Atlas-based Manifold Representations for Interpretable Riemannian Machine Learning

Ryan A. Robinett
University of Chicago
Computer Science (CS)

Sophia A. Madejski
University of Chicago
Molecular Engineering (PME)

Kyle Ruark
Harvard University
SEAS

Samantha J. Riesenfeld*
University of Chicago, PME & Medicine
NSF-Simons NITMB, Chicago
Biohub, Chicago

Lorenzo Orecchia*
University of Chicago, CS
NSF-Simons NITMB, Chicago

Abstract

Despite the popularity of the manifold hypothesis, current manifold-learning methods do not support machine learning directly on the latent d-dimensional data manifold, as they primarily aim to perform dimensionality reduction into $\mathbb{R}^D$, losing key manifold features when the embedding dimension D approaches d. On the other hand, methods that directly learn the latent manifold as a differentiable atlas have been relatively underexplored. In this paper, we aim to give a proof of concept of the effectiveness and potential of atlas-based methods. To this end, we define and implement a generic data structure `Atlas` to maintain a differentiable atlas that enables Riemannian optimization over the manifold. We complement this with an unsupervised heuristic `Atlas-Learn` that learns a differentiable atlas from point cloud data. We experimentally demonstrate that this approach has advantages in terms of efficiency and accuracy in selected settings. Moreover, in a supervised classification task over the Klein bottle and in RNA velocity analysis of hematopoietic data, we showcase the improved interpretability and robustness of our approach.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

1 INTRODUCTION

By extending concepts from flat Euclidean space to more complex, curved spaces, Riemannian manifolds provide a convenient mathematical model to abstractly represent nonlinear data dependencies in high-dimensional datasets. The related manifold hypothesis (Bengio et al., 2013), stating that real-world data are predominantly concentrated on low-dimensional manifolds within a high-dimensional embedding space, has been both theoretically (Carlsson et al., 2008) and experimentally (Pope et al., 2021; Brown et al., 2022) validated for image data and is considered a key ingredient for the success of deep learning (Bengio et al., 2013; Bronstein et al., 2021). Similarly, natural scientists increasingly encounter empirical data that are well characterized by low-dimensional, often non-Euclidean, Riemannian manifolds. For example, for several biological systems profiled by single-cell RNA-sequencing (scRNA-seq), which estimates the transcript counts of thousands of genes in tens of thousands of individual cells, the data were shown to live near low-dimensional manifolds ($d \leq 5$) and display nontrivial geometry with statistically significant scalar curvature (Sritharan et al., 2021).

As a result, there is now great interest in designing algorithms and simulating dynamics that eschew the high-dimensional ambient space and directly work with a representation of the low-dimensional manifold. Examples include recently proposed Riemannian variants of generative diffusion models (Bortoli et al., 2022; Jo and Hwang, 2024; Mangoubi et al., 2025) that have been applied to protein generation (Yim et al., 2023). Stochastic models and dynamic optimal

*Co-corresponding authors, contributed equally.

transport algorithms have also been applied to learned manifold representations of transcriptomic spaces to study dynamic processes, such as cell-fate decisions in embryogenesis (Schiebinger et al., 2019; Moon et al., 2018; Huguet et al., 2022).

Although Riemannian optimization techniques are well developed (Absil et al., 2008), most commonly employed methods for manifold representation are ill-suited to their application, preventing the deployment of general-purpose machine learning techniques to the low-dimensional manifolds underlying much real-world data. Indeed, the steps of even the simplest Riemannian first-order methods are applications of the manifold's exponential map (or more generally of a retraction) (Hosseini and Sra, 2020b), which require knowledge of the manifold's tangent planes and differentiable structure. By contrast, the large majority of current manifold learning techniques do not attempt to learn representations that constrain downstream tasks to work within the manifold and its intrinsic geometry. Rather, they focus heavily on dimensionality reduction, i.e., constructing embeddings of data that are naturally represented in a high-dimensional space $\mathbb{R}^n$ into a lower-dimensional space $\mathbb{R}^D$, where $D << n$ is still higher (often substantially) than the intrinsic dimension of the latent data manifold (Meilă and Zhang, 2023). Such methods often aim to preserve fundamental intrinsic topological invariants (such as homology) and geometric invariants (such as geodesic distances), but not to support optimization. For instance, even the simple task of evolving a given ODE by forward-Euler discretization over the manifold cannot be realized in a generic dimensionality-reduced embedding, as the evolving point may leave the manifold and travel freely in the D-dimensional ambient space, making it hard to interpret the inferred trajectories. This issue arises in scientific applications where experimental data naturally include a velocity vector field over the manifold, such as RNA velocity (Bergen et al., 2021; Qiu et al., 2022; Wang et al., 2025; Gao et al., 2024). Section 1.2 includesk further discussion of different manifold learning methods.

1.1 Our Contribution

The previous discussion immediately suggests considering manifold representations that directly parametrize the latent manifold via diffeomorphic charts. Because most real-world latent manifolds are not diffeomorphic to Euclidean space (Cooley et al., 2020), this approach explicitly requires the use of multiple charts and differentiable transition maps between overlapping charts, i.e., the maintenance of a differentiable atlas (Lovett, 2010). While some atlas-learning methods have been proposed, such as chart auto-encoders (Schonsheck et al., 2019; Sidheekh et al., 2023), the approach has been relatively under-explored, with current methods performing poorly at neighborhood preservation (Cohn et al., 2022) and finding only limited practical application. A common objection to the deployment of atlas-learning methods is the conceptual and computational complexity of constructing and maintaining a large number of charts and overlaps (Meilă and Zhang, 2023) and the inherent difficulty in deploying machine learning methods on such a complex representation.

The main goal of this paper is to provide a proof of concept demonstrating that practical methods that explicitly learn and maintain an (approximate) atlas can enable scalable analysis of low-dimensional manifold structure that is more faithful and interpretable than that afforded by current methods, while preserving the ease of use associated with Euclidean representations. To this end, we define the specification of a straightforward data structure, named `Atlas`, which which requires the implementation of a small number of simple computational primitives associated with a differentiable atlas, such as chart identification and computation of chart differentials. Given access to a generic implementation for `Atlas`, we provide methods that enable the execution of higher-level algorithms, such as numerical approximation of exponential maps and geodesics, vector transports, and homology computation, which yield immediate insights in the geometric structure of the underlying manifold. As we demonstrate in our empirical study, these methods can also be easily combined, as in the Euclidean setting, to implement fundamental machine learning routines over the manifold. Effectively, our approach reduces many fundamental geometric and machine-learning tasks over the manifold to the task of providing an implementation for a small number of simple primitives in the `Atlas` data structure. In contrast with existing computational packages, our implementation can represent and enable optimization over both known atlases of algebraic manifolds and atlases learned from empirical manifolds given as high-dimensional point-cloud data. For the latter application, which is crucial in many data-analysis workflows, we also present a simple heuristic, `Atlas-Learn`, for constructing an `Atlas` implementation via *approximate quadratic coordinate charts* (Sritharan et al., 2021). The rest of the paper is dedicated to exploring the theoretical properties and empirical behavior of `Atlas` and `Atlas-Learn`, as well as showcasing the practical advantages of combining these two methods in the analysis of point-cloud data. Our empirical analyses demonstrate the efficiency of this approach, its capability to recover manifold structure and its enabling of machine-learning routines to work directly in the

inherent geometry of the data. In particular, we show that `Atlas` can both speed up first-order Riemannian optimization over manifolds with closed-form algebraic structure and enable it in the first place over manifolds learned from empirical point cloud data. The following are our main contributions in this direction:

1. In Section 3.1, we show that the `Atlas` scheme can speed up first-order Riemannian optimization for a benchmark online subspace learning problem on the Grassmann manifold, a classical target of Riemannian optimization routines (Absil et al., 2008).

2. In Section 3.2, we compare `Atlas-Learn` with existing manifold-learning techniques on the task of reconstructing from point cloud data an established manifold parametrization of Carlsson's high-contrast, natural image patches (Carlsson et al., 2008; Sritharan et al., 2021). We show that our scheme outperforms competitors in preserving homology and approximate geodesic path lengths.

3. In Section 3.3, we use `Atlas`-based primitives to implement a Riemannian optimization algorithm – the Riemannian principal boundary (`RPB`) algorithm (Yao and Zhang, 2020) – to solve a classification problem over the Klein bottle, a manifold with nontrivial topology. The classification results are more accurate and interpretable than those from existing manifold learning methods.

4. Finally, in Section 3.4, we apply `Atlas-Learn` and `Atlas` to the reconstruction of transcriptional dynamics using a well-studied hematopoietic scRNA-seq dataset (Qiu et al., 2022). Compared with flows predicted by previous approaches based on higher-dimensional embeddings, the flows predicted by the `Atlas`-based approach adhere much more closely to the latent manifold, improving biological interpretability of the results.

As prototype solutions to the challenge of learning an effective atlas-like manifold representation, both `Atlas` and `Atlas-Learn` can likely be improved upon significantly, particularly by more closely co-designing the two methods and by affording `Atlas-Learn` more adaptivity in choosing chart parameters. Indeed, the main point of our paper is not their algorithmic optimization, but the finding that such a simple framework already improves upon the inadequacies of existing manifold learning methods. We hope that our paper can inspire others to design improved algorithms and to use our approach to verify empirical results obtained by other methods.

1.2 Related Work

Manifold Learning Previous work on manifold learning has been centered on the problem of reducing the dimension n of the ambient space $\mathbb{R}^n$ to a target dimension D, while preserving the structure of the d-dimensional latent manifold, with the goal of saving space and processing time (Meilă and Zhang, 2023). The earliest example of such methods is principal component analysis (`PCA`) (Jolliffe, 2002), which simply projects onto the D-dimensional subspace preserving the most variance in the data. `Isomap` (Tenenbaum et al., 2000) and its variations (Zha and Zhang, 2003; Weinberger and Saul, 2006) attempt to construct *metric embeddings* of the latent manifold into $\mathbb{R}^d$, i.e., embeddings of the data in which the Euclidean geodesic (straight-line) distances in $\mathbb{R}^D$ equal (or approximate) the manifold geodesic distances. Generally, this is only possible for flat manifolds, no matter how large D is chosen[1](Robinson, 2006). Graph-based methods, such as diffusion maps (Coifman and Lafon, 2006) and Laplacian eigenmaps (Belkin and Niyogi, 2001, 2006) successfully approximate the Laplace-Beltrami operator of the latent manifold and yield embeddings that capture the diffusion distance along the manifold. However, they may require large embedding dimension D and do not enable execution of optimization routines over the manifold. Popular visualization-oriented graph-based methods, such as t-SNE and UMAP, are known in practice to greatly distort topological information, as shown in the case of scRNA-seq data (Cooley et al., 2020). More advanced methods (Koelle et al., 2022; Donoho and Grimes, 2003; Roweis and Saul, 2000) learn local representations, akin to charts, but combine them into one global Euclidean representation, rather than constructing an atlas.

Finally, there exists a growing body of latent variable models that attempt to construct low-dimensional manifold representations via deep learning approaches, including normalizing flows (Brehmer and Cranmer, 2020) and variational autoencoders (Chen et al., 2020). In this vein, a natural approach is to endow latent variable models with pullback metrics that better reflect the geometry of the data (Tosi et al., 2014; Arvanitidis et al., 2018), potentially including manifold structure within the data. However, Hauberg (2019) shows that the recovery of manifold structures within this framework requires careful uncertainty quantification of the geometry away from the data, which is a subject of current study (Syrota et al., 2025).

Ultimately, all these methods aim to produce a faith-

[1]Isometric embeddings, such as those of Nash (1954, 1956), preserve the Riemannian metric and the geodesic distance along the manifold, not with respect to the metric structure of the ambient space.

ful embedding of the latent manifold into $\mathbb{R}^D$, which is known to exist for $D \geq 2d$ by Whitney's celebrated embedding result (Whitney, 1936). However, even Whitney's embedding, a gold standard of manifold learning (Meilă and Zhang, 2023), would not let us work directly on the latent manifold. To address this issue, all the above methods are often run with $D \approx d$ in the attempt to construct a global single chart for the data manifold. However, this is liable to cause significant distortions in the resulting embedding for manifolds with non-trivial topology, which are not homeomorphic to $\mathbb{R}^D$.

In light of these shortcomings, several authors have attempted to learn topologically nontrivial manifolds by means of chart autoencoders (Schonsheck et al., 2019; Korman, 2021; Sidheekh et al., 2023), which explicitly consider the construction of an atlas. However, these methods were shown to sacrifice local structure more than much simpler methods (Cohn et al., 2022) and have otherwise seen little rigorous practical evaluation. Further, the use of deep learning, as opposed to simpler models for chart construction, can obscure the geometric properties of the atlas construction, making downstream results harder to interpret and debug.

There is, however, some good news in that many crucial manifold invariants can be accurately estimated from samples in a local fashion. For instance, Little et al. (2017) prove that if a manifold $\mathcal{M} \subset \mathbb{R}^D$ is observed as a finite sample perturbed by sub-Gaussian noise, then multiscale singular value decomposition (`mSVD`) can be used to compute both local tangent plane approximations and the intrinsic dimensionality of $\mathcal{M}$. In more empirical work, Sritharan et al. (2021) describe a method to estimate the Riemannian curvature tensor via the Gauss-Codazzi equations by computing local quadratic approximations to the manifold. These successful local algorithms, `mSVD` and local quadratic approximations, are also the main workhorses of our manifold learning approach.

Riemannian Optimization Many Riemannian optimization routines are intuitive adaptations of Euclidean first- and second-order methods (Absil et al., 2008; Hosseini and Sra, 2020b). However, it is far more complex to compute analogous first- and second-order primitives on Riemannian manifolds. For example, the action of a gradient on a point in Euclidean space is simply vector addition, whereas in a Riemannian manifold, this action requires the computation of the exponential map, which is the solution to a specific second-order ordinary differential equation. While closed-form solutions are known for some algebraic manifolds (Absil et al., 2008; Boumal et al., 2016), they are generally unavailable, even for cases as simple as oblate ellipsoids in $\mathbb{R}^3$ (Gan'shin, 1967). This difficulty persists even when an oracle for the Riemannian metric at a point is available, as in the case of pullback metric approaches (e.g., Arvanitidis et al., 2018). Addressing this challenge is often discussed in terms of retractions, i.e., smooth maps that assign to each point a map from the tangent space to the manifold that agrees with the exponential map in the zero-th and first derivatives. Euclidean "step-and-project" methods, which make updates on a manifold using Euclidean gradients in an ambient space before mapping the update back into the manifold, are prototypical examples of retractions, but can suffer from slow running time due to numerical condition issues (Absil et al., 2008). Because finding sufficiently fast, accurate retractions is a challenge, they are often discussed in theory rather than practice (Hosseini and Sra, 2020b), with the notable exception of the Bürer-Monteiro scheme for convex optimization on the PSD elliptope (Cifuentes, 2021).

For general manifolds observed empirically as point clouds, existing Riemannian optimization techniques have seen limited application, as the differential-geometric primitives lack known closed forms. In these cases, Euclidean optimization methods are usually deployed on low-dimensional Euclidean representations learned via the dimensionality reduction techniques discussed above (e.g., Huguet et al., 2022). However, downstream machine learning tasks, such as clustering and inference of pseudo-time, have been shown to be sensitive to the choice of low-dimensional representation (Patruno et al., 2020), emphasizing the need for a more robust approach to Riemannian optimization over learned manifolds.

2 METHODS

Throughout the paper, alphabetically indexed sections refer to the Supplementary Materials. This section describes the specification of the `Atlas` data structure and motivation for its design (Sec. 2.1). While most implementation details appear in the Supplementary Materials (Sec. B), we describe here how `Atlas` uses simple Euclidean updates to quickly compute retractions, i.e., approximations to the exponential map, and vector transports (Sec. 2.2). Finally, we introduce the method `Atlas-Learn` for constructing an `Atlas` data structure from point-cloud data (Sec. 2.3).

2.1 The `Atlas` Data Structure

Differentiable Atlases Since the work of Whitney, the notion of a differentiable atlas has been a crucial component of the modern mathematical definition of a smooth manifold (Lovett, 2010; Kobayashi and No-

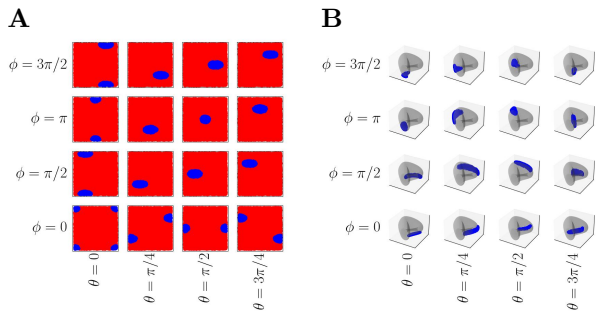

Figure 1: **Chart domains for a differentiable atlas of the Klein bottle A.** Charts depicted in polar coordinates show points belonging (blue) or not (red) to the chart, and indexed by the (θ, ϕ) coordinates of the chart center $k(\theta, \phi)$ (as in Sec. 3.2). **B.** The same coordinate charts shown in the Karcher representation.

mizu, 1996). A differentiable atlas for a manifold $\mathcal{M}$ of dimension d is a collection $\left\{\varphi_i : \mathcal{V}_i \subseteq \mathbb{R}^d \to \mathcal{U}_i\right\}_i$ of bijective coordinate charts[2] whose images cover $\mathcal{M}$, i.e., $\bigcup_i \mathcal{U}_i = \mathcal{M}$ (example in Fig. 1). For coordinate charts φ_i and φ_j whose images overlap, i.e., $\mathcal{U}_i \bigcap \mathcal{U}_j \neq \emptyset$, an atlas also specifies bijective *transition maps*

$$\psi_{ij} := \left(\varphi_j^{-1} \circ \varphi_i\right)\Big|_{\varphi_i^{-1}(\mathcal{U}_i \cap \mathcal{U}_j)}. \tag{1}$$

The atlas is differentiable if all the transition maps are differentiable. This definition neatly allows $\mathcal{M}$ to be an abstract topological space, by shifting all the differentiability requirements to the transition maps. A collection of bijective differentiable transition maps uniquely identifies a differentiable manifold, up to diffeomorphism, even in the absence of corresponding charts. Motivated by this property, we remove the requirement of Equation 1 and define transition maps separately from coordinate chart images. This significantly simplifies the computational work needed to handle empirical manifolds.

The Atlas Data Structure We assume, without loss of generality, that the manifold $\mathcal{M}$ of interest is a subset of $\mathbb{R}^D$. For applications to point-cloud data, this inclusion will be an embedding that may have been preprocessed by `PCA`-based dimensionality reduction to accelerate computation. An instance of the `Atlas` data structure maintains n coordinate charts of the form $\{\varphi_i : \mathcal{V}_i \subseteq \mathbb{R}^d \to \mathbb{R}^D\}_{i \in [n]}$, which are differentiable injections. An undirected atlas graph $G = ([n], E)$ describes which charts overlap, i.e., $\{i, j\} \in E$ if charts i and j overlap. In this case, a differentiable injective transition map ψ_{ij} is defined from a subset $\mathcal{V}_{ij} \subset \mathcal{V}_i$ to $\mathcal{V}_j$. The basic attributes and methods of the data structure are given in Algorithm 1.

Patching Together Local Charts The definition of `Atlas` is motivated by the desire to accommodate not only atlases for manifolds with known structure, but also atlases learned from point-cloud data, which may not exactly conform to the definition of a differentiable atlas. To understand this challenge, let us briefly consider the behavior of `Atlas-Learn` (further detailed in Sec. 2.3). `Atlas-Learn` starts by locally learning tangent planes via PCA and defining charts in tangential coordinates by performing quadratic regression in the normal space. An immediate obstacle is that different charts are solving separate regression problems: for example, it may happen that no pair of images $\phi_i(\mathcal{V}_i)$ and $\phi_j(\mathcal{V}_j)$ even overlap, so that transition maps of the form of Equation 1 become undefined. To accommodate this possibility, while still being able to work on the manifold, we weaken the requirement on transition maps by allowing the `Atlas` data structure to contain arbitrary differentiable injective transition maps between chart domains, which may fail to obey Equation 1. In other words, coordinate charts will serve just as canonical maps from each domain $\mathcal{V}_i$ to the embedding space $\mathbb{R}^D$, while the global differentiable structure is determined by the transition maps.

Discrepancy Between Charts Naturally, our `Atlas-Learn` heuristic enforces some compatibility between coordinate charts φ_i and transition maps. In particular, for any point $\vec{\xi} \in \mathcal{V}_{ij}$, the discrepancy

$$\|\varphi_i(\vec{\xi}) - \varphi_j(\psi_{ij}(\vec{\xi}))\|_{\mathbb{R}^D}$$

goes to 0 as the number n of charts and the number of points sampled from a smooth, nonnegative probability measure go to infinity (Sec. C.4). Notice that if the discrepancy is 0 for all points $\vec{\xi}$, Equation 1 holds and the `Atlas` data structure actually yields a differentiable atlas (Sec. A).

In contrast, when a discrepancy exists, error may be introduced and accumulate over multiple transitions. The closer ψ_{ji} is to ψ_{ij}^{-1}, and the fewer times a transition boundary needs to be crossed, the less error is introduced. Despite error from transition maps, our results empirically demonstrate numerical robustness of `Atlas`-enabled computations on various problems.

2.2 Building Retractions via `Atlas`

In Riemannian optimization, hard-to-compute exponential maps are often replaced by retractions (e.g., Hosseini and Sra, 2020b). A retraction $R_{\vec{p}}$ at a point $\vec{p} \in \mathcal{M}$ is a local diffeomorphism from the tangent

[2] The coordinate charts are also homeomorphisms. For simplicity here, we omit standard topological details.

Algorithm 1 `Atlas`
Generic specification for `Atlas` data structure

Require: target manifold dimension d, ambient dimension D, number n of charts, atlas graph $G = ([n], E)$
  for $i \in [n]$ **require**
    $\texttt{in\_domain}_i : \vec{\xi} \in \mathbb{R}^d \to \{\mathbb{T}, \mathbb{F}\}$ ▷ Implicitly defines the domain $\mathcal{V}_i$ for chart ϕ_i
    $\texttt{identify\_new\_chart}_i : \vec{\xi} \in \mathcal{V}_i \to 2^{[n]}$ ▷ Identifies the charts j for which $\psi_{ij}(\vec{\xi})$ is defined, defining $\mathcal{V}_{ij}$
    $\varphi_i : \vec{\xi} \in \mathcal{V}_i \to \mathbb{R}^D$ ▷ Coordinate chart mapping $\vec{\xi}$ to ambient coordinates $\varphi(\vec{\xi})$
    $\mathrm{D}\varphi_i : \vec{\xi} \in \mathcal{V}_i \to \mathbb{R}^{D\times d}$ ▷ Maps $\vec{\xi}$ to the differential operator $D\varphi_i(\vec{\xi})$
  for $\{i, j\} \in E$ **require**
    $\psi_{ij} : \mathcal{V}_{ij} \to \mathcal{V}_j$ ▷ Transition map from $\mathcal{V}_{ij}$ to $\mathcal{V}_j$
    $\mathrm{D}\psi_{ij} : (\vec{\xi}, \vec{\tau}) \in \mathcal{V}_{ij} \times \mathbb{R}^d \to \mathbb{R}^d$ ▷ Maps tangent vector $\vec{\tau}$ at $\vec{\xi} \in \mathcal{V}_i$ to $[D\psi_{ij}]_{\vec{\xi}}(\vec{\tau})$

plane $\mathcal{T}_{\vec{p}}\mathcal{M}$ at $\vec{p}$ to $\mathcal{M}$ which maps $\vec{0}$ to $\vec{p}$ and has the identity as its differential at $\vec{0}$. This ensures that the retraction is a first-order approximation to the true exponential map. Given an `Atlas` data structure, we can exploit our coordinate charts to give an efficient *local* retraction by the following simple construction (Absil et al., 2008). Let $\vec{p} = \varphi_i(\vec{\xi})$ and define:

$$R_{\vec{p}}(\vec{\tau}) := \varphi_i\left(\vec{\xi} + \left[\mathrm{D}\varphi_i(\vec{\xi})\right]^{-1}\vec{\tau}\right) \tag{2}$$

Elementary differentiation shows that $R_{\vec{p}}(\tau)$ is a valid retraction at $\vec{p}$ as long as $\vec{\xi} + \left[\mathrm{D}\varphi_i(\vec{\xi})\right]^{-1}\vec{\tau} \in \mathcal{V}_i$. Otherwise, we are forced anyway to transition to a different chart. When the coordinate is barely past the transition boundary, we simply transition charts after the update[3]. Otherwise, we perform the update by truncating $\vec{\tau}$ to $\vec{\tau}' = c \cdot \vec{\tau}$, for the largest c such that $\vec{q} = R_{\vec{p}}(\vec{\tau}')$ still lands in $\mathcal{V}_i$. We then transition $\vec{q}$ and tangent vector $(1-c)\cdot\tau$ to the next chart. Hereafter, we refer to updates performed via the retraction of Equation 2 as *quasi-Euclidean updates*, as, in the coordinate chart representation, they simply add a linear transformation of the update direction $\vec{\tau}$ to the current point $\vec{\xi}$. We rely on basic primitives of the `Atlas` data structure to implement other fundamental manifold methods, including retraction-logarithms (the local inverses of retractions), geodesic computations, and vector transports. (Details in Section B.)

2.3 The `Atlas-Learn` Heuristic

We assume that the dimensionality of the manifold representation be specified *a priori*; this can be learned using dimensionality estimation methods, e.g., multiscale singular value decomposition (mSVD) (Little et al., 2017). The `Atlas-Learn` starts by using n-medoids clustering (Schubert and Rousseeuw, 2021) to partition the N-point cloud $X \in \mathbb{R}^{N\times D}$ into n subsets $X_1, \ldots, X_n$, which will correspond to the images of the coordinate charts. For each X_i, we compute the mean $\vec{m}_i$ of points in X_i and learn a local tangent plane by computing the top d principal components of the data X_i centered at m_i. This yields a Stiefel matrix $L_i \in \mathbb{R}^{d\times D}$, which projects onto the tangent space passing through m_i, and another Stiefel matrix $M_i \in \mathbb{R}^{(D-d)\times D}$, which projects onto the orthogonal normal space. With this representation of the tangent space in hand, we parametrize the manifold over X_i by performing a least-squares quadratic regression from the tangent coordinates $L_i^\top X_i$ to the normal coordinates $M_i^\top X_i$. Let $f_i : \mathbb{R}^d \mapsto \mathbb{R}^{D-d}$ be the resulting quadratic function. Then, we can implement part of specification of `Atlas` by the standard parametrization $\varphi_i(\vec{\xi}) := (\vec{\xi}, f_i(\vec{\xi}))$ of a graph manifold (do Carmo, 2016) and its differential $\mathrm{D}\varphi_i$. The transition map is constructed as:

$$\psi_{ij}(\vec{\xi}) = L_j^\top\left[\frac{1}{2}M_i K_i\left(\vec{\xi}\otimes\vec{\xi}\right) + \left(L_i\vec{\xi} + \vec{m}_i\right) - \vec{m}_j\right]$$

with $\mathrm{D}\psi_{ij}$ being again its differential. We also note that the coordinate representation at $\vec{\xi}$ of the Riemannian metric induced by the ambient embedding of the manifold can be easily expressed as the matrix $I_d + \nabla f_i(\vec{\xi})^\top \nabla f_i(\vec{\xi})$. It remains to define a mechanism to determine chart membership. For this purpose, we make use of a two-step procedure. For each subset X_i, we learn an approximate minimum-volume enclosing ellipsoid (MVEE) of X_i, by solving a second-order conic program (Nie and Demmel, 2005). We use membership in this ellipsoid as a criterion to decide whether a transition should be considered, i.e., the MVEE implicitly defines $\mathcal{V}_i \setminus \left(\bigcup_j \mathcal{V}_{ij}\right)$. When we exit the MVEE, we evaluate potential transitions by considering which charts have the least regression error for the current point. Section C details how the primitives in Algorithm 1 are implemented by `Atlas-Learn`.

[3] In our specific experiments, due to the relatively small magnitude of our first-order updates, we only have to consider this case.

3 RESULTS

3.1 Online Subspace Learning

An `Atlas` data structure can be constructed explicitly for a manifold with a known atlas. We do this for the (n, k)-Grassmann manifold $\mathbf{Gr}_{n,k}$ with its canonical Ehresmann atlas. The implementation is a simple exercise in numerical algebra (Sec. D). We use the resulting `Atlas` to compute the Grassmann inductive Fréchet expectation estimator (`GiFEE`) (Chakraborty and Vemuri, 2015), a standard task in online subspace learning. Our experiments show that our approach, based on quasi-Euclidean retractions on the `Atlas` data structure, outperforms state-of-the-art methods in runtime, with no loss in accuracy.

Fréchet Expectation Estimation Given a stream of samples $\mathcal{X}_1, \ldots, \mathcal{X}_i$ from a probability distribution on $\mathbf{Gr}_{n,k}$, the `GiFEE` estimator is computed by inductively applying the update rule

$$\vec{v}_{i+1} \leftarrow \mathbf{Log}_{M_i}(\mathcal{X}_i) \tag{3}$$

$$M_{i+1} \leftarrow \mathbf{Exp}_{M_i}\left(\frac{1}{i+1}\vec{v}_{i+1}\right), \tag{4}$$

where $M_1 = \mathcal{X}_1$ and the tangent vector $\vec{v}_i \in T_{M_i}\mathbf{Gr}_{n,k}$ is initialized to $\vec{v}_1 = \vec{0}$. Chakraborty and Vemuri show that the `GiFEE` estimator converges in probability to the Fréchet mean of any distribution on $\mathbf{Gr}_{n,k}$ under certain limitations on support and Riemannian L^2-moment (Chakraborty and Vemuri, 2015). In this experiment, we use geodesic power distributions $\mathbf{GPD}(\mathcal{X}, p)$ with Fréchet mean $\mathcal{X}$ for $p > 1$ (formally defined in Section D). To compute first-order updates on the `Atlas` representation, we replace invocations of **Exp** with invocations of quasi-Euclidean updates on the `Atlas` instance and invocations of **Log** with an `Atlas`-based approximation of the Riemannian logarithm, which are detailed in Section D.

Competitors We compare our algorithm against these Grassmannian optimization routines: (1) the original GiFEE algorithm (Chakraborty and Vemuri, 2015); (2) `MANOPT`, the state-of-the-art package for manifold optimization (Boumal et al., 2016; Hosseini and Sra, 2020a; Luchnikov et al., 2021); and 3) `MANOPT-RET`, a method in the `MANOPT` package that uses a faster retraction than the exponential map. In contrast, `GiFMEE` and `MANOPT` use closed forms for logarithms and exponential maps in $\mathbf{Gr}_{n,k}$, but differ in how the numerical steps are organized, leading to different performance.

Experimental Results All methods have similar, high accuracy, measured by geodesic distance between

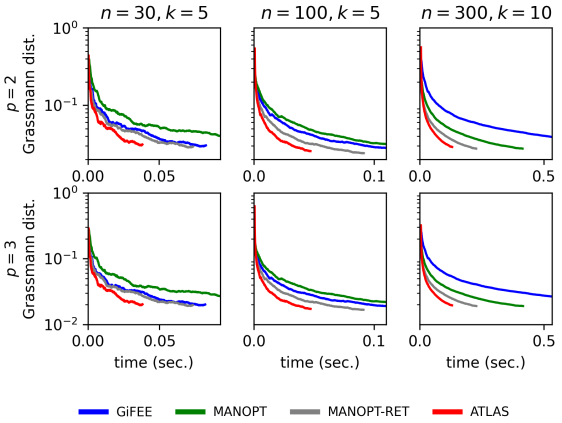

Figure 2: **Quasi-Euclidean updates in the `Atlas` representation of the Grassmannian $\mathbf{Gr}_{n,k}$ converge faster to the population Fréchet mean than do other first-order update schemes.** Logarithms and retractions are executed by each method one observation at a time for 1,000 iterations, each observation being a $\mathbb{R}^{n,k}$ Stiefel matrix. Stiefel matrices $X \in \mathbb{R}^{n\times k}$ drawn from the distribution $\mathbf{GPD}(\mathcal{X}, p)$ with fixed Fréchet mean $\mathcal{X}$ and $p \in \{2, 3\}$ (Sec. D.6). $\mathbf{Gr}_{n,k}$-distance to $\mathcal{X}$ from each method's iterate is plotted against cumulative runtime.

the `GiFEE` estimator and the true population Fréchet mean as a function of the number of iterations. The experimental results show runtime superiority of the `Atlas` approach over the other first-order routines for experiments with (n, k) set to $(30, 5)$, $(100, 5)$, or $(300, 10)$ (Fig. 2).

3.2 Preservation of Manifold Geometry

Here, we evaluate for some practical data how well the `Atlas` data structure constructed by `Atlas-Learn` preserves topological and geometric features of the ground-truth manifold. We leverage a well-studied space of 3×3 high-contrast, natural image patches (van Hateren and van der Schaaf, 1998), the underlying manifold of which Carlsson elegantly parametrized and showed was homeomorphic to the Klein bottle (Carlsson et al., 2008) (Fig. 1). Focusing on the k^0 parametrization (Sritharan et al., 2021), which induces a Riemannian metric through a specific embedding into $\mathbb{R}^{3\times3}$, we construct meshes of different cardinalities on the manifold to use as inputs for `Atlas-Learn` and the dimensionality-reduction methods `UMAP`, `TSNE`, `LLE`, `Isomap` and `PCA`. All methods are run with target dimension $D = 2$, the intrinsic dimension of the manifold. We then compute persistent homology and geodesic distances on the resulting representations.

The results show that geodesic distances between pairs of points are better preserved by the `Atlas` represen-

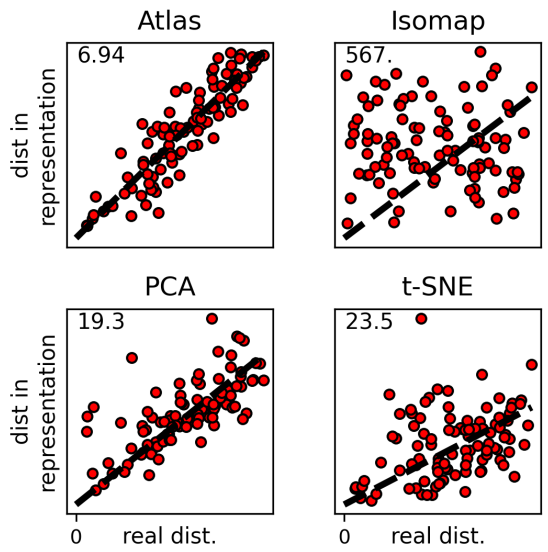

Figure 3: **The `Atlas` data structure preserves geodesic distances better than common dimensionality-reduction techniques.** `Isomap`, `PCA` and t-`SNE` were computed on a mesh of 1 million points. `Atlas-Learn` was asked to produce 64 charts, given only a 10,000-points mesh. For 100 randomly sampled pairs of points on the Klein bottle, each scatter plot shows each pair (dot) according to a precise estimate of their true geodesic distance (x axis, "real dist.") versus the embedding distance (y axis, "dist. in representation") for each embedding (panel title) (details in Sec. E). Metric distortion given in top left corner. Dashed line, linear fit with y-intercept zero.

tation than by the other methods (Fig. 3). In particular, `Isomap`, which explicitly aims to preserve geodesic distances, fares very poorly. These findings, robust even when larger dimension budget $D > 2$ is allowed (Sec. E), likely stem from the nontrivial topology and curvature of the Klein bottle. Indeed, we find that while competitors do not preserve homological features, as measured by an aggregate bottleneck distance, the `Atlas` representation are almost perfectly recapitulates these features (Sec. E).

3.3 Classification of Image Patches

To test the ability of a learned `Atlas` to enable Riemannian optimization and machine learning, we implemented and applied the Riemannian principal boundary (`RPB`) algorithm (Yao and Zhang, 2020), which generalizes support vector machines (`SVM`) to Riemannian geometry. The natural image patches have a natural principal boundary between convex and concave patches, which we aim to identify using the `RPB` algorithmic approach (Fig. 4, Sec. E). For a 2-dimensional manifold $\mathcal{M}$, the `RPB` algorithm learns a binary classifier by first learning one-dimensional "boundary" submanifolds $\Gamma, \Gamma' \subset \mathcal{M}$, for each of the two classes, and then "interpolating" between these two boundaries to create a 1-dimensional separating submanifold. The interpolation is done by characterizing Γ and Γ' as solutions to ordinary differential equations (ODEs) called *principal flows* parameterized by the class label. At each iteration of the algorithm, the solution curve to each principle flow is approximated locally as a short geodesic update in the direction of the first derivative given by the ODE. The boundary curve is simultaneously updated by parallel-transporting the first derivatives from the two principle flows, taking their weighted Fréchet mean, and taking a small geodesic update in the direction of this average. The resulting boundary curve, which is referred to as the *principal boundary*, serves as a binary classifier.

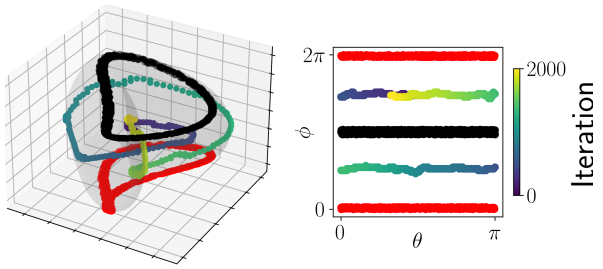

Figure 4: **The `RPB` algorithm effectively learns a discriminating boundary on the manifold of high-contrast image patches.** A 64-chart `Atlas` data structure was generated by `Atlas-Klein` from a 10,000-point mesh of the Klein bottle manifold for high-contrast image patches. The `RPB` implementation was then applied to learn a discriminating boundary (colored by discretized ODE iterations of the algorithm) between the convex (red) and concave (black) patches, as illustrated in Karcher (left) and polar (right) coordinates.

Our `RPB` algorithm implementation applies Euler's method over 2,000 iterations to simultaneously integrate the principal flows for the convex and concave patches, as well as the principal boundary ODE. For easier visualization in polar coordinates, the `Atlas` is built using `Atlas-Klein`, a deterministic variant of `Atlas-Learn` with 64 chart centers and spherical bounding ellipsoids that are specified manually using a Cartesian mesh in k^0-polar coordinates (Sec. E).

Our analyses show that `RPB` successfully learns a boundary curve between the convex and concave patches, solely in terms of the geometry encoded in the `Atlas` representation (Fig. 4). The result is a robust separator that is interpretable, i.e., entirely supported on the manifold composed of high-contrast patches and unconstrained by the ambient geometry, which

other techniques fail to produce (Sec. F).

3.4 Integrating Vector Flows Over Transcriptomic Space

In this section, we explore the utility of a learned `Atlas` representation in the study of dynamics over real-world single-cell transcriptomic data. Specifically, we consider an application in which vector field information known as "RNA velocities" (Bergen et al., 2021; Qiu et al., 2022; Wang et al., 2025; Gao et al., 2024), i.e., estimated time derivatives of the transcriptional states of cells, have been inferred, and we aim to simulate the resulting RNA dynamics.

For this task, we use a benchmark dataset from Qiu et al. (2022) containing scRNA-seq profiles of CD34+ hematopoietic stem cells (HSCs), progenitor cells, and differentiated cells, enhanced by metabolic labeling to distinguish nascent from pre-existing mRNA. The accompanying `Dynamo` computational framework uses the metabolic labels to more accurately estimate RNA velocity vectors for each datum (cell). Sparse kernel regression is used to extend these velocities to a vector field $\vec{V}_{\texttt{amb}}$ over the ambient space $\mathbb{R}^{30}$, determined by the top 30 principal components. Finally, the authors integrate this vector field as an ODE in ambient space to reveal transcriptional trajectories characterizing the differentiation of HSCs into terminal cell types.

An explicit assumption behind this approach is that integration over the vector field, starting at HSC data points, should maintain proximity to the latent lower-dimensional manifold $\mathcal{M}$ of mRNA expression. To verify this assumption, we compared the ODE solution yielded by `Dynamo`'s ambient-space integration against that yielded by integrating the ODE over an `Atlas` learned via `Atlas-Learn`. To perform this integration, we endow our `Atlas` with a vector field $\vec{V}_{\texttt{atlas}}$ given by the projection of $\vec{V}_{\texttt{amb}}$ onto the tangent spaces produced by `Atlas-Learn`. The complete setup is detailed in the Supplementary Materials (Sec. G).

The results show that the iterates of the `Atlas`-restricted ODE lie significantly closer to datapoints than ambient ODE iterates do (Fig. 5), suggesting that the ambient ODE veers away from the manifold. Moreover, the average distance between an `Atlas`-restricted iterate and the closest datum is relatively small, falling below the global average distance between a data point and the learned `Atlas` manifold. These findings suggest that the learned `Atlas` instance serves as a constraint that enables the integration along $\vec{V}_{\texttt{amb}}$ to reconstruct transcriptional dynamics in HSC differentiation more faithfully, which can improve interpretations of the underlying biological processes.

In Section G, visualizing this phenomenon on a UMAP

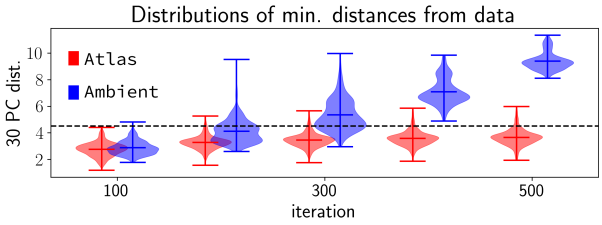

Figure 5: **Integrating the RNA velocity vector field over the `Atlas` representation rather than the ambient space more closely matches observations in hematopoietic scRNA-seq data.** For 309 distinct initial conditions, corresponding to labeled HSCs, a vector field over human hematopoietic cells is integrated, either over the ambient space (blue) or over a 5-dimensional `Atlas` representation with 30 charts (red). At every iteration, distances from each of the 309 evolutions to the closest datum in 30-PC space is computed; these are shown as violin plots. Dashed line indicates average distance between a datum and nearest `Atlas` point.

reveals that, unlike the ambient ODE, the `Atlas`-restricted ODE steadily spreads through progenitor cell states to terminal cell states (Sec. G, Fig. 12).

4 DISCUSSION

Since real-world data may not conform to a constant dimension, one weakness of our `Atlas` definition is its lack of support for varifolds, generalizations of manifolds whose intrinsic dimensionality may change from location to location. Another opportunity for improvement lies in the construction of charts within `Atlas-Learn`, which may benefit from adaptively choosing chart centers by considering the goodness of fit of the resulting quadratic approximations. Finally, robust Riemannian optimization on an atlas-based representation requires careful consideration of the error introduced through both geodesic approximation and transition maps, which remains to be evaluated rigorously.

ACKNOWLEDGMENTS

SJR and LO gratefully acknowledge support for this work by grants from the NSF (DMS-2235451) and Simons Foundation (MPS-NITMB-00005320) to the NSF-Simons National Institute for Theory and Mathematics in Biology (NITMB). SJR is a Biohub Investigator. RAR was supported in part by a National Science Foundation Graduate Research Fellowship. The authors wish to thank Shmuel Weinberger for his feedback and for many inspiring conversations.

References

P.-A. Absil, R. Mahoney, and Rodolphe Sepulchre. Optimization algorithms on matrix manifolds. Princeton University Press, 2008.

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: On the curvature of deep generative models. In Proceedings of International Conference on Learning Representations, January 2018.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In Advances in Neural Information Processing Systems, volume 14. MIT Press, 2001.

Mikhail Belkin and Partha Niyogi. Convergence of Laplacian eigenmaps. In B. Schölkopf, J. Platt, and T. Hoffman, editors, Advances in Neural Information Processing Systems, volume 19. MIT Press, 2006.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. IEEE Trans. Pattern Anal. Mach. Intell., 35(8):1798–1828, August 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2013.50.

Volker Bergen, Ruslan A Soldatov, Peter V Kharchenko, and Fabian J Theis. RNA velocity—current challenges and future perspectives. Molecular Systems Biology, 17(8):e10282, 2021. doi: https://doi.org/10.15252/msb.202110282.

Valentin De Bortoli, Emile Mathieu, Michael John Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. Riemannian score-based generative modelling. In Advances in Neural Information Processing Systems, October 2022.

Nicolas Boumal, Vladislav Voroninski, and Afonso S. Bandeira. The non-convex Burer–Monteiro approach works on smooth semidefinite programs. In Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16, page 2765–2773, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. In Advances in Neural Information Processing Systems, volume 33, pages 442–453, 2020.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, May 2021.

Bradley CA Brown, Anthony L. Caterini, Brendan Leigh Ross, Jesse C. Cresswell, and Gabriel Loaiza-Ganem. Verifying the union of manifolds hypothesis for image data. In The Eleventh International Conference on Learning Representations, September 2022.

Gunnar Carlsson, Tigran Ishkhanov, Vin de Silva, and Afra Zomorodian. On the local behavior of spaces of natural images. International Journal of Computer Vision, 76:1–12, 2008.

Rudrasis Chakraborty and Baba C. Vemuri. Recursive Fréchet mean computation on the Grassmannian and its applications to computer vision. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 4229–4237, 2015. doi: 10.1109/ICCV.2015.481.

Nutan Chen, Alexej Klushyn, Francesco Ferroni, Justin Bayer, and Patrick Van Der Smagt. Learning flat latent manifolds with VAEs. In Proceedings of the 37th International Conference on Machine Learning, volume 119 of ICML'20, pages 1587–1596. JMLR.org, July 2020.

Diego Cifuentes. On the Burer–Monteiro method for general semidefinite programs. Optimization Letters, 15:2299–2309, 2021.

Thomas Cohn, Nikhil Devraj, and Odest Chadwicke Jenkins. Topologically-informed atlas learning. In 2022 International Conference on Robotics and Automation (ICRA), page 3598–3604. IEEE Press, 2022. doi: 10.1109/ICRA46639.2022.9812311.

Ronald R. Coifman and Stéphane Lafon. Diffusion maps. Applied and Computational Harmonic Analysis, 21(1):5–30, July 2006. ISSN 1063-5203. doi: 10.1016/j.acha.2006.04.006.

Shamus M. Cooley, Timothy Hamilton, J. Christian J. Ray, and Eric J. Deeds. A novel metric reveals previously unrecognized distortion in dimensionality reduction of scRNA-seq data, 2020. URL `https://www.biorxiv.org/content/10.1101/689851v6`.

Manfredo P. do Carmo. Differential Geometry of Curves and Surfaces: Revised and Updated Second Edition, chapter 5, page 435. Dover Publications, 2016.

David L. Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. Proceedings of the National Academy of Sciences, 100(10):5591–5596, May 2003. doi: 10.1073/pnas.1031596100.

V. N. Gan'shin. Geometry of the Earth Ellipsoid, chapter 5. Nedra Publishers, 1967. Translated by J. M. Willis, Technical Translation Unit, Aeoronautical Chart and Information Center, 1969.

Cheng Frank Gao, Suriyanarayanan Vaikuntanathan, and Samantha J. Riesenfeld. Dissection and integration of bursty transcriptional dynamics for complex

systems. Proceedings of the National Academy of Sciences, 121(18):e2306901121, 2024. doi: 10.1073/pnas.2306901121.

Søren Hauberg. Only Bayes should learn a manifold (on the estimation of differential geometric structure from data), 2019. URL `https://arxiv.org/abs/1806.04994`.

Reshad Hosseini and Suvrit Sra. An alternative to EM for Gaussian mixture models: batch and stochastic Riemannian optimization. Mathematical Programming, 181:187–223, 2020a.

Reshad Hosseini and Suvrit Sra. Recent Advances in Stochastic Riemannian Optimization, pages 527–554. Springer, Cham, Switzerland, 2020b.

Guillaume Huguet, D.S. Magruder, Alexander Tong, Oluwadamilola Fasina, Manik Kuchroo, Guy Wolf, and Smita Krishnaswamy. Manifold interpolating optimal-transport flows for trajectory inference. In Advances in Neural Information Processing Systems, volume 35, pages 29705–29718, December 2022.

Jaehyeong Jo and Sung Ju Hwang. Generative modeling on manifolds through mixture of Riemannian diffusion processes. In Proceedings of the 41st International Conference on Machine Learning, volume 235 of ICML'24, pages 22348–22370, Vienna, Austria, July 2024. JMLR.org.

I. T. Jolliffe. Principal Component Analysis. Springer Series in Statistics. Springer-Verlag, New York, 2002. ISBN 978-0-387-95442-4. doi: 10.1007/b98835.

Shoshichi Kobayashi and Katsumi Nomizu. Foundations of Differential Geometry, volume 1. Wiley-Interscience, new edition, 1996.

Samson J. Koelle, Hanyu Zhang, Marina Meilă, and Yu-Chia Chen. Manifold coordinates with physical meaning. J. Mach. Learn. Res., 23(1):133:5898–133:5954, January 2022. ISSN 1532-4435.

Eric O. Korman. Atlas based representation and metric learning on manifolds, 2021. URL `https://arxiv.org/abs/2106.07062`.

Anna V. Little, Mauro Maggioni, and Lorenzo Rosasco. Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature. Applied and Computational Harmonic Analysis, 43(3):504–567, 2017. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2015.09.009.

Stephen T. Lovett. Differential Geometry of Manifolds. A K Peters/CRC Press, Natick, Mass, 1st edition edition, June 2010. ISBN 978-1-56881-457-5.

Ilia A. Luchnikov, Alexander Ryzhov, Sergey N. Filippov, and Henni Ouerdane. QGOpt: Riemannian optimization for quantum technologies. SciPost Phys., 10:079, 2021. doi: 10.21468/SciPostPhys.10.3.079.

Oren Mangoubi, Neil He, and Nisheeth K. Vishnoi. Efficient diffusion models for symmetric manifolds. In Forty-Second International Conference on Machine Learning, June 2025.

Marina Meilă and Hanyu Zhang. Manifold learning: What, how, and why, November 2023. URL `https://arxiv.org/abs/2311.03757`.

Kevin R. Moon, Jay S. Stanley, Daniel Burkhardt, David van Dijk, Guy Wolf, and Smita Krishnaswamy. Manifold learning-based methods for analyzing single-cell RNA-sequencing data. Current Opinion in Systems Biology, 7:36–46, 2018. ISSN 2452-3100. doi: https://doi.org/10.1016/j.coisb.2017.12.008.

John Nash. C1 isometric imbeddings. Annals of Mathematics, 60(3):383–396, 1954. ISSN 0003-486X. doi: 10.2307/1969840.

John Nash. The imbedding problem for Riemannian manifolds. Annals of Mathematics, 63(1):20–63, 1956. ISSN 0003-486X. doi: 10.2307/1969989.

Jiawang Nie and James W. Demmel. Minimum ellipsoid bounds for solutions of polynomial systems via sum of squares. Journal of Global Optimization, 33:511–525, 2005.

Lucrezia Patruno, Davide Maspero, Francesco Craighero, Fabrizio Angaroni, Marco Antoniotti, and Alex Graudenzi. A review of computational strategies for denoising and imputation of single-cell transcriptomic data. Briefings in Bioinformatics, 22(4):bbaa222, 10 2020. ISSN 1477-4054. doi: 10.1093/bib/bbaa222.

Phil Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. In International Conference on Learning Representations, 2021.

Xiaojie Qiu, Yan Zhang, Jorge D. Martin-Rufino, Chen Weng, Shayan Hosseinzadeh, Dian Yang, Angela N. Pogson, Marco Y. Hein, Kyung Hoi (Joseph) Min, Li Wang, Emanuelle I. Grody, Matthew J. Shurtleff, Ruoshi Yuan, Song Xu, Yian Ma, Joseph M. Replogle, Eric S. Lander, Spyros Darmanis, Ivet Bahar, Vijay G. Sankaran, Jianhua Xing, and Jonathan S. Weissman. Mapping transcriptomic vector fields of single cells. Cell, 185(4):690–711.e45, February 2022. ISSN 0092-8674. doi: 10.1016/j.cell.2021.12.045.

P. L. Robinson. The sphere is not flat. The American Mathematical Monthly, 113(2):171–173, 2006. ISSN 0002-9890. doi: 10.2307/27641870.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. Science, December 2000. doi: 10.1126/science.290.5500.2323.

Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, Lia Lee, Jenny Chen, Justin Brumbaugh, Philippe Rigollet, Konrad Hochedlinger, Rudolf Jaenisch, Aviv Regev, and Eric S Lander. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. Cell, 176(4):928–943.e22, February 2019.

Stefan C. Schonsheck, Jie Chen, and Rongjie Lai. Chart auto-encoders for manifold structured data, 2019. URL `https://arxiv.org/abs/1912.10094`.

Erich Schubert and Peter J. Rousseeuw. Fast and eager k-medoids clustering: O(k) runtime improvement of the PAM, CLARA, and CLARANS algorithms. Information Systems, 101:101804, November 2021. ISSN 0306-4379. doi: 10.1016/j.is.2021.101804.

Sahil Sidheekh, Chris B. Dock, Tushar Jain, Radu Balan, and Maneesh K. Singh. VQ-Flows: Vector quantized local normalizing flows. PMLR, pages 1835–1845, 2023.

Duluxan Sritharan, Shu Wang, and Sahand Hormoz. Computing the Riemannian curvature of image patch and single-cell RNA sequencing data manifolds using extrinsic differential geometry. Proceedings of the National Academy of Sciences, 118(29):e2100473118, 2021. doi: 10.1073/pnas.2100473118.

Stas Syrota, Yevgen Zainchkovskyy, Johnny Xi, Benjamin Bloem-Reddy, and Søren Hauberg. Identifying metric structures of deep latent variable models, 2025. URL `https://arxiv.org/abs/2502.13757`.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. Science, December 2000. doi: 10.1126/science.290.5500.2319.

Alessandra Tosi, Sören Hauberg, Alfredo Vellido, and Neil D. Lawrence. Metrics for probabilistic geometries. In Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI'14, page 800–808, Arlington, Virginia, USA, 2014. AUAI Press. ISBN 9780974903910.

J H van Hateren and A van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. Proceedings of the Royal Society B: Biological Sciences, 265(1394):359–366, March 1998. ISSN 0962-8452. doi: 10.1098/rspb.1998.0303.

Yajunzi Wang, Jing Li, Haoruo Zha, Shuhe Liu, Daiyun Huang, Lei Fu, and Xin Liu. Paradigms, innovations, and biological applications of RNA velocity: a comprehensive review. Briefings in Bioinformatics, 26(4):bbaf339, 07 2025. ISSN 1477-4054. doi: 10.1093/bib/bbaf339.

Kilian Q. Weinberger and Lawrence K. Saul. Unsupervised learning of image manifolds by semidefinite programming. International Journal of Computer Vision, 70(1):77–90, October 2006. ISSN 1573-1405. doi: 10.1007/s11263-005-4939-z.

Hassler Whitney. Differentiable manifolds. Annals of Mathematics, pages 645–680, 1936.

Zhigang Yao and Zhenyue Zhang. Principal boundary on Riemannian manifolds. Journal of the American Statistical Association, 115(531):1435–1448, 2020. doi: 10.1080/01621459.2019.1610660.

Jason Yim, Brian L. Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. SE(3) diffusion model with application to protein backbone generation, May 2023. URL `https://arxiv.org/abs/2302.02277`.

Hongyuan Zha and Zhenyue Zhang. Isometric embedding and continuum ISOMAP. AAAI, 2003.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model.
   Yes, we give a clear description of the mathematical setting and assumptions.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm.
   Yes

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results.
   Yes, we carefully consider underlying assumptions in all our theoretical results.
   (b) Complete proofs of all theoretical results.
   Yes
   (c) Clear explanations of any assumptions.
   Yes

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL).
Yes

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen).
Yes

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times).
Yes

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider).
Yes, only applicable for Grassmann experiments.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets.
Yes

(b) The license information of the assets, if applicable.
Not Applicable

(c) New assets either in the supplemental material or as a URL, if applicable.
Not Applicable

(d) Information about consent from data providers/curators.
Not Applicable

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content.
Not Applicable

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots.
Not Applicable

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable.
Not Applicable

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation.
Not Applicable

Supplementary Materials

CODE REPOSITORY

All code is available at the repository `https://github.com/RiesenfeldGroup/atlas_graph_learning.git`.

A DETAILS AND NUMERICAL RESULTS ON ATLAS

A.1 `Atlas` Discrepancy

Recall the following definition:

Definition 1. *Given an `Atlas` data structure $\mathcal{A}$ with charts $\varphi_i : \mathcal{V}_i \to \mathbb{R}^D$ and transition maps $\psi_{ij} : \mathcal{V}_{ij} \to V_j$, the discrepancy of $\mathcal{A}$ is defined as*

$$\sup_{i,j} \sup_{\vec{\xi} \in \mathcal{V}_{ij}} \|\varphi_i(\vec{\xi}) - \phi_j(\psi_{ij}(\vec{\xi}))\|_{\mathbb{R}^D}.$$

As claimed in the body, if the discrepancy of an `Atlas` $\mathcal{A}$ is 0, then $\mathcal{A}$ is a differentiable atlas for a manifold $\mathcal{M} \subset \mathbb{R}^D$.

Lemma 1. *If the discrepancy of `Atlas` $\mathcal{A}$ equals 0, then for all charts i, j*

$$\psi_{ij} = \varphi_j^{-1} \circ \varphi_i$$

on $\mathcal{V}_{ij}$.

Proof. By the definition of discrepancy, for all charts i, j, for all $\vec{\xi} \in \mathcal{V}_{ij}$, we have

$$\|\varphi_i(\vec{\xi}) - \phi_j(\psi_{ij}(\vec{\xi}))\|_{\mathbb{R}^D} = 0 \implies \varphi_i(\vec{\xi}) = \phi_j(\psi_{ij}(\vec{\xi})) \implies \psi_{ij}(\vec{\xi}) = (\varphi_j^{-1} \circ \varphi_i)(\vec{\xi}).$$

□

In Section C.4, we sketch the proof that, given n samples from the uniform distribution over a smooth manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$, `Atlas-Learn` builds an `Atlas` data structure $\mathcal{A}$ whose discrepancy goes to 0 as n goes to infinity, showing that $\mathcal{A}$ converges to a differentiable atlas.

A.2 Representation of the Riemannian Metric on `Atlas`

Let $(\mathcal{V}, \varphi)$ be a coordinate chart of Riemannian manifold $\mathcal{M}$. If $\mathfrak{g}$ is the Riemannian metric on $\mathcal{M}$, then $\mathfrak{g}_p$ can be thought of as a linear isomorphism from $T_p\mathcal{M}$ to $T_p^*\mathcal{M}$ (i.e., the vector space of linear functionals on $T_p\mathcal{M}$). For $\vec{\xi} = \varphi(p)$, we can define a distinct Riemannian metric $\overline{\mathfrak{g}}_{\vec{\xi}} : T_{\varphi(p)}\mathcal{V} \to T^*_{\varphi(p)}\mathcal{V}$ in terms of local coordinate charts such that

$$\overline{\mathfrak{g}}_{\vec{\xi}}(\vec{\tau}) : \vec{\tau}' \mapsto \mathfrak{g}_p \left(\left[D\varphi^{-1} \right]_{\varphi(p)} (\vec{\tau}) \right) \left(\left[D\varphi^{-1} \right]_{\varphi(p)} (\vec{\tau}') \right), \tag{5}$$

where $D\varphi^{-1}$ denotes the differential of φ^{-1}.

Similarly, for an `Atlas` data structure, we approximate the metric $\overline{\mathfrak{g}}_{\vec{\xi}}(\vec{\tau})$ for $\vec{\xi} \in \mathcal{V}_i$ as:

$$\overline{\mathfrak{g}}_{\vec{\xi}}(\vec{\tau}) : \vec{\tau}' \mapsto \tau^\top (\mathrm{D}\varphi_i(\vec{\xi})^{-1})^\top (\mathrm{D}\varphi_i(\vec{\xi})^{-1}) \tau'$$

where we take $\mathfrak{g}_p$ to be the identity, i.e., the standard metric for the Euclidean space $\mathbb{R}^D$ in which the manifold $\mathcal{M}$ is embedded.

B APPROXIMATION OF DIFFERENTIAL-GEOMETRIC PRIMITIVES NEAR THE ORIGIN IN GRAPHS OF QUADRATIC APPROXIMATIONS

In this section, we provide details for our implementations of the high-level methods supported by the `Atlas` data structure. These are:

- computation of **retractions**, i.e., approximations to the exponential map, described in Section 2.2.
- computation of the associated retraction logarithm, described below in Section B.3.
- computation of approximate vector transports, described below in Section B.1.
- computation of approximate geodesic distances, described below in Section B.2.
- computation of persistent homology is described in Section E.1 for the case of the Klein bottle.

B.1 Approximation Error of Vector Transport on `Atlas`

When a vector field V acts on a differentiable manifold $\mathcal{M}$ to take a point p to q, vectors in $T_p\mathcal{M}$ should be transformed into vectors in $T_q\mathcal{M}$. There are many linear isomorphisms between $T_p\mathcal{M}$ and $T_q\mathcal{M}$, and the "correct" choice of isomorphism should be uniquely determined by the path $\gamma : [0, 1] \to \mathcal{M}$ along which V takes p to q. This isomorphism $T_p\mathcal{M} \to T_q\mathcal{M}$ determined by γ is called the *parallel transport*. Its fundamental role in first-order methods over manifolds is described in Hosseini and Sra (2020).

For `Atlas`, within a chart, we approximate parallel transport $\mathcal{P}^{\mathbf{Ret}}_{p\to\mathbf{Ret}_p(\vec{v})}\vec{w}$ of tangent vector $\vec{w} \in T_p\mathcal{M}$ to $T_{\mathbf{Ret}_p(\vec{v})}\mathcal{M}$ along $\mathbf{Ret}_p$ with the identity vector transport

$$\mathcal{T}_{\vec{\xi}\to\vec{\xi}+\vec{\tau}} : \vec{\sigma} \mapsto \vec{\sigma} \tag{6}$$

It is used to approximate Riemannian logarithms (Secs. D and F), to approximate geodesic path lengths (Sec. E) and implement the Riemannian principal boundary (RPB) algorithm (Sec. F).

Here, we look at the error term of the identity vector transport of Equation 6 as an approximation of the parallel transport of a tangent vector along a quasi-Euclidean update.

Theorem 1. *The vector transport of Equation 6 approximates parallel transport along quasi-Euclidean updates up to order* $O\left(\|\vec{\sigma}\|_{\mathfrak{g}}\|\vec{\tau}\|^2_{\mathfrak{g}}\right)$.

Proof. Let $(\mathcal{U}, \mathcal{V}, \varphi)$ be a coordinate chart of Riemannian manifold $\mathcal{M}$ with points $p, q \in \mathcal{U}$ satisfying $p \neq q$. Further, let $\tau := \varphi(q) - \varphi(p)$, and let there be a tangent vector $\vec{w} \in T_p\mathcal{M}$ with representative tangent vector $\vec{\sigma}_0 := [D\varphi]_p(\vec{w})$. Lastly, let $\Gamma^{\lambda}_{\mu\nu}$ be the Christoffel symbol of the second kind. The parallel transport of vector field σ along path the quasi-Euclidean update from $\varphi(p)$ to $\varphi(q)$ is given (Absil et al., 2008) by

$$\tau^j\partial_j\left(\sigma^l\right)\partial_l + \tau^j\sigma^k\Gamma^l_{jk}\partial_l = 0.$$

Recognizing that the time derivative of the jth component of σ, which we denote by $\dot{\sigma}^j$, is equal to $\tau^l\partial_l\left(\sigma^j\right)$, we see that the jth component of $\mathcal{P}^{\mathbf{Ret}_p}_{p\to q}\vec{w}$ is given by

$$\left[\mathcal{P}^{\mathbf{Ret}_p}_{p\to q}\vec{w}\right]^j = \left(\left[D\varphi^{-1}\right]_{\varphi(q)}\right)^j_k\sigma^k_0 - \left(\left[D\varphi^{-1}\right]_{\varphi(q)}\right)^j_k\int_0^1\sigma^l\Gamma^k_{ml}\tau^m dt. \tag{7}$$

Therefore, the following series of deductions ends with a convenient form for the jth component of the difference between $\mathcal{P}^{\mathbf{Ret}_p}_{p\to q}\vec{w}$ and $\mathcal{T}_{p\to q}\vec{w}$.

$$\begin{aligned}
\left[\mathcal{T}_{p\to q}\vec{w} - \mathcal{P}^{\mathbf{Ret}_p}_{p\to q}\vec{w}\right]^j &= \left(\left[D\varphi^{-1}\right]_{\varphi(q)}\right)^j_k\int_0^1\sigma^l\Gamma^k_{ml}\tau^m dt \\
&= \left(\left[D\varphi^{-1}\right]_{\varphi(q)}\right)^j_k\tau^m\int_0^1\sigma^l\Gamma^k_{ml}dt \\
&= \left(\left[D\varphi^{-1}\right]_{\varphi(q)}\right)^j_k\left(\tau^m\sigma^l_0\int_0^1\Gamma^k_{ml}dt + \tau^m\int_0^1\left(\sigma^l - \sigma^l_0\right)\Gamma^k_{ml}dt\right)
\end{aligned}$$

The $\left[D\varphi^{-1}\right]$ term is absorbed according to Equation 5 when computing the Riemannian metric, and the term $\tau^m \sigma_0^l \int_0^1 \Gamma_{ml}^k dt$ is of order $O\left(\|\vec{\sigma}_0\|_{\overline{\mathfrak{g}}} \|\vec{\tau}\|_{\overline{\mathfrak{g}}}\right)$. To see that $\mathcal{T}_{p\to q}\vec{w}$ approximates $\mathcal{P}^{\mathbf{Ret}_p}_{p\to q}\vec{w}$ up to order $O\left(\|\vec{\sigma}_0\|_{\overline{\mathfrak{g}}} \|\vec{\tau}\|^2_{\overline{\mathfrak{g}}}\right)$, it remains to show that this is the order of the term $\tau^m \int_0^1 \left(\sigma^l - \sigma_0^l\right) \Gamma_{ml}^k dt$. This is accomplished in the following series of deductions, which make use of Lemma 2 and the Taylor expansion of the integrand.

$$\begin{aligned}
\tau^m \int_0^1 \left(\sigma^l - \sigma_0^l\right) \Gamma_{ml}^k dt &= \tau^m \int_0^1 \left(t\dot{\sigma}_0^l + \sum_{d=2}^{\infty} \frac{t^d}{d!} \left(\sigma_0^{(d)}\right)^l \right) \Gamma_{ml}^k dt \\
&= \int_0^1 \left(tO\left(\|\vec{\sigma}_0\|_{\overline{\mathfrak{g}}} \|\vec{\tau}\|^2_{\overline{\mathfrak{g}}}\right) + \sum_{d=2}^{\infty} \frac{t^d}{d!} O\left(\|\vec{\sigma}_0\|_{\overline{\mathfrak{g}}} \|\vec{\tau}\|^{d+1}_{\overline{\mathfrak{g}}}\right) \right) dt \\
&= O\left(\|\vec{\sigma}_0\|_{\overline{\mathfrak{g}}} \|\vec{\tau}\|^2_{\overline{\mathfrak{g}}}\right)
\end{aligned}$$

□

Lemma 2. *Let $\vec{\sigma}_0^{(d)}$ be the dth derivative of $\vec{\sigma}$ with respect to time, evaluated at time $t = 0$. For all $d \in \mathbb{N}_{>0}$, it holds that $\vec{\sigma}_0^{(d)}$ is linear in σ_0 and homogeneously of order d in both $\vec{\tau}$ and the Christoffel symbols of the second kind.*

Proof. This is easy to prove by induction, with the base case $d = 1$ being given by the parallel transport equation. □

B.2 Approximate Geodesic Distances via Quasi-Euclidean Updates

In Riemannian optimization algorithms that iterate over manifold-valued data, a loss function may weigh the contribution of each datum according to its distance from the current iterate (e.g., stochastic Riemannian gradient descent, Hosseini and Sra, 2020). In regimes where quasi-Euclidean updates approximate the exponential map well, the paths traversed by quasi-Euclidean updates approximate geodesic paths well. Therefore, the lengths of quasi-Euclidean paths with respect to the metric inherited from the ambient space can be used to estimate (albeit overestimate[4]) geodesic distances. For $\vec{\xi}_0, \vec{\xi}_1 \in \mathcal{V}_i$, we refer to

$$\tilde{d}_i\left(\vec{\xi}_0, \vec{\xi}_1\right) = \int_0^1 \left(\left(\vec{\xi}_1 - \vec{\xi}_0\right)^\top \left(\vec{\xi}_1 - \vec{\xi}_0\right) + \sum_{j=1}^{D-d} \left(\left(\vec{\xi}_1 - \vec{\xi}_0\right)^\top K_{ij} \left((1-t)\vec{\xi}_0 + t\vec{\xi}_1\right) \right)^2 \right)^{\frac{1}{2}} dt \tag{8}$$

as the *naïve approximate distance* between $\vec{\xi}_0$ and $\vec{\xi}_1$.

To define the naive approximate distances between two points $\vec{\xi}_1 \in \mathcal{V}_1, \vec{\xi}_2 \in \mathcal{V}_2$ that are not in the same coordinate chart, we use the shortest-path distance between their ambient representations $\vec{x}_1$ and $\vec{x}_2$, respectively. These will be an accurate representation of the corresponding manifold distance only when $\vec{x}_1$ and $\vec{x}_2$ are close .

We construct a graph $G_{\delta,\epsilon}$ whose vertices are the union of δ-nets of each chart domain $\mathcal{V}_i$. We connect vertices by an edge when their naïve approximate distances are less than ϵ and assign the distance to be the edge length. Finally, we compute approximate geodesics between two points $\vec{\xi}_1 \in \mathcal{V}_1$ and $\vec{\xi}_2 \in \mathcal{V}_2$ by rounding each point to the nearest representative in that chart's δ-net and performing a shortest-path computation in $G_{\delta,\epsilon}$ between the resulting vertices.

B.3 Approximating the Riemannian Retraction Logarithm

When performing variants of Riemannian gradient descent on a loss function determined by manifold-valued data (Hosseini and Sra, 2020), the contribution of each datum to the Riemannian gradient of the loss function involves computing retraction logarithms. For the quasi-Euclidean retraction on `Atlas`, retraction logarithms can be computed easily. Specifically, for points $\vec{\xi}_1, \vec{\xi}_2$ belonging to the same $\mathcal{V}$, the retraction logarithm of $\vec{\xi}_2$

[4]Overestimation results from the definition of geodesics as length-minimizing paths on a Riemannian manifold

at $\vec{\xi}_1$, or $\mathbf{Log}^{\mathbf{Ret}}_{\vec{\xi}_1}(\vec{\xi}_2)$, is simply $\vec{\xi}_2 - \vec{\xi}_1$; for points $\vec{\xi}_1, \vec{\xi}_2$ that do not belong to the same $\mathcal{V}$, we approximate $\mathbf{Log}^{\mathbf{Ret}}_{\vec{\xi}_1}(\vec{\xi}_2)$ using the shortest path between $\vec{\xi}_1$ and $\vec{\xi}_2$ on the nearest neighbor graph $G_{\delta,\epsilon}$ of densely sampled points from `Atlas` structure (Sec. B.2), by iteratively summing and vector-transporting the edgewise retraction logarithms from the endpoint of the path back to the start.

C METHODS FOR INFERRING COORDINATE CHART STRUCTURE FROM POINT CLOUDS

C.1 Time Complexity of `Atlas-Learn`

The `Atlas-Learn` algorithm is described in Figure 2.

Algorithm 2 `Atlas-Learn` (Learn `Atlas` from point cloud data)

Require: Point cloud $X \in \mathbb{R}^{N \times D}$ in D dimensions
Require: Estimated intrinsic dimension d
Require: Number k of charts to learn; number ν neighbors in nearest neighbors graph
  $G \leftarrow \texttt{NearestNeighbors}(X, \nu)$ ▷ l_2-weighted, ν-nearest neighbors graph
  $X_1, \ldots, X_k \leftarrow \texttt{KMedoids}(G, k)$ ▷ Partition X by k-medoids
  $\mathcal{A} \leftarrow \varnothing$ ▷ Store charts
  for $j \in \{1, \ldots, k\}$ **do**
    $\vec{x}, L, M, K \leftarrow \texttt{LocalQuadraticApproximation}\,(X_j)$ ▷ Follows methodology described in Sec. C.2
    $A, \vec{b}, c \leftarrow \texttt{MVEE}\,(X_j)$ ▷ minimum-volume ellipsoid enclosing X_j; takes form $\vec{x}^\top A\vec{x} + \vec{b}^\top \vec{x} + c = 0$
    $\mathcal{A} \leftarrow \mathcal{A} \cup \left\{ \left(\vec{x}, L, M, K, A, \vec{b}, c \right) \right\}$ ▷ $\vec{x}, L, M, K$ encode local Riem. structure, $A, \vec{b}, c$ transition boundary
  return $\mathcal{A}$

Given a point cloud $X \in \mathbb{R}^{N \times D}$ and $\nu \in \mathbb{N}_{>0}$, the time complexity of constructing an l_2-weighted, ν-nearest neighbors graph G from X is $O(Dn^2)$, though this can be sped up through approximate constructions (e.g., Zhao et al., 2022). Running k-medoids using the shortest path distance on G takes $O(kN^2)$ time using FasterPAM; variants of k-medoids yielding approximate solutions can reduce this to $O(kN)$ (Schubert and Rousseeuw, 2021). The method `LocalQuadraticApproximation`, outlined in Section C.2, takes the following time complexities in different subroutines:

Learning Stiefel matrices L and M by SVD: $O(NDd)$ (Talwalkar et al., 2013)

Learning mean $\vec{x}$: $O(ND)$

Learning trilinear form K: $O(Nd^2)$ time to construct $\mathbf{t}$, $O(N(D-d))$ time to construct $\mathbf{n}$, $O(Nd^4)$ to construct the Moore-Penrose inverse $\mathbf{t}^\dagger$ of $\mathbf{t}$ by SVD (Li et al., 2019), and $O(N(D-d)d^2)$ time to multiply $\mathbf{t}^\dagger$ and $\mathbf{n}$; therefore, $O(Nd^4 + N(D-d)d^2)$ time overall.

C.2 Quadratic Approximations of Point Clouds

For local quadratic approximation of point clouds, there exists a procedure (Section C.1, Sritharan et al., 2021), which we include here for completeness, with minor modifications in notation. Let $X \in \mathbb{R}^{N \times D}$ be a matrix whose rows are observations in $\mathbb{R}^D$. In practice, we assume that X only contains points sufficiently close to one another for a local quadratic approximation to be reasonable. Further, let $\tilde{X} \in \mathbb{R}^{N \times D}$ be the matrix such that, if $\vec{x}_i$ is the ith row of X, then the ith row of $\tilde{X}$ is equal to $\vec{x}_i - \overline{\vec{x}}$, where $\overline{\vec{x}}$ is the mean of $\vec{x}_i$. We can construct the local covariance matrix

$$\Sigma = \frac{1}{N} \tilde{X}^\top \tilde{X}. \tag{9}$$

We can get an eigendecomposition $V \Lambda V^\top = \Sigma$ such that the diagonal entries of Λ are decreasing. Fixing $d < D$, we can define $L \in \mathbb{R}^{D \times d}, M \in \mathbb{R}^{D \times (D-d)}$ satisfying $V = (L \mid M)$.

The decomposition $V = (L \mid M)$ allows us to decompose $\mathbb{R}^D$ into *tangential coordinates* with basis given by the columns of L and *normal coordinates* with basis given by the columns of M. The mean-centered data $\tilde{X}$ similarly decompose into a *tangential component* $\tau = \tilde{X}L$ and a *normal component* $\mathbf{n} = \tilde{X}M_{\vec{c},r}$. If the data X are sampled from a sufficiently small neighborhood of a d-dimensional manifold with sufficiently mild curvature, and N is sufficiently large, then the relationship between $\mathbf{n}$ and τ should be that of a quadratic polynomial in terms of tangential coordinates. This relationship takes the form $\mathbf{n} \approx \mathbf{th}$, where $\mathbf{h}$ is a matrix of quadratic coefficients[5] and $\mathbf{t}$ is a matrix of ones and quadratic monomials given by

$$\mathbf{t} = \begin{pmatrix} 1 & \tau_1^{(1)}\tau_1^{(1)} & \dots & \tau_1^{(1)}\tau_d^{(1)} & \tau_2^{(1)}\tau_1^{(1)} & \dots & \tau_d^{(1)}\tau_d^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & \tau_1^{(N)}\tau_1^{(N)} & \dots & \tau_1^{(N)}\tau_d^{(N)} & \tau_2^{(N)}\tau_1^{(N)} & \dots & \tau_d^{(N)}\tau_d^{(N)} \end{pmatrix}, \tag{10}$$

as in Sritharan et al. (2021). Regression by least squares solves for $\mathbf{h}$ by $\hat{\mathbf{h}} = \left(\mathbf{t}^\top \mathbf{t}\right)^{-1} \mathbf{t}^\top \mathbf{n}$. Instead of storing quadratic coefficients and constant terms in the matrix $\hat{\mathbf{h}}$, we can represent the relationship between tangential coordinate vector $\vec{\xi}$ and normal coordinate vector $\vec{\nu}$ as

$$\vec{\nu} \approx \frac{1}{2} K \left(\vec{\xi} \otimes \vec{\xi}\right) + \vec{a}, \tag{11}$$

where K is a matrix of quadratic coefficients, $\vec{a}$ is a vector of constants, and "$\otimes$" is the Kronecker product. Accordingly, points $\vec{x}$ in the ambient space obey the relationship

$$\vec{x} \approx \frac{1}{2} MK \left(\vec{\xi} \otimes \vec{\xi}\right) + L\vec{\xi} + \vec{x}_0, \tag{12}$$

where $\vec{x}_0 = \vec{c} + \vec{a}$.

C.3 Injectivity of Naïve Transition Maps

For a function $f : X \to Y$, let $\overline{f} : X \to X \times Y$ denote the function $\overline{f} : x \mapsto (x, f(x))$ (conventionally, $\overline{f}$ is called the *graph* of f).

Claim 1. *Let $\mathcal{V}, \mathcal{V}' \subset \mathbb{R}^N$ be d-dimensional affine spaces with angle θ between them, and ψ, ψ' be the affine projections onto $\mathcal{V}$ and $\mathcal{V}'$, respectively. If $F : \mathcal{V} \to \mathcal{V}^\perp$ is K-Lipschitz for $K < \tan\left(\frac{\pi}{2} - \theta\right)$, then $\psi' \circ \overline{F}$ is injective.*

Proof. Assume to the contrary that there exist distinct $a, b \in \mathcal{V}$ such that $(\psi' \circ \overline{F})(a) = (\psi' \circ \overline{F})(b)$. The map $\overline{F}$ is injective, so $\alpha := \overline{F}(a), \beta := \overline{F}(b)$ are distinct, and ψ' maps the line $l_{\alpha\beta}$ passing through α, β to a single point p. The line $l_{\alpha\beta}$ is the graph of some affine function $\varphi : \psi(l_{\alpha\beta}) \to \mathcal{V}^\perp$ obeying the relation

$$\frac{\|\varphi(x) - \varphi(y)\|_{\mathbb{R}^{N-d}}}{\|x - y\|_{\mathbb{R}^d}} \geq \tan\left(\frac{\pi}{2} - \theta\right)$$

for distinct $x, y \in \psi(l_{\alpha\beta})$. So

$$\frac{\|F(a) - F(b)\|_{\mathbb{R}^{N-d}}}{\|a - b\|_{\mathbb{R}^d}} \geq \tan\left(\frac{\pi}{2} - \theta\right),$$

which contradicts our assumption that F is K-Lipschitz. □

C.4 Convergence of Learned `Atlas` to Differentiable Atlas

Here, we prove under realistic assumptions that in limit of infinite samples, when learning a manifold $\mathcal{M} \subset \mathbb{R}^n$ with compact closure from using `Atlas-Learn`, the discrepancy between the resulting `Atlas` object and $\mathcal{M}$. It remains as future work to give a non-asymptotic result relating the learned `Atlas` structure and $\mathcal{M}$ (Niyogi et al., 2008, i.e., something in the spirit of).

[5] By our definition of L, we assume that linear dependence of $\mathbf{n}$ on $\mathbf{t}$ is negligible.

Claim 2. *Let $\iota : \mathcal{M} \hookrightarrow \mathbb{R}^D$ be a smooth embedding of a manifold $\mathcal{M}$ into $\mathbb{R}^D$. We assume that ι is smooth and that $\mathcal{M}$ has compact closure in $\mathbb{R}^D$. We further assume that $\mathcal{M}$ is δ-discernably embedded into $\mathbb{R}^D$ for $\delta > 0$ (i.e., for all $\vec{x} \in \operatorname{Im} \iota$ in the image of ι, the intersection of the D-dimensional ball of radius δ centered at $\vec{x}$ is homotopic to a point (e.g., Munkres, 2000). Lastly, we assume that $\mathcal{M}$, as a submanifold of $\mathbb{R}^D$, has κ-uniformly bounded sectional curvature. We use $\mathfrak{g}$ to denote the Riemannian metric that $\mathcal{M}$ inherits as a submanifold of $\mathbb{R}^D$.*

Say we have a measure μ on $\mathcal{M}$ that is measurable with respect to the uniform measure on $\mathcal{M}$. As the number of points n sampled from μ approaches infinity, the discrepancy

$$\|\varphi_i(\vec{\xi}) - \varphi_j(\psi_{ij}(\vec{\xi}))\|_{\mathbb{R}^D}$$

goes to zero for all $\vec{\xi} \in \mathcal{V}_{ij}$ whenever $\mathcal{V}_{ij}$ is defined.

Proof. The proof consists of four steps. Together, they allow us to relate the number n of points sampled from μ to both the number of charts k and the smallest radius r_0 of random balls centered at $\vec{x}_i$ to cover $\mathcal{M}$ with high probability. The proof is complete in establishing a relationship between n, k, and r_0 that results in vanishing discrepancy in probability.

1. First, we note that ι induces a metric space structure on $\mathcal{M}$, with distance $d_{\mathfrak{g}}$ given by the length of geodesics. Note that, for $\vec{x}, \vec{y} \in \mathcal{M}$, the distance $d_{\mathfrak{g}}(\vec{x}, \vec{y})$ is bounded below by $\|\vec{x} - \vec{y}\|_2$. By compactness of the closure of $\mathcal{M}$, if we take a fixed radius $r_0 > 0$ and a probability measure μ supported a.e. w.r.t. the uniform measure on $\mathcal{X}$, the expected number of $d_{\mathfrak{g}}$-balls of radius r_0 with centers sampled from μ needed to cover $\mathcal{X}$ is finite. Further, the variance of this number is bounded in Proposition 2.1 from Aldous (2022). In this way, as the number of balls generated by the process grows, the probability $p(n, \mu)$ that the closure of $\mathcal{M}$ is not covered vanishes.

2. By the above, for any positive $r_0 < \delta$, there exist $n > k > 0$ with $X = \{\vec{x}_1, \ldots, \vec{x}_n\}, \vec{x}_i \sim \mu$ and $\overline{X} \subset X, |\overline{X}| = k$ selected using a k-medoids algorithm variant such that the nearest distance from point $\vec{x}_i \in X$ to $\vec{m}_j \in \overline{X}$ is less than r_0 in expectation, with probability $p(k, \mu)$ that goes to zero as n goes to infinity and k goes to infinity more slowly than n. Specifically, we require k to grow with respect to n such that there are enough points in each subset S_i of the k-medoids partition that the quadratic fit learned in `Atlas-Learn` is fully determined with high probability for each subset. In this way, not only are the quadratic fits always determined, but the radius of each S_i is bounded above by $r_0 < \delta$.

3. By the κ-uniform bound on sectional curvature, condition i) in Section 2.1 of Little et al. (2017) is met for sufficiently small $r_0 < \delta$. Therefore, there is a sufficiently small $r_0 < \delta$ and sufficiently high n such that points within each S_i satisfy Assumption I of Section 3.3 of Little et al. (2017). In this way, a local tangent plane approximation L can be learned according to Theorem 1 of Little et al. (2017).

4. By 3), we have that $\mathcal{V}_i$ and φ_i are well defined for each S_i. Let $\mathcal{V}_j$ be such that $\mathcal{V}_{ij}$ is well defined, and let $\vec{\xi}$ be some element of $\mathcal{V}_{ij}$. Let $\mathcal{E}_{ij}$ denote the intersection of the MVEEs $\mathcal{E}_i$ and $\mathcal{E}_j$ defined by $A_i, \vec{b}_i, c_i$ and $A_j, \vec{b}_j, c_j$, respectively. By definition, $\varphi_i(\vec{\xi})$ is an $O(r_0^3)$ approximation of a point $\vec{x} \in \mathcal{E}_{ij}$, and $\varphi_j\left(\psi_{ij}(\vec{\xi})\right)$ is an $O(r_0^3)$ approximation of a point $\vec{y} \in \mathcal{E}_{ij}$. Because $\vec{x}$ and $\vec{y}$ both belong to $\mathcal{E}_{ij}$, the distance $\|\vec{x} - \vec{y}\|_2$ is of order $O(r_0^2)$. By the triangle inequality, the following series of deductions gives us the approximation order of the discrepancy between $\varphi_i(\vec{\xi})$ and $\varphi_j(\psi_{ij}(\vec{\xi}))$.

$$\begin{aligned}\left\|\varphi_i(\vec{\xi}) - \varphi_j\left(\psi_{ij}(\vec{\xi})\right)\right\|_2 &\leq \left\|\varphi_i(\vec{\xi}) - \vec{x}\right\|_2 + \|\vec{x} - \vec{y}\|_2 + \left\|\varphi_j\left(\psi_{ij}(\vec{\xi})\right) - \vec{y}\right\|_2 \\ &= O(r_0^3) + O(r_0^2) + O(r_0^3) \\ &= O(r_0^2)\end{aligned}$$

Taking the limit as r_0 goes to zero, and scaling n and k according to 2) above, completes the proof.

□

C.5 Time Complexity of `Atlas` Primitives Constructed via `Atlas-Learn`

In this section, we discuss the time complexity of base methods (i.e., the ones listed in Algorithm 1) implemented on `Atlas` objects, as implemented in one or more of our experiments. Methods used only the Grassmann experiments (Section 3.1) are discussed separately in Section D. Similarly, methods used only for computations on the Klein bottle—namely, length-adjusted retraction updates and logarithms on `Atlas` instances learned by `Atlas-Learn`—are discussed separately in Section E.

C.5.1 `in_domain`

In `Atlas-Learn` objects, the `in_domain` method evaluates the expression

$$\texttt{in\_domain}(\vec{\xi}) = \left[\varphi(\vec{\xi})^\top A\varphi(\vec{\xi}) + \vec{b}^\top\varphi(\vec{\xi}) + c \leq 0\right]. \tag{13}$$

Naive evaluation of this expression is done in $O(D^2)$ time. This complexity be improved upon by leveraging Equation (12); namely, we define

$$\begin{aligned} A_4 &= K^\top M^\top AMK, & A_3 &= L^\top AMK, & A_2 &= L^\top AL, \\ \vec{b}_2 &= K^\top M^\top A\vec{x}_0 + \frac{1}{2}KM\vec{b}, & \vec{b}_1 &= 2L^\top A\vec{x}_0 + L^\top\vec{b}, & c_0 &= \vec{x}_0^\top A\vec{x}_0 + \vec{b}^\top\vec{x}_0 + c, \end{aligned}$$

and we implement the computation as

$$\texttt{in\_domain}(\vec{\xi}) = \left[\vec{\xi}_{\otimes 2}^\top A_4\vec{\xi}_{\otimes 2} + \vec{\xi}^\top A_3\vec{\xi}_{\otimes 2} + \vec{\xi}^\top A\vec{\xi} + \vec{b}_2^\top\vec{\xi}_{\otimes 2} + \vec{b}_1^\top\vec{\xi} + c_0 \leq 0\right], \tag{14}$$

where $\vec{\xi}_{\otimes 2} = \vec{\xi}\otimes\vec{\xi}$. In this way, the complexity becomes $O(d^4)$, which is an improvement in the case that d is of order $o(\sqrt{D})$, at the cost of storing more arrays than A, $\vec{b}$, and c per chart.

C.5.2 `identify_new_chart`

When a retraction `Ret` takes a point $\vec{\xi}\in\mathcal{V}_i$ along a tangent vector $\vec{\tau}\in T_{\vec{\xi}}\mathcal{V}_i$ to or past its boundary, identifying the chart to which $\texttt{Ret}_{\vec{\xi}}(\vec{\tau})$ best belongs can be assessed using different heuristics, each possessing different tradeoffs. Using the $\vec{x}_i$ learned from the local quadratic approximation in each $\mathcal{V}_i$, locally sensitive hashing can be used to identify the k nearest charts to query for constant k. The goodness of fit of a new chart $\vec{V}_j$ can be assessed by how well a point lies within the MVEE of that chart, as defined by A_i, $\vec{b}_i$, and c_i according to the lefthand side of Inequality 13; this can be done after mapping $\texttt{Ret}_{\vec{\xi}}(\vec{\tau})$ to the ambient space in $O(D^2)$ time, or within $\mathcal{V}_j$ in $O(d^4)$ time. Alternatively, the goodness of fit can be assessed by the error of the local quadratic approximation of $\mathcal{V}$, which can be done with the same time complexities for ambient and $\mathcal{V}_j$ evaluation; we assess goodness of fit in this second way in our `Atlas` objects learned by `Atlas-Learn`. Either way, `identify_new_chart` can be done in $O(cD^2 + cDd^2)$ or $O(cd^4)$ time.

C.5.3 The Coordinate Chart Map φ_i

Chart coordinates $\vec{\xi}\in\mathcal{V}_i$ are mapped by φ_i into the ambient space by evaluating the righthand side of Approximation 12, which is done in $O(Dd^2)$ time.

C.5.4 The Differential $\mathrm{D}\varphi_i$

For $\vec{\xi}\in\mathcal{V}_i$, if a tangent vector $\tau\in T_{\varphi_i(\vec{\xi})}\mathrm{Im}\varphi_i$ is represented in the ambient space explicitly, implementing the retraction $R_{\varphi_i(\vec{\xi})}$ from Equation 2 takes $O(Dd) + \kappa(D,d) + \tilde{\kappa}(D,d)$ time, where D is the ambient dimension, d is the dimension of the `Atlas` object, $\kappa(D,d)$ is the time complexity of evaluating $\mathrm{D}\varphi_i$ as a matrix, and $\tilde{\kappa}(D,d)$ is time complexity of (pseudo-) inverting this matrix. When learned using `Atlas-Learn`, an `Atlas` object implements $\mathrm{D}\varphi_i$ as

$$\mathrm{D}\varphi_i(\vec{\xi}) = M_iK_i\left(\vec{\xi}\otimes\vec{1}\right) + L_i \tag{15}$$

when wanting to act on tangent vectors $\vec{v} \in T_{\varphi_i(\vec{\xi})}\mathcal{M}$ in ambient coordinates, but is the identity for tangent vectors $\vec{\tau} \in T_{\vec{\xi}}\mathcal{V}_i$ in chart coordinates (see Sec. C.2). When acting on tangent vector $\vec{v} \in T_{\varphi_i(\vec{\xi})}\mathcal{M}$ in ambient coordinates[6], $\kappa(D,d) = O(D^2(D-d))$, and $\tilde{\kappa}(D,d) = O(Dd^2)$[7]. This is the time complexity relevant for implementing $\mathrm{D}\varphi_i$ in the Dynamo vector field integration (Section 3.4). When a tangent vector is represented implicitly by an element of the tangent space $T_{\vec{\xi}}\mathcal{V}$, then $\mathrm{D}\varphi_i(\vec{\xi})$ is an implicit identity operation, and the retraction takes $O(d)$ time to compute for the vector addition. This representation of the tangent vector as an element of $T_{\vec{\xi}}\mathcal{V}$ is done in the RPB experiment (Section 3.3).

When there are too few samples for `Atlas-Learn` to determine a unique quadratic fit within a neighborhood, but enough to determine a linear approximation, we learn L and $\vec{x}_0$ as described in Algorithm 2.3, but set the trilinear form K to the zero map. This is the case in running `Atlas-Learn` with 30 charts on the Dynamo data in Section 3.4. In this scenario, the runtime $\kappa(D,d)$ of $\mathrm{D}\varphi$ becomes $O((D-d)d)$, as the quadratic term is always zero.

C.5.5 The Transition Map ψ_{ij}

The transition map ψ_{ij} is evaluated as the equation

$$\psi_{ij}(\vec{\xi}) = L_j^\top \left[\frac{1}{2} M_i K_i \left(\vec{\xi} \otimes \vec{\xi}\right) + \left(L_i \vec{\xi} + \vec{x}_i\right) - \vec{x}_j\right]. \tag{16}$$

When the products $L_j^\top M_i K_i$, $L_j^\top L_i$, and $L_j^\top (\vec{x}_i - \vec{x}_j)$ are precomputed, this evaluation can be done in $O(d^3)$ time. At first computation, the overhead times for computing these products are $O(Dd(D-d))$, $O(Dd^2)$, and $O(Dd)$, respectively. When $L_j^\top M_i K_i$ is known to be the zero matrix, as may happen when there are too few points for a quadratic fit in a neighborhood, the time complexity reduces to $O(d^2)$.

C.5.6 The Differential $\mathrm{D}\psi_{ij}$

To map a tangent vector $\vec{\tau}$ from $T_{\vec{\xi}}\mathcal{V}_i$ to $T_{\vec{\eta}}\mathcal{V}_j$ for any $\vec{\eta} \in \mathcal{V}_j$, we evaluate

$$\mathrm{D}\psi_{ij}(\vec{\xi}, \vec{\eta}, \vec{\tau}) = L_j^\top \left(M_i K_i \left(\vec{\xi} \otimes \vec{1}\right) + L_i\right) \vec{\tau}. \tag{17}$$

If $L_j^\top M_i K_i$ and $L_j^\top L_i$ are precomputed, as discussed in Section C.5.5, this computation can be done in $O(d^3)$ time in the case of nonzero K_i and $O(d^2)$ time otherwise.

D METHODS FOR PERFORMING ONLINE GRASSMANN FRÉCHET MEAN ESTIMATION

Experiments were performed using one core on an Intel Xeon Gold-6448Y processor, 100GB RAM, and the Linux 4.18.0-305.3.1.el8.x86_64 operating system.

D.1 An `Atlas` Representation of the (n,k)-Grassmannian

The `Atlas` structure is constructed from a conventional atlas of $\mathbf{Gr}_{n,k}$ derived from a cell complex presented by Ehresmann (1934), which we adapt to our notation for convenience. Our `Atlas` has charts from the Ehresmann atlas and also permits *ad hoc* creation of new coordinate charts centered at any point in the manifold. We use *ad hoc* chart creation to maintain proximity to the origin in compressed charts and, hence, accuracy of quasi-Euclidean updates, which results in fast online learning.

In addition to the Ehresmann atlas, there exists another canonical representation of the Grassmann manifold as a quotient of the Stiefel matrices (Bendokat et al., 2011). We find the Ehresmann atlas simpler to present, but also use the Stiefel construction (Algorithm 7 and Fig. 6) to enable direct comparisons with existing Riemannian

[6] Here, we assume the product $M_i K_i$ is cached.

[7] $\tilde{\kappa}(D,d) = O(Dd^2)$ here assuming the Moore-Penrose inverse is computed by singular value decomposition (SVD) Li et al. (2019).

optimization approaches that use it, such as Pymanopt (Townsend et al., 2016) and GiFEE (Chakraborty and Vemuri, 2015).

D.2 Coordinate Charts of the Ehresmann Atlas

We begin with some intuition for the Ehresmann atlas construction. The (n,k)-Grassmannian can be understood as the manifold of $n \times n$ orthogonal projection matrices of rank k. There are $\binom{n}{k}$ such matrices whose entries are all zero, save for exactly k diagonal entries that are equal to one. These matrices are in one-to-one correspondence with sets of k fixed integers $i_1, \ldots, i_k$ satisfying $1 \leq i_1 < \ldots < i_k \leq n$, and therefore with the permutations[8]

$$\begin{aligned} \pi_{i_1,\ldots,i_k} : \{1,\ldots,n\} &\to \{1,\ldots,n\} \\ j &\mapsto \begin{cases} i_j, & j \leq k \\ \min_{(j-k)} \left(\{1,\ldots,n\} \setminus \{i_1,\ldots,i_k\}\right), & j > k \end{cases}. \end{aligned} \tag{18}$$

Let $P_{i_1,\ldots,i_k}$ be the $n \times n$ permutation matrix corresponding to $\pi_{i_1,\ldots,i_k}$. The aforementioned $\binom{n}{k}$ matrices take the form

$$P_{i_1,\ldots,i_k} \begin{pmatrix} I_k \\ \mathbf{0}_{n-k,k} \end{pmatrix} \begin{pmatrix} I_k \\ \mathbf{0}_{n-k,k} \end{pmatrix}^\top P_{i_1,\ldots,i_k}^\top.$$

These $\binom{n}{k}$ points are the centerpoints of each coordinate chart in the Ehresmann atlas. Further, for every $P \in \mathbf{Gr}_{n,k}$, there exists a permutation $\pi_{i_1,\ldots,i_k}$ and a matrix $A \in \mathbb{R}^{(n-k)\times k}$ such that $P = P_{i_1,\ldots,i_k} \left(\frac{I_k}{A}\right) \left(P_{i_1,\ldots,i_k} \left(\frac{I_k}{A}\right)\right)^\dagger$, where $\dagger$ denotes the Moore-Penrose pseudoinverse.

For the coordinate charts in the Ehresmann atlas, let $\mathbf{colproj} : \mathbb{R}^{n\times k} \to \mathbb{R}^{n\times n}$ be the map that takes a matrix A to the orthogonal projector onto the column space of A. The coordinate chart $\varphi_{i_1,\ldots,i_k} : \mathcal{V}_{i_1,\ldots,i_k} \to \mathcal{U}_{i_1,\ldots,i_k}$ is then defined as:

- $\varphi^{-1}_{i_1,\ldots,i_k} : A \mapsto \mathbf{colproj}\left(P_{i_1,\ldots,i_k} \left(\frac{I_k}{A}\right)\right)$
- $\mathcal{V}_{i_1,\ldots,i_k} = \mathbb{R}^{(n-k)\times k}$, and
- $\mathcal{U}_{i_1,\ldots,i_k} = \operatorname{im}\left(\varphi^{-1}_{i_1,\ldots,i_k}\right)$.

For convenience, we denote the coordinate chart corresponding to the identity permutation as $\varphi_0 : \mathcal{V}_0 \to \mathcal{U}_0$. In Claims 5 and 6 (Appendix D.7.2), we demonstrate that a point $\vec{\xi}$ in some chart $\mathcal{V}_{i_1,\ldots,i_k}$ of the Ehresmann atlas is closer to the center of another chart than to the center of $\mathcal{V}_{i_1,\ldots,i_k}$ if any coordinate of $\vec{\xi}$ exceeds one (Section D.4). We use this result to determine when to generate new *ad hoc* charts. The creation of *ad hoc* coordinate charts allows for points to always be close to the origin within a coordinate chart, thereby reducing the error of quasi-Euclidean updates. In the case that $P \in \mathbf{Gr}_{n,k}$ is closer to the center of $\mathcal{V}$ than to the center of any Ehresmann chart, we say that $\mathcal{V}$ is the "best" Ehresmann chart for P.

D.3 Ad Hoc Formation of Coordinate Charts

To perform *ad hoc* chart formation, we begin with the case of generating charts centered at projection matrices belonging to $\mathcal{U}_0$. This case generalizes to projection matrices belonging to $\mathcal{U}_{i_1,\ldots,i_k}$ by simple conjugation by $P_{i_1,\ldots,i_k}$. The group $\mathbf{O}(n)$ of $n \times n$ orthogonal matrices acts transitively on $\mathbf{Gr}_{n,k}$ according to the action $Q : P \mapsto QPQ^\top$. Therefore, for all $A \in \mathcal{V}_0$, there exists an orthogonal matrix Q_A such that

$$\varphi_0^{-1}(A) = Q_A \left(\begin{array}{c|c} I_k & \mathbf{0}_{k,n-k} \\ \hline \mathbf{0}_{n-k,k} & \mathbf{0}_{n-k,n-k} \end{array}\right) Q_A^\top. \tag{19}$$

Claim 3. *An orthogonal matrix Q_A which satisfies Equation 19 is given by*

$$Q_A = \left(\begin{array}{c|c} \sqrt{(I_k + A^\top A)^{-1}} & -A^\top \sqrt{(I_{n-k} + AA^\top)^{-1}} \\ \hline A\sqrt{(I_k + A^\top A)^{-1}} & \sqrt{(I_{n-k} + AA^\top)^{-1}} \end{array}\right).$$

[8] For finite $S \subset \mathbb{R}$, the notation $\min_{(m)} S$ denotes the mth smallest element of S.

Algorithm 3 `QA_from_A` (Q_A for $A \in \mathcal{V}_0$ according to Claim 3)

Require: $A \in \mathbb{R}^{n-k,k}$

$$Q_A \leftarrow \left(\begin{array}{c|c} \sqrt{(I_k + A^\top A)^{-1}} & -A^\top \sqrt{(I_{n-k} + AA^\top)^{-1}} \\ \hline A\sqrt{(I_k + A^\top A)^{-1}} & \sqrt{(I_{n-k} + AA^\top)^{-1}} \end{array}\right) \qquad \triangleright\ O(n^3 + n^2k + nk^2 + k^3) \text{ time}$$

return Q_A

Proof. See Section D.7.2. □

Claim 3 allows us to change coordinate charts so that quasi-Euclidean updates approximate the exponential map more accurately, as done in Algorithm 7. While this specific method only works for points in the Ehresmann chart $\varphi_0 : \mathcal{V}_0 \to \mathcal{U}_0$, it generalizes to charts $\varphi_{i_1,\ldots,i_k} : \mathcal{V}_{i_1,\ldots,i_k} \to \mathcal{U}_{i_1,\ldots,i_k}$ by replacing the action in Equation 19 with the action

$$\varphi^{-1}_{i_1,\ldots,i_k}(A) = Q_A P_{i_1,\ldots,i_k} \left(\begin{array}{c|c} I_k & \mathbf{0}_{k,n-k} \\ \hline \mathbf{0}_{n-k,k} & \mathbf{0}_{n-k,n-k} \end{array}\right) P^\top_{i_1,\ldots,i_k} Q_A^\top. \tag{20}$$

This creates coordinate charts $\varphi_{A,i_1,\ldots,i_k} : \mathcal{V}_{A,i_1,\ldots,i_k} \to \mathcal{U}_{A,i_1,\ldots,i_k}, \varphi_{A,i_1,\ldots,i_k}$ for all $A \in \mathbb{R}^{(n-k)\times k}$ and all $P_{i_1,\ldots,i_k}$, with the transition map $\psi_{A,i_1,\ldots,i_k \to A',j_1,\ldots,j_k}$ taking $B \in \mathcal{V}_{i_1,\ldots,i_k}$ to

$$(\mathbf{0}_{n-k,k} \mid I_{n-k})\, R\, ((I_k \mid \mathbf{0}_{k,n-k})\, R)^{-1}, \tag{21}$$

where $R = P^\top_{j_1,\ldots,j_k} Q^\top_{A'} Q_A P_{i_1,\ldots,i_k} \left(\frac{I_k}{B}\right)$. Within any coordinate chart $\mathcal{V}_{A,i_1,\ldots,i_k}$, transition maps are invoked whenever any element of the coordinates B exceeds one (Section D.4), transitioning into a chart centered at $\varphi_{A,i_1,\ldots,i_k}(B)$.

D.4 Transition Maps in the `Atlas` Representation of the Grassmannian

Here, we explain when we invoke transition maps between charts on the `Atlas` representation of the Grassmann manifold $\mathbf{Gr}_{n,k}$ given in Sec. D.1. For a given Ehressman chart, the adjacent charts are determined by the permutation Eq. 18, one existing for every dimension of the manifold. The condition under which we change charts within this `Atlas` representation is given in Claim 4. For this reason, a transition boundary exists for each coordinate such that, if the coordinate exceeds an absolute value of one, the transition map is invoked.

Claim 4. *An element A of a fixed Ehressman chart is closer to the center of the chart than to the center of any other Ehressman chart if all of the elements of A have absolute value less than one.*

Proof. The claim is proved for the Ehressman chart $\varphi_0 : \mathcal{V}_0 \to \mathcal{U}_0$ by Claims 5 and 6. For the remaining Ehressman charts, the Claim is proved by observing we can conjugate elements of $\mathcal{U}_0$ by the permutation matrices used to define the remaining Ehressman charts. □

In the online Fréchet mean experiment, transition maps between coordinate charts are implemented according to Algorithm 6, whose runtime scales with complexity $O(n^3 + n^2k + nk^2 + k^3)$ but is constant with respect to the number of compressed charts traversed. Since the quasi-Euclidean updates have $O(nk)$ time complexity, while the first-order update schemes for GiFEE, MANOPT, and MANOPT-RET have $\Omega(nk)$ time-complexity, the ATLAS framework can outspeed the other schemes when it can make infrequent use of high-cost chart transitions.

D.5 Miscellaneous Subroutines

The online Fréchet mean estimation on the `Atlas` structure(Algorithm 7) depends on the subroutines `Atlas_Grassmann_identify_chart` (Algorithm 4), which identifies the closest Ehressman chart, and `Atlas_Grassmann_ingest_matrix` (Algorithm 5), which gives the representation of matrix in the current coordinate chart of the `Atlas`.

(a) `GifeeLog`

Require: $\mathcal{X}, \mathcal{Y} \in \mathbf{Gr}_{n,k}$
Require: $X \in \mathbb{R}^{n\times k}, \mathbf{colproj}(X) = \mathcal{X}$
Require: $Y \in \mathbb{R}^{n\times k}, \mathbf{colproj}(Y) = \mathcal{Y}$
$A \leftarrow \left(I - X\left(X^\top X\right)^{-1} X^\top\right) Y \left(X^\top Y\right)^{-1}$
$U, \Sigma, V^\top \leftarrow \texttt{ThinSVD}(A)$
return $U, \Sigma, V^\top$

(b) `ManoptLog`

Require: $\mathcal{X}, \mathcal{Y} \in \mathbf{Gr}_{n,k}$
Require: $X \in \mathbb{R}^{n\times k}, \mathbf{colproj}(X) = \mathcal{X}$
Require: $Y \in \mathbb{R}^{n\times k}, \mathbf{colproj}(Y) = \mathcal{Y}$
$A \leftarrow \left(I - X\left(X^\top X\right)^{-1} X^\top\right) Y \left(X^\top Y\right)^{-1}$
$U, \Sigma, V^\top \leftarrow \texttt{ThinSVD}(A)$
$\Theta \leftarrow \arctan \Sigma$
return $U\Theta V^\top$

(c) `GifeeExp`

Require: $\mathcal{X} \in \mathbf{Gr}_{n,k}$
Require: $X \in \mathbb{R}^{n\times k}, \mathbf{colproj}(X) = \mathcal{X}$
Require: $U, \Sigma, V^\top$ from `GifeeLog`
Require: iteration number $i > 0$
$\Theta \leftarrow \arctan \Sigma$
$\tilde{Z} \leftarrow XV \cos(\Theta/i) + U \sin(\Theta/i)$
$Z, R \leftarrow$ QR decomposition of $\tilde{Z}$
return Z

(d) `ManoptExp`

Require: $\mathcal{X} \in \mathbf{Gr}_{n,k}$
Require: $X \in \mathbb{R}^{n\times k}, \mathbf{colproj}(X) = \mathcal{X}$
Require: L from `ManoptLog`
Require: iteration number $i > 0$
$U, \Sigma, V^\top \leftarrow \texttt{ThinSVD}(L/i)$
$\tilde{Z} \leftarrow XV \cos(\Sigma)V^\top + U \sin(\Sigma)V^\top$
$Z, R \leftarrow$ QR decomposition of $\tilde{Z}$
return Z

(e) `ManoptRet`

Require: $\mathcal{X} \in \mathbf{Gr}_{n,k}$
Require: $X \in \mathbb{R}^{n\times k}, \mathbf{colproj}(X) = \mathcal{X}$
Require: L from `ManoptLog`
Require: iteration number $i > 0$
$U, \Sigma, V^\top \leftarrow \texttt{ThinSVD}(X + L/i)$
$\tilde{Z} \leftarrow UV^\top$
$Z, R \leftarrow$ QR decomposition of $\tilde{Z}$
return Z

Figure 6: **Riemannian logarithm and retraction algorithms used by the non-ATLAS first-order update schemes in Sec. 3.1.**

D.6 Non-Uniform Sampling of the Grassmann Manifold by Geodesic Power Scaling

Because of the lack of existing benchmark datasets, we introduce the *geodesic power distribution* $\mathbf{GPD}(X, p)$ with Fréchet mean $\mathcal{X} \in \mathbf{Gr}_{n,k}$ and scaling exponent $p > 1$, which inversely controls the entropy of the distribution (formally defined in Section D.6). The **GPD** is a natural, efficiently samplable distribution, which guarantees existence and uniqueness of the Fréchet mean $\mathcal{X}$. For the Grassmann experiments, points for online sampling were generated by sampling randomly from the Grassmann manifold. To generate points from a distribution with a well-defined Fréchet mean, such as the uniform measure on balls of a fixed radius on the Grassmannian, we used the following method, which we call *geodesic power scaling*.

Fix $p > 1$ and $\mathcal{X} \in \mathbf{Gr}_{n,k}$. Points are sampled from the *geodesic power distribution* $\mathbf{GPD}(\mathcal{X}, p)$ as follows:

1. A point $\mathcal{Y}$ is sampled uniformly from $\mathbf{Gr}_{n,k}$.
2. The Grassmann distance δ between $\mathcal{X}$ and $\mathcal{Y}$ is computed.
3. A new point $\mathcal{Y}'$ is computed as $\mathcal{Y}' = \exp_{\mathcal{X}}\left(\left(\frac{\delta}{\delta_{\max}}\right)^p \log_{\mathcal{X}} \mathcal{Y}\right)$, where $\delta_{\max}$ is the largest possible Grassmann distance between two points on $\mathbf{Gr}_{n,k}$.[9]

Whenever we sample from $\mathbf{GPD}(\mathcal{X}, p)$, we assume that the sample takes the form of a matrix $X \in \mathbb{R}^{n\times k}$ such that $\mathcal{X}$ is the column space of X. Increasing the scaling exponent p reduces the entropy of the distribution by concentrating probability around $\mathcal{X}$; as $p \to \infty$, $\mathbf{GPD}(\mathcal{X}, p)$ approaches the Dirac delta based at $\mathcal{X}$. Note that the distribution is invariant under action by the special orthogonal group, as long as the action fixes $\mathcal{X}$, giving the distribution "rotational symmetry" about $\mathcal{X}$. Moreover, $\mathbf{GPD}(\mathcal{X}, p)$ has unique population Fréchet mean $\mathcal{X}$ for all $p > 1$.

[9] $\delta_{\max} = \frac{\pi}{2}\sqrt{\max\{k, n-k\}}$

Algorithm 4 `Atlas_Grassmann_identify_chart` (identify chart in Ehresmann atlas whose center is closest to **colspan**X)

Require: $X \in \mathbb{R}^{n \times k}$, full rank
  $P \leftarrow X \left(X^\top X\right)^{-1} X^\top$ ▷ $P = \mathbf{colproj} X$; $O(n^2k + nk^2 + k^3)$ time
  for $j \in \{1, \ldots, k\}$ **do** ▷ $O(nk)$ time
    $i_j \leftarrow i$ such that P_{ii} is jth largest diagonal entry of P
    $i_1, \ldots, i_k \leftarrow i_1, \ldots, i_k$ in increasing order
  return $i_1, \ldots, i_k$

Algorithm 5 `Atlas_Grassmann_ingest_matrix` (ingest Grassmann element represented as full-rank matrix into Ehresmann chart)

Require: $X \in \mathbb{R}^{n \times k}$, full rank
Require: $1 \leq i_1 \leq \ldots \leq i_k \leq n$ specifying Ehresmann chart
  $X_U \leftarrow$ restriction of X to rows in $\{i_1, \ldots, i_k\}$ ▷ $O(1)$ time
  $X_L \leftarrow$ restriction of X to rows not in $\{i_1, \ldots, i_k\}$ ▷ $O(1)$ time
  $A \leftarrow X_L X_U^{-1}$ ▷ $A = \varphi_{i_1,\ldots,i_k}\left(\mathbf{colproj}\left(X\right)\right)$; $O(nk^2 + k^3)$
  return A

A **GPD** satisfies the L^2-moment constraint for $p > 1$; while it does not satisfy the support constraint, the probability density function $\mathbf{GPD}(\mathcal{X}, p)$ is close to zero for most points $\mathcal{X} \in \mathbf{Gr}_{n,k}$ for sufficiently high p. For the sake of fair comparison, points are sampled from $\mathbf{GPD}\left(\mathcal{X}, p\right)$ not as orthogonal projection matrices P, but as Stiefel matrices X satisfying $P = XX^\top$. This is done by implementing the procedure of sampling from $\mathbf{GPD}\left(\mathcal{X}, p\right)$ (Section D.6) in Pymanopt, which represents elements of $\mathbf{Gr}_{n,k}$ in terms of Stiefel matrices.

D.7 Theoretical Results for the Grassmann Manifold

D.7.1 Ingesting Columnspanning Matrix into the `Atlas` Representation of the Grassmannian

Say $X \in \mathbb{R}^{n \times k}$ has full rank. Thinking of $\mathbf{Gr}_{n,k}$ as the manifold of $n \times n$ orthogonal projection matrices of rank k, we know that the columnspace of X is uniquely represented in $\mathbf{Gr}_{n,k}$ by $X \left(X^\top X\right)^{-1} X^\top$. Finding the chart to which X belongs is tantamount to finding the "centerpoint" projection matrix to which $X \left(X^\top X\right)^{-1} X^\top$ is closest. This, in turn, is equivalent to finding the centerpoint projection matrix with which $X \left(X^\top X\right)^{-1} X^\top$ has the highest Frobenius inner product. This is accomplished by finding the k largest diagonal entries of $X \left(X^\top X\right)^{-1} X^\top$, as demonstrated by the following series of deductions.

$$\begin{aligned}
\left\langle X \left(X^\top X\right)^{-1} X^\top, \sum_{j=1}^{k} \vec{e}_{i_j} \vec{e}_{i_j}^\top \right\rangle_{\text{Fr}} &= \mathbf{Tr}\left[X \left(X^\top X\right)^{-1} X^\top \sum_{j=1}^{k} \vec{e}_{i_j} \vec{e}_{i_j}^\top \right] \\
&= \sum_{j=1}^{k} \mathbf{Tr}\left[X \left(X^\top X\right) X^\top \left(\vec{e}_{i_j} \vec{e}_{i_j}^\top \right) \right] \\
&= \sum_{j=1}^{k} \left[X \left(X^\top X\right)^{-1} X^\top \right]_{i_j i_j}
\end{aligned}$$

Note that the diagonal entries of $X \left(X^\top X\right)^{-1} X^\top$ are guaranteed to be nonnegative by the positive-semidefiniteness of $X \left(X^\top X\right)^{-1} X^\top$.

More generally, let Q and P be orthogonal matrices such that

$$Q \left(\begin{array}{c|c} I_k & \mathbf{0}_{k,n-k} \\ \hline \mathbf{0}_{k,n-k} & \mathbf{0}_{n-k,n-k} \end{array} \right) Q^\top = P.$$

Algorithm 6 `Atlas_Grassmann_transition_map` (transition map on Grassmann `Atlas`, used in Alg. 7)

Require: $A \in \mathbb{R}^{(n-k)\times k}$, permutation indices $i_1, \ldots, i_k$
Require: Q_A ▷ output by `QA_from_A`(A), Alg. 3
$P \leftarrow$ `permutation_matrix` $(i_1, \ldots, i_k)$ ▷ $O(n)$ time
$Y \leftarrow Q_A P \begin{pmatrix} I_k \\ A \end{pmatrix}$ ▷ $O(n^3 + n^2 k)$ time
$i_1, \ldots, i_k \leftarrow$ `ATLAS_identify_chart` (Y) ▷ Alg. 4; $O(n^2 k + nk^2 + k^3)$ time
$\tilde{A} \leftarrow$ `ATLAS_ingest_matrix` $(Y, i_1, \ldots, i_k)$ ▷ Alg. 5; $O(nk^2 + k^3)$ time
$\tilde{Q}_A \leftarrow$ `QA_from_A` $\left(\tilde{A}\right)$ ▷ Alg. 3; $O(n^3 + n^2 k + nk^2 + k^3)$ time
$Q_A \leftarrow P \tilde{Q}_A P^\top$ ▷ $O(n^3)$ time
$A \leftarrow \mathbf{0}_{n-k,k}$ ▷ $O(1)$ time
return A, Q_A

Algorithm 7 `Atlas_Grassmann_Frechet`
Online Fréchet mean estimation on $\mathbf{Gr}_{n,k}$ using quasi-Euclidean updates on `Atlas`

Require: Probability distribution $\mathcal{D}$ on $\mathbf{Gr}_{n,k}$
Require: Fréchet stream of samples $X_1, X_2, \ldots \sim \mathcal{D}$, sampled as Stiefel matrices
$i_1, \ldots, i_k \leftarrow$ `ATLAS_identify_chart` (X_1) ▷ Best Ehresmann chart (Sec. D.2); Alg. 4
$A \leftarrow$ `ATLAS_ingest_matrix` $(X_1, i_1, \ldots, i_k)$ ▷ $A = \varphi_{i_1,\ldots,i_k}(\mathbf{colproj}(X_1))$; Alg. 5
$Q_A \leftarrow I_n$ ▷ Q_A is used to define the map (19)
$Q_{A,U} \leftarrow (I_k \mid \mathbf{0}_{k,n-k})^\top$ ▷ restriction of Q_A to columns in $\{i_1, \ldots, i_k\}$
$Q_{A,L} \leftarrow (I_{n-k} \mid \mathbf{0}_{n-k,k})^\top$ ▷ restriction of Q_A to columns not in $\{i_1, \ldots, i_k\}$
$n \leftarrow 1$
while streaming **do**
  $n \leftarrow n + 1$
  $\tilde{A} \leftarrow Q_{A,L}^\top X_n \left(Q_{A,U}^\top X_n\right)^{-1}$ ▷ $\tilde{A} = \varphi_{i_1,\ldots,i_k}\left(Q_A^\top \mathbf{colproj}(X_n) Q_A\right)$; Eq. (20)
  $A \leftarrow A + \left(\tilde{A} - A\right)/n$ ▷ Update online Fréchet mean estimator
  if any entry a of A violates $|a| < 1$ **then** ▷ change chart if necessary; Claim 5
    $A, Q_A \leftarrow$ `Atlas_Grassmann_transition_map`$(A, i_1, \ldots, i_k, Q_A)$ ▷ Alg. 6
    $Q_{A,U} \leftarrow$ restriction of Q_A to columns in $\{i_1, \ldots, i_k\}$
    $Q_{A,L} \leftarrow$ restriction of Q_A to columns not in $\{i_1, \ldots, i_k\}$
return $A, i_1, \ldots, i_k, Q_A$

We ingest a full-rank matrix $X \in \mathbb{R}^{n\times k}$ into the chart centered at P by the map

$$X \mapsto X_L X_U^{-1}, \tag{22}$$

where X_U, X_L are given by

$$X_U = (I_k \mid \mathbf{0}_{k,n-k})\, Q^\top X, \qquad X_L = (\mathbf{0}_{n-k,k} \mid I_{n-k})\, Q^\top X.$$

D.7.2 Distances on the Grassmann Manifold

Claim 5. *Let φ_0 be the Ehresmann coordinate chart map (Sec. D.1). For $t \in \mathbb{R}$, we define $V_t := \varphi_0(t\vec{e}_{n-k}\vec{e}_k) = \begin{pmatrix} I_k \\ t\vec{e}_{n-k}\vec{e}_k^\top \end{pmatrix}$. The Grassmann distance between $\mathbf{colproj}(V_t)$ and $\mathbf{colproj}(U)$, where $U := \varphi_0(\mathbf{0}_{n-k,k}) = \begin{pmatrix} I_k \\ \mathbf{0}_{n-k,k} \end{pmatrix}$, is equal to $|\arctan t|$.*

Proof. The square of the Grassmann distance $\mathbf{dist}_{\mathbf{Gr}}$ between two projection matrices $P, Q \in \mathbf{Gr}_{n,k}$ is given by the sum of squares of Jordan angles between their subspaces. From Lemma 5 in Neretin (2001), if matrices $U_P, U_Q \in \mathbb{R}^{n\times k}$ satisfy $P = \mathbf{colproj} U_P$ and $Q = \mathbf{colproj} U_P$, the squares of cosines of the Jordan angles between P and Q are the eigenvalues of the matrix $\left(U_P^\top U_P\right)^{-1} U_P^\top U_Q \left(U_Q^\top U_Q\right)^{-1} U_Q^\top U_P$. Therefore, the following series

of deductions holds.

$$\begin{aligned}
\mathbf{dist_{Gr}}\left(\varphi_0\left((t\vec{e}_k\vec{e}_k^{\top})\right), \varphi_0\left(\mathbf{0}_{n-k,k}\right)\right) &= \sqrt{\mathbf{Tr}\left[\arccos\left(\sqrt{(U^\top U)^{-1}U^\top V_t\left(V_t^\top V_t\right)^{-1}V_t^\top U}\right)^2\right]} \\
&= \sqrt{\mathbf{Tr}\left[\arccos\left(\sqrt{\left(I_k + t^2\vec{e}_k\vec{e}_k^{\top}\right)^{-1}}\right)^2\right]} \\
&= \sqrt{\mathbf{Tr}\left[\arccos\left(\sqrt{I_k - \frac{t^2}{1+t^2}\vec{e}_k\vec{e}_k^{\top}}\right)^2\right]} \\
&= \sqrt{\mathbf{Tr}\left[\left(\begin{array}{c|c} \mathbf{0}_{k-1,k-1} & \mathbf{0}_{k-1,1} \\ \hline \mathbf{0}_{1,k-1} & \arccos\left(\sqrt{\frac{1}{1+t^2}}\right)\end{array}\right)^2\right]} \\
&= |\arctan t|
\end{aligned}$$

□

Claim 6. *Let* $Q := \left(\begin{array}{c|c} \mathbf{0}_{n-1,1} & I_{n-1} \\ \hline 1 & \mathbf{0}_{1,n-1}\end{array}\right)$ *be the permutation matrix which moves up the indices of row vectors. Further, let* U, V_t *be as in Claim 5, and let* $U_\varnothing = QU$*. The Grassmann distance between* $\mathbf{col}\left(V_t\right)$ *and* $\mathbf{col}U_\varnothing$ *is equal to* $|\operatorname{arccot} t|$*.*

Proof. Following the discussion of Jordan angles in Neretin (2001), we get:

$$\begin{aligned}
\mathbf{dist_{Gr}}\left(\mathbf{col}V_t, \mathbf{col}U_\varnothing\right) &= \left(\mathbf{Tr}\left[\arccos\left(\left[\left(U_\varnothing^\top U_\varnothing\right)^{-1}U_\varnothing^\top V_t\left(V_t^\top V_t\right)^{-1}V_t^\top U_\varnothing\right]^{1/2}\right)^2\right]\right)^{1/2} \\
&= \left(\mathbf{Tr}\left[\arccos\left(\left[\left(I_k \mid \mathbf{0}_{k,n-k}\right)Q^\top\left(\frac{I_k}{t\vec{e}_{n-k}\vec{e}_k^{\top}}\right)\left(I_k - \frac{t^2}{1+t^2}\vec{e}_k\vec{e}_k^{\top}\right)\right.\right.\right. \\
&\qquad \left.\left.\left. \cdot\left(I_k \mid t\vec{e}_k\vec{e}_{n-k}^{\top}\right)Q\left(\frac{I_k}{\mathbf{0}_{n-k,k}}\right)\right]^{1/2}\right)^2\right]\right)^{1/2} \\
&= \left(\mathbf{Tr}\left[\arccos\left(\left[\left(\begin{array}{c|c} \mathbf{0}_{1,k-1} & t \\ \hline I_{k-1} & \mathbf{0}_{k-1,1}\end{array}\right)\left(I_k - \frac{t^2}{1+t^2}\vec{e}_k\vec{e}_k^{\top}\right)\right.\right.\right. \\
&\qquad \left.\left.\left. \cdot\left(\begin{array}{c|c} \mathbf{0}_{k-1,1} & I_{k-1} \\ \hline t & \mathbf{0}_{1,k-1}\end{array}\right)\right]^{1/2}\right)^2\right]\right)^{1/2} \\
&= \left(\mathbf{Tr}\left[\arccos\left(\left[\left(\begin{array}{c|c} t^2 & \mathbf{0}_{1,k-1} \\ \hline \mathbf{0}_{k-1,1} & I_{k-1}\end{array}\right) - \frac{t^4}{1+t^2}\vec{e}_1\vec{e}_1^{\top}\right]^{1/2}\right)^2\right]\right)^{1/2} \\
&= \left(\mathbf{Tr}\left[\arccos\left(\left(\begin{array}{c|c} \frac{t^2}{1+t^2} & \mathbf{0}_{1,k-1} \\ \hline \mathbf{0}_{k-1,1} & I_{k-1}\end{array}\right)^{1/2}\right)^2\right]\right)^{1/2} \\
&= \left(\mathbf{Tr}\left[\left(\begin{array}{c|c} \arccos\left(\sqrt{\frac{t^2}{1+t^2}}\right) & \mathbf{0}_{1,k-1} \\ \hline \mathbf{0}_{k-1,1} & \mathbf{0}_{k-1,k-1}\end{array}\right)^2\right]\right)^{1/2} \\
&= |\operatorname{arccot} t|
\end{aligned}$$

□

Proof. (of Claim 3) We first show that Q_A is orthogonal. Observe that $Q_A Q_A^\top$ is equal to

$$\left(\begin{array}{c|c} \left(I_k + A^\top A\right)^{-1} + A^\top \left(I_{n-k} + AA^\top\right)^{-1} A & \left(I_k + A^\top A\right)^{-1} A^\top - A^\top \left(I_{n-k} + AA^\top\right)^{-1} \\ \hline A\left(I_k + A^\top A\right)^{-1} - \left(I_{n-k} + AA^\top\right)^{-1} A & A\left(I_k + A^\top A\right)^{-1} A^\top + \left(I_{n-k} + AA^\top\right)^{-1} \end{array}\right).$$

To show Q_A is orthogonal, then, it suffices to show that:

1) $\left(I_k + A^\top A\right)^{-1} A^\top - A^\top \left(I_{n-k} + AA^\top\right)^{-1} = \mathbf{0}_{k,n-k}$;
2) $\left(I_k + A^\top A\right)^{-1} + A^\top \left(I_{n-k} + AA^\top\right)^{-1} A = I_k$; and
3) $A\left(I_k + A^\top A\right)^{-1} A^\top + \left(I_{n-k} + AA^\top\right)^{-1} = I_{n-k}$.

These are shown in Lemmas, 3, 4, and 5, respectively.

It remains to show that $Q_A \left(\begin{array}{c|c} I_k & \mathbf{0}_{k,n-k} \\ \hline \mathbf{0}_{n-k,k} & \mathbf{0}_{n-k,n-k} \end{array}\right) Q_A^\top = \tilde{\varphi}_0(A)$. By definition of φ_0,

$$\varphi_0(A) = \left(\begin{array}{c|c} \left(I + A^\top A\right)^{-1} & \left(I + A^\top A\right)^{-1} A^\top \\ \hline A\left(I + A^\top A\right)^{-1} & A\left(I + A^\top A\right)^{-1} A^\top \end{array}\right),$$

and so completing the proof is a straightforward computation. □

Lemma 3.

$$\left(I_k + A^\top A\right)^{-1} A^\top - A^\top \left(I_{n-k} + AA^\top\right)^{-1} = \mathbf{0}_{k,n-k}$$

Proof. Using a Neumann series representation of the matrix inverse (e.g., Stewart, 1998), the lemma is proved by the following series of deductions.

$$\begin{aligned} A^\top \left(I_{n-k} + AA^\top\right)^{-1} &= A^\top \sum_{j=0}^{\infty} (-1)^j \left(AA^\top\right)^j \\ &= \left(\sum_{j=0}^{\infty} (-1)^j \left(A^\top A\right)^j\right) A^\top \\ &= \left(I_k + A^\top A\right)^{-1} A^\top \end{aligned}$$

□

Lemma 4.

$$\left(I_k + A^\top A\right)^{-1} + A^\top \left(I_{n-k} + AA^\top\right)^{-1} A = I_k$$

Proof. By Lemma 3, the following series of deductions holds.

$$\begin{aligned} \left(I_k + A^\top A\right)^{-1} + A^\top \left(I_{n-k} + AA^\top\right)^{-1} A &= \left(I_k + A^\top A\right)^{-1} + \left(I_k + A^\top A\right)^{-1} A^\top A \\ &= \left(I_k + A^\top A\right)\left(I_k + A^\top A\right)^{-1} \\ &= I_k \end{aligned}$$

□

Lemma 5.

$$A\left(I_k + A^\top A\right)^{-1} A^\top + \left(I_{n-k} + AA^\top\right)^{-1} = I_{n-k}$$

Proof. A proof for this Lemma is easily recreated from the method used to prove Lemma 4. □

Lemma 6.

$$A\sqrt{I_k + A^\top A} = \sqrt{I_{n-k} + AA^\top}A,$$
$$A^\top\sqrt{I_{n-k} + AA^\top} = \sqrt{I_k + A^\top A}A^\top$$

Proof. This proof relies on a Neumann series representation of the square root of a matrix (e.g., Stewart, 1998).

$$\begin{aligned}
A\sqrt{I_k + A^\top A} &= A\left(I_k - \sum_{j=1}^{\infty}\left|\binom{1/2}{j}\right|\left(I_k - I_k - A^\top A\right)^j\right) \\
&= A\left(I_k - \sum_{j=1}^{\infty}\left|\binom{1/2}{j}\right|\left(-A^\top A\right)^j\right) \\
&= A - \sum_{j=1}^{\infty}\left|\binom{1/2}{j}\right|A\left(-A^\top A\right)^j \\
&= A - \sum_{j=1}^{\infty}\left|\binom{1/2}{j}\right|\left(-AA^\top\right)^j A \\
&= \left(I_{n-k} - \sum_{j=1}^{\infty}\left|\binom{1/2}{j}\right|\left(-AA^\top\right)^j\right)A \\
&= \sqrt{I_{n-k} + AA^\top}A
\end{aligned}$$

The remainder of the claim is proved by replacing I_k with I_n and A with $A^\top$. □

Claim 7. *Let $\mathfrak{g}$ be the Riemannian metric on $\mathbf{Gr}_{n,k}$ inherited from the Euclidean metric in $\mathbb{R}^{n\times n}$. The retraction on our `Atlas` representation of $\mathbf{Gr}_{n,k}$ approximates geodesics $\gamma : [0,1] \to \mathbf{Gr}_{n,k}$ with initial conditions $\gamma(0) = \varphi(\vec{0}), \dot{\gamma}(0) = \vec{\tau}$ with error of order $O\left(\|\vec{\tau}\|_{\mathfrak{g}}^3\right)$.*

Proof. From Claim 5, we know that for charts $\varphi : \mathcal{V} \to \mathcal{U}$ in the `Atlas` object, lengths of paths constrained to coordinate axes in $\mathcal{V}$ are preserved by the automorphism

$$\begin{aligned}
\alpha : \mathcal{V} &\to \mathcal{V} \\
\vec{\xi} &\mapsto A\vec{\xi},
\end{aligned}$$

where A is a diagonal matrix whose nonzero entries are in $\{-1, 1\}$. For this reason, the partial derivatives $g_{\mu\mu,\nu}$ are equal to their additive inverses at $\vec{\xi} = \vec{0}$, meaning they must be zero. By the parallelogram law, the partial derivatives $g_{\mu\nu,\lambda}$ must also vanish at $\vec{\xi} = \vec{0}$. Because these partial derivatives all vanish, the Christoffel symbols $\Gamma^\lambda_{\mu\nu}$ must also vanish at $\vec{\xi} = \vec{0}$.

We can exploit this fact by considering the third-order Taylor expansion of the geodesic γ:

$$\gamma(t) = \gamma(0) + t\dot{\gamma}(0) + \frac{t^2}{2}\ddot{\gamma}(0) + \frac{t^3}{6}\dddot{\gamma}(0) + O(t^4). \tag{23}$$

The constant and first-order terms are given by the initial conditions $\gamma(0) = \varphi(\vec{0})$ and $\dot{\gamma}(0) = \vec{\tau}$. The second-order term is given by the geodesic equation

$$\ddot{\gamma}^i + \Gamma^i_{jk}\dot{\gamma}^j\dot{\gamma}^k = 0. \tag{24}$$

As we know that Γ^i_{jk} vanishes at $t = 0$, and so $\ddot{\gamma}(0) = \vec{0}$. So, the approximation error of the retraction at $\vec{\xi} = \vec{0}$ must have $O\left(\|\vec{\tau}\|_{\mathfrak{g}}^3\right)$ approximation error. □

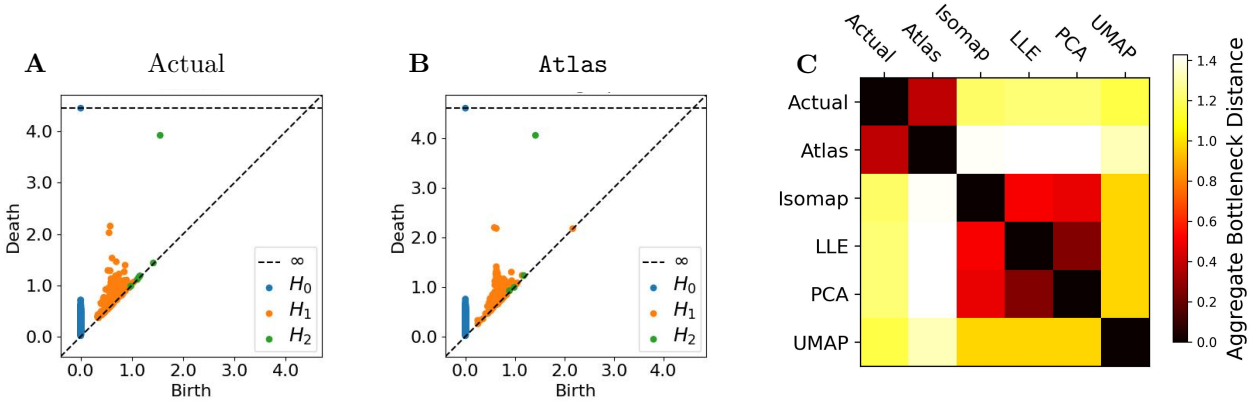

Figure 7: **Our `Atlas` representation of the high-contrast image patches manifold preserves its H1 and H2 homology groups. A)** Representations of the H0, H1, and H2 homology groups of the k^0 Klein bottle computed using Vietoris-Rips persistent homology on 640 points sampled uniformly in polar coordinates and mapped to the k^0 geometry. **B)** Representations of the H0, H1, and H2 homology groups of an 64-chart `Atlas-Learn` instance computed using Vietoris-Rips persistent homology on 640 points, sampled by taking 10 points from each chart using the method `atlas_grid_sample`. **C)** The aggregate bottleneck distance between Vietoris-Rips persistence diagrams generated by different 2D representions using 640 sampled points

E ATLAS METHODS APPLIED TO THE k_0 KLEIN BOTTLE

E.1 Persistence Diagram Computation (Fig. 7A,B)

We compute persistent homology on point clouds generated from our `Atlas-Learn` representation of high-contrast image patches, as well as on points sampled from a parametrization from Sritharan et al. (2021) of the same manifold (Fig. 7).

For the Sritharan parametrization of the Klein bottle, a point is sampled by sampling coordinates (θ, ϕ) from the uniform measure on $[0, \pi] \times [0, 2\pi]$, mapping the coordinates into $\mathbb{R}^{3\times 3}$ by the restriction of the map $k_{\theta,\phi}$ to $\{-1, 0, 1\} \times \{-1, 0, 1\}$, and reshaping the result as an element of $\mathbb{R}^9$.

Points from `Atlas` were generated deterministically from a mesh in tangential coordinates for each of the 64 charts $(\mathcal{V}, \varphi)$ in our `Atlas` representation as follows: for $\epsilon = 0.1$, we generate the Cartesian ϵ-mesh[10] of all tangential coordinates $\vec{\xi} \in \mathcal{V}$ such that $\vec{\xi}$ is within the MVEE of $\mathcal{V}$ according to Equation 14.

Ten points were sampled for each of the 64 charts in the `Atlas`, for a total of 640 points, and 640 points were sampled from Sritharan's parametrization of the Klein bottle.

Persistent homology was computed on the samples by Vietoris-Rips complex using the Python package Ripser (Bauer, 2021).

E.2 Pairwise Aggregate Bottleneck Distance Computation (Fig. 7C)

Let P_0, P_1, P_2 be persistence diagrams generated from data X corresponding to H_0, H_1, and H_2 features, respectively. Let Q_0, Q_1, Q_2 be generated similarly from data Y. We define the *aggregate bottleneck distance* between diagram triplets (P_0, P_1, P_2) and (Q_0, Q_1, Q_2) as $\sqrt{\sum_{i=0}^{2} d_{\mathrm{B}}^2 (P_i, Q_i)}$, where d_{B} denotes the bottleneck distance (e.g., Oudot, 2015). This distance function reflects the bottleneck distance between persistence diagrams in each dimension, making it a natural way to measure the preservation of H_0, H_1, and H_2 features by different dimensionality reduction measures.

To create the heatmap in Fig. 7C, the same points were used as described in Section E.1. PCA (restricted to the

[10]The Cartesian ϵ-mesh in $\mathbb{R}^2$ is the set of all points $\vec{\xi} = (m\epsilon, n\epsilon)$ for $m, n \in \mathbb{Z}$.

top two or five principal components) and a two-dimensional UMAP were each computed on the points sampled from the Sritharan parametrization. We excluded t-SNE from the bottleneck distance computations, due to its very high aggregate bottleneck distances to all other methods.

E.3 Geodesic Distance Computations (Fig. 3)

For the experiment comparing geodesic distances, we approximated true geodesic distance as follows. First, we generated sample points from the Sritharan parametrization by mapping a 1000×1000 grid of evenly spaced points in the set $[0, \pi] \times [0, 2\pi]$ into $\mathbb{R}^{3\times 3}$ through the restriction of k^0 to $\{-1, 0, 1\} \times \{-1, 0, 1\}$. For `Isomap` specifically, we uniformly subsampled these points down to 20,000 in order to keep memory requirements under 4GB. These points in $\mathbb{R}^{3\times 3}$ were then reshaped these as vectors in $\mathbb{R}^9$. These points were used to create a 15-nearest-neighbors graph, with each edge weighted by the Euclidean distance between its endpoints. True geodesic distance between a given pair of points was then approximated by computing a shortest path in this graph.

For `Isomap`, `PCA`, and t-`SNE`, geodesic distances were computed via an analogous approach. That is, each of these transformations was used to map the sampled grid points into a new, separate representation, on which 15-nearest-neighbor graphs with distance-weighted edges were computed. Note that these graphs have one million nodes, while `Atlas` uses a graph $G_{\epsilon,\delta}$ of only 28,700 nodes for the values of $\epsilon = 0.6$ and $\delta = 0.1$ (Sec. B.2); therefore, naively, one might expect geodesic distances to be better preserved by `PCA`, t-`SNE`, and `Isomap` in the transformed space than by the `Atlas` representation.

To investigate how well geodesic distances were preserved by `Atlas`, PCA, t-SNE, and UMAP, 100 pairs of points were randomly sampled without replacement from the one million grid points in the Sritharan parametrization. For `Atlas`, these 100 pairs of points were ingested into the `Atlas`, and the geodesic distance between each pair of points was approximated as the naïve approximate distance between the points (Sec. B.2).

E.3.1 State-of-the-Art Dimensionality Reduction Techniques Fail to Preserve Geodesic Distances Even at the Whitney Dimension

At least four dimmensions are needed to embed the Klein bottle into Euclidean space. In Figure 8, we show that `PCA` finds a five-dimensional linear subspace onto which data sampled from the k^0 Klein bottle can be preserved with little metric distortion, whereas `Isomap` and t-`SNE` fail to preserve the metric structure even when having enough dimensions. The methodology for sampling points, performing dimensionality reduction, and computing geodesic distances is as described in Section 3. For t-`SNE`, four dimensions were used instead of five due to runtime constraints.

E.3.2 Preservation of Geodesic Distances with `Atlas-Learn`-Generated Objects is Stable for Some Number of Charts Given a Fixed Number of Observations from the Klein Bottle

Individual metric distortion estimates do not indicate the variance of the distortion over different samples used for estimation. We did not compute variance in metric distortion over different samples for `Isomap`, t-`SNE`, or `UMAP` due to prohibitive runtime. Further, we did not compute variance in metric distortion over different samples for `PCA` due to the strict linearity of PCA-learned subspaces and the highly nonlinear nature of the Klein bottle, our manifold of choice for both our homology preservation experiments and our metric preservation experiments. For this reason, we only look at the variance of l_∞- and l_2-metric distortions for `Atlas` objects generated by `Atlas-Learn`.

We generate `Atlas` instances from $N \in \{500, 1000, 5000, 10000\}$ samples from the Sritharan parameterization of the Klein bottle and $k \in \{4, 8, 16, 32, 64\}$ charts using `Atlas-Learn`. For each N, N points were sampled from the Sritharan parametrization of the Klein bottle as described in Section E.1. For each of 10 fixed random seeds, 100 pairs of distinct points on the Klein bottle were randomly generated using a 1000 x 1000 grid in polar coordinates as described in Section E.3. The geodesic distance between a pair of points was closely estimated using a dense, 15-nearest neighbor graph, whereas the `Atlas` approximation of the geodesic distance is computed using a graph[11] $G_{\epsilon,\delta}$ for $\epsilon = 0.6$ and $\delta = 0.1$, as described in Section E.3. We compute standard metric distortion

[11] We write in Section E.3 that $G_{\epsilon,\delta}$ has 28,700 nodes for $N = 10,000$, $k = 64$ to emphasize how `Atlas` results in less metric distortion using fewer points. For different N and k, this number of nodes varies.

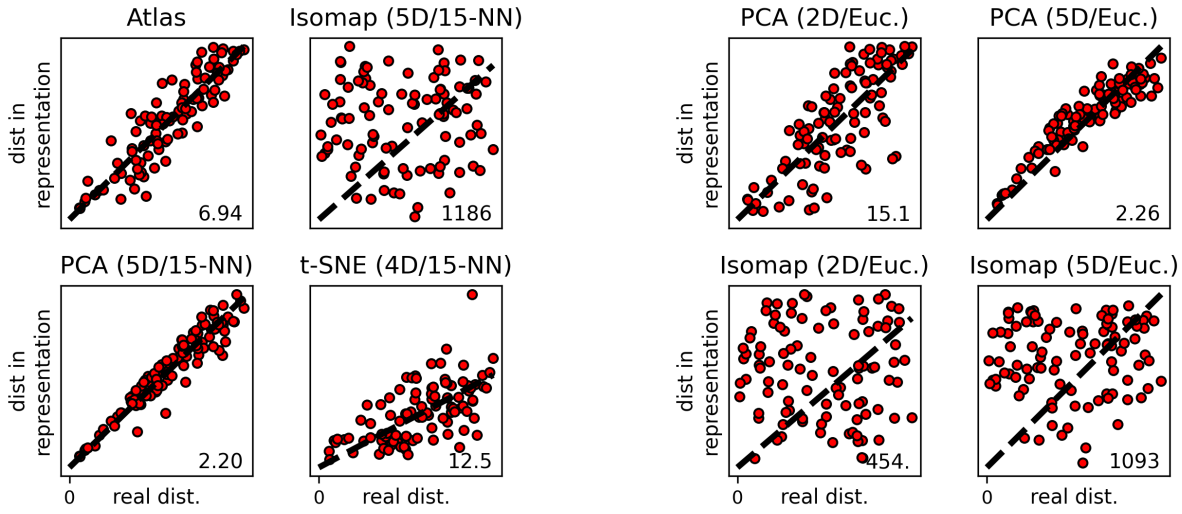

Figure 8: **Even at the maximum embedding dimension from the Whitney Embedding Theorem, state-of-the-art, nonlinear dimensionality reduction techniques fail to preserve geodesic distances.** `Isomap`, `PCA` and t-`SNE` were computed on a mesh of 1 million points. `Atlas-Learn` was asked to produce 64 charts, given only a 10,000-points mesh. For 100 randomly sampled pairs of points on the Klein bottle, each scatter plot shows each pair (dot) according to a precise estimate of their true geodesic distance (x axis, "real dist.") versus the embedding distance (y axis, "dist. in representation") for each embedding (panel title). `Atlas` is a 2D representation. For all other methods, we specify parameters in the form (a/b) where a indicates the dimensionality of the representation used and b the method used for computing geodesics. For b, the options are either *15-NN*, indicating that shortest path were computed on a 15-nearest-neighbor graph, or *Euc.*, meaning that distances were estimated by the Euclidean distance in the represented embedding. Metric distortion is given in bottom right corner. Dashed line is fit with y-intercept zero.

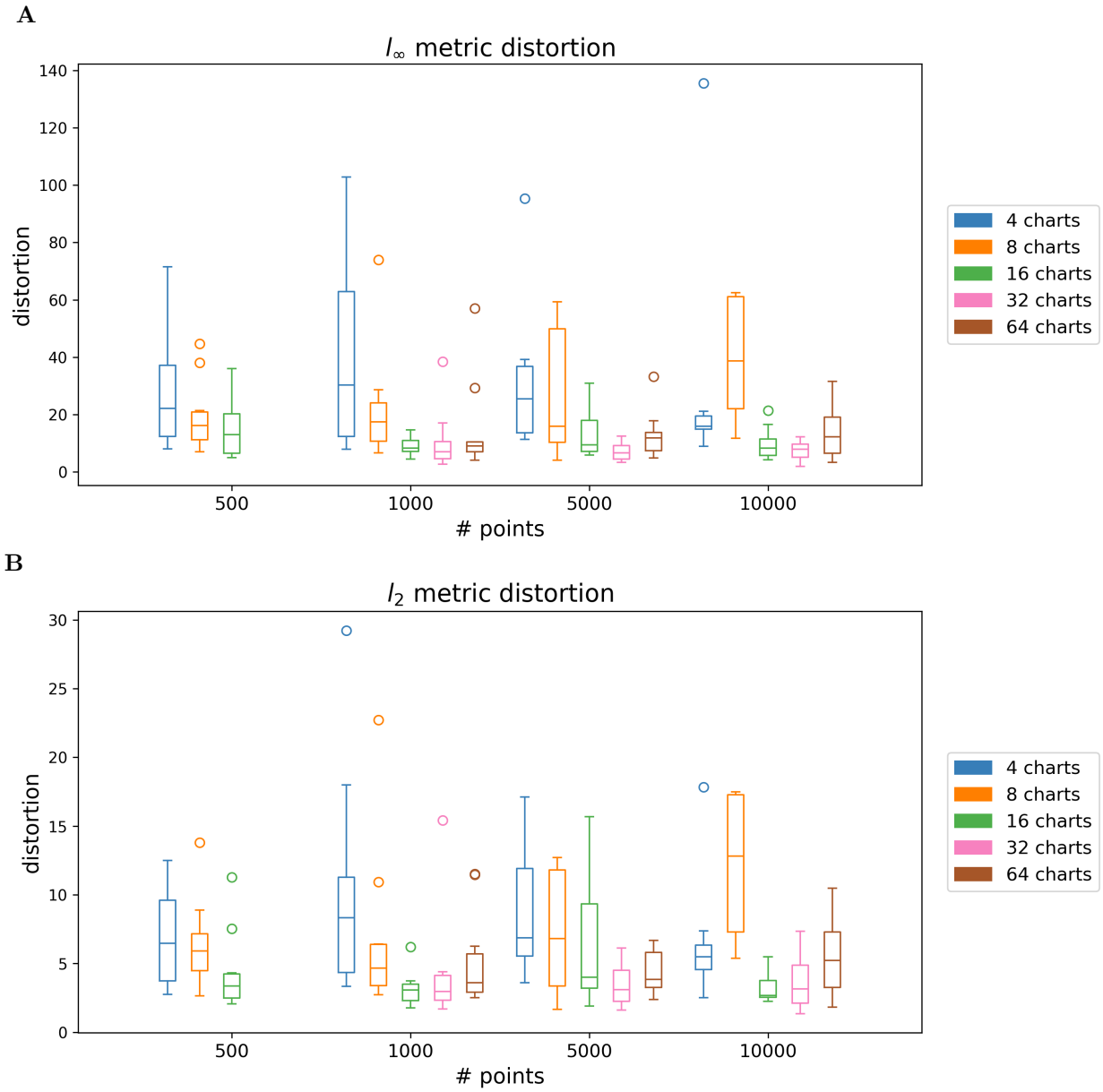

Figure 9: **The optimal number of charts to pass to `Atlas-Learn` for a fixed dataset for purposes of minimizing metric distortion varies with the number of samples.** `Atlas-Learn` was used to learn `Atlas` representations of the Klein bottle from an N-point mesh, for $N \in \{500, 1000, 5000, 10000\}$, using $k \in \{4, 8, 16, 32, 64\}$ charts. For ten distinct random seeds, 100 pairs of points were randomly sampled from on the Klein bottle, and the l_∞- (**A**) and l_2- (**B**) metric distortion were computed by comparing a precise estimate of the true geodesic distance versus the embedding distance for each pair of points. Each box plot depicts the distribution of these ten distortion computations for the `Atlas` object learned using N samples (x-axis) and k charts (color). Distortion values are given on the y-axis for each of l_∞- (**A**) and l_2- (**B**) distortion. Note the difference in scales on the y-axis between **A** and **B**. Results not shown for $N = 500$, $k \in 32, 64$ due to insufficient samples for learning charts. Outlier distortion values indicated by open circles. From top to bottom, for each box plot, ticks indicate the upper quartile plus 1.5 times the interquartile range (IQR), the upper quartile, the median, the lower quartile, and the lower quartile minus 1.5 times the IQR, respectively.

(i.e., l_∞-metric distortion) as the maximum ratio of true geodesic distance over `Atlas` distance, divided by the minimum ratio. We additionally compute the l_2-metric distortion: the average ratio of true geodesic distance over `Atlas` distance, divided by the minimum ratio.

We see in our results that selecting a number of charts to minimize metric distortion varies as a function of the number of samples passed to `Atlas-Learn` (Figure 9). For the smallest numbers of charts, large metric distortion is observed across all numbers of samples. At the same time, even for the largest number of samples explored, having 64 charts almost always increases both the expectation and the variance of the metric distortion, for both l_∞- and l_2-metric distortion. The number of charts yielding the least distortion increases with the number of points sampled from Klein bottle, while maintaining the convex relationship between number of charts and metric distortion. For 32 charts, l_∞-distortion is kept below 10 on average and below 20 at maximum across ten random seeds, giving evidence that, distributionally, metric distortions of two-dimensional, `Atlas-Learn`-generated objects significantly undercut those of state-of-the-art, nonlinear dimensionality reduction techniques using two dimensions (Figures 3, 8). Given a fixed number of point cloud observations, this sort of metric distortion analysis over numbers of charts might help avoid overfitting due to an excess number of charts.

E.4 Packages Used for Non-`Atlas` Dimensionality Reduction

PCA and t-SNE were computed using scikit-learn (version 1.3.2, Pedregosa et al., 2011) . We ran t-SNE with `perplexity=5.0` and otherwise default arguments.

F COMPUTING RIEMANNIAN PRINCIPAL BOUNDARIES

F.1 Parameter Choices and Modifications

We implemented the RPB algorithm as described in Yao and Zhang (2020), with the exceptions described here.

In Yao and Zhang (2020), Sec. 2.2, a univariate kernel κ_h is used to define which points are included in the computation of the local covariance matrix Σ_h. For this kernel, we use the indicator function for the ball of radius h.

The original algorithm computes a weighted average of the first derivatives of the two principal flows, where $\lambda_\delta(t)$ is the weight at iteration t (Eq. 10, Yao and Zhang, 2020). In our implementation, we assume that $\lambda_\delta(t) = 1/2$ for all t. This choice was motivated by both simplicity and practical considerations, i.e., it helped avoid issues where the principal boundary could collapse into one of the principal flows.

The differential equation for a principal flow γ^+ (induced by Equation 5 of Yao and Zhang, 2020) effectively follows the vector field defined by the top eigenvector of the local covariance matrix Σ_h. To avoid oscillations in the principal flow direction, due either to the insensitivity of eigenvector computations to multiplication by -1 or non-smooth changes in the top eigenvectors, we enforce a positive inner product between tangent vectors in adjacent iterates. Further, if the support of the data is sufficiently sparse, in practice, the top eigenvector field for the principal flow will be dominated by noise and will eventually cause the boundary curve to move away from the data, which also causes Σ_h to become undefined. To prevent this from happening, we use the following modification to correct the principal flow solution by moving it towards the mean of the local data:

Let W be the top eigenvector field of the local covariance matrix Σ_h (as in Equation 4 of Yao and Zhang, 2020). Instead of following the update rule $\dot\gamma = W(\gamma)$, we instead follow the rule

$$\dot\gamma = W(\gamma) + \alpha \left(I - W(\gamma)W(\gamma)^\top\right) \left(\frac{1}{\sum_i \kappa_h\left(x_i, \gamma\right)} \sum_i \log_\gamma x_j \right), \tag{25}$$

where $\alpha > 0$ is a correction factor that moves $\dot\gamma$ toward the mean Riemannian logarithm of nearby sample points, projected onto the orthocomplement $I - W(\gamma)W(\gamma)^\top$ of the top eigenvector of the local covariance matrix. We found $\alpha = 5$ to be a helpful value for avoiding divergence.

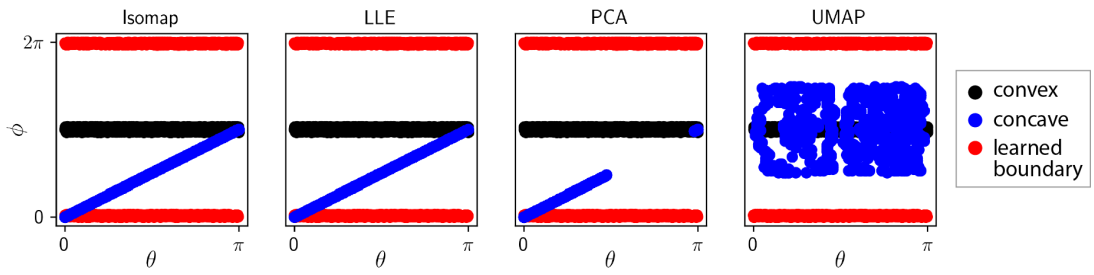

Figure 10: **Separating boundaries between convex and concave patches learned within 2D representations from state-of-the-art dimensionality reduction techniques forfeit the topology of the k^0 Klein bottle.** Dimensionality reduction techniques are applied to 20,000 points (Isomap) or one million points (LLE, PCA, UMAP) from the k^0 Klein bottle in order to learn 2D representations of the data. Then, $1,000$ convex (black) patches and $1,000$ concave (red) patches are mapped into the learned representation. Linear SVM is applied in order to learn a separator between convex and concave patches. The $1,000$ points closest to the boundary in the learned space are then visualized in the polar representation of the Klein bottle (blue).

F.1.1 State-of-the-Art Dimensionality Reduction Methods Fail to Learn Intelligible Separator Between Convex and Concave Patches in Two Dimensions

To highlight the significance of the `RPB` algorithm learning an interpretable, intrinsic separator between convex and concave patches on a 2D `Atlas` object in Section E, we compare against other 2D representations learned using state-of-the-art dimensionality reduction techniques. Specifically, we take 1 million points from a grid in the polar representation of the Klein bottle, and subsample $20,000$ for `Isomap`, as in Section E.3. After transforming these polar points into $\mathbb{R}^9$, we apply `Isomap`, t-`SNE`, and `PCA` as described in the same section, and additionally apply `UMAP` from the `scikit_learn` implementation using `n_neighbors=5` and othwerwise default parameters. We then map $1,000$ convex and $1,000$ concave patches into the learned representations. Using the linear `SVM` classifier implementation from `scikit_learn`, we learn linear boundaries separating convex and concave patches in each representation (Pedregosa et al., 2011). Recall that the `RPB` algorithm is intended to be a Riemannian-geometric generalization of linear `SVM`; this use of linear `SVM` is the closest analog of the `RPB` algorithm in these learned, one-chart representations of the data, and is chosen for the sake of fair comparison (Yao and Zhang, 2020).

When the boundaries learned in the 2D representations are mapped back into the ambient space—by identifying the preimages of the $1,000$ points closest to the learned boundary in each represenation—we see that `Isomap`, `PCA`, and t-`SNE` capture a local separator between the convex and concave patches. However, these local separators take into account neither the compactness nor the unorientability of the Klein bottle.

G RNA EXPERIMENTS

G.1 Using `Atlas-Learn` to Create 5-Dimensional Representation of Hematopoietic Single-Cell Transcriptomic Data

Single-cell RNA-sequencing data (with metabolically labeled nascent RNA) from hematopoietic cells were retrieved using the `dynamo` Python package, following the "scNT-seq human hematopoiesis dynamics" tutorial notebook, which creates a data representation based on the top 30 principal components (PCs) (Qiu et al., 2022a,b). Our preliminary, multiscale-singular vector decomposition (mSVD) analysis of these data suggests that, at several points of the manifold, the data are well approximated by a five-dimensional submanifold (Figure 11). An `Atlas` data structure is learned on the 30-PC representation using `Atlas-Learn` (Algorithm 2) with dimensionality 5 and 30 coordinate charts. For several random seeds, there are enough points in each randomly assigned chart for the tangent Stiefel matrix L to be learned, but not enough for the quadratic coefficients in $\hat{\mathbf{h}}$ to be determined. For this reason, we set M to be an arbitrary, Stiefel orthocomplement of L and all quadratic coefficients in K to be zero.

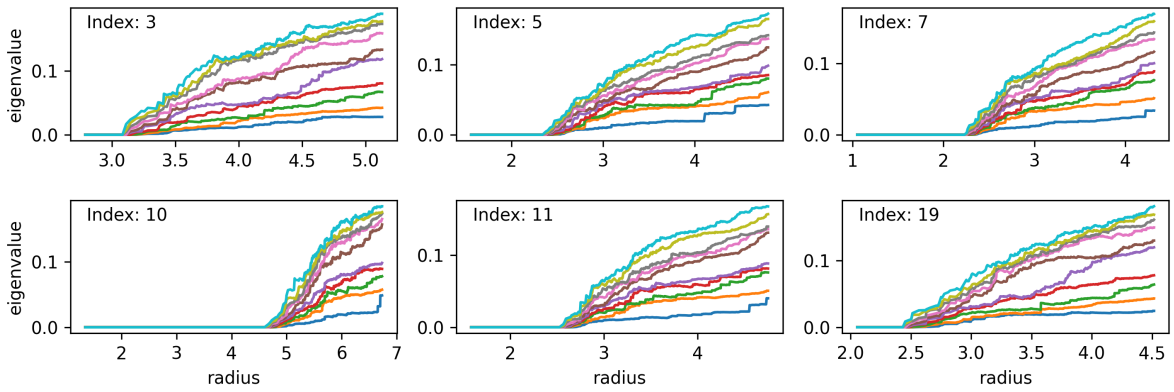

Figure 11: **mSVD eigenvalue plots based at chosen centerpoints on the Dynamo HSC data in 30-PC space suggest the data lie close to a submanifold of dimension five.** For each centerpoint $\vec{x}$, all other data were sorted by distance from $\vec{x}$. At each radius r, the eigenvalues of the covariance matrix Σ containing points within distance r of $\vec{x}$ are plotted. This is done for the 1200 points closest to $\vec{x}$. These heuristic, graphical adaptations of the mSVD technique from Little et al. (2017) demonstrate a prominent gap between the fifth and sixth eigenvalues.

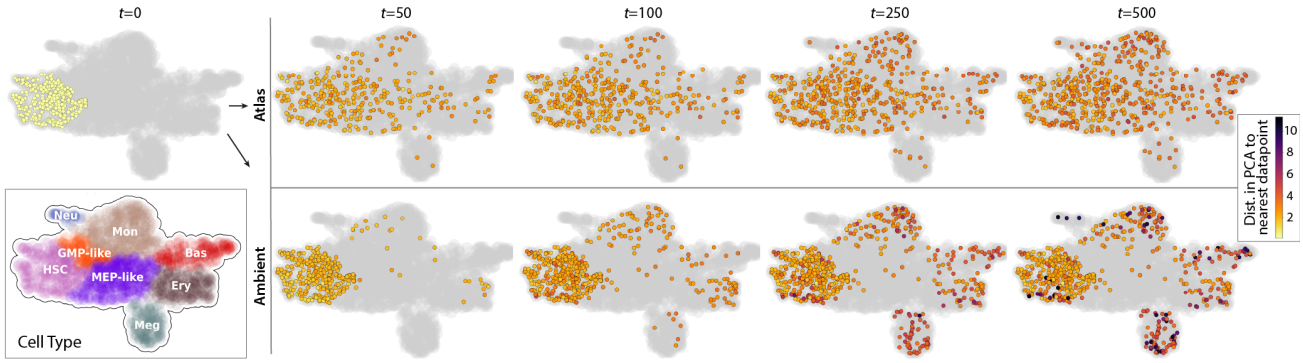

Figure 12: **Integrating the vector field in Section 3.4 in the tangent space of an `Atlas` object preserves proximity to the data while capturing all cell types present.** Each plot shows a UMAP representation of single-cell RNA-sequencing data (with metabolically labeled nascent RNA) from 1,947 hematopoietic cells (colored by cell type, bottom left, or in gray) (Qiu et al., 2022b). The RNA velocity vector field $\vec{V}_{\texttt{amb}}$ was integrated as an ODE both in the tangent plane of an `Atlas-Learn` structure ("Atlas", top row) and in 30-dimensional PCA space ("Ambient", bottom row). For 309 initial conditions ($t = 0$, top left, in yellow), each corresponding to a HSC, integration was performed for 500 time steps t in each space. At each labelled time step $t \in \{50, 100, 250, 500\}$ (panels), the iterate is visualized in the UMAP by overplotting the datapoint that is closest to the iterate in color, representing its Euclidean distance from the iterate in 30-PC space. Abbreviations: HSC: hematopoietic stem cell; GMP: granulocyte-monocyte progenitor; MEP: megakaryocyte-erythroid progenitor; Neu: neutrophil; Mon: monocyte; Bas: basophil; Ery: erythroid; Meg: megakaryocyte.

G.2 Computation, Integration, and Interpretation of an RNA Velocity Vector Field

We isolate the 309 points labeled as hematopoietic stem cells (HSCs), treating each as the initial condition to an ODE. We consider two ODEs: the one induced by the `Dynamo` vector field $\vec{V}_{\texttt{amb}}$ in the ambient space and the vector field $\vec{V}_{\texttt{atlas}}$ induced by projecting $\vec{V}_{\texttt{amb}}$ onto the learned tangent bundle. The $\vec{V}_{\texttt{amb}}$ ODE is integrated in the ambient space by the 4(5) Runge-Kutta method using the `Dynamo` function `dyn.vf.VectorField.integrate`, with arguments `interpolation_num=500` and `t_end=t_term`[12]. The $\vec{V}_{\texttt{atlas}}$ is integrated by the forward-Euler method with stepsize 1 for 500 iterations, using the method $\mathbf{Ret}_{\cdot}\cdot$ in Algorithm 1 as the update step. Integrating $\vec{V}_{\texttt{amb}}$ results in iterates $\vec{x}_i^{\texttt{amb}} \in \mathbb{R}^{30}$ for $i \in \{0, \ldots, 500\}$. Similarly, integrating $\vec{V}_{\texttt{atlas}}$ results in iterates $\left(\vec{\xi}_i, j_i\right) \in \mathbb{R}^5 \times \{1, \ldots, 30\}$ for $i \in \{0, \ldots, 500\}$, where $\vec{\xi}_i$ are tangential coordinates and j_i are chart indices. For the sake of comparison, we define $\vec{x}_i^{\texttt{atlas}} = \varphi_{j_i}\left(\vec{\xi}\right)$.

The results show that iterates $\vec{x}_i^{\texttt{atlas}}$ keep close to the manifold for all iterations, whereas iterates $\vec{x}_i^{\texttt{amb}}$ gradually depart from the manifold over time (Figure 12). Moreover, iterates $\vec{x}_i^{\texttt{atlas}}$ gradually pass through canonical intermediate cell states (i.e., granulocyte-macrophage progenitor-like and megakaryocyte-erythroid progenitor-like cells) before entering terminal states (i.e., neutrophils, monocytes, basophils, erythroid cells, megakaryocytes). On the other hand, iterates $\vec{x}_i^{\texttt{amb}}$ rarely pass through intermediate states, including in time steps that are not shown. Taken together, these data suggest that using the learned atlas constrains the integration of an RNA-velocity transcriptomic vector field ODE to better reflect the data, improving the biological plausiblity of the resulting trajectories.

G.3 Average Distance of ODE Iterates from Data Decreases with Number of Charts

While cursory inspection of our results in the Dynamo ODE integration suggests that the `Atlas` object made using `Atlas-Learn` is serving as a useful regularizer for integrating the ODE, it is possible that both our better proximity to actual data (Figure 5) and our more biologically plausible movement through progenitor states (Figure 12) are artifacts of overfitting. To assess this possibility, we repeated the vector field experiment from Section 3.4, specifying different numbers of charts for `Atlas-Learn`. We integrated the same ODE within each learned `Atlas` object for 500 iterations as described in Section G.2. We see for only two charts that the average distance from an iterate to its nearest datum at 300 iterations is greater than the average distance from a datum to the `Atlas` (Figure 13). For five and ten charts, at the same number of iterations, the average distance from iterates to nearest data roughly matches the average distance from a datum to the `Atlas`, while these same number of charts result in relatively higher average distance from iterate to nearest datum as compared to the baseline for 500 iterations. For 20 or more charts, the average distance of an iterate to its nearest datum is always below the average distance from a datum to the `Atlas` for all 500 iterations.

Maintaining the average distance of an iterate to its nearest datum below the average distance of a datum to the `Atlas` indicates that the ODE iterates are moving towards particular data within the approximation error of the `Atlas` object, as opposed to moving away into regions which poorly approximate the data. We see from Figure 12 that the intermediate and terminal states captured by the 30-chart `Atlas` indeed recreate a more biologically plausible progression than the ODE integrator in the "ambient" 30-PC space. Together, it would seem that the increased closeness of ODE iterates to actual data is indeed capturing signal robustly as opposed to overfitting. We see, further, that the average distance from a datum to the `Atlas` stabilizes as the number of charts approaches 30, suggesting the ability of our `Atlas` object to represent the data faithfully using only linear charts saturates. We truncated our experiments at 30 charts due to overfitting of individual charts for mildly larger numbers of charts, due to lack of data.

References

P.-A. Absil, R. Mahoney, and Rodolphe Sepulchre. Optimization algorithms on matrix manifolds. Princeton University Press, 2008.

David J. Aldous. Covering a compact space by fixed-radius or growing random balls. Latin American Journal of Probability and Mathematical Statistics, 19:755–767, 2022.

[12] Here, `t_term = 2 * dynamo.tools.utils.getTend(adata_hsc.obsm["X_pca"], adata_hsc.obsm["velocity_pca"])`.

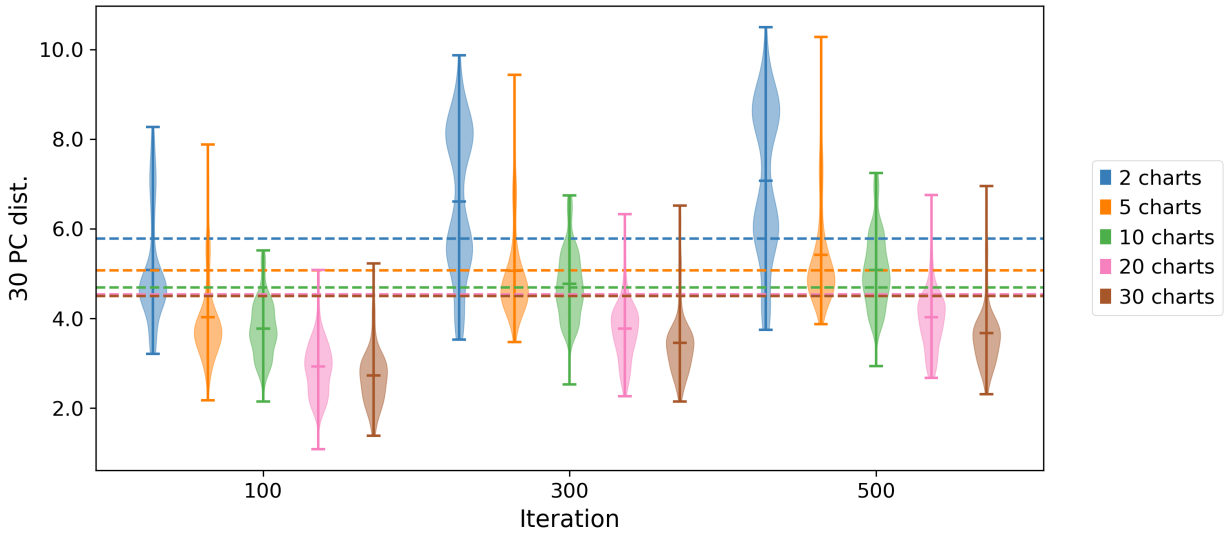

Figure 13: **Average distance of ODE iterates from data decreases with number of charts.** `Atlas` representations of the 30-PC hematopoiesis dataset were created using `Atlas-Learn` with only linear coefficients (Section G.1), for numbers of charts $k \in \{2, 5, 10, 20, 30\}$. For each resulting `Atlas` object, the average distance from a datum to the `Atlas` is indicated by a horizontal dashed line. The Dynamo ODE was integrated within each `Atlas` object (Section G.2) with 309 initial conditions—one for each datum labeled as a hemotopoeitic stem cell—for 500 iterations. For each number of iterations $t \in \{100, 300, 500\}$ and each number of charts, a violin plot indicates the distribution of minimum distance from an ODE iterate to the nearest datum over all 309 iterates at t iterations. Distances are given in the "ambient", Euclidean distance within the 30-PC space containing the data. Upper, central, and lower ticks indicate maximum, mean, and minimum distance to the nearest datum, respectively.

Ulrich Bauer. Ripser: efficient computation of Vietoris-Rips persistence barcodes. J. Appl. Comput. Topol., 5 (3):391–423, 2021. ISSN 2367-1726. doi: 10.1007/s41468-021-00071-5.

Thomas Bendokat, Ralf Zimmermann, and P.-A. Absil. A Grassmann manifold handbook: Basic geometry and computational aspects, 2011. URL `https://arxiv.org/pdf/2011.13699.pdf`.

Rudrasis Chakraborty and Baba C. Vemuri. Recursive Fréchet mean computation on the Grassmannian and its applications to computer vision. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 4229–4237, 2015. doi: 10.1109/ICCV.2015.481.

Charles Ehresmann. Sur la topologie de certains espaces homogènes. Annals of Mathematics, pages 396–443, 1934.

Reshad Hosseini and Suvrit Sra. Recent Advances in Stochastic Riemannian Optimization, pages 527–554. Springer, Cham, Switzerland, 2020.

Xiaocan Li, Shuo Wang, and Yinghao Cai. Tutorial: Complexity analysis of singular value decomposition and its variants. arXiv: Numerical Analysis, 2019.

Anna V. Little, Mauro Maggioni, and Lorenzo Rosasco. Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature. Applied and Computational Harmonic Analysis, 43(3):504–567, 2017. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2015.09.009.

James R. Munkres. Topology. Prentice Hall, 2 edition, 2000.

Yurii A. Neretin. On jordan angles and the triangle inequality in grassmann manifolds. Geometriae Dedicata, 86, 2001.

Partha Niyogi, Stephen Smale, and Shmuel Weinberger. Finding the homology of submanifolds with high confidence from random samples. Discrete and Computational Geometry, 39:419–441, 2008.

Steve Y. Oudot. Persistence Theory: From Quiver Representations to Data Analysis. American Mathematical Society, Providence, RI, 2015.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

Xiaojie Qiu, Yan Zhang, Ke, Sebastian Bischoff, and Sichao. dynamo-tutorials, 2022a. URL `https://github.com/aristoteleo/dynamo-tutorials/tree/master`.

Xiaojie Qiu, Yan Zhang, Jorge D. Martin-Rufino, Chen Weng, Shayan Hosseinzadeh, Dian Yang, Angela N. Pogson, Marco Y. Hein, Kyung Hoi (Joseph) Min, Li Wang, Emanuelle I. Grody, Matthew J. Shurtleff, Ruoshi Yuan, Song Xu, Yian Ma, Joseph M. Replogle, Eric S. Lander, Spyros Darmanis, Ivet Bahar, Vijay G. Sankaran, Jianhua Xing, and Jonathan S. Weissman. Mapping transcriptomic vector fields of single cells. Cell, 185(4):690–711.e45, February 2022b. ISSN 0092-8674. doi: 10.1016/j.cell.2021.12.045.

Erich Schubert and Peter J. Rousseeuw. Fast and eager k-medoids clustering: O(k) runtime improvement of the PAM, CLARA, and CLARANS algorithms. Information Systems, 101:101804, November 2021. ISSN 0306-4379. doi: 10.1016/j.is.2021.101804.

Duluxan Sritharan, Shu Wang, and Sahand Hormoz. Computing the Riemannian curvature of image patch and single-cell RNA sequencing data manifolds using extrinsic differential geometry. Proceedings of the National Academy of Sciences, 118(29):e2100473118, 2021. doi: 10.1073/pnas.2100473118.

Gilbert W. Stewart. Matrix Algorithms, volume 1. Society for Industrial and Applied Mathematics, 1998.

Ameet Talwalkar, Sanjiv Kumar, Mehryar Mohri, and Henry Rowley. Large-scale svd and manifold learning. Journal of Machine Learning Research, 14(96):3129–3152, 2013. URL `http://jmlr.org/papers/v14/talwalkar13a.html`.

James Townsend, Niklas Koep, and Sebastian Weichwald. Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation. Journal of Machine Learning Research, 17(137):1–5, 2016. URL `http://jmlr.org/papers/v17/16-177.html`.

Zhigang Yao and Zhenyue Zhang. Principal boundary on Riemannian manifolds. Journal of the American Statistical Association, 115(531):1435–1448, 2020. doi: 10.1080/01621459.2019.1610660.

Wan-Lei Zhao, Hui Wang, and Chong-Wah Ngo. Approximate k-nn graph construction: A generic online approach. IEEE Transactions on Multimedia, 24:1909–1921, 2022. doi: 10.1109/TMM.2021.3073811.