodesic with AWGN $\sigma = 10^{-2}$. When $\ell \geq k$ we can estimate the subspace with the SVD on just those ℓ samples. The geodesic model can estimate the subspaces even when $\ell < k$ and leverages all of the data to produce lower error.

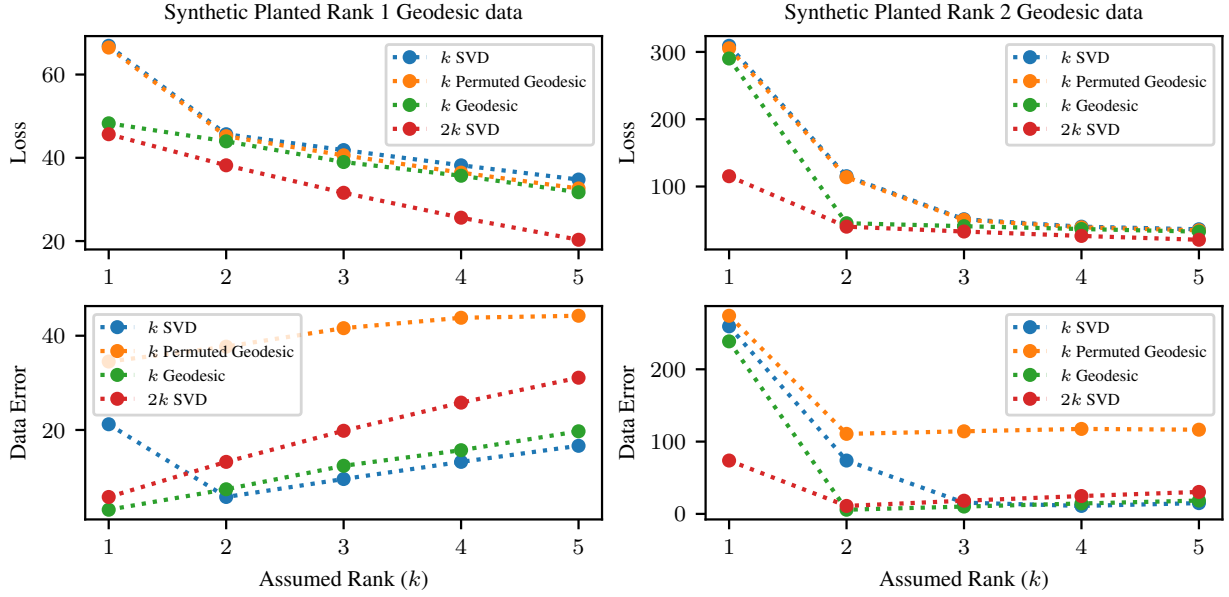


Figure 5: Loss on synthetic data for a rank-1 geodesic (left) and a rank-2 geodesic (right). The loss of the proposed method is lower-bounded and upper-bounded by rank- $2k$ and rank- k SVD, respectively. When the assumed rank is equal to the true rank, then the loss of the proposed method is much closer to that of rank- $2k$ SVD, while permuting the data significantly increases the loss. From this, it is easy to deduce the true rank. The second row shows the “Data Error,” which is the norm of the residual between the projected noisy data and the noiseless data. We can see that rank- $2k$ SVD was overfitting noise to obtain a lower training loss, but this increases its error. The proposed method has a lower error than rank- $2k$ SVD as long as the assumed rank is greater than or equal to the true rank on geodesic data.

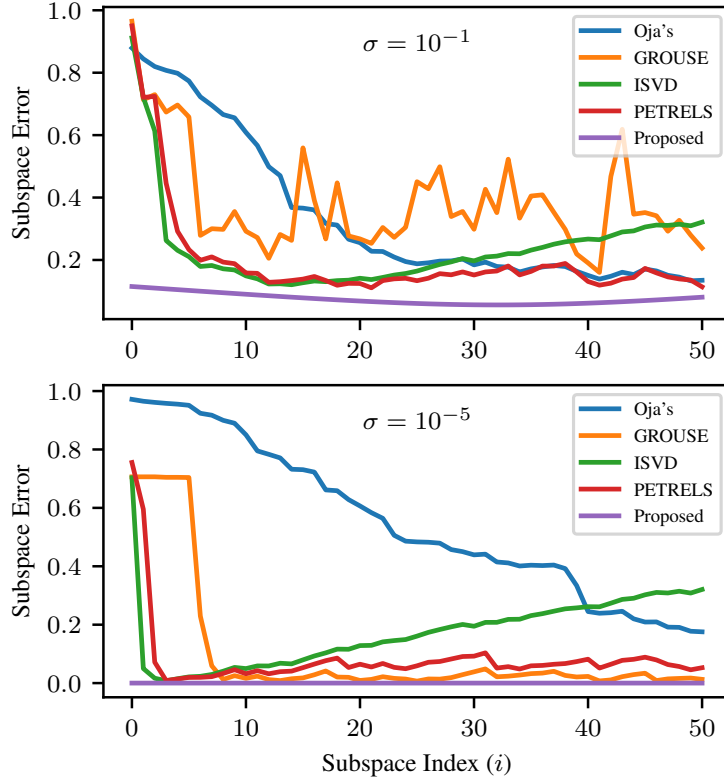


Figure 6: Subspace error at each time point for the proposed method compared to several online subspace tracking algorithms at two different noise levels σ . We sampled a single vector from 51 time points of a rank-2 geodesic for both noise cases. Note that the online algorithms estimate a single time point at a time, while the proposed method considers all time points.

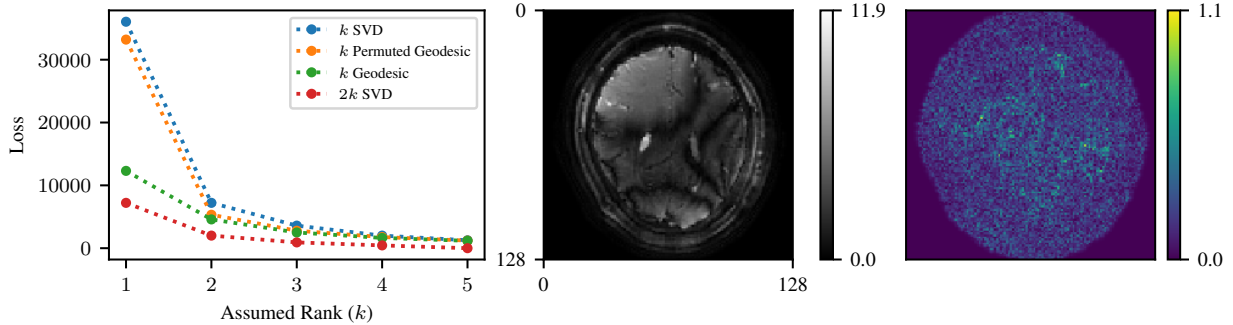


Figure 7: **(Left)** Loss for OSSI dynamic fMRI data. In the $k = 1$ case, the proposed method has a loss close to rank- $2k$ SVD and permuting the data produces a loss closer to rank- k SVD, which is similar to Figure 5 (top, left). From this behavior, we infer that the data likely has rank-1 geodesic structure. **(Center)** Magnitude of reconstructed OSSI image at a single slow and fast test time point. For each pixel, fast time points were collected in a vector and a rank-1 geodesic was fit across odd slow time points for training. Even slow time points were then projected onto the geodesic for test. **(Right)** The magnitude difference map of the reconstruction (center) against the true test point. The geodesic model appears to have smoothed the image, possibly removing noise. The NRMSD across the test points was 0.10. This is slightly larger than using a rank-2 SVD across the batched data which has a NRMSD of 0.08. While imposing a strict temporal structure, the proposed approach fit nearly as well as an unconstrained subspace estimation. Figure 13 shows ground truth and SVD reconstructions.

a rank- $2k$ and rank- k SVD respectively. Additionally, if the data has geodesic structure, then it is ordered. For comparison we also show the loss of the proposed method on data that was generated from a geodesic and then permuted. We see that when the assumed rank is equal to the true rank of the underlying geodesic, the proposed method produces a loss much closer to that of a rank- $2k$ SVD while the proposed method applied to permuted data produces a loss much closer to that of a rank- k SVD. For permuted unordered data, the proposed model learns a geodesic with small values of Θ , approximating a static rank- k subspace similar to a rank- k SVD. For comparison, Figure 5 also shows the data error, which is the norm of the residual between the projected noisy data and the noiseless data. These plots show that rank- $2k$ SVD overfits the noise and has a higher error. The proposed method has a lower error than rank- $2k$ SVD for any assumed rank greater than or equal to the true rank on geodesic data.

Figure 6 compares the proposed geodesic algorithm to several online subspace tracking algorithms, including GROUSE (Balzano et al., 2010), Oja’s algorithm Oja (1982), ISVD (Bunch and Nielsen, 1978), and PETRELS (Chi et al., 2013). We used $d = 20$, $k = 2$, $\ell = 1$ and two noise levels. The proposed method consistently shows better performance (lower error) on geodesic data. See B.1 for details. Appendix B.1 also contains additional synthetic experiments exploring a 2D loss surface and the effect of rank on the rate of convergence.

3.2 fMRI Data

We show the effectiveness of the proposed method by applying it to (fully anonymized) dynamic functional MRI data collected with institutional review board approval. An effective data model for fMRI could be applied as part of an advanced image reconstruction algorithm, allowing for reduced scan times and higher temporal resolution without sacrificing image quality. We leave a full investigation of joint reconstruction and modeling as future work and, here, show only the viability of this model on fMRI data. In particular, we apply the proposed method on data collected with an oscillating steady state imaging (OSSI) (Guo and Noll, 2020) acquisition on a 3T GE MR750 scanner. Appendix B.2 and Guo and Noll (2020) provide more details on acquisition and reconstruction parameters and example data. The OSSI acquisition rapidly cycles through 10 (t_{fast}) different acquisition settings and then repeats this (t_{slow}) for the duration of the scan. During the scan, subject breathing and scanner drift lead to slowly varying subspace changes that we hypothesized are suitable for a geodesic model. The scanner drift is approximately linear in time, so equally

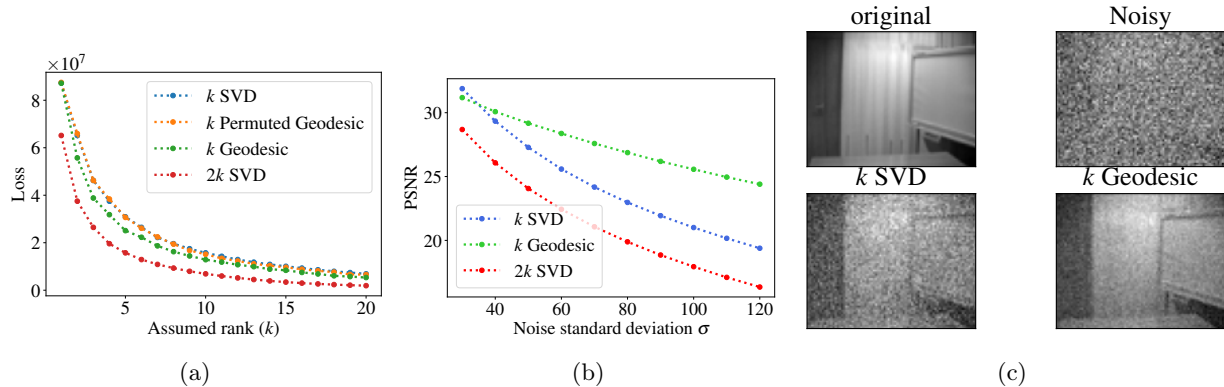


Figure 8: Quantitative evaluation of geodesic subspace model for video data. In (a) loss from (5) is plotted for a video sequence containing 260 frames/images. Loss is plotted against different values of assumed rank of data k . In (b) we added AWGN to the video data and then apply rank- k SVD, rank- $2k$ SVD, and the geodesic model to denoise the noisy version of video with $k = 10$ and $\ell = 4$. (c) Visual example of denoising frame 125 in the Curtain video sequence with AWGN of $\sigma = 110$. The geodesic model was able to denoise the noisy image more effectively than SVD.

spaced t_i values seems reasonable. The measurements are dynamic, high dimensional, and show redundant anatomical structure. As is common for image subspace models, we model a spatial patch of data.

Figure 7 shows the loss of applying the proposed method for a variety of ranks. Similar to Figure 5, we show a comparison to rank- k and rank- $2k$ SVD and the proposed method on the data permuted. From this figure, we can see that the OSSI data appears to be well modeled by a rank 1 geodesic; the proposed method with rank 1 performs similarly to rank- k SVD and permuting the data significantly increases the loss to that of rank- k SVD.

3.3 Video denoising

In this section we apply the geodesic data model for a video denoising application. The video sequence used in this experiment is the first 260 frames of the Curtain video dataset (Li et al., 2004, Section V-A). The video sequence has variations due to a moving curtain and a person entering the scene at the end of the video. See more details and results for other videos in Appendix B.3.

We start by showing that the geodesic model is a good choice for this video data. Along similar lines as the fMRI data, we achieve this goal by computing loss (representation error) for approximating the video using rank- k SVD, rank- $2k$ SVD, the proposed geodesic method, and applying the geodesic method after reordering/permuting the frames in video sequence. Again, the rationale behind permuting the frames is that if there is no temporal correlation in the frames then permuting the data would not have any negative impact on the loss. Figure 8 shows the training loss as a function of the rank k and the PSNR of the denoised video as a function of added noise level. The training loss for the geodesic model lies in between k and $2k$ cases, similar to the simulated data. More importantly, applying the geodesic model to permuted data degrades the loss, confirming that there is temporal correlation for the geodesic model to exploit. We add additive white Gaussian noise (AWGN) with different values of standard deviation σ to the video sequence and applied rank- k SVD, rank- $2k$ SVD, and the proposed geodesic subspace model to denoise the video sequence. The quality of the denoised image is measured using the peak signal to noise ratio (PSNR), defined as:

$$\text{PSNR} = 20 \log_{10} \left(\frac{255}{\frac{1}{\sqrt{d \times \ell}} \|\mathbf{X}_i - \hat{\mathbf{X}}_i\|_F} \right).$$

Figure 8 shows that rank- $2k$ SVD has the worst denoising performance (due to overfitting the very large amount of noise) while the proposed geodesic method has the best performance in the noisiest regime. For visual evidence, Figure 8c illustrates the denoising performance of rank- k SVD and geodesic model

for frame 125 in the sequence when noise of $\sigma = 110$ is added. Again, similar to the simulated data, the superior denoising performance of geodesic model is evident.

4 Conclusion, Discussion, and Future Work

This work proposed a model and algorithm for dynamic subspace estimation, or batch-computed subspace tracking. The proposed method is sample efficient and the optimization requires no hyperparameters beyond the assumed rank of the data. The model and method are applicable to real data, as shown on dynamic fMRI data and video data, and the single geodesic we studied here represents a major building block for a very general piecewise geodesic model. While we can apply the proposed methods on chunks of data to learn a series of geodesics, we leave it as future work to develop a more efficient way to learn a continuous piecewise geodesic and its change points.

The proposed method also requires knowledge of the sample time points t_i . This is reasonable for temporal data with actual sample times, but there may be problems where we do not know the sample times or we want to model an unknown varying velocity across the geodesic. In these cases, we would need a method to estimate suitable t_i which we leave for future work.

The time complexity of our algorithm is dominated by the computation of the $d \times 2k$ SVD in the $\mathbf{H}$, $\mathbf{Y}$ update each iteration. It is possible that a tighter majorizer could increase the efficiency of this update. We leave it as future work to further accelerate the algorithm.

The loss function proposed here is nonconvex. Major open questions of interest are therefore understanding the landscape properties of this loss function, algorithmic convergence to stationary points or minimizers, and algorithmic initialization. Figure 2 exhibits some brighter patches in the $T > 2k$ regime where a few instances appear to have converged to poor local minima. These instances seem relatively uncommon for the geodesics considered here. We leave it as future work to develop theory for the loss landscape, theoretical bounds for geodesic recovery, and related initialization techniques.

Finally, subspace tracking is often applied to problems where data points are modeled as they arrive, e.g., in array processing and communications. It would therefore also be of great interest to develop a streaming algorithm for the piecewise geodesic model.

5 Acknowledgements

The authors thank Shouchang Guo and Doug Noll for sharing the dynamic fMRI data. This work was supported in part by AFOSR YIP award FA9550-19-1-0026, NSF Grant IIS 1838179, and NIH Grants U01 EB026977 and R01 EB023618.

References

- Absil, P.-A., Mahony, R., and Sepulchre, R. (2004). Riemannian geometry of grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematica*, 80(2):199–220.
- Allen-Zhu, Z. and Li, Y. (2017). Follow the compressed leader: faster online learning of eigenvectors and faster mmwu. In *Proc. 34th Int. Conf. Mach. Learning*, pages 116–125. JMLR. org.
- Balzano, L. (2022). On the equivalence of oja’s algorithm and grouse. In *International Conference on Artificial Intelligence and Statistics*, pages 7014–7030. PMLR.
- Balzano, L., Chi, Y., and Lu, Y. (2018). Streaming PCA and subspace tracking: The missing data case. *Proc. of IEEE Special Issue on Rethinking PCA for Modern Datasets: Theory, Algorithms, and Applications*.
- Balzano, L., Nowak, R., and Recht, B. (2010). Online identification and tracking of subspaces from highly incomplete information. In *2010 48th Annual allerton conference on communication, control, and computing (Allerton)*, pages 704–711. IEEE.
- Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2010). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. & Trends in Machine Learning*, 3(1):1–122.
- Breloy, A., Kumar, S., Sun, Y., and Palomar, D. P. (2021). Majorization-minimization on the Stiefel manifold with application to robust sparse PCA. *IEEE Trans. Sig. Proc.*, 69:1507–20.

- Bunch, J. R. and Nielsen, C. P. (1978). Updating the singular value decomposition. *Numerische Mathematik*, 31(2):111–129.
- Chi, Y., Eldar, Y. C., and Calderbank, R. (2013). Petrels: Parallel subspace estimation and tracking by recursive least squares from partial observations. *IEEE Trans. Signal Process.*, 61(23):5947–5959.
- Comon, P. and Golub, G. H. (1990). Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE*, 78(8):1327–1343.
- Edelman, A., Arias, T. A., and Smith, S. T. (1998). The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix. Anal. Appl.*, 20(2):303–53.
- Fessler, J. A. (2016). Michigan image reconstruction toolbox (MIRT) for Matlab. Available from <http://web.eecs.umich.edu/~fessler>.
- Fuhrmann, D. R. (1997). A geometric approach to subspace tracking. In *Asilomar Conf. Signals, Syst. and Comput.*, volume 1, pages 783–787. IEEE.
- Funai, A. K., Fessler, J. A., Yeo, D. T. B., Olafsson, V. T., and Noll, D. C. (2008). Regularized field map estimation in MRI. *IEEE Trans. Med. Imag.*, 27(10):1484–94.
- Golub, G. and Pereyra, V. (2003). Separable nonlinear least squares: the variable projection method and its applications. *Inverse Prob.*, 19(2):R1–26.
- Guo, S. and Noll, D. C. (2020). Oscillating steady-state imaging (OSSI): A novel method for functional MRI. *Mag. Res. Med.*, 84(2):698–712.
- Haghighatshoar, S. and Caire, G. (2018). Low-complexity massive MIMO subspace estimation and tracking from low-dimensional projections. *IEEE Trans. Signal Process.*, 66(7):1832–1844.
- Hong, Y., Kwitt, R., Singh, N., Vasconcelos, N., and Niethammer, M. (2016). Parametric regression on the grassmannian. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(11):2284–2297.
- Huang, D., Niles-Weed, J., and Ward, R. (2021). Streaming k-pca: Efficient guarantees for oja’s algorithm, beyond rank-one updates. In *Conference on Learning Theory*, pages 2463–2498. PMLR.
- Lake, D. E. and Keenan, D. (1998). Maximum likelihood estimation of geodesic subspace trajectories using approximate methods and stochastic optimization. In *IEEE Signal Process. Workshop Statis. Signal and Array Process.*, pages 148–151. IEEE.
- Li, L., Huang, W., Gu, I. Y.-H., and Tian, Q. (2004). Statistical modeling of complex backgrounds for foreground object detection. *IEEE Trans. Image Process.*, 13(11):1459–1472.
- Narayanamurthy, P. and Vaswani, N. (2018). Provable dynamic robust PCA or robust subspace tracking. *IEEE Trans. Inform. Theory*, 65(3):1547–1577.
- Oja, E. (1982). Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3):267–273.
- Otazo, R., Candès, E., and Sodickson, D. K. (2015). Low-rank plus sparse matrix decomposition for accelerated dynamic MRI with separation of background and dynamic components. *Magnetic Resonance in Medicine*, 73(3):1125–1136.
- Srivastava, A. and Klassen, E. (2004). Bayesian and geometric subspace tracking. *Advances in Applied Probability*, 36(1):43–56.
- Sun, Y., Babu, P., and Palomar, D. P. (2017). Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Trans. Sig. Proc.*, 65(3):794–816.
- Uecker, M., Lai, P., Murphy, M. J., Virtue, P., Elad, M., Pauly, J. M., Vasanawala, S. S., and Lustig, M. (2014). ESPIRiT—an eigenvalue approach to autocalibrating parallel MRI: Where SENSE meets GRAPPA. *Mag. Res. Med.*, 71(3):990–1001.
- Vaswani, N., Bouwmans, T., Javed, S., and Narayanamurthy, P. (2018). Robust subspace learning: Robust pca, robust subspace tracking, and robust subspace recovery. *IEEE Signal Process. Mag.*, 35(4):32–55.
- Yang, B. (1995). Projection approximation subspace tracking. *IEEE Trans. Signal process.*, 43(1):95–107.

Supplementary Material for “Dynamic Subspace Estimation with Grassmannian Geodesics”

A Additional Algorithmic Derivations and Details

A.1 Derivation of (H, Y) Update as Majorize Minimize Step

Let $\mathbf{Q} \triangleq [\mathbf{H} \ \mathbf{Y}]$ and $\mathbf{Z}_i \triangleq [\cos(\boldsymbol{\Theta}t_i); \sin(\boldsymbol{\Theta}t_i)]$. Then our model can be written as

$$\mathbf{U}_i = \mathbf{H} \cos(\boldsymbol{\Theta}t_i) + \mathbf{Y} \sin(\boldsymbol{\Theta}t_i) = \mathbf{Q} \mathbf{Z}_i. \quad (18)$$

To form a linear majorizer for loss with respect to $\mathbf{Q}$, we first derive its unconstrained gradient

$$\mathcal{L}(\mathbf{Q}) = - \sum_{i=1}^T \|\mathbf{X}_i^\top \mathbf{Q} \mathbf{Z}_i\|_F^2 + c = - \sum_{i=1}^T \text{trace} \left\{ \mathbf{Q}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Q} \mathbf{Z}_i \mathbf{Z}_i^\top \right\} + c \quad (19)$$

$$\nabla_{\mathbf{Q}} \mathcal{L}(\bar{\mathbf{Q}}) = - \sum_{i=1}^T 2 \mathbf{X}_i \mathbf{X}_i^\top \bar{\mathbf{Q}} \mathbf{Z}_i \mathbf{Z}_i^\top. \quad (20)$$

We can form a linear majorizer for the loss

$$g(\mathbf{Q}; \bar{\mathbf{Q}}) \triangleq \text{trace} \left\{ \mathbf{Q}^\top \nabla_{\mathbf{Q}} \mathcal{L}(\bar{\mathbf{Q}}) \right\} + c \geq \mathcal{L}(\bar{\mathbf{Q}}). \quad (21)$$

Note that $g(\bar{\mathbf{Q}}; \bar{\mathbf{Q}}) = \mathcal{L}(\bar{\mathbf{Q}})$, and it is linear and continuous. The above inequality only needs to hold for $\mathbf{Q} \in \mathcal{V}^{d \times 2k}$. It currently holds for $\mathbb{R}^{d \times 2k}$ so there is room for a tighter majorizer.

Following the work of Breloy et al. (2021), we can write $g(\mathbf{Q}; \bar{\mathbf{Q}}) = -2 \text{trace} \left\{ \mathbf{Q}^\top R(\bar{\mathbf{Q}}) \right\} + c$ for matrix function $R(\bar{\mathbf{Q}}) = \sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \bar{\mathbf{Q}} \mathbf{Z}_i \mathbf{Z}_i^\top$. We can then minimize the linear majorizer g with a Stiefel manifold constraint simply by projecting its negative gradient $R(\bar{\mathbf{Q}})$ onto the Stiefel manifold. The update is then given by

$$\hat{\mathbf{Q}} = \arg \min_{\mathbf{Q} \in \mathcal{V}^{d \times 2k}} \|\mathbf{R}(\bar{\mathbf{Q}}) - \mathbf{Q}\|_F^2 \quad (22)$$

$$= \arg \min_{\mathbf{Q} \in \mathcal{V}^{d \times 2k}} \left\| \left(\sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \bar{\mathbf{Q}} \mathbf{Z}_i \mathbf{Z}_i^\top \right) - \mathbf{Q} \right\|_F^2 \quad (23)$$

$$= \mathbf{W} \mathbf{V}^\top, \quad (24)$$

$$\begin{aligned} \text{where } \mathbf{W} \boldsymbol{\Sigma} \mathbf{V}^\top &= \sum_{i=1}^T \mathbf{X}_i \mathbf{X}_i^\top \bar{\mathbf{Q}} \mathbf{Z}_i \mathbf{Z}_i^\top \\ &= \sum_{i=1}^T \begin{bmatrix} \mathbf{X}_i \hat{\mathbf{G}}_i^\top \cos(\boldsymbol{\Theta}t_i) & \mathbf{X}_i \hat{\mathbf{G}}_i^\top \sin(\boldsymbol{\Theta}t_i) \end{bmatrix}, \end{aligned}$$

where in the last line we have let $\hat{\mathbf{G}}_i^\top = \mathbf{X}_i^\top \mathbf{U}_i^{(n)} = \mathbf{X}_i^\top \bar{\mathbf{Q}} \mathbf{Z}_i$.

We can derive this same update as a block coordinate update on $\mathbf{Q}$ and $\{\mathbf{G}_i\}_{i=1}^T$. We start with our loss (5) without projecting out $\{\mathbf{G}_i\}_{i=1}^T$ and substitute our geodesic model (2). The $(\mathbf{H}, \mathbf{Y})$ update with fixed $\boldsymbol{\Theta}$ and $\{\mathbf{G}_i\}_{i=1}^T$ can be minimized by recognizing it as a generalized Procrustes problem

$$\hat{\mathbf{H}}, \hat{\mathbf{Y}} = \arg \min_{\mathbf{H}, \mathbf{Y} \in \mathcal{V}^{d \times k}, \mathbf{H}^\top \mathbf{Y} = \mathbf{0}} \sum_{i=1}^T \|\mathbf{X}_i - (\mathbf{H} \cos(\boldsymbol{\Theta}t_i) + \mathbf{Y} \sin(\boldsymbol{\Theta}t_i)) \mathbf{G}_i\|_F^2 \quad (25)$$

$$[\hat{\mathbf{H}} \ \hat{\mathbf{Y}}] = \arg \min_{[\mathbf{H} \ \mathbf{Y}] \in \mathcal{V}^{d \times 2k}} \sum_{i=1}^T \left\| \mathbf{X}_i - [\mathbf{H} \ \mathbf{Y}] \begin{bmatrix} \cos(\boldsymbol{\Theta}t_i) \\ \sin(\boldsymbol{\Theta}t_i) \end{bmatrix} \mathbf{G}_i \right\|_F^2 \quad (26)$$

$$= \mathbf{W} \mathbf{V}^\top, \quad (27)$$

$$\begin{aligned} \text{where } \mathbf{W} \boldsymbol{\Sigma} \mathbf{V}^\top &= \sum_{i=1}^T \mathbf{X}_i \left(\begin{bmatrix} \cos(\boldsymbol{\Theta}t_i) \\ \sin(\boldsymbol{\Theta}t_i) \end{bmatrix} \mathbf{G}_i \right)^\top \\ &= \sum_{i=1}^T \begin{bmatrix} \mathbf{X}_i \mathbf{G}_i^\top \cos(\boldsymbol{\Theta}t_i) & \mathbf{X}_i \mathbf{G}_i^\top \sin(\boldsymbol{\Theta}t_i) \end{bmatrix}. \end{aligned}$$

This Procrustes step involves a single SVD of the $d \times 2k$ matrix shown in the last line. While derived on different losses, Updates (24) and (27) yield the same update for $(\mathbf{H}, \mathbf{Y})$.

A.2 Derivation of Θ Update

We derive a majorize minimize iteration for Θ . First we simplify the loss and highlight the separability of the loss with respect to the diagonal elements of Θ . We then construct majorizers for each term in the simplified loss using a translated Huber majorizer. The update is then given by minimizing the sum of these majorizers.

We note that while we may refer to Θ as arc distances, we do not constrain the elements of Θ to be non-negative. Conceptually, the negative values of Θ represent walking in the opposite direction on the surface of the Grassmannian, *i.e.*, they are *signed* arc distances.

A.2.1 Simplifying the Loss

We start by simplifying the loss

$$\hat{\Theta} = \arg \min_{\Theta} \min_{\{G_i\}} \sum_{i=1}^T \|X_i - (H \cos(\Theta t_i) + Y \sin(\Theta t_i)) G_i\|_F^2 \quad (28)$$

$$= \arg \min_{\Theta} - \sum_{i=1}^T \|X_i^\top (H \cos(\Theta t_i) + Y \sin(\Theta t_i))\|_F^2 \quad (29)$$

$$= \arg \min_{\Theta} - \sum_{i=1}^T \text{trace} \left\{ X_i^\top \left(H \cos^2(\Theta t_i) H^\top + 2 \text{real} \left\{ H \cos(\Theta t_i) \sin(\Theta t_i) Y^\top \right\} + Y \sin^2(\Theta t_i) Y^\top \right) X_i \right\} \quad (30)$$

$$= \arg \min_{\Theta} - \sum_{i=1}^T \text{trace} \left\{ \cos^2(\Theta t_i) H^\top X_i X_i^\top H + 2 \cos(\Theta t_i) \sin(\Theta t_i) \text{real} \left\{ Y^\top X_i X_i H \right\} + \sin^2(\Theta t_i) Y^\top X_i X_i^\top Y \right\} \quad (31)$$

$$= \arg \min_{\Theta} - \sum_{i=1}^T \sum_{j=1}^k \cos^2(\theta_j t_i) \left[H^\top X_i X_i^\top H \right]_{j,j} + 2 \cos(\theta_j t_i) \sin(\theta_j t_i) \left[\text{real} \left\{ Y^\top X_i X_i H \right\} \right]_{j,j} + \sin^2(\theta_j t_i) \left[Y^\top X_i X_i^\top Y \right]_{j,j}. \quad (32)$$

We solve the problem of optimizing Θ by updating each of its diagonal elements θ_j separately. We define the following constants

$$\alpha_{i,j} = [H^\top X_i X_i^\top H]_{j,j} \quad (33)$$

$$\beta_{i,j} = \text{real} \left\{ [Y^\top X_i X_i^\top H]_{j,j} \right\} \quad (34)$$

$$\gamma_{i,j} = [Y^\top X_i X_i^\top Y]_{j,j}. \quad (35)$$

Our optimization problem for each $j = 1, \dots, k$ is now

$$\hat{\theta}_j = \arg \min_{\theta_j} - \sum_{i=1}^T \alpha_{i,j} \cos^2(\theta_j t_i) + 2\beta_{i,j} \cos(\theta_j t_i) \sin(\theta_j t_i) + \gamma_{i,j} \sin^2(\theta_j t_i). \quad (36)$$

Using the following trigonometric identities

$$2 \cos(x) \sin(x) = \sin(2x) \quad (37)$$

$$\cos^2(x) + \sin^2(x) = 1 \quad (38)$$

$$\cos^2(x) = \frac{1}{2} (\cos(2x) + 1) \quad (39)$$

$$a \cos(x) + b \sin(x) = \sqrt{a^2 + b^2} \cos(x - \arctan2(b, a)), \quad (40)$$

we further simplify the loss:

$$\hat{\theta}_j = \arg \min_{\theta_j} - \sum_{i=1}^T \alpha_{i,j} \cos^2(\theta_j t_i) + \beta_{i,j} \sin(2\theta_j t_i) + \gamma_{i,j} \sin^2(\theta_j t_i) \quad (41)$$

$$= \arg \min_{\theta_j} - \sum_{i=1}^T (\alpha_{i,j} - \gamma_{i,j}) \cos^2(\theta_j t_i) + \beta_{i,j} \sin(2\theta_j t_i) + \gamma_{i,j} (\cos^2(\theta_j t_i) + \sin^2(\theta_j t_i)) \quad (42)$$

$$= \arg \min_{\theta_j} - \sum_{i=1}^T (\alpha_{i,j} - \gamma_{i,j}) \frac{1}{2} (\cos(2\theta_j t_i) + 1) + \beta_{i,j} \sin(2\theta_j t_i) + \gamma_{i,j} \quad (43)$$

$$= \arg \min_{\theta_j} - \sum_{i=1}^T r_{i,j} \cos(2\theta_j t_i - \phi_{i,j}) + b_{i,j}, \quad (44)$$

where

$$r_{i,j} = \sqrt{\left(\frac{\alpha_{i,j} - \gamma_{i,j}}{2}\right)^2 + \beta_{i,j}^2} \quad (45)$$

$$\phi_{i,j} = \arctan2\left(\beta_{i,j}, \frac{\alpha_{i,j} - \gamma_{i,j}}{2}\right) \quad (46)$$

$$b_{i,j} = \frac{\alpha_{i,j} + \gamma_{i,j}}{2}. \quad (47)$$

A.2.2 Constructing a majorizer

To majorize our loss function, we construct a quadratic majorizer for each term of the form

$$f_{i,j}(\theta_j) \triangleq -r_{i,j} \cos(2\theta_j t_i - \phi_{i,j}) + b_{i,j}. \quad (48)$$

To be a majorizer at a point $\bar{\theta}_j$, we require $q_{i,j}(\bar{\theta}_j; \bar{\theta}_j) = f_{i,j}(\bar{\theta}_j)$ (equal at the point of construction) and $q_{i,j}(\theta_j; \bar{\theta}_j) \geq f_{i,j}(\theta_j)$ (greater than or equal to the loss everywhere). This can be achieved with a quadratic of the form

$$q_{i,j}(\theta_j; \bar{\theta}_j) = f_{i,j}(\bar{\theta}_j) + \dot{f}_{i,j}(\bar{\theta}_j)(\theta_j - \bar{\theta}_j) + \frac{1}{2} w_{f_{i,j}}(\bar{\theta}_j) (\theta_j - \bar{\theta}_j)^2, \quad (49)$$

where $\dot{f}_{i,j}(\theta_j) = 2r_{i,j} t_i \sin(2\theta_j t_i - \phi_{i,j})$ is the derivative of $f_{i,j}$ and $w_{f_{i,j}}$ is an appropriate curvature (or “weighting”) function. A simple option is $w_{f_{i,j}}(\theta_j) = L_{\dot{f}_{i,j}}$ the Lipschitz constant of the derivative. Minimizing the resulting majorizer yields the standard fixed step size gradient descent algorithm. A tighter majorizer will touch our original function at two or more points. Note that $\frac{\phi_{i,j}}{2t_i}$ is a minimizer and our function is symmetric and quasi-convex on the interval $\left[\frac{\phi_{i,j} - \pi}{2t_i}, \frac{\phi_{i,j} + \pi}{2t_i}\right]$ about this point. Our approach will be to construct a curvature function $\bar{w}_{f_{i,j}}$ for points in this interval and periodically extend it to construct the final curvature function $w_{f_{i,j}}$. Because $f_{i,j}$ is symmetric about $\frac{\phi_{i,j}}{2t_i}$, our majorizer will touch at two points when the axis (and minimizer) of $q_{i,j}$ is $\frac{\phi_{i,j}}{2t_i}$, equivalently when its gradient at this point equals zero

$$\dot{q}_{i,j}\left(\frac{\phi_{i,j}}{2t_i}; \bar{\theta}_j\right) = 0. \quad (50)$$

Solving for $\bar{w}_{f_{i,j}}(\bar{\theta}_j)$ yields

$$\bar{w}_{f_{i,j}}(\bar{\theta}_j) = \frac{\dot{f}_{i,j}(\bar{\theta}_j)}{\bar{\theta}_j - \frac{\phi_{i,j}}{2t_i}}, \quad (51)$$

which can be recognized as a (translated) Huber curvature function. For the case when $\bar{\theta}_j = \frac{\phi_{i,j}}{2t_i}$, we define $w_{f_{i,j}}\left(\frac{\phi_{i,j}}{2t_i}\right) = 4t_i^2 r_{i,j}$, which is its limit point.

Forming $w_{f_{i,j}}$ by periodically extending $\bar{w}_{f_{i,j}}$ only requires periodically extending the denominator, since the numerator is already periodic. The resulting periodic version of the curvature function is

$$w_{f_{i,j}}(\bar{\theta}_j) = \begin{cases} \frac{\dot{f}_{i,j}(\bar{\theta}_j)}{\text{mod}\left(\bar{\theta}_j - \frac{\phi_{i,j}}{2t_i}, \frac{2\pi}{2t_i}\right) - \frac{\pi}{2t_i}} & \bar{\theta}_j \neq \frac{\phi_{i,j} + 2\pi m}{2t_i}, m \in \mathbb{Z} \\ 4t_i^2 r_{i,j} & \bar{\theta}_j = \frac{\phi_{i,j} + 2\pi m}{2t_i}, m \in \mathbb{Z}. \end{cases} \quad (52)$$

The final majorizer for our loss function of θ_j is then

$$q_j(\theta_j; \bar{\theta}_j) = \sum_{i=1}^T q_{i,j}(\theta_j; \bar{\theta}_j). \quad (53)$$

Figure 9 shows an example loss and the constructed majorizer.

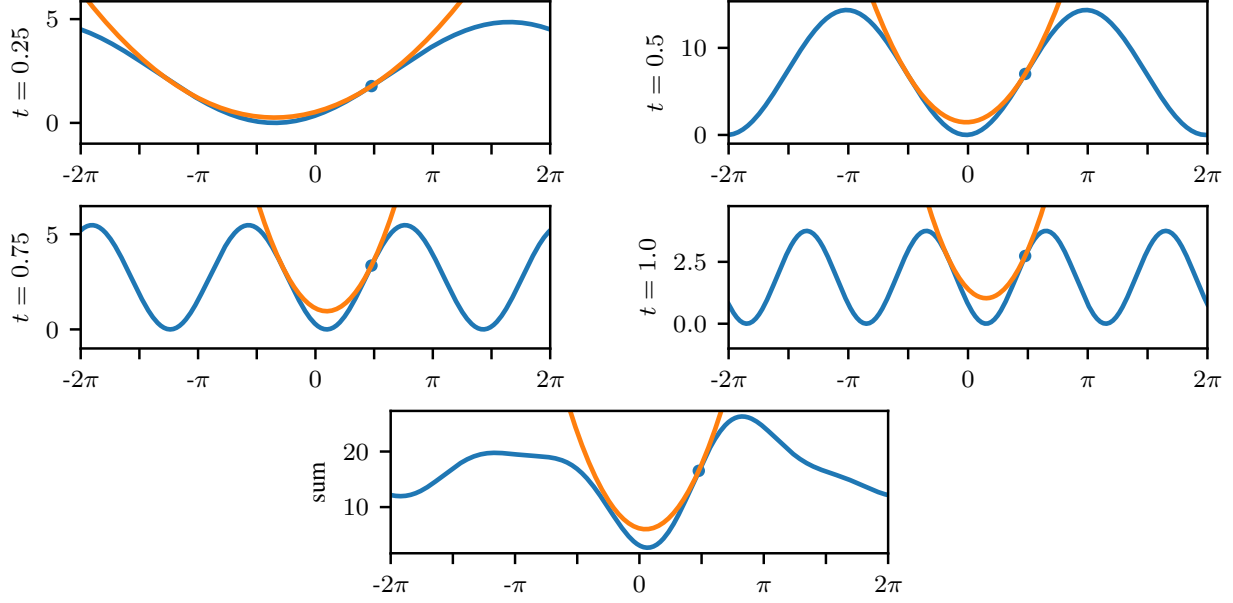


Figure 9: An example of four cosines (top two rows, blue) that sum to form the (nonconvex) loss for a single θ_j (bottom row, blue). For each cosine function, we construct a quadratic majorizer (top two rows, orange) at a point θ_j (blue dot). The sum of these individual quadratic majorizers form a quadratic majorizer for the loss (bottom, orange) that has a closed-form minimizer. Although the loss is non-convex, distance-minimizing geodesics will have $\theta_j \in [-\pi/2, \pi/2]$. On this interval, the loss is often well-behaved (here, quasi-convex).

A.2.3 Minimizing the Majorizer

Because the majorizer is a sum of one-dimensional quadratics, we minimize it by setting its derivative to zero and solving for θ_j

$$\dot{q}_j(\theta_j; \bar{\theta}_j) = \sum_{i=1}^T \dot{q}_{i,j}(\theta_j; \bar{\theta}_j) \quad (54)$$

$$0 = \sum_{i=1}^T \dot{f}_{i,j}(\bar{\theta}_j) + w_{f_{i,j}}(\bar{\theta}_j)(\theta_j - \bar{\theta}_j) = \left(\sum_{i=1}^T \dot{f}_{i,j}(\bar{\theta}_j) \right) + \left(\sum_{i=1}^T w_{f_{i,j}}(\bar{\theta}_j) \right) (\theta_j - \bar{\theta}_j) \quad (55)$$

$$\theta_j = \bar{\theta}_j - \frac{\sum_{i=1}^T \dot{f}_{i,j}(\bar{\theta}_j)}{\sum_{i=1}^T w_{f_{i,j}}(\bar{\theta}_j)}. \quad (56)$$

Iteratively constructing majorizers and minimizing them yields the following descent scheme

$$\theta_j^{(n+1)} = \theta_j^{(n)} - \frac{\sum_{i=1}^T \dot{f}_{i,j}(\theta_j^{(n)})}{\sum_{i=1}^T w_{f_{i,j}}(\theta_j^{(n)})}. \quad (57)$$

B Additional Experiments and Details

B.1 Synthetic Experiments - Details and Further Investigation

Rank-1 subspace in $\mathbb{R}^2$ To provide some simple intuition for our problem, we present the algorithm applied to learning a rank-1 subspace in two dimensions. In this special case, Θ is a scalar θ , and $\mathbf{H}$ and $\mathbf{Y}$ can be parameterized by a single scalar rotation ω (up to a sign flip in $\mathbf{Y}$, which we can absorb into θ). Then $\mathbf{H} = [\cos(\omega); \sin(\omega)]$ and $\mathbf{Y} = [-\sin(\omega); \cos(\omega)]$. Our loss (5) then simplifies to a two-dimensional function of θ and ω

$$\mathcal{L}(\mathbf{U}) = - \sum_{i=1}^T r_{i,1} \cos(2\theta t_i - \phi_{i,1} + 2\omega) + b_{i,1} + c, \quad (58)$$

Algorithm 1 Geodesic Subspace Estimation

Require: $\{\mathbf{X}_i, t_i\}_{i=1}^T$, $\mathbf{H}^{(0)}, \mathbf{Y}^{(0)}, \boldsymbol{\Theta}^{(0)}$, $N = \#$ of outer iterations, $M = \#$ of inner MM iterations
for $n = 1, \dots, N$ **do**
 # $\mathbf{H}, \mathbf{Y}$ update
 $\mathbf{U}_i^{(n-1)} = \mathbf{H}^{(n-1)} \cos(\boldsymbol{\Theta}^{(n-1)} t_i) + \mathbf{Y}^{(n-1)} \sin(\boldsymbol{\Theta}^{(n-1)} t_i) \quad \forall i$
 $\mathbf{G}_i = (\mathbf{U}_i^{(n-1)})^\top \mathbf{X}_i \quad \forall i$
 $\mathbf{M} = \sum_{i=1}^T [\mathbf{X}_i \mathbf{G}_i^\top \cos(\boldsymbol{\Theta}^{(n-1)} t_i) \quad \mathbf{X}_i \mathbf{G}_i^\top \sin(\boldsymbol{\Theta}^{(n-1)} t_i)]$
 $\mathbf{W}, \boldsymbol{\Sigma}, \mathbf{V}^\top = \text{SVD}(\mathbf{M})$
 $[\mathbf{H}^{(n)} \quad \mathbf{Y}^{(n)}] = \mathbf{W} \mathbf{V}^\top$
 # $\boldsymbol{\Theta}$ update
 $\boldsymbol{\Theta}^{(n,0)} = \boldsymbol{\Theta}^{(n-1)}$
 for $j = 1, \dots, k$ **do**
 $\alpha_{i,j} = [\mathbf{H}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{H}]_{j,j} \quad \forall i$
 $\beta_{i,j} = \text{real}\{[\mathbf{Y}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{H}]_{j,j}\} \quad \forall i$
 $\gamma_{i,j} = [\mathbf{Y}^\top \mathbf{X}_i \mathbf{X}_i^\top \mathbf{Y}]_{j,j} \quad \forall i$
 $\phi_{i,j} = \arctan2\left(\beta_{i,j}, \frac{\alpha_{i,j} - \gamma_{i,j}}{2}\right) \quad \forall i$
 $r_{i,j} = \sqrt{\left(\frac{\alpha_{i,j} - \gamma_{i,j}}{2}\right)^2 + \beta_{i,j}^2} \quad \forall i$
 for $m = 1, \dots, M$ **do**
 $z_j = \sum_{i=1}^T 2r_{i,j} t_i \sin(2\theta_j^{(n,m-1)} t_i - \phi_{i,j})$
 $w_j = \sum_{i=1}^T \frac{2r_{i,j} t_i \sin(2\theta_j^{(n,m-1)} t_i - \phi_{i,j})}{\text{mod}\left((\theta_j^{(n,m-1)} - \frac{\phi_{i,j}}{2t_i}) + \frac{\pi}{2t_i}, \frac{2\pi}{2t_i}\right) - \frac{\pi}{2t_i}}$
 $\theta_j^{(n,m)} = \theta_j^{(n,m-1)} - \frac{z_j}{w_j}$
 end for
 end for
 $\boldsymbol{\Theta}^{(n)} = \boldsymbol{\Theta}^{(n,M)}$
end for

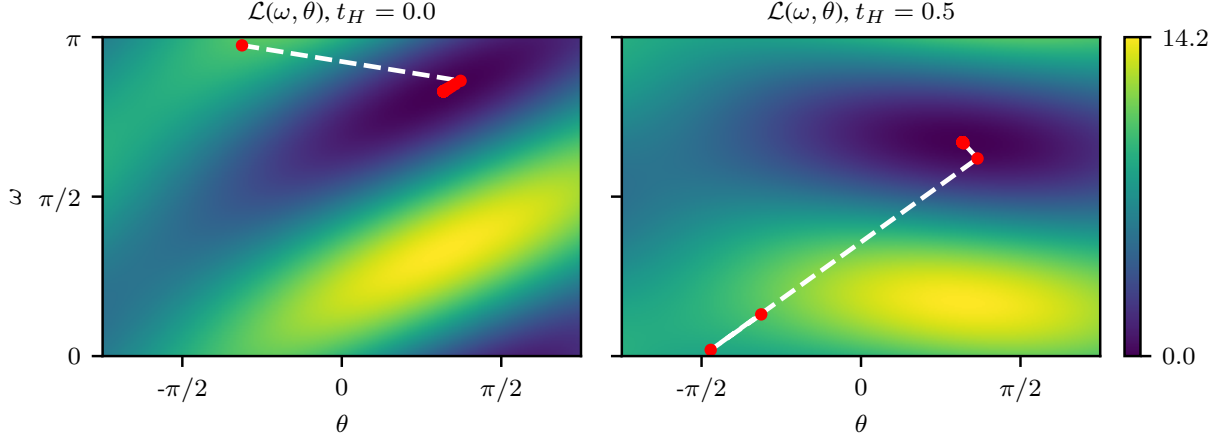


Figure 10: The loss function for learning a rank-1 geodesic in 2D, where we have made the transformation $\mathbf{H} = [\cos(\omega); \sin(\omega)]$ and $\mathbf{Y} = [-\sin(\omega); \cos(\omega)]$. Note that the y-axes are π -periodic. (Left) The unaltered loss function and associated algorithm iterates (red). The iterates approach the minimum slowly, as the structure of the loss is not well aligned with the coordinate directions. (Right) The loss after making the transformation $\tilde{t}_i = t_i - t_{\mathbf{H}}$ for $t_{\mathbf{H}} = 0.5$. The iterates fully converge in only a few iterations. Both sets of iterations are initialized at equivalent points.

where $r_{i,1}, \phi_{i,1}, b_{i,1}$ are defined as previously, but with $[\mathbf{H}\mathbf{Y}] = \mathbf{I}$ and c is the same constant from (5).

Figure 10 (left) shows this loss function on some noisy synthetic data with iterates of the proposed algorithm shown in red. Many iterations take small steps as the minimizers of θ and ω are very interdependent.

Intuitively, this behavior may be because the optimal starting subspace $\mathbf{H}$ is very dependent on the arc length θ of the geodesic, as the method will naturally want to center the geodesic to minimize error. If we instead parameterized our geodesic by its center subspace, the resulting $\mathbf{H}$ minimizer would hopefully be more independent of arc length. This can be done by applying the proposed method after first transforming the time points by letting $\tilde{t}_i = t_i - t_{\mathbf{H}}$, where $t_{\mathbf{H}} \in [0, 1]$ is the point along the geodesic equal to $\mathbf{H}$. Figure 10 (right) shows the loss and associated iterates of setting $t_{\mathbf{H}} = 1/2$. The proposed algorithm converges in only a few iterations.

Comparison to streaming subspace estimation methods Here we give more details about Figure 6, which compares the proposed geodesic algorithm to several online subspace tracking algorithms, including GROUSE (Balzano et al., 2010), Oja’s algorithm Oja (1982); Huang et al. (2021), Incremental SVD (ISVD) (Bunch and Nielsen, 1978; Balzano et al., 2018), and PETRELS (Chi et al., 2013). We used the code for these algorithms provided by the authors of Balzano et al. (2018). We used $d = 20, k = 2, \ell = 1$ and two noise levels, $\sigma = 10^{-1}, 10^{-5}$. The proposed method consistently shows better performance (lower error) on geodesic data.

Each of these comparison methods performs subspace updates in a streaming way, one vector at a time, and so their subspace estimate is time-varying. However, without an explicit time-varying model, there is no natural way to pass over the data multiple times and improve the estimate for each time point, and so this comparison only uses one pass over the data for the streaming algorithms.

For Oja’s algorithm, we used step-size 0.1, and we can see it requires several vectors before it begins to converge. For the GROUSE algorithm, we used the “greedy step-size” that is default in the implementation provided for Balzano et al. (2018). For this step-size choice, GROUSE performs quite poorly with high noise but converges somewhat close to the true subspace with a small amount of noise. We note that Oja’s algorithm and GROUSE have been shown to be equivalent Balzano (2022) for certain step-sizes, so these two lines are essentially showing behavior for different choices of step-size on the same algorithm.

The ISVD algorithm performs the incremental singular value decomposition with truncation of the smallest singular values at every step. It therefore quickly estimates the subspace at early iterations, but then drifts from the true subspace since it is not adapting to the new data. The PETRELS algorithm, for which we used a forgetting factor of 0.8, finds a very nice balance between quick estimation of the early subspace followed by adaptation.

Empirical Error Convergence Figure 11 shows the geodesic error per iteration of the proposed algorithm applied to several synthetically generated planted models. Generally, the proposed algorithm for rank-1 geodesics

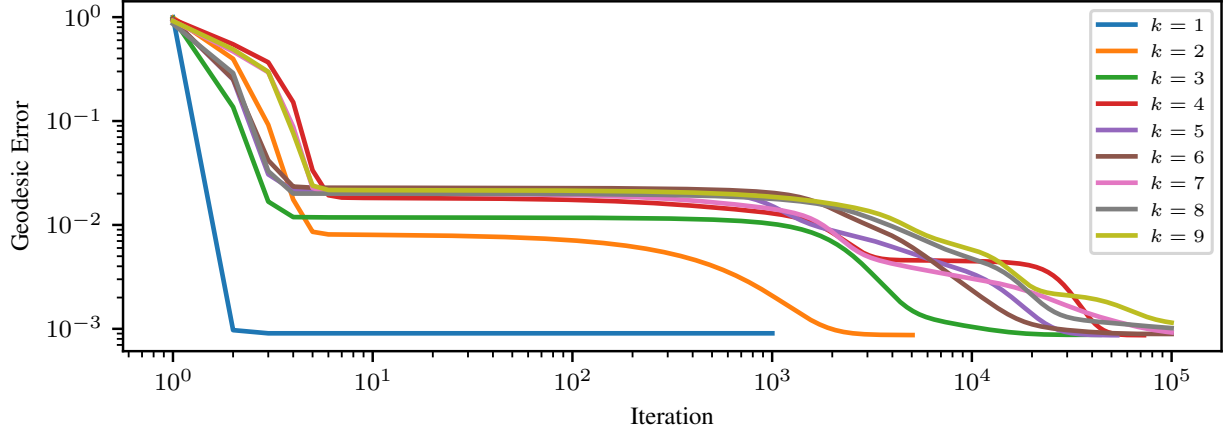


Figure 11: Convergence of the proposed algorithm in geodesic error to the true geodesic for nine planted geodesic models with varying ranks in $\mathbb{R}^{40}$. The planted models were used to generate 100 sample points ($T = 100, \ell = 1$) and AWGN with standard deviation $\sigma = 10^{-3}$ was added. The algorithm was initialized with a random geodesic.

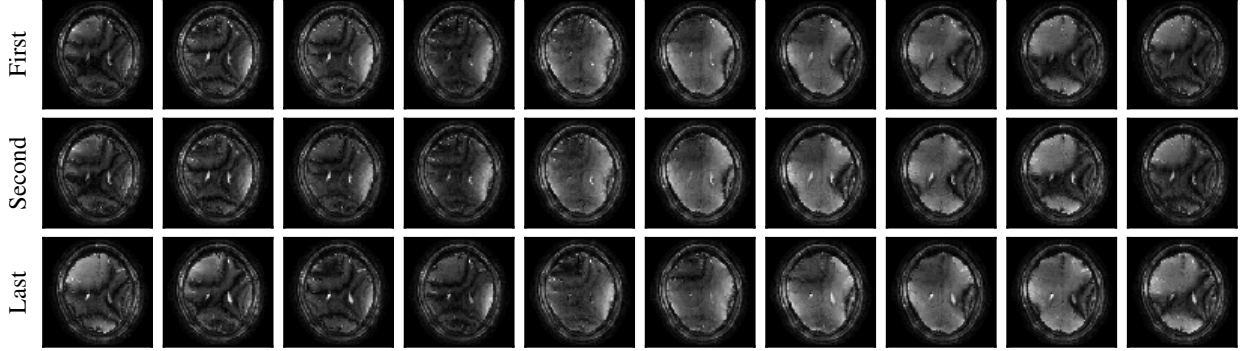


Figure 12: A sample of OSSI acquisitions. Each column, referred to as fast time, represents a certain set of acquisition parameters. Each row, referred to as slow time, is a complete cycle through these acquisitions, which is done many times. Here we only show the first, second, and last slow time set. We see that there is little difference between neighboring slow time points, but that over the course of the scan they change more significantly.

converges in only a few iterations, while larger k requires an increasing number of iterations to converge. These experiments were initialized randomly, but were still able to recover the true geodesic with error at the level of the additive noise. We occasionally see the algorithm converge to poor local minima and fail to recover the true geodesic. We leave it a future work to make the algorithm more robust to these instances and to provide theoretical bounds on geodesic recovery.

B.2 OSSI Dynamic fMRI Dataset Details

The OSSI dynamic fMRI dataset was acquired on a 3T GE MR750 scanner with a 32-channel head coil and is comprised of 167 slow time, 10 fast time and 128×128 spatial samples. The complex data was fully sampled with a variable-density spiral trajectory with $n_i = 8$ interleaves, a densely sampled core, and spiral-out readouts. Detailed OSSI acquisition parameters (TR, n_c , TE, and flip angle) can be found in Guo and Noll (Guo and Noll, 2020). The volunteer was given a left versus right reversing-checkerboard visual stimulus (20 s Left / 20 s Right $\times$ 5 cycles) for 200 s in total. For reconstruction, the k-space data was compressed to 16 virtual coils and ESPIRiT SENSE maps were generated using the BART toolbox (Uecker et al., 2014). Finally, the images were reconstructed using conjugate gradient SENSE with a Huber potential via the MIRT toolbox (Fessler, 2016). Figure 12 shows the

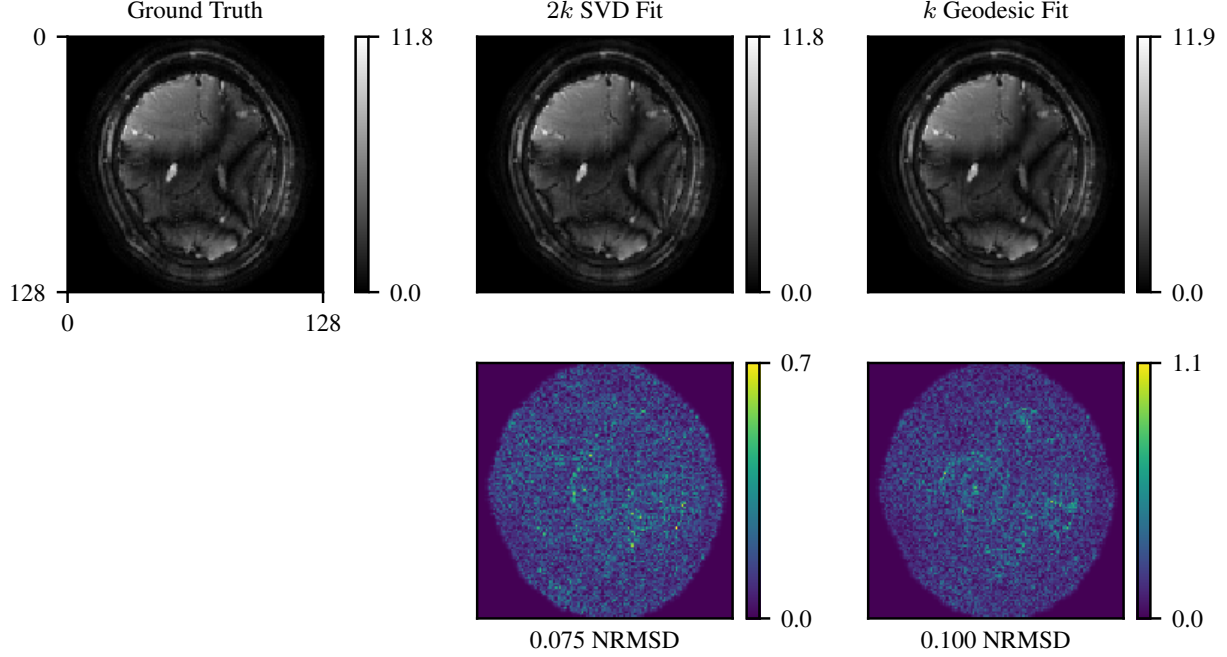


Figure 13: Comparison of OSSI magnitude images reconstructed with a static $2k$ subspace *vs.* the geodesic model for a particular slow and fast time point (See Figure 7). Despite being more temporally constrained, the proposed method fits nearly as well.

magnitude of a sample of the reconstructed images.

B.2.1 Additional Figures for OSSI data

Figure 13 provides comparison to ground truth and rank- $2k$ SVD for the geodesic OSSI reconstruction presented in Figure 7 on a single slow and fast time point. The proposed geodesic model fits nearly as well despite being a more constrained model.

Piecewise Geodesic While the OSSI data can be modeled well by a rank-1 geodesic, Figure 14 provides some initial evidence that a piecewise rank-1 geodesic may be a better model. In this figure, we first modeled it with a single rank-1 geodesic (as in the main body of the paper), and plotted value of the loss function at each data point $\mathbf{X}_i$ (on the left). From this we can see that the loss varies in a W shape across the data points, which may indicate that a piecewise geodesic is more appropriate. Then we tried two rank-1 geodesics, fitting the first half and second half of the data to separate rank-1 geodesics. When we plot the loss for this model, the shape of the loss is more flat. A more detailed investigation of methods for identifying knots in a piecewise geodesic model is of great interest for future work.

Even when we know the knots of a piecewise geodesic model, applying the described geodesic estimation algorithm on each segment individually is not guaranteed to produce a continuous piecewise geodesic. If we let $\mathbf{U}_j(t)$ represent the j th geodesic segment, then we would like $\text{span}(\mathbf{U}_j(1)) = \text{span}(\mathbf{U}_{j+1}(0))$, or equivalently $\mathbf{U}_j(1)\mathbf{U}_j(1)^\top = \mathbf{U}_{j+1}(0)\mathbf{U}_{j+1}(0)^\top$, $\forall j = 1 \dots J-1$. Combining this constraint with our loss yields the following optimization problem for piecewise geodesics

$$\begin{aligned} \min_{\mathbf{U}_j} & - \sum_{j=1}^J \sum_{i=1}^T \left\| \mathbf{X}_{j,i}^\top \mathbf{U}_j(t_i) \right\|_F^2 \\ \text{s.t. } & \mathbf{U}_j(1)\mathbf{U}_j(1)^\top = \mathbf{U}_{j+1}(0)\mathbf{U}_{j+1}(0)^\top \quad \forall j = 1 \dots J-1. \end{aligned} \quad (59)$$

We can simplify this problem by relaxing the constraint to a penalty. If we optimize the loss using a BCD

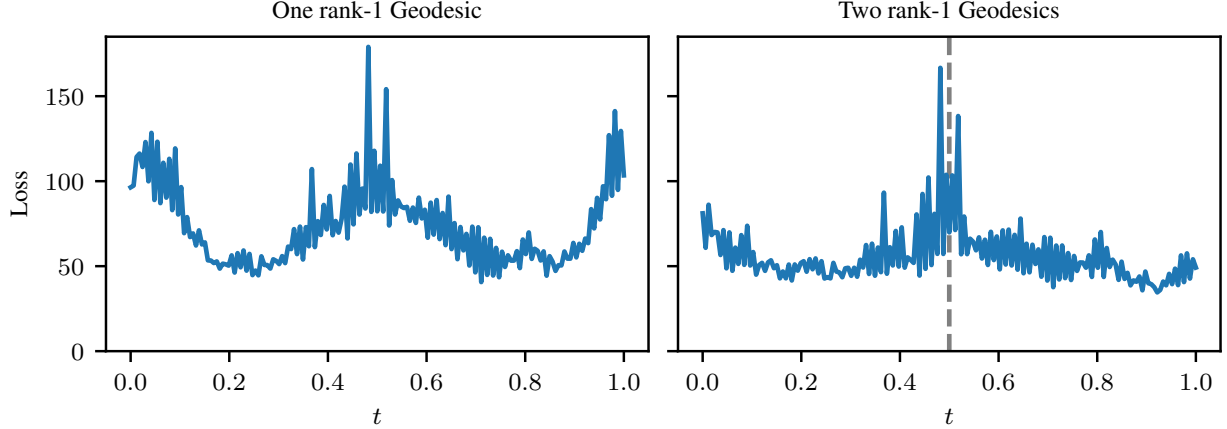


Figure 14: **(Left)** The loss broken down for each time point t for fitting a rank-1 geodesic on OSSI fMRI data. In addition to a seemingly noisy component, the loss also exhibits systematic variations. We hypothesize that these systematic variations result from the limitations of a single geodesic to fit a curve. **(Right)** Instead of fitting a single rank-1 geodesic, we model the data as a piecewise geodesic by fitting a rank-1 geodesic on each half of the data. The resulting loss retains the noisy variations from the single geodesic model, but removes most of the systematic variation.

approach on each segment, then the penalized loss for the j th segment is

$$\min_{\mathbf{U}_j} - \sum_{i=1}^T \left\| \mathbf{X}_i^\top \mathbf{U}_j(t_i) \right\|_{\mathbb{F}}^2 + \lambda \left\| \mathbf{U}_j(0) \mathbf{U}_j(0)^\top - \mathbf{U}_{j-1}(1) \mathbf{U}_{j-1}(1)^\top \right\|_{\mathbb{F}}^2 + \lambda \left\| \mathbf{U}_j(1) \mathbf{U}_j(1)^\top - \mathbf{U}_{j+1}(0) \mathbf{U}_{j+1}(0)^\top \right\|_{\mathbb{F}}^2 \quad (60)$$

$$\min_{\mathbf{U}_j} - \sum_{i=1}^T \left\| \mathbf{X}_i^\top \mathbf{U}_j(t_i) \right\|_{\mathbb{F}}^2 - \lambda \left\| \mathbf{U}_{j-1}(1)^\top \mathbf{U}_j(0) \right\|_{\mathbb{F}}^2 - \lambda \left\| \mathbf{U}_{j+1}(0)^\top \mathbf{U}_j(1) \right\|_{\mathbb{F}}^2, \quad (61)$$

where $\mathbf{X}_i$ is the data on just the j th segment. Intuitively, this loss can be minimized using the same updates described previously, but with extra data $\tilde{\mathbf{X}}_1 \triangleq \sqrt{\lambda} \mathbf{U}_{j-1}(1)$ and $\tilde{\mathbf{X}}_T \triangleq \sqrt{\lambda} \mathbf{U}_{j+1}(0)$ at each end of the geodesic, $t_1 = 0$ and $t_T = 1$ respectively.

This formulation will perform better on a continuous piecewise geodesic than just applying the proposed method to each geodesic individually, but this approach will only approximate a continuous piecewise geodesic when λ is sufficiently large (though too large of λ will cause the BCD algorithm to disregard the data and stall suboptimally). An effective approach may be to start with $\lambda = 0$ and slowly increase it as the algorithm converges. This approach may be sufficient since we are only approximating real data as being derived from a piecewise geodesic and still are able to leverage the extra data. In a large data regime, enforcing the constraint too strictly may only lead to larger modeling error. Thus, a penalized formulation allows us to balance the fit to the model and the data. Alternatively, an ADMM (Boyd et al., 2010) approach could be developed to satisfy the constraint strictly in a similar manner.

B.3 Video Denoising Additional Experiments and Details

The link to the video data from (Li et al., 2004, Section V-A) has been broken for a while, but the videos can still be found using the internet archive. We downloaded the videos from https://web.archive.org/web/20080118111318/http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html.

We provide further denoising results for video data in this section. We perform experiments on waterfall video data (in addition to the curtain video we provided in the main text), and the results are presented in Figure 15 and Figure 16b. Same as before, we start by showing that the geodesic model is a good choice for this video data. Figure 15a shows the training loss as a function of the rank k . The training loss for the geodesic model lies in between k and $2k$ cases, as expected. It also has smaller error as compared to permuted data, which is strong evidence that the geodesic model is a reasonable model to consider for the waterfall video sequence. Next, we study the denoising capabilities of rank- k SVD, rank- $2k$ SVD, and rank- k geodesic model by adding AWGN with different values of standard deviation σ to the video sequence and applying these three approaches to remove noise. The quality of the denoised image is measured using the peak signal to noise ratio (PSNR). Figure 15b shows the PSNR

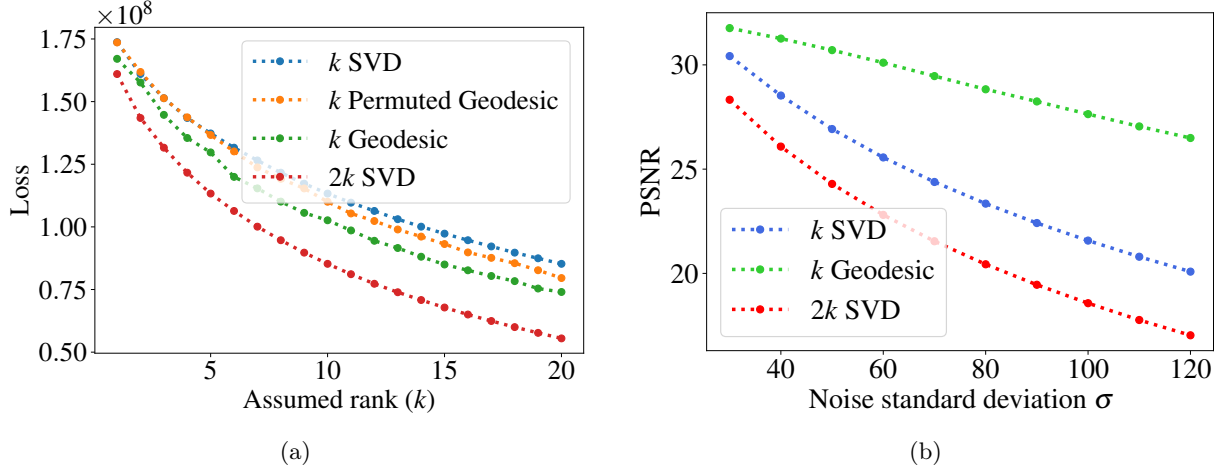


Figure 15: Quantitative evaluation of geodesic subspace model for waterfall video sequence. In (a) loss from (5) is plotted for a video sequence containing 260 frames/images. Loss is plotted against different values of assumed rank of data k . In (b) we added AWGN to the video data and then applied rank- k SVD, rank- $2k$ SVD, and the geodesic model to denoise the noisy version of video with $k = 10$ and $\ell = 4$.

of the denoised video as a function of added noise level. Finally, we provide visual evidence of denoising in Figure 16. In Figure 16a frame 125 is shown for denoising the video corrupted with AWGN of $\sigma = 110$. Each image shown is the reconstruction of that frame using each model for denoising, and the PSNR is given at the top of each image. Notice that both PSNR and the perceptual quality of image denoised by the rank- k geodesic model is better than the other two methods, which is a similar trend we observed in Figure 8b for the Curtain dataset. Next, in Figure 16b we show results from a similar experiment but the noise added is AWGN with $\sigma = 30$, which is significantly lower than the previous experiment. From this experiment we can conclude that the three methods have almost similar performance in low noise settings.

Finally, we display frame 125 for curtain and waterfall sequences for a lower noise (higher SNR) regime in Figure 17. These results show that in lower noise settings rank- $2k$ SVD is able to learn more structure in data and hence has better denoising performance than rank- k SVD and rank- k geodesic model. In contrast, in higher noise settings the structure in the images described by smaller singular values is overwhelmed by noise and rank- $2k$ SVD ends up learning a lot of noise, which diminishes its denoising capabilities.

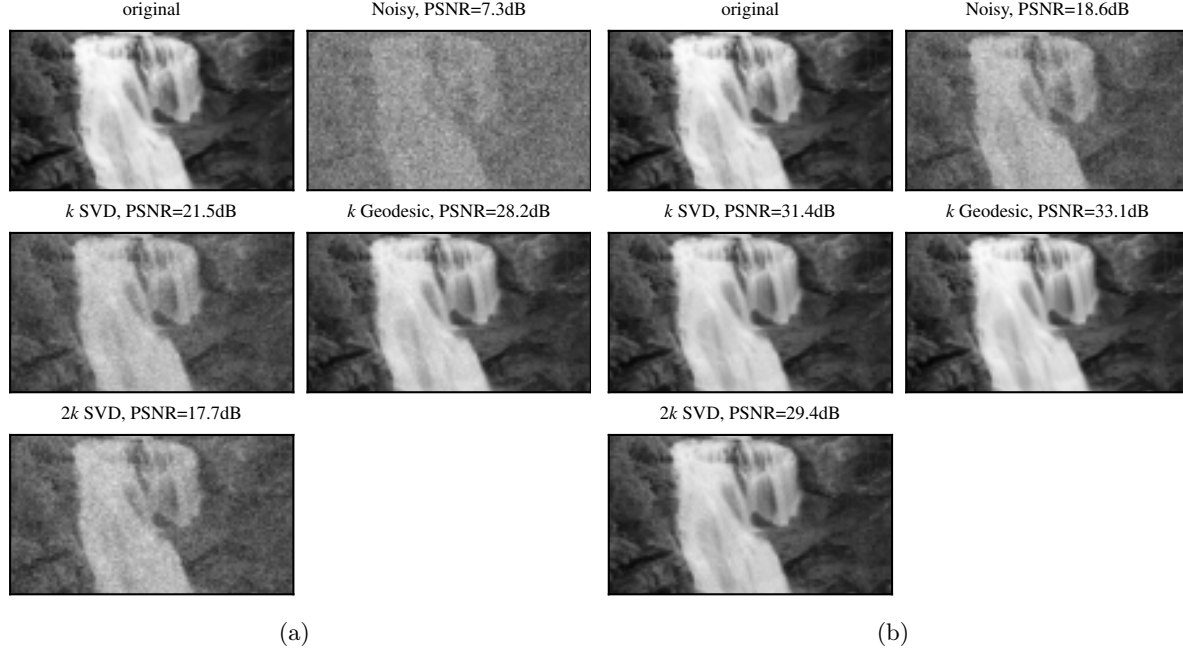


Figure 16: Visual example of denoising frame 125 in the waterfall video sequence with AWGN of $\sigma = 110$ in (a) and $\sigma = 30$ in (b). The geodesic model was able to denoise the noisy image more effectively than rank- k SVD and rank- $2k$ SVD in high noise regime in (a). On the other hand for lower noise regime in (b) the denoising performance is quite similar.

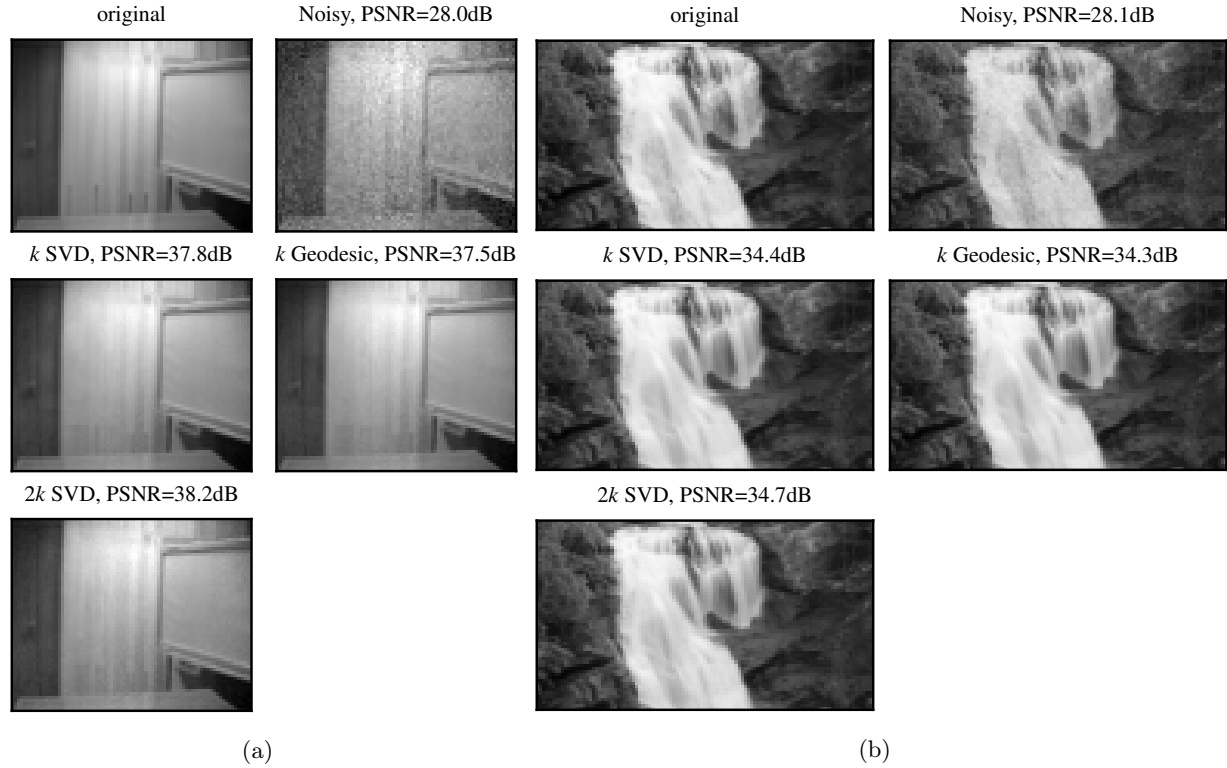


Figure 17: When the SNR is higher, the geodesic model and PCA model are more comparable. This figure shows the same frames from both the curtain and waterfall videos and their reconstructions with $\sigma = 10$.