Journal of Machine Learning Research 27 (2026) 1-51 Submitted 4/25; Revised 7/26; Published 9/26

Atlas Gaussian processes on restricted domains and point clouds

Mu Niu MU.NIU@GLASGOW.AC.UK
Yue Zhang 2416950Z@STUDENT.GLA.AC.UK
School of Mathematics and Statistics, University of Glasgow

Ke Ye KEYK@AMSS.AC.CN
KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, China

Pokman Cheung POKMAN@ALUMNI.STANFORD.EDU

Yizhu Wang 2603214W@STUDENT.GLA.AC.UK
Xiaochen Yang XIAOCHEN.YANG@GLASGOW.AC.UK
School of Mathematics and Statistics, University of Glasgow

Editor: Mingyuan Zhou

Abstract

In real-world applications, data often reside in restricted domains with unknown boundaries, or as high-dimensional point clouds lying on a lower-dimensional, nontrivial, unknown manifold. Traditional Gaussian Processes (GPs) struggle to capture the underlying geometry in such settings. Some existing methods assume a flat space embedded in a point cloud, which can be represented by a single latent chart (latent space), while others exhibit weak performance when the point cloud is sparse or irregularly sampled. The goal of this work is to address these challenges. The main contributions are twofold: (1) We establish the Atlas Brownian Motion (BM) framework for estimating the heat kernel on point clouds with unknown geometries and nontrivial topological structures; (2) Instead of directly using the estimated heat kernel as the GP covariance function, we construct a Riemannian-corrected kernel in which the estimated heat kernel acts as a geometry-aware correction to a positive-definite local base kernel. This leads to the formulation of Riemannian-corrected Atlas Gaussian Processes (RC-AGPs), which substantially reduce the BM simulation cost by requiring simulations only from atlas representatives rather than from all data points. The resulting RC-AGPs are applied to regression tasks across synthetic and real-world data sets. These examples demonstrate that our method outperforms existing approaches in both heat kernel estimation and regression accuracy. It improves statistical inference by effectively bridging the gap between complex, high-dimensional observations and manifold-based inferences.

Keywords: Gaussian process, Atlas, Brownian motion, Restricted domain, Heat kernel

©2026 Mu Niu, Yue Zhang, Ke Ye, Pokman Cheung, Yizhu Wang, Xiaochen Yang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v27/25-0786.html.

1. Introduction

In recent years, data collection has increasingly involved complex, constrained spaces with unknown boundaries or high-dimensional point clouds residing on a lower-dimensional, non-trivial, unknown manifold. For example, pollution data may be collected within geographic regions like lakes, where the exact boundaries are complex and unknown. Similarly, object rotations and movements in images or videos can form lower-dimensional manifolds embedded in high-dimensional pixel spaces. In this work, we focus on the following regression model for such point cloud data:

$$y_j = f(s_j) + \epsilon_j, \quad \epsilon_j \sim \mathcal{N}(0, \sigma_\epsilon^2) \tag{1}$$

where $f : \mathbb{M} \to \mathbb{R}$ is an unknown regression function. $y_j \in \mathbb{R}$ is a response variable. $s_j = (s_j^1, \ldots, s_j^p) \in \mathbb{M} \subseteq \mathbb{R}^p$ is an observed predictor on a complete compact and nontrivial Riemannian manifold $\mathbb{M}$ embedded in $\mathbb{R}^p$. Here, nontrivial topology means that the manifold cannot be globally represented by a single Euclidean chart and therefore requires multiple local charts. For a point cloud, significant persistent-homology features provide empirical evidence of such nontrivial topology. The manifold $\mathbb{M}$ can be parameterised by an atlas, which is a collection of some q-dimensional local coordinate systems(charts), where $q \leq p$. Let $\mathcal{D} = \{(s_j, y_j), j = 1, \ldots, n_d\}$ denote the labelled training data set where $n_d \geq 1$. Suppose we also have an unlabeled points $\mathcal{U} = \{s_j, j = 1, \ldots, n_u\}$, where $n_u \geq 1$. These points correspond to prediction (or test) locations for which the responses are unobserved. The total number of points in the point cloud is $n = n_d + n_u$. Considering the regression model in Equation 1, our goal is to make inferences about f using the labeled data set $\mathcal{D}$ and to predict y values for the unlabeled data set $\mathcal{U}$. Gaussian Processes (GPs) have been widely used as prior distributions for unknown functions. The GP with squared exponential (a.k.a. Radial Basis Function) kernel is a commonly used prior (Rasmussen and Williams, 2006). However, covariance functions relying on Euclidean distance can be inadequate when the input space $\mathbb{M}$ deviates from Euclidean geometry. On manifolds, such measures fail to reflect true relationships between points on $\mathbb{M}$, highlighting the need for approaches that incorporate the manifold's intrinsic geometry to support inference on complex data structures.

A major challenge in developing GPs on manifolds lies in constructing valid covariance kernels for these complex spaces. Recent advances in this area assume known manifold structure or simple topology that can be covered by a single chart. For instance, Niu et al. (2019) defined intrinsic GPs on known manifolds by simulating Brownian Motion(BM) paths and estimating the transition densities. Niu et al. (2023) extended this method to unknown manifolds, though limited to flat, trivial topologies characterised by a single chart (latent space). Lin et al. (2019) proposed extrinsic framework for GP modeling on manifolds, which relies on embedding the manifold into a Euclidean space. However such embeddings are only available for certain manifolds with known geometry.

Alternatively, several works approximate the manifold by constructing graphs over point clouds (Li et al., 2023; He et al., 2026; Bolin et al., 2024; Dunson et al., 2020), leading to Graph Laplacian GPs (GL-GPs). Here, kernels are estimated from eigenpairs of a graph Laplacian built using scaled Euclidean distances. While GL-GPs offer improvements over standard Euclidean GPs for manifold structured data, they can suffer when the point cloud

is sparse or irregularly sampled, and are sensitive to hyperparameters such as bandwidth and neighbor counts (García Trillos et al., 2020). Moreover, smoothing spline approaches (Wood et al., 2008; Ramsay, 2002) address noisy observations within regions of known boundaries in $\mathbb{R}^2$. While cases with unknown boundaries were discussed, they remain an open problem. It is well known that a nontrivial manifold cannot be covered by a single chart unless it is homeomorphic to an open subset of $\mathbb{R}^q$ (Lee, 2013, Section 1). Schonsheck et al. (2019) and Stolberg-Larsen and Sommer (2022) emphasize the necessity of using multiple charts to properly capture the structure of manifolds with nontrivial topology. Mapper (Singh et al., 2007), a tool from topological data analysis, has been widely used to partition complex point clouds into locally simple subsets. Mapper has successfully uncovered global structures (e.g., Trefoil knots) (Floryan and Graham, 2022; Paik et al., 2022), while Chart Autoencoders (Schonsheck et al., 2019, 2022) have also been employed for learning multiple charts in molecular dynamics data. In this work, we follow Singh et al. (2007) and adopt Mapper as a preprocessing step to partition the point cloud into subsets.

Main Contribution: We propose a novel methodology for constructing Gaussian processes on constrained domains with unknown boundaries and point clouds characterised by unknown geometry and nontrivial topology. Our key contributions are: (1) We develop an *Atlas Brownian Motion framework* to estimate the heat kernel on point clouds via simulation of BM paths across multiple learned charts. (2) Instead of directly using the estimated heat kernel, we construct a *Riemannian-corrected kernel*, combining the global heat kernel and local RBF kernel, and leading to the formulation of *Riemannian-corrected Atlas Gaussian Processes (RC-AGPs)*.

The RC-AGP is constructed through three steps. First, we build a probabilistic atlas by learning multiple local charts from subsets of the point cloud, where the subsets are generated using the TDA Mapper (Section 3). Second, we simulate BM paths on the Atlas by enabling probabilistic coordinate transitions across these charts, and estimate the heat kernel from the transition densities on the Atlas (Section 4). Third, we construct RC-AGPs by integrating the estimated heat kernel with the local RBF kernel, proving its semipositive definiteness to guarantee well-posed Gaussian process inference (Section 5).

The Atlas BM framework improves upon existing methods—such as single-chart BM and graph Laplacian approximations—in terms of heat kernel estimation accuracy. It is robust to variations in point cloud distribution and requires significantly fewer points to achieve accurate kernel estimates. By using the estimated heat kernel as a Riemannian correction to a local base kernel, the proposed RC-AGP incorporates intrinsic geometric information while retaining a stable positive-definite covariance structure for regression. This construction is particularly useful when the heat kernel is estimated from a finite number of BM paths, since the estimated heat kernel modulates the local covariance rather than entirely determining the GP covariance. We apply RC-AGPs on four regression tasks: two domains in $\mathbb{R}^2$ with unknown boundaries (the U-shape and the Aral Sea), a torus point cloud in $\mathbb{R}^3$, and a high-dimensional shark prey image cloud in $\mathbb{R}^{900}$. Across all tasks, RC-AGPs consistently outperform Euclidean GPs and GL-GPs.

2. Background of Riemannian Geometry

A manifold is a curved space that generalises the flat Euclidean space. A classic example is the unit sphere $\mathbb{S} \subseteq \mathbb{R}^3$. In general, a point cloud in $\mathbb{R}^p$ sampled from an unknown manifold can be described by probabilistic mapping functions and the corresponding latent spaces. In the context of differential geometry, the latent spaces can be interpreted as coordinate charts of the unknown manifold to be learned.

2.1 Atlas and Metrics of Manifolds

Let $\mathbb{M} \subseteq \mathbb{R}^p$ be a q-dimensional manifold where $q < p$. An *atlas* consists of charts that, roughly speaking, locally parametrizes the manifold. It is a collection $\mathcal{A} = \{(V_i, \varphi_i)\}_{i \in I}$, where I is an index set, $V_i \subset \mathbb{M}$ is an open subset for $i \in I$ such that $\bigcup_{i \in I} V_i = \mathbb{M}$. φ_i is a mapping (homeomorphism) from an open subset of $\mathbb{R}^q$ to the corresponding region V_i on the manifold. For example, Figure 1 illustrates two overlapping coordinate charts of a 2-dimensional manifold, namely $\varphi_1 : X \to V_1 \subset \mathbb{M}$ (blue) and $\varphi_2 : W \to V_2 \subset \mathbb{M}$ (red). Given a point s_j in the intersection $V_1 \cap V_2$ (yellow), it is represented via φ_1 by a point $x_j = \varphi_1^{-1}(s_j)$ in X and also via φ_2 by a point $w_j = \varphi_2^{-1}(s_j)$ in W. The two local representations are related by the transition operation $w_j = (\varphi_2^{-1} \circ \varphi_1)(x_j)$, also known as the *change of coordinates*. The BM path on $\mathbb{M}$ can be interpreted as the two stochastic processes in X and W.

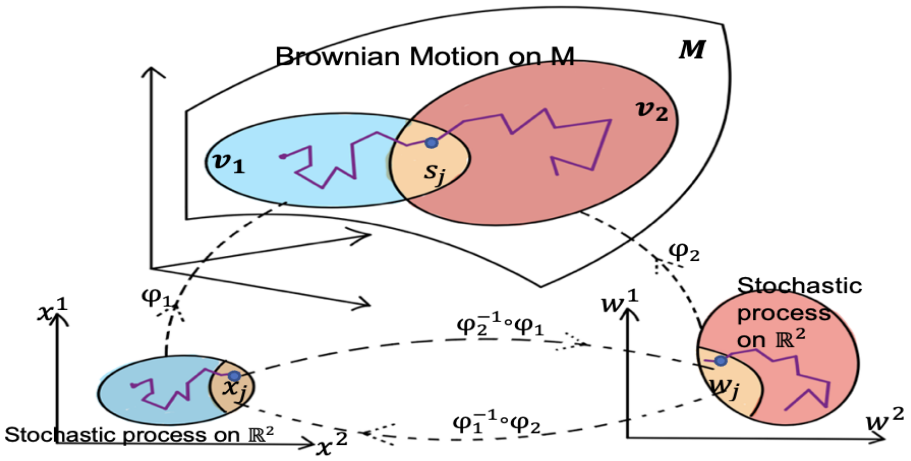


Figure 1: Atlas with two charts on $\mathbb{M}$ and change of coordinates operations.

The *tangent space* of $\mathbb{M}$ at a point $s \in \mathbb{M}$ is the q-dimensional subspace $\mathbb{T}_s\mathbb{M} \subseteq \mathbb{R}^p$ such that $s + \mathbb{T}_s\mathbb{M}$ is tangent to $\mathbb{M}$ at s. The (induced Riemannian) *metric* (Lee, 2018) on $\mathbb{M} \subseteq \mathbb{R}^p$ is defined for each $s \in \mathbb{M}$ as the inner product on $\mathbb{T}_s\mathbb{M} \subseteq \mathbb{R}^p$, obtained by restricting the standard one on $\mathbb{R}^p$. This metric plays a crucial role in simulating BM on $\mathbb{M}$, as it encodes the local geometry and governs the stochastic process. It also defines the geodesic distance function, which is in general not equal to the Euclidean distance. In a chart (V, φ) of $\mathbb{M}$, the metric is represented by a symmetric positive definite matrix-valued

function $\mathbf{g}$ on V. If $\mathcal{J}$ denotes the Jacobian of φ, then we have

$$\mathbf{g} = \mathcal{J}^T \mathcal{J}, \quad \mathcal{J}^{l,k} = \frac{\partial \varphi^l}{\partial x^k}, \tag{2}$$

where the superscripts indicate the k^{th} dimension of the chart and the l^{th} dimension of the observation space. The determinant of $\mathbf{g}$ can express how much a part of chart(latent space) is stretched or bend when mapped to $\mathbb{M}$. Moreover, the Laplace-Beltrami operator Δ_s of $\mathbb{M}$ can be expressed in local coordinates as $\Delta_s f = \sum_{l,k=1}^{q} \frac{1}{\sqrt{G}} \frac{\partial}{\partial x^k} \left(\sqrt{G} \mathbf{g}^{lk} \frac{\partial f}{\partial x^l} \right)$, where G is the determinant of $\mathbf{g}$, $\mathbf{g}^{lk}$ is the (l,k) element of its inverse and f is a smooth function on $\mathbb{M}$. In the special case where $\mathbb{M} = \mathbb{R}^q$ with its standard coordinate chart, $\mathbf{g}$ is the $q \times q$ identity matrix at every point and the Laplace-Beltrami operator Δ_s becomes the Laplace operator $\Delta = \sum_{l=1}^{q} \frac{\partial^2}{\partial^2 x^l}$.

2.2 Heat Kernel and BM on $\mathbb{M}$

Consider the heat equation on $\mathbb{M}$, given by $\frac{\partial}{\partial t} h(s_0, s_j, t) = \Delta_s h(s_0, s_j, t),\ s_0, s_j \in \mathbb{M}$, where s_0 and s_j are points on $\mathbb{M}$, $t \in \mathbb{R}^+$ is the diffusion time, and Δ_s is the Laplacian-Beltrami operator on $\mathbb{M}$. A heat kernel of $\mathbb{M}$ is a smooth function $h(s_0, s_j, t)$ on $\mathbb{M} \times \mathbb{M} \times \mathbb{R}^+$ that satisfies the heat equation and the initial condition $\lim_{t \to 0} h(s_0, s_j, t) = \delta(s_0, s_j)$, where δ is the Dirac delta function. It can be interpreted as the amount of heat that is transferred from s_0 to s_j in time t via diffusion. The heat kernel becomes unique when we impose a suitable condition along the boundary $\partial\mathbb{M}$, such as the Neumann boundary condition. If $\mathbb{M}$ is a Euclidean space $\mathbb{R}^q$, the heat kernel has a closed form expression corresponding to a time-varying Gaussian function: $h(s_0, s_j, t) = \frac{1}{(2\pi t)^{q/2}} \exp\left\{ -\frac{||s_0 - s_j||^2}{4t} \right\},\ s_j \in \mathbb{R}^q$. The diffusion time t controls the rate of decay of the covariance.

Solving the heat equation on arbitrary manifolds presents substantial challenges due to the complexity of their geometric structure (Chavel, 1984). To avoid directly solving the heat equation, Niu et al. (2019) proposed estimating the heat kernel by simulating BM paths on a known manifold and using the transition density as an approximation. However, their approach is limited to manifolds that can be described with a single chart.

3. Learn the Probabilistic Atlas $\mathcal{A}$ of $\mathbb{M}$

We construct the probabilistic atlas in two steps. First, the point cloud is decomposed into overlapping subsets that form an cover of the manifold. Second, a local coordinate chart and mapping function are learned on each subset. This section describes both steps and explains how the resulting charts are combined into the atlas used for BM simulation.

3.1 Construction of the Atlas Cover

We use tools from topological data analysis (TDA) to guide this construction. TDA extracts topological features of point clouds, such as connected components, loops, and voids, often through Vietoris–Rips filtrations and persistent homology (Carlsson, 2009; Edelsbrunner and Harer, 2010). Persistent homology summarizes these features in a persistence diagram: points close to the diagonal are usually treated as topological noise, while points far from

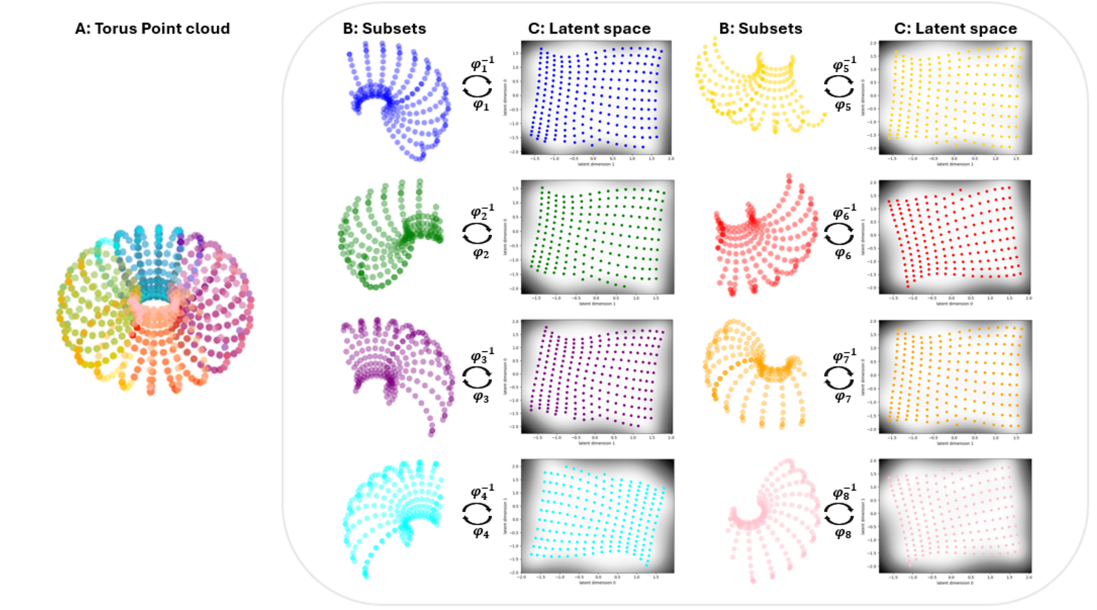


Figure 2: The torus atlas and point cloud.

the diagonal indicate persistent features that should not be represented by a single local chart.

Let $\mathcal{S} = \{s_j | j = 1, \ldots, n\}$, with $s_j \in \mathbb{R}^p$ and $n = n_d + n_u$, denote all point-cloud inputs, including labeled points from $\mathcal{D}$ and unlabeled points from $\mathcal{U}$. To handle manifolds that cannot be represented by a single chart, we decompose $\mathcal{S}$ into overlapping subsets, each of which is simple enough to be represented by one local chart. We use Mapper (Singh et al., 2007; van Veen et al., 2019) as the main tool for this decomposition. Mapper projects the data to a low-dimensional filter space, divides the filter space into overlapping intervals or bins, and then clusters the original observations within each bin. This produces a collection of overlapping subsets $\mathcal{S} = \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_{n_v}$, where $\mathcal{S}_i = \{s_{ij} : j = 1, \ldots, ns_i\}$ and neighbouring subsets have nonempty intersections. The union of the subsets covers the observed point cloud. An example of torus subsets is shown in column B of Figure 2, with each subset marked in a different color.The corresponding learned latent spaces are shown in column C.

To check whether a subset can be represented by a single chart, we compute persistent homology for each subset. If a subset contains significant persistent features, it is further partitioned. This refinement is repeated until each subset has no significant persistent topological features. The resulting subsets form the empirical atlas cover used in the subsequent BM simulation. Persistent diagrams for the Torus, U-shape, Aral Sea, and Shark Prey image data sets, together with their corresponding subsets, are provided in Appendix J, where the individual subsets exhibit no significant persistent topological features. Appendix J also provides further implementation details, including two possible procedures for constructing overlapping subsets whose persistent diagrams indicate trivial topology.

3.2 Learning Local Charts

Given a subset $\mathcal{S}_i$, we can learn the coordinate chart and mapping function. For each subset $\mathcal{S}_i$, we can define a latent variable model, which introduces a set of unobserved variables $\mathcal{X}_i = \{x_{ij}|j = 1, \cdots, ns_i\}$, $x_{ij} \in \mathbb{R}^q$. The latent $\mathcal{X}_i$ is associated with $\mathcal{S}_i$ observed in a higher-dimensional space as follows:

$$s_{ij} = \varphi_i(x_{ij}) + \epsilon_{ij}, \quad s_{ij} \in \mathcal{S}_i, \tag{3}$$

where ϵ_{ij} is an iid noise term. x_{ij} is a latent variable. i is the index of subset. If p, the dimension of the data space, is larger than q, the dimension of the latent space, Equation 3 can be seen as a model for dimension reduction. Various approaches can be used to learn the mapping φ_i and the latent space $\mathcal{X}_i$, such as the Gaussian process latent variable model (GPLVM,Lawrence 2005) and autoencoder (AE,Baldi 2012). Both approaches have been used in this work. We present the construction of GPLVM atlas in the main paper. The AE atlas is explained in Appendix B. If p is the same as q, the latent space becomes a subset of $\mathbb{R}^p$. This lead to a natural chart for $\mathbb{M}$. In the U-shape example in Section 6.2, we have $q = p = 2$. $\mathbb{R}^2$ becomes a natural chart. The mapping becomes an identity mapping. The intrinsic dimension q is known in the synthetic examples. For real point-cloud data, q can be selected using data-driven diagnostics. In this work, we consider automatic relevance determination (ARD) within the GPLVM framework (Lawrence, 2000; Zwiessele, 2017) and cross-validation based on reconstruction performance, following the noise-splitting idea of Rasines and Young (2023). Appendix K reports these diagnostics for the torus point cloud and the shark-prey image data.

GPLVM (Lawrence, 2005) is a GP-based dimensionality reduction method designed to learn a probabilistic mapping between a low-dimensional latent space and observed data. The probabilistic model is computed by placing a GP prior over the mapping, $\varphi_i(x) \sim \mathcal{GP}(0, k(x, x'))$, where k is the RBF kernel. The latent variables $\{x_{ij}|j = 1, \cdots, ns_i\}$ are inferred by maximizing the log-likelihood of the latent variable model. The predictive distribution of GPLVM acts as the forward map, denoted by φ_i, enabling the transformation of a latent point x_{i*} into its corresponding observation space point s_{i*}. Conversely, the backward mapping φ_i^{-1} facilitates the estimation of a latent point x_{i*} given an observation s_{i*}. In practice, the backward map is approximated via an optimization procedure that maximizes the log-likelihood conditioned on the observed data. Using both the forward and backward maps, we can map points between different charts. Given two charts i and j, the *transition map* is defined as $\varphi_i \circ \varphi_j^{-1}$. Further details regarding the forward and backward mappings in the GPLVM framework, are provided in Section C of the Appendix.

An example of Torus atlas is shown in Figure 2. Eight subsets are created from the torus point cloud consisting of 625 points. Given a three dimensional torus subset $\mathcal{S}_i$ in column B of Figure 2, we learn a two-dimensional latent space $\mathcal{X}_i$ and the corresponding mapping function φ_i in column C. The background shading in the latent spaces represents the uncertainty of mapping φ_i, with darker areas indicating higher variance. Applying GPLVM to each subset separately results in multiple charts. The collection of all learned charts becomes the probabilistic atlas $\mathcal{A}$ of $\mathbb{M}$. The intersections of the subsets $\mathcal{S}_i \cap \mathcal{S}_j$ on $\mathbb{M}$ are used to identify the overlapping regions within the latent space (see Appendix F for details). We further assess the quality of the learned atlas using reconstruction error in

the observation space and. When ground truth is available, geodesic-distance comparisons; these diagnostics are reported in Appendix L.

To learn the associated metric tensor of the i^{th} chart (latent space), we also need to estimate the Jacobian of φ_i. Since the RBF kernel is differentiable, Rasmussen and Williams (2006) show that the derivative of a GP is also a GP. Therefore, given a point in the latent space, we can derive the distribution of the Jacobian $p(\mathcal{J}_i|\mathcal{X}_i, S_i) = \prod_{l=1}^{p} \mathcal{N}(\mu^l_{J_i}, \Sigma_{J_i})$. The expressions for $\mu^l_{J_i}$ and Σ_{J_i} are shown in Appendix C. Given Equation 2, the resulting metric $\mathbf{g}_i$ of the i^{th} chart follows a non-central Wishart distribution (Anderson, 1946). The expected metric tensor $\mathbb{E}(\mathbf{g}_i)$ is defined in Equation 4.

$$\mathbb{E}[\mathbf{g}_i] = \mathcal{G}_i = \mathbb{E}(\mathcal{J}_i^T)\mathbb{E}(\mathcal{J}_i) + p\Sigma_{J_i}. \tag{4}$$

4. Estimate the Heat Kernel via BM on $\mathbb{M}$

Given the learned atlas, we will estimate the heat kernel as the BM transition density on $\mathbb{M}$. BM paths are simulated using the learned metrics of the atlas from Section 3.

4.1 BM on $\mathbb{M}$ as the SDE on the Atlas $\mathcal{A}$

For a single chart manifold, simulating BM sample paths on $\mathbb{M} \subset \mathbb{R}^p$ is equivalent to simulating the stochastic process in the chart in $\mathbb{R}^q$, $q \leq p$. BM on a Riemannian manifold in a local coordinate system (chart) is given as a system of stochastic differential equations (SDEs) in Itô form (Hsu, 1988, 2008). Given the atlas $\mathcal{A} = \{(V_i, \varphi_i)|i \in \{1, \cdots, n_v\}\}$ on $\mathbb{M}$, where n_v is the number of the charts and $\cup_{i=1}^{n_v} V_i = \mathbb{M}$, we can define a group of SDEs on $\mathcal{A}$ that is equivalent to the BM on $\mathbb{M}$. For each φ_i in $\mathcal{A}$, we can compute the expected metric $\mathcal{G}_i = \mathbb{E}[\mathbf{g}_i]$ as in Equation 4 and the SDE in the i^{th} chart can be derived as

$$dx_i^l(t) = \frac{1}{2} {G_i}^{-1/2} \sum_{r=1}^{q} \frac{\partial}{\partial x_i^r} \left({\mathcal{G}_i}^{lr} {G_i}^{1/2} \right) dt + \left({\mathcal{G}_i}^{-1/2} dB_i(t) \right)_l, \tag{5}$$

where x_i^l represents the coordinate in the l^{th} dimension of chart i (latent space i). $\mathcal{G}_i$ is defined in Equation 4, G_i is the determinant of $\mathcal{G}_i$, and $B_i(t)$ represents an independent BM in the Euclidean space.

Since each chart covers only a part of the original manifold, we introduce the probabilistic change of coordinates operations for transitioning SDEs between charts within an atlas. Specifically, BM trajectories are simulated by transitioning between different local coordinates using Equation 5 and the proposed transition map. Figure 3 serves as a demonstrative example. Consider a point cloud representation of the manifold $\mathbb{M}$ shown in the middle panel. Two overlapping subsets (green and brown) on $\mathbb{M}$ are used to learn two charts. While the atlas $\mathcal{A}$ may consist of multiple charts to cover $\mathbb{M}$, we illustrate the process with two charts for simplicity. Chart 1 (left panel) parameterizes the brown region, and Chart 2 (right panel) parameterizes the green region. The intersection of these regions on $\mathbb{M}$ is non-empty, defining an identifiable overlapping region. This overlapping region serves as the switching domain in the latent space. In the middle panel, the red star denotes the starting point of the BM path. It can be mapped to chart 1 via φ_1^{-1}. In Chart 1, the stochastic process is depicted as the purple line. Upon entering the switching domain, the

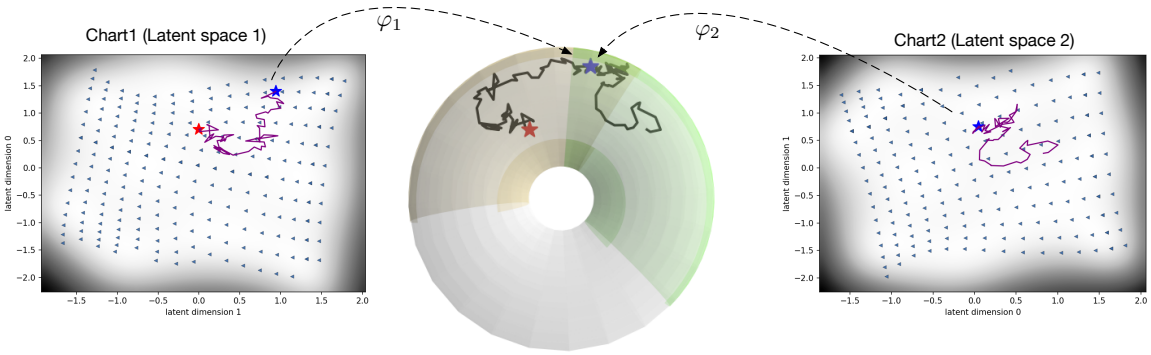


Figure 3: A demonstration of simulating a BM path by switching the stochastic process in different charts (or latent spaces). The collection of charts forms the atlas of the torus manifold. The black trajectory on the torus (middle panel) represents a BM path with a red star indicating the starting point. The green region of the torus is described by Chart 2 on the right. The brown region of the torus is describe by Chart 1 on the left. The purple trajectory in Chart 1 represents a stochastic process simulated by Equation 5 with $i = 1$. Once the path reaches the overlapping region, indicated by the blue star x_{1j} on the left panel, the blue star are transferred to Chart 2 using the change of coordinates operation: $x_{2j} = \varphi_2^{-1} \circ \varphi_1(x_{1j})$. The stochastic process continues in Chart 2, as shown by the purple trajectory in the right panel (simulated by Equation 5 with $i = 2$). The projection of both segments of purple lines from the charts onto torus generates the BM path shown in the middle panel.

step of SDE, marked by a blue star in Chart 1, is mapped back to $\mathbb{M}$ (the overlapping region) using φ_1 and subsequently to Chart 2 via φ_2^{-1}. The transition map from Chart 1 to Chart 2 is given by $\varphi_2^{-1} \circ \varphi_1$. The ensuing trajectory is simulated as the stochastic process(purple line) originating from the blue star in Chart 2 in the right panel. When we map the purple trajectories from Chart 1 and Chart 2 to $\mathbb{M}$, they form the black BM path in the middle panel.

While this example involves two charts covering the overlapping region, in general, multiple charts may cover the same region. In such cases, there are several target charts to which the simulated steps can transit to. According to Theorem 1, simulations using any target chart result in equivalent steps on $\mathbb{M}$. We can randomly pick one of the target charts and use the transition map to switch the chart. The proof is given in Appendix A.1.

Theorem 1. *If the region on $\mathbb{M}$ can be parameterised by multiple charts, the stochastic process defined in Equation 5 is chart independent. With a given diffusion time, simulations in any choice of charts are equivalent to the same step in $\mathbb{M}$.*

The predictive variance $\sigma^2(x)$ of the GPLVM mapping $\varphi_i(x)$ provides a natural measure of chart reliability. Since GPLVMs are Gaussian-process models, $\sigma^2(x)$ is typically small near observed data, increases away from the data, and approaches the prior variance $\sigma^2_{\max}$ in regions where the mapping is weakly constrained. Large predictive variance therefore indicates that the local coordinate representation may be unreliable. To prevent BM trajectories from propagating through such regions, we define uncertainty-aware chart boundaries

using a fixed fraction of $\sigma^2_{\max}$. Proposed BM steps that leave the reliable domain of the current chart are rejected and resampled. The complete simulation procedure is summarised in Algorithm 1.

Algorithm 1 Simulating Brownian-motion sample paths on $\mathbb{M}$

Step 1: Construct a probabilistic atlas $\{(\widehat{\mathcal{X}}_i, \varphi_i)\}_{i=1}^{n_v}$ from point-cloud subsets. $\widehat{\mathcal{X}}_i$ is the reliable domain of chart i, defined using the GPLVM mapping uncertainty. Identify the overlapping regions $\widehat{\mathcal{O}}_{ij}$ from the subset intersections. These are overlap regions that admit valid coordinate representations in both charts and lie within their reliable domains.
Step 2: Simulate SDE sample paths on the probabilistic atlas.
for $\kappa = 1, \ldots, N_s$ **do** ▷ N_s starting points
  for $j = 1, \ldots, N_{bm}$ **do** ▷ N_{bm} paths per starting point
    Select an initial active chart i containing s_κ.
    Set $x_i(0) = \varphi_i^{-1}(s_\kappa)$.
    for $\tau = 1, \ldots, T$ **do** ▷ $T\Delta t$ is the maximum diffusion time
      accepted $\leftarrow$ false.
      while not accepted **do**
        Propose a candidate step using Equation 5
        if $x_i^*(\tau) \notin \widehat{\mathcal{X}}_i$ **then** ▷ Outside the reliable chart domain
          Reject and resample.
        else
          Accept the proposal $x_i(\tau) \leftarrow x_i^*(\tau)$.
          if $x_i(\tau) \in \widehat{\mathcal{O}}_{ij}$ for one or more neighbouring charts **then**
            Let $\mathcal{C}\{x_i(\tau)\}$ be the set of charts containing valid representations of $x_i(\tau)$.
            Select an active chart $i_s \in \mathcal{C}\{x_i(\tau)\}$. ▷ The current chart may be retained
            if $i_s \neq i$ **then**
              $x_{i_s}(\tau) = \varphi_{i_s}^{-1} \circ \varphi_i\{x_i(\tau)\}$. ▷ Change of coordinates
              $i \leftarrow i_s$. ▷ Theorem 1
          accepted $\leftarrow$ true.
return simulated paths and active chart indices.

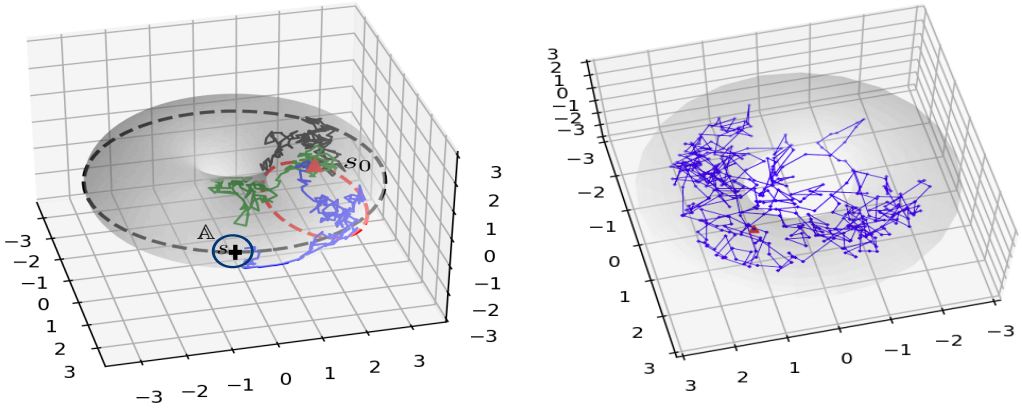
While a trajectory remains in a valid overlap region, its active chart may be retained or changed at each simulation step. Once it leaves the overlap region, the simulation continues in the chart that is active at the time of exit. The trajectory may subsequently re-enter an overlap region and switch to another valid chart representation. By Theorem 1, all such representations correspond to the same Brownian-motion path on $\mathbb{M}$.

In practice, the valid overlap regions $\widehat{\mathcal{O}}_{ij}$ are identified from observations shared by the corresponding point-cloud subsets during atlas construction. For each chart, a classifier is used to distinguish chart-specific regions from overlap regions with neighbouring charts. The resulting overlap regions are then restricted to the reliable domains of the participating charts, ensuring that chart transitions occur only where both coordinate representations are reliable.

4.2 Heat Kernel Estimation as Transition Density of BM on the Atlas $\mathcal{A}$

We estimate the heat kernel by simulating BM sample paths on $\mathbb{M}$ and numerically evaluating the transition density. Let $\{S(t) : t > 0\}$ denote BM on $\mathbb{M}$, starting at $S(0) = s_0$. We generate N sample paths by Equation 5 and transition maps. For any $t > 0$ and $s \in \mathbb{M}$,

the transition probability $p(S(t) \in \mathbb{A} \mid S(0) = s_0)$ is approximated by the proportion of BM paths that reach $\mathbb{A}$ at time t, where $\mathbb{A}$ is a small neighborhood of s. An illustrative diagram is presented in Figure 4(a). The red triangle denotes the starting point of the BM paths. Three paths are simulated using the SDEs on the atlas of torus as in Figure 2. Of these paths, only one reaches the neighborhood of the blue cross (target point), resulting in a transition probability of 1/3. For comparison, Figure 4(b) presents a trajectory simulated using SDEs on a single-chart torus, following the approach in Niu et al. (2023). The single-chart approach results in the path leaving the torus surface and crossing through the central void, illustrating the limitations of the single-chart approach.



(a) Sample paths of atlas BM

(b) Sample path of single chart SDE

Figure 4: BM paths on $\mathbb{M}$: (a)s_0 (red triangle) is the starting point of three BM paths (black, green and blue solid lines). Only the blue path reaches the neighbourhood $\mathbb{A}$ (blue circle) of s (black cross) at time t. Therefore the estimate of the transition probability $p(S(t) \in \mathbb{A} \mid S(0) = s_0) = \frac{1}{3}$. (b) The blue path, generated using the SDEs on a single chart (Niu et al., 2023), starts at the red triangle and traverses through the central void of the torus, demonstrating that it does not correspond to a proper BM on the torus.

The transition density of BM at s can be approximated as:

$$\hat{h}^t(s_0, s) \approx \frac{p(S(t) \in \mathbb{A} \mid S(0) = s_0)}{V(\mathbb{A})} \approx \frac{1}{V(\mathbb{A})} \cdot \frac{N_A}{N} \tag{6}$$

where $\hat{h}^t(s_0, s)$ is the heat kernel estimator, N is the number of simulated BM paths and N_A is the number of BM paths which reach $\mathbb{A}$ at time t, $V(\mathbb{A})$ is the Riemannian volume of $\mathbb{A}$. Note that the BM diffusion time t operates akin to the smoothing parameter in the RBF kernel. A larger t implies a higher probability of BM reaching the neighbourhood of the target point, consequently leading to increased covariance, and vice versa. Given Theorem 2, this estimator is asymptotically unbiased and consistent.

Theorem 2. *The estimator $\hat{h}^t(s_0, s)$ in Equation 6 is asymptotically unbiased.*

The proof of Theorem 2 is given in Appendix A.2. We can also derive the error bound in Corollary 3.

Corollary 3 (Finite-sample error bound)**.** *Let A be a neighbourhood of s with window size w so that* $\mathrm{Vol}(A) = O(w^d)$ *where d is the intrinsic dimension of the manifold. $\widehat{h}_t(s_0, s)$ is the heat kernel estimator. Then*

$$Var(\hat{h}_t(s_0, s)) = O(N^{-1}w^{-d}),$$

and

$$\hat{h}_t(s_0, s) - h_t(s_0, s) = O_p(N^{-1/2}w^{-d/2}) + O(w^2).$$

The convergence rate $O_p(N^{-1/2}w^{-d/2})$ corresponds to the standard Monte Carlo convergence rate, while the $O(w^2)$ term arises from the local approximation of the heat kernel over the neighbourhood set A. Theorem 2 and Corollary 3 establish consistency of the estimated heat kernel. Consequently, the covariance matrix used by AtlasGP converges to that obtained from the population heat kernel. Existing theoretical work on Gaussian processes over manifolds provides supporting evidence for the asymptotic behaviour of the resulting predictor. Castillo et al. (2014) established posterior contraction rates for heat-kernel Gaussian process priors on compact manifolds, while Yang and Dunson (2016) showed that Gaussian-process regression on manifolds can achieve convergence rates determined by the intrinsic manifold dimension. Although these results assume the manifold geometry is known and therefore do not directly apply to AtlasGP, they suggest that if the atlas approximation error and heat-kernel estimation error are asymptotically negligible, AtlasGP should recover the asymptotic behavior of the corresponding oracle manifold GP.

When the estimated manifold closely approximates the true manifold, the resulting BM transition density estimates will resemble those obtained through analytical parameterization. We utilize the Torus as a case study. By assuming the Torus's geometry is known, we can adopt the methodology outlined by Niu et al. (2019) to assess the heat kernel $h^t(s_0, s)$ through BM transition density using the analytical metric tensor. The derivation of the analytical metric and the parameterization of the Torus are detailed in Appendix I. Consider the red triangle in Figure 4 as our starting point for the BM simulation. The BM transition density is evaluated with 60 evenly distributed target points along the black horizontal dashed circle in Figure 4. The hyper-parameter t (diffusion time) remains constant at 4. Results are depicted as solid red lines in Figure 5, where the horizontal axis represents the outer circle angle θ of the Torus, and the vertical axis represents the transition density.

In the case of the point cloud, the geometry and topology of the Torus are unknown. Following the approach in Section 3, the atlas can be learned using GPLVM and AE. BM trajectories are simulated on the estimated atlas, and the corresponding heat kernel estimates with GPLVM metrics are represented by blue dashed line in Figure 5. The estimates using AE metric are plotted as the purple dashed line in Figure 5. It is evident that these GPLVM results closely match the red solid line. Heat kernel estimates derived from the Graph Laplacian (GL) method (Dunson et al., 2020) across various data scenarios are depicted in Figure 5. The hyperparameters of the GL method are selected using a grid search that maximizes the marginal likelihood. Initially, when the Torus data set comprises 625 points, the GL kernel estimates are represented by the green dash dotted line, noticeably deviating from the solid red line for analytical kernel estimates. However, as the number of grid points escalates to 2500 and 6400, the GL kernel estimates shift to the black dotted line and brown dashed line, respectively. With this increase in the number of points, the

estimates progressively converge toward the solid red line. Compared to the GL method, kernel estimates utilising GPLVM metrics exhibit superior performance even with fewer points on the Torus. We also applied the single chart BM method (Niu et al., 2023), with the estimates plotted as the yellow dotted line. The results demonstrate that the single chart method does not align with the analytical estimates. Examples of the latent space and BM trajectories for the single chart method are available in Appendix E. To further illustrate the robustness of our BM-on-atlas approach, we modified the configuration of the Torus to generate a non-regular, non-uniformly distributed point cloud. Even under these conditions, the kernel estimates produced by our method closely match the analytical results. Additional results are provided in Appendix D. The numerical accuracy of the heat kernel estimates for the Torus is examined by varying the number of simulated BM paths, as shown in Figure 22 of Appendix I. With a number of simulated paths on the order of 10^4, the estimates closely match the analytical solution.

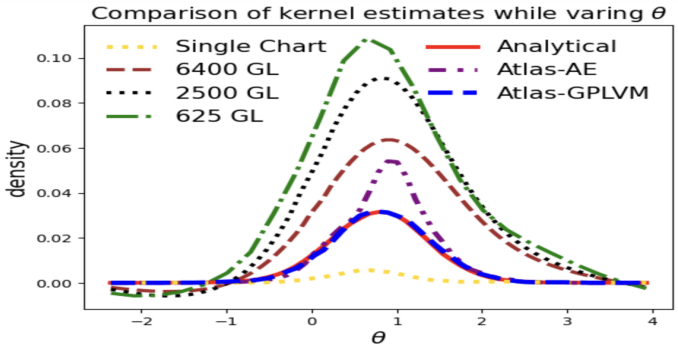


Figure 5: Kernel estimates comparison. The analytical kernel density is plotted as a solid red line. The kernel estimates using BM transition density with the GPLVM atlas are represented by a dashed blue line. The kernel estimates using BM transition density with the AE atlas are shown as a dashed purple line. Both atlases are constructed using 625 points on the Torus. The GL kernel estimates are plotted as a green dash-dotted line when the number of points is 625. We also plot the GL estimates as a black dotted line and a brown dashed line when the number of points increases to 2,500 and 6,400, respectively. The kernel estimates using single chart BM are shown in yellow dotted line.

5. Construct the Riemannian-Corrected Atlas GP

A straightforward approach to constructing the atlas GPs is to use the heat kernel estimates in Equation 6 for the covariance function in GPs. However, this method requires simulating BM paths for all grid points. Let N_{BM} denote the number of BM paths simulated from each point, and let n be the total number of points on $\mathbb{M}$. The direct intrinsic Atlas GP(AGP) therefore requires approximately $n \times N_{\text{BM}}$ simulated paths, which can be computationally expensive. It is also sensitive in finite samples to Monte Carlo error and numerical asymmetry in the estimated heat-kernel matrix.

To address these issues, we propose the Riemannian-Corrected Atlas GP (RC-AGP). Instead of using the estimated heat kernel as the entire covariance function, RC-AGP uses

it as a geometric correction to a standard positive-definite base kernel. In this work, we use the RBF kernel as the base kernel. The Brownian-motion correction downweights covariance between points that are close in Euclidean space but geometrically distant on the manifold, while the base RBF kernel provides a stable local covariance model. Thus, the RBF component should not be interpreted as increasing the sample-path smoothness of a heat-kernel GP; rather, it provides a stable baseline covariance, making the regression less sensitive to finite-sample Monte Carlo errors in the estimated heat kernel. This construction also substantially reduces the BM simulation cost. Instead of simulating BM paths from all n data points, RC-AGP simulates BM paths only from n_v atlas representatives, where n_v is the number of charts and typically $n_v \ll n$. The simulation cost is therefore reduced from approximately $n \times N_{\mathrm{BM}}$ to $n_v \times N_{\mathrm{BM}}$. Once these chart-level heat-kernel estimates have been constructed, they can be reused for different training data sets on the same manifold. A detailed discussion of the BM simulation cost, together with standardized runtime benchmarks for the data sets considered in this paper, is provided in Appendix O.

As a result, RC-AGP provides a computationally efficient and numerically stable way to incorporate intrinsic geometry into GP regression. For comparison, we also developed the Sparse Atlas GP (S-AGP), which integrates the Atlas BM framework with the subset of regressors approximation (Smola and Bartlett, 2000). While this section focuses on RC-AGP, details on S-AGP can be found in Appendix G. As demonstrated in subsequent regression experiments, the RC-AGP consistently outperforms all other comparison methods across most tasks.

Let the atlas of $\mathbb{M}$ consist of n_v charts, with the associated data cloud $\mathcal{S}$ represented as the union of n_v subsets: $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cdots \cup \mathcal{S}_{n_v}$. We select the centre points (median points) from each $\mathcal{S}_i$ as the starting points for the BM trajectories simulated by Equation 5. The transition density between these center points is then estimated using Equation 6. Given a diffusion time t, we have an $n_v \times n_v$ covariance matrix, named K_h^t, the heat kernel matrix. Let K_{rbf} be a $n \times n$ covariance matrix computed from a standard Euclidean RBF kernel. The RC-AGP can be constructed by first expanding the $n_v \times n_v$ matrix K_h^t to $n \times n$ matrix $\tilde{K}_h^t$ with repeating elements, and then taking hadamard product with K_{rbf}.

For example, let $\mathbb{M}$ be represented by a point cloud consisting of n points. Assume the point cloud can be partitioned into three overlapped subsets $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$. Three charts are learned from the subsets. By initiating BM paths from the center points of these subsets and simulating the process over a diffusion time t, we can derive a 3×3 matrix K_h^t which is the heat kernel evaluated at the three center points of $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$:

$$K_{h_u}^t = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix},$$

where h_{ij} represents the heat kernel estimated at point i and j by Equation 6. Let n_{S_i} be the number of points in $\mathcal{S}_i$. We can expand K_h and create a $n \times n$ matrix $\tilde{K}_h$ by repeating elements.

$$\tilde{K}_h^t = \begin{bmatrix} h_{11} * \mathbf{1_{S1}} & h_{12} * \mathbf{1_{S12}} & h_{13} * \mathbf{1_{S13}} \\ h_{21} * \mathbf{1_{S21}} & h_{22} * \mathbf{1_{S2}} & h_{23} * \mathbf{1_{S23}} \\ h_{31} * \mathbf{1_{S31}} & h_{32} * \mathbf{1_{S32}} & h_{33} * \mathbf{1_{S3}} \end{bmatrix} \tag{7}$$

$\mathbf{1_{S_i}}$ represent a $n_{S_i} \times n_{S_i}$ matrix of ones. $\mathbf{1_{S_{ij}}}$ is a $n_{S_i} \times n_{S_j}$ matrix of ones. The Riemannian corrected kernel matrix can be calculated as:

$$K_{RC} = K_{rbf} \odot \tilde{K}_h^t, \tag{8}$$

where K_{rbf} is an $n \times n$ matrix created from the RBF kernel. $\odot$ stand for the Hadamard product. Given Theorem 4, K_{RC} is semi-positive definite and can be used to construct GP on manifolds. The proof of Theorem 4 is given in Appendix A.3. The Riemannian corrected kernel is also differentiable. This framework is very flexible. The RBF kernel can be replaced by other valid covariance kernels.

The exact heat kernel is a positive-semidefinite kernel on $\mathbb{M}$. In practice, AtlasGP uses a Monte Carlo estimator obtained from simulated Brownian-motion trajectories. Consequently, finite-sample estimates of the heat-kernel matrix may exhibit small violations of positive semidefiniteness.

Theorem 4. *Let K_h denote the population heat-kernel matrix and let K_{local} be a positive-semidefinite local kernel matrix. Then the population Riemannian-corrected kernel matrix defined in Equation 8 is positive semidefinite.*

Remark. Theorem 4 is stated for the population heat-kernel matrix. In practice, AtlasGP uses the estimator $\widehat{K}_h$ obtained from Brownian-motion simulations. By Theorem 2 and Corollary 3, $\widehat{K}_h$ converges to K_h as the number of simulated paths increases. Consequently, the estimated Riemannian-corrected kernel converges to its population counterpart, which is positive semidefinite by Theorem 4. Any finite-sample violations of positive semidefiniteness are attributable to Monte Carlo approximation error, become asymptotically negligible as the number of simulated paths increases, and are eliminated numerically by eigenvalue truncation.

Under an RC-AGP prior for the unknown regression function as $f \sim GP(0, K_{RC}(.,.))$,

$$p(\mathbf{f}|s_1, s_2, ..., s_n) \sim \mathcal{N}(0, \Sigma_{\mathbf{ff}}),$$

where $\mathbf{f}$ is the discretisation of f over the labeled points $s_1, s_2, \cdots, s_{n_d}$ so that $f_i = f(s_i)$. It is straightforward to identify which subsets the labeled points belong to and therefore construct the corresponding $n_d \times n_d$ $\tilde{K}_h^t$ as in Equation 7. The covariance matrix $\Sigma_{\mathbf{ff}}$, also of size $n_d \times n_d$, is computed using the Riemannian corrected kernel from Equation 8. The (i,j)-th entry of $\Sigma_{\mathbf{ff}}$ is given by $\Sigma_{\mathbf{ff}_{i,j}} = \sigma_r^2 K_{RC}(s_i, s_j)$. There are three hyper-parameters in this construction – the diffusion time t of K_h, the length-scale l of K_{rbf} and the rescaling parameter σ_r. Optimisation of the hyper-parameters can be done by maximising the log marginal likelihood in Equation 9:

$$\log p(\boldsymbol{y}|s) = \log \int p(\boldsymbol{y}|\mathbf{f}) p(\mathbf{f}|s) d\mathbf{f} = -\frac{1}{2}\boldsymbol{y}^T(\Sigma_{\mathbf{ff}} + \sigma_{noise}^2 I)^{-1}\boldsymbol{y} - \frac{\log|\Sigma_{\mathbf{ff}} + \sigma_{noise}^2 I|}{2} - \frac{n \log 2\pi}{2} \tag{9}$$

Let $\mathbf{f}_*$ denote the function values at the unlabelled test points. $\Sigma_{\mathbf{f}_*\mathbf{f}_*}$ and $\Sigma_{\mathbf{ff}_*}$ can also be derived from the Riemannian corrected kernel by checking which subset x_* belongs to. The predictive distribution is

$$p(\mathbf{f}_*|\boldsymbol{y}) = \mathcal{N}(\Sigma_{\mathbf{f}_*\mathbf{f}}(\Sigma_{\mathbf{ff}} + \sigma_{noise}^2 I)^{-1}\boldsymbol{y}, \Sigma_{\mathbf{f}_*\mathbf{f}_*} - \Sigma_{\mathbf{f}_*\mathbf{f}}(\Sigma_{\mathbf{ff}} + \sigma_{noise}^2 I)^{-1}\Sigma_{\mathbf{ff}_*}). \tag{10}$$

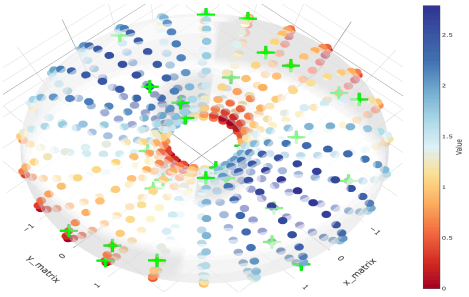
The Hadamard product is not the only possible composition. An alternative approach is to combine the expanded heat-kernel matrix and the local covariance additively,

$$K_{\text{add}} = \sigma^2 \left\{ \alpha \overline{K}_h + (1-\alpha)\overline{K}_{\text{local}} \right\} + \sigma_n^2 I.$$

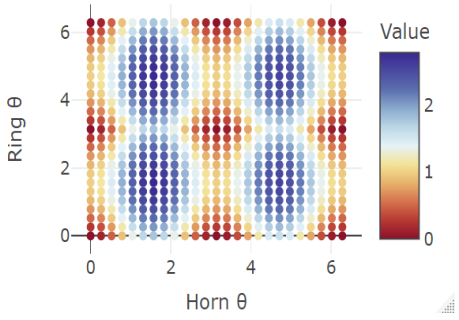
where, $\overline{K}_h$ and $\overline{K}_{\text{local}}$ are normalized matrices. The additive weight and kernel hyperparameters are selected by maximizing marginal likelihood. The comparison of the productive construction and additive construction are reported in Appendix M. It shows that the additive construction performs similarly to, but slightly worse than, the Hadamard product on the torus and is substantially less accurate on the U-shape. The product construction is particularly beneficial when Euclidean proximity conflicts with intrinsic connectivity, because the heat-kernel component suppresses geometrically inappropriate local correlations.

6. Applications

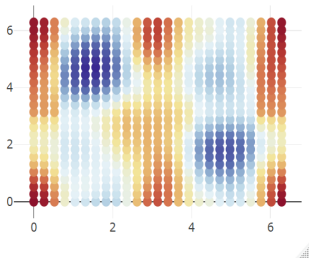
6.1 Torus



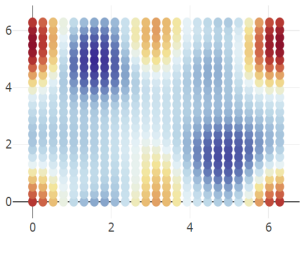
(a) True function on the torus in $\mathbb{R}^3$



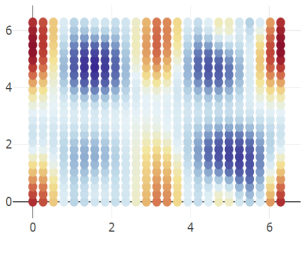
(b) True function on the torus (2D)



(c) RC-AGP prediction



(d) Euclidean GP prediction



(e) GL-GP prediction

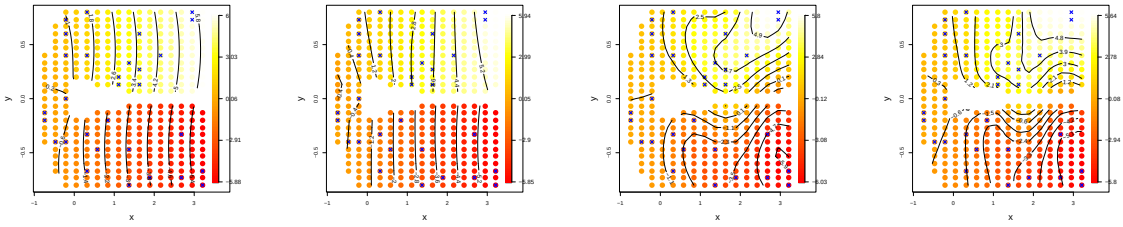
Figure 6: Comparison of different methods on the torus. (a) The true function on the torus point cloud in $\mathbb{R}^3$, where the green crosses denote the training observations. (b) The true function values on the 2D parameter domain. (c) RC-AGP prediction. (d) Euclidean GP prediction using the RBF kernel. (e) GL-GP prediction. In (b)–(e), the red-to-blue color scale represents the function values at the grid points.

SNR	*Euclidean* $\mathbb{R}^3$ *GP*	*GL-GP*	*RC-AGP*	*S-AGP*
30db	0.42(0.09)	0.42(0.04)	0.33(0.06)	0.41(0.08)
20db	0.43(0.10)	0.42(0.04)	0.34(0.05)	0.44(0.13)
10db	0.48(0.09)	0.44(0.04)	0.40(0.05)	0.48(0.12)

Table 1: Comparison of the RMSE of predictive means for various methods on Torus point cloud.

In this section, we conduct a simulation study for a regression model with synthetic data on a Torus, a two-dimensional manifold depicted by a point cloud in $\mathbb{R}^3$. The point cloud and the true function values, indicated by color codes, are plotted in Figure 6(a). The point cloud comprises $n = 625$ points, with $n_d = 30$ labeled points marked as green crosses in Figure 6(a). The analytical chart is parameterised by the angles of the outer and inner circles of the torus. The regression function varies according to these angles. However, in the context of the point cloud, these angles are unknown. Only the three-dimensional coordinates of the point cloud and the function values at the labeled points are observed. To learn the atlas, the torus cloud is divided into overlapping subsets, as shown in Figure 2. GPLVM is employed to learn the charts for each subset. The union of these learned charts constitutes the atlas of the Torus point cloud in Figure 2C. Appendix K shows that both ARD and cross-validation estimate the latent dimension to be two, which is also consistent with the known intrinsic dimension of the torus. We therefore set $q = 2$ for each local GPLVM chart. Appendix L reports additional atlas diagnostics for this data set, including geodesic-distance comparisons and reconstruction errors.

For clearer visualisation, we also plot the true function on the torus in two dimensions, as illustrated in Figure 6(b), where the coordinates correspond to the outer and inner circular angles of the Torus. We first applied the standard Euclidean $\mathbb{R}^3$ GP (following Rasmussen and Williams (2006)), utilising an RBF kernel based on $\mathbb{R}^3$ Euclidean distance. This method neglects the torus's intrinsic geometry. The predictive means at the unlabeled points are shown in the 2D Torus coordinates in Figure 6(d). In the center of Figure 6(d), the prediction appears predominantly blue, which is completely different from the true function, where the centre is dark red. A similar discrepancy is observed in the predictive mean of the GL-GP in Figure 6(e). Both the GL-GP and Euclidean GP fail to incorporate the manifold structure during regression. Conversely, the RC-AGP predictive mean, depicted in Figure 6(c), closely aligns with the true function. The color distribution in 6(c) is consistent with that in Figure 6(b). The S-AGP results are shown in Section H of the Appendix. It outperforms those of the GL-GP and Euclidean GP but does not match the accuracy of the RC-AGP. We further evaluated all methods across a range of signal-to-noise ratios(SNRs) by perturbing the true function with Gaussian noise at levels of 10 dB, 20 dB, and 30 dB, performing 10 replicates for each noise level. Each method was subsequently applied to estimate the test function at the grid points for each replicate. The mean and standard deviation of the root mean squared error(RMSE) across the replicates at different noise levels are summarised in Table 1. Values in brackets are the standard deviation. The RC-AGP is significantly better than all other methods under all conditions.



(a) True function in U-shape (b) RC-AGP prediction (c) Euclidean GP prediction (d) GL-GP prediction

Figure 7: Comparison of different methods on the U-shape data set: (a) true function in the U-shape (blue crosses denote the training observations); (b) RC-AGP prediction; (c) Euclidean GP prediction using the RBF kernel; (d) GL-GP prediction.

6.2 U-shape point cloud

We defined a U-shape point cloud as shown in Figure 7(a). A smooth regression function is applied to the points (Wood et al., 2008). The values (i.e., the color of the points) vary smoothly from the lower right-hand corner to the upper right-hand corner of the domain, ranging from -6 to 6. The blue crosses represent 30 observations randomly located within the domain of interest. The boundaries of the domain are unknown. The goal is to estimate the regression function and make predictions at equally spaced grid points within the domain. There are 355 grid points in the U-shaped domain. All grid points are divided into four subsets, as shown in Figure 18(a) in Appendix. Since the U-shaped point cloud is a subset of $\mathbb{R}^2$, each chart is represented as a domain of $\mathbb{R}^2$ that covers the subset. The BM within a chart becomes the standard BM in $\mathbb{R}^2$. The change of coordinates operation is also straightforward. The BM steps can be transferred to a different chart in the overlapping region of subsets.

In this study, we compared the RC-AGP, Euclidean $\mathbb{R}^2$ GP, GL-GP, and S-AGP at various levels of signal-to-noise ratio. The values of the true function were perturbed by Gaussian noise (with SNRs of 10db, 20db and 30db, respectively), with 10 replicates for each noise level. For each replicate, the different methods were applied to estimate the test function at the grid points. As a demonstration in the high SNR case, the predictive mean of the RC-AGP at the grid points is shown in Figure 7(b). The colored contours of the prediction closely resemble those of the true function in Figure 7(a). In contrast, the contours of the Euclidean $\mathbb{R}^2$GP predictive mean in Figure 7(c) smooth over the gap between the upper and lower arms of the point cloud, as these two arms are close in Euclidean distance. The $\mathbb{R}^2$GP using the RBF kernel assigns strong covariance between these two regions. However, the BM simulation starting from the lower arm needs to traverse different charts to reach the upper arm. This results in lower covariance between these two regions and more accurate predictions by the RC-AGP. This example was also used to evaluate the performance of GL-GP, as shown in Figure 7(d). The result of GL-GP is similar to the $\mathbb{R}^2$GP.

The mean and standard deviation of the root mean-squared error for the replicates at different noise levels are reported in Table 2. The RC-AGP is significantly better than all

SNR	*Euclidean* $\mathbb{R}^2$ *GP*	*GL-GP*	*RC-AGP*	*S-AGP*
30db	1.73(0.25)	1.58(0.20)	0.168(0.05)	0.282(0.02)
20db	1.73(0.26)	1.61(0.22)	0.265(0.05)	0.312(0.04)
10db	1.83(0.30)	1.78(0.47)	0.682(0.13)	0.630(0.13)

Table 2: Comparison of the RMSE of predictive means for various methods on U shape point cloud

other methods in almost all cases. When the SNR level is low (10db), the difference between RC-AGP and S-AGP is not significant. More results of S-AGP is shown in Section H of the Appendix. We also compared RC-AGP with a direct intrinsic AGP constructed by using the estimated heat kernel itself as the covariance function. For the U-shape data set, the intrinsic AGP required BM simulations from all 30 training locations, whereas RC-AGP simulated Brownian motion only from the four atlas representatives. Despite reducing the simulation cost by more than an order of magnitude, RC-AGP achieved a lower RMSE (0.16 versus 0.24), suggesting that the Riemannian-corrected construction provides a robust finite-sample approximation while substantially reducing computational cost.

6.3 Shark Prey LEGO Image Cloud

We analyze a data set comprising 576 images depicting a dynamic scenario involving a LEGO shark and its prey(Watkins, 2012). In this sequence, the small LEGO fish rotates around the shark while both entities simultaneously follow a circular trajectory within the background. The system's movement resembles a double rotation, analogous to the Earth's rotation and revolution. The regression function is defined as a function of the angles of revolution and rotation. The dimension of the original image space is $p = 900$. 12 image examples are given in Figure 8. Despite the high dimensionality, the movements of the LEGO shark prey occur on low-dimensional manifolds. Eight subsets are generated from the original set of 576 images. For each subset, we estimated a latent space with $q = 2$ dimensions using GPLVM. Appendix K provides ARD and cross-validation diagnostics supporting this two-dimensional choice. Since the true manifold geometry is unavailable for this image data set, Appendix L reports reconstruction-error diagnostics for the learned atlas. An example of the resulting latent space is shown in Figure 19 of the Appendix, where the dark regions indicate areas of high uncertainty. The atlas for the LEGO images is composed of these latent spaces.

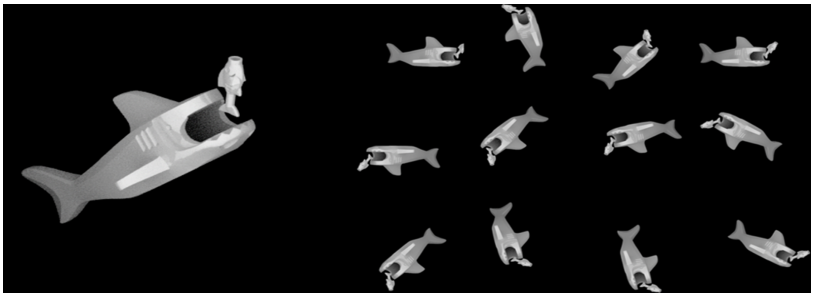
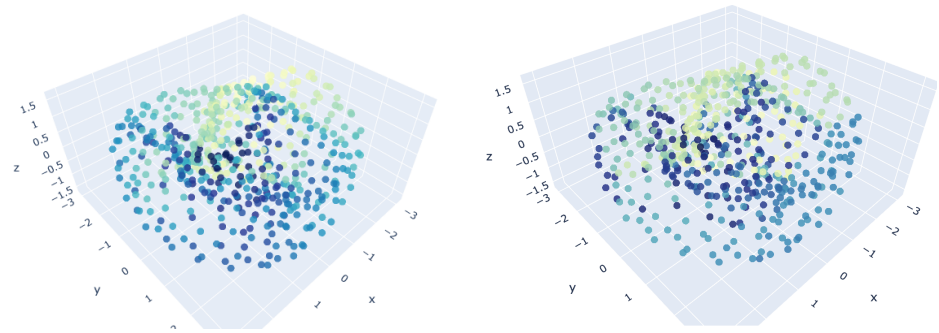


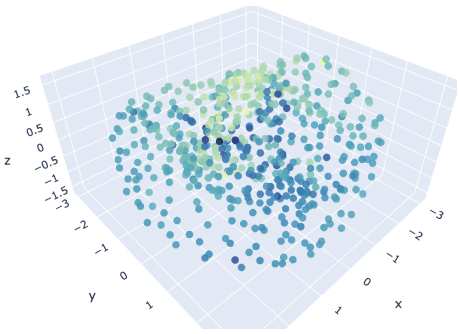
Figure 8: Shark Prey image examples.

In this study, a set of $n_d = 35$ images are randomly selected and used as the training set (labeled data). The remaining 541 images are used as the test set (unlabeled data). Different methods have been applied to estimate the regression function. Given the high dimensionality of the image space, visualising the 900-dimensional point cloud is impractical. To facilitate visualisation, we plotted all 576 images in a three-dimensional space using PCA, as shown in Figure 9(a). The color density represents the regression function values at each point. The predictive mean of the RC-AGP is illustrated in Figure 9(b), displaying a pattern closely aligned with the true function depicted in Figure 9(a). In contrast, the predictive means of the GL-GP, shown in Figure 9(d), reveal a notable discrepancy: the central region of the cloud is dominated by bright yellow and green hues, whereas the true function indicates dark blue in the same area. The predictive means of the $\mathbb{R}^{900}$ GP, utilising the RBF kernel with $\mathbb{R}^{900}$ Euclidean distance, are presented in Figure 9(c). The results display a color distribution that deviates from the true function. We also created ten different training and test sets through random selection. The mean and standard deviation of the root mean square errors calculated over the ten test sets are shown in Table 3. The RC-AGP achieves the smallest mean RMSE. It is significantly better than all other methods (Euclidean GP, GL-GP and S-AGP). The plot of S-AGP results is shown in Section H of the Appendix.

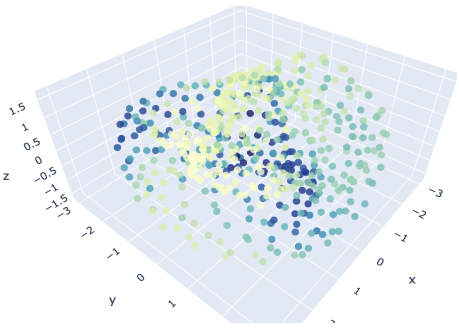


(a) True function on image cloud in 3D

(b) RC-AGP prediction



(c) Euclidean $\mathbb{R}^{900}$ GP prediction



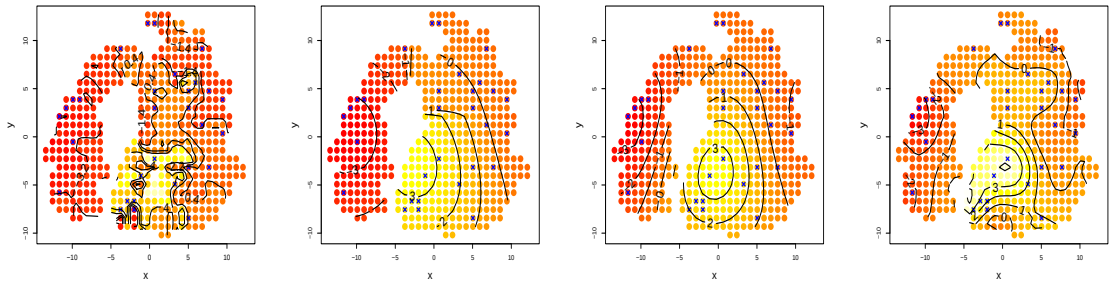
(d) GL-GP prediction

Figure 9: Comparison of different methods in Shark prey LEGO image cloud: (a) The true function values are depicted on the 3D points, with colors ranging from blue to green to represent varying magnitudes; (b) RC-AGP prediction; (c) Euclidean $\mathbb{R}^{900}$ GP prediction; (d) GL-GP prediction.

Euclidean $\mathbb{R}^{900}$ *GP*	*GL-GP*	*RC-AGP*	*S-AGP*
1.09(0.06)	0.98(0.17)	0.62 (0.04)	0.81(0.07)

Table 3: Comparison of the RMSE of predictive means for various methods on Shark image cloud

6.4 Aral Sea Point Cloud



(a) Truth in Aral sea (b) RC-AGP prediction (c) Euclidean GP prediction (d) GL-GP prediction

Figure 10: Comparison of different methods in Aral sea: (a) True function in Aral sea(blue crosses represent the training observations); (b) RC-AGP prediction at the grid points; (c) Euclidean GP prediction using RBF kernel; (d) GL-GP prediction.

The chlorophyll concentration at 485 locations in the Aral sea is plotted in Figure 10(a) (Wood, 2025). The concentration level is indicated by the intensity of the color. The remote sensing data is noisy and not as smooth as the U-shape example. Although the east and west part of the Aral sea is very close in the Euclidean distance, they are separated by the isthmus of the peninsula. The west part has a lower concentration level compared to the east part. The blue crosses represent 30 observations randomly located within the Aral sea. We applied different methods to estimate the spatial pattern of the chlorophyll density. The scaled chlorophyll concentration level is modeled as a function of the location (latitude and longitude).

All grid points are divided into three subsets, as shown in Figure 21(a) of the Appendix. The predictive mean of the RC-AGP at the grid points is displayed in Figure 10(b). The colored contours indicate that the prediction does not smooth across the isthmus of the peninsula. The prediction in the western part of the Aral Sea is lower and remains unaffected by the observations in the eastern part. The overall pattern closely resembles the data in Figure 10(a). In contrast, the contours of the Euclidean GP predictive mean in Figure 10(c) smooth over the gap. The Euclidean GP with an RBF kernel assigns a high covariance between these regions. A similar effect can be observed in the GL-GP prediction, as shown in Figure 10(d). We randomly create 10 batches of training data. Each batch contains 30 training points. The mean and standard deviation of the RMSE for the 10 replicates are listed in Table 4. The RC-AGP achieves the minimum RMSE. The difference between the Euclidean GP and GL-GP is not significant. We also considered a Euclidean GP with a Matérn-5/2 covariance function. The same training and test sets were used, and all

Euclidean $\mathbb{R}^2$ *GP*	*GL-GP*	*RC-AGP*	*S-AGP*
2.65(0.42)	2.49(0.32)	2.31 (0.23)	2.33(0.16)

Table 4: Comparison of the RMSE of predictive means for various methods on Aral sea point cloud

covariance hyperparameters were estimated by maximum likelihood. The results, reported in Appendix N.1, show that the Matérn covariance improves the Euclidean baseline in some examples, but RC-AGP remains more accurate in most cases.

7. Discussion

In this paper, we introduce the Riemannian corrected Atlas Gaussian processes, a novel framework for modeling data on unknown manifolds with nontrivial topologies. Our key contributions are twofold: (1) we developed an efficient method for heat kernel estimation on unknown manifolds by constructing a probabilistic atlas and simulating BM paths on the atlas; (2) we introduced Riemannian-corrected Atlas GPs, which leverage the estimated heat kernel combined with an RBF kernel to enhance computational efficiency and accuracy in manifold-based inference. Our method outperforms single-chart BM and graph Laplacian approximations, requiring fewer data points for accurate kernel estimates and maintaining robustness to variations in point cloud distributions. We also proved the semi-positive definiteness of the Riemannian-corrected kernel. In regression tasks on data sets such as U-Shape, Aral Sea, Torus, and high-dimensional image clouds, RC-AGPs consistently outperformed Euclidean GPs and GL-GPs, demonstrating their effectiveness. We further evaluate the Fast Graph Laplacian Gaussian Process (FLGP) proposed by He et al. (2026) on two representative datasets, the Torus and U-shape, with the results presented in Appendix N.2. RC-AGPs outperform FLGP in both cases. Future work could extend this framework to dynamic and time-varying manifolds and explore adaptive atlas construction to handle large-scale data, further broadening the applicability of RC-AGPs.

Appendix A. Proof of Theorem

A.1 Proof of Theorem 1 in the Main Paper

For each chart in the atlas, the BM can be defined as the SDE in Equation 5. Let $\mathbb{M}$ be a manifold of dimension q with a Riemannian metric $g_{\mathbb{M}}$. $\varphi_1 : \mathbb{U}_1 \to \mathbb{M}$ and $\varphi_2 : \mathbb{U}_2 \to \mathbb{M}$ are two local coordinate charts of $\mathbb{M}$ where $\mathbb{U}_1$ and $\mathbb{U}_2$ are open subsets of $\mathbb{R}^q$.

- $\psi = {\varphi_2}^{-1} \circ \varphi_1 : \mathbb{U}_1 \to \mathbb{U}_2$ is the change of coordinate. $\psi^{-1} = \varphi_1^{-1} \circ \varphi_2 : \mathbb{U}_2 \to \mathbb{U}_1$ is the inverse change of coordinate. (assume $\varphi_1(\mathbb{U}_1) = \varphi_2(\mathbb{U}_2)$ without loss of generality.)

- $\left(x^1, \ldots, x^q\right)$ denotes the standard coordinates in $\mathbb{U}_1 \in \mathbb{R}^q$. $\left(\bar{x}^1, \ldots, \bar{x}^q\right)$ denotes the standard coordinates in $\mathbb{U}_2 \in \mathbb{R}^q$.

- g_1 denotes matrix representation of $g_{\mathbb{M}}$ in $\mathbb{U}_1$ (via φ_1) i.e. $g_{1,ij} = g_{\mathbb{M}}\left(\varphi_{1,*}\frac{\partial}{\partial x^i}, \varphi_{1,*}\frac{\partial}{\partial x^j}\right)$. g_1^{ij} denotes the element of g_1^{-1}. $G_1 = det(g_1)$. Similarly for g_2 , $g_{2,ij}$, g_2^{ij}, G_2.

- $D\psi$ is the matrix derivative of $\psi = \left(\psi^1\left(x^1,\dots x^q\right),\dots,\psi^q\left(x^1,\dots,x^q\right)\right)$, i.e. $(D\psi)_i^j = \frac{\partial\psi^j}{\partial x^i}$. Using the chain rule $\Rightarrow D\psi^{-1}\cdot D\psi = I_q$.
- $\frac{\partial}{\partial x^i} = \sum_{j=1}^q \frac{\partial\psi^j}{\partial x^i}\frac{\partial}{\partial\bar{x}^j} = \sum_{j=1}^q (D\psi)_i^j\frac{\partial}{\partial\bar{x}^j} \quad \Rightarrow \quad g_{1,ij} = (D\psi)_i^k g_{2,k\ell}(D\psi)_j^\ell \quad \Rightarrow g_1^{-1} = \left(D\psi^{-1}\right)^\top g_2^{-1}D\psi^{-1}$, $G_1 = (\det\mathcal{D}\psi)^2 G_2$.

Proof Consider a stochastic process in $\mathbb{U}\in\mathbb{R}^q$ defined by:

$$dx^i = \frac{1}{2}\frac{1}{\sqrt{G_1}}\sum_{j=1}^q\frac{\partial}{\partial x^j}\left(\sqrt{G_1}g_1^{ij}\right)dt + \left(g_1^{-\frac{1}{2}}dB\right)^i, \quad i=1,\dots,q. \tag{11}$$

$$dx^i dx^j = \left(g_1^{-\frac{1}{2}}dB\right)^i\left(g_1^{-\frac{1}{2}}dB\right)^j = \left(\sum_{k=1}^q\left(g_1^{-\frac{1}{2}}\right)^{ik}dB^k\right)\cdot\left(\sum_{l=1}^q\left(g_1^{-\frac{1}{2}}\right)^{jl}dB^l\right) = g_1^{ij}dt$$

ψ maps the above process in $\mathbb{U}_1$ to a process in $\mathbb{U}_2$ defined by:

$$\begin{aligned} d\bar{x}^i = d\psi^i\left(x^1,\dots,x^q\right) &= \sum_{j=1}^q\frac{\partial\psi^i}{\partial x^j}dx^j + \frac{1}{2}\sum_{j=1}^q\sum_{k=1}^q\frac{\partial^2\psi^i}{\partial x^j\partial x^k}dx^j dx^k \\ &= \frac{1}{2}\sum_{j=1}^q\sum_{k=1}^q\left[(D\psi)_j^i\cdot\frac{1}{\sqrt{G_1}}\frac{\partial}{\partial x^k}\left(\sqrt{G_1}g_1^{jk}\right) + \frac{\partial^2\psi^i}{\partial x^j\partial x^k}g_1^{jk}\right]dt + \sum_{j=1}^q(D\psi)_j^i\left(g_1^{-\frac{1}{2}}dB\right)^j. \end{aligned} \tag{12}$$

The dt term in Equation 12 can be written as follows

$$\begin{aligned} &\frac{1}{2}\sum_{j=1}^q\sum_{k=1}^q\left[\frac{(D\psi)_j^i}{detD\psi}\cdot\frac{1}{\sqrt{G_2}}\cdot\sum_{\ell=1}^q(D\psi)_k^l\cdot\frac{\partial}{\partial\bar{x}^\ell}\left(detD\psi\cdot\sqrt{G_2}\sum_{r=1}^q\sum_{s=1}^q\left(D\psi^{-1}\right)_r^j g_2{}^{rs}(D\psi^{-1})_s^k\right)\right. \\ &\left.+\left(\sum_{l=1}^q(D\psi)_j^l\frac{\partial}{\partial\bar{x}^\ell}(D\psi)_k\right)\left(\cdot\sum_{r=1}^q\sum_{s=1}^q\left(D\psi^{-1}\right)_r^j g_2^{rs}\left(D\psi^{-1}\right)_s^k\right)\right]dt \\ =&\frac{1}{2}\sum_{l=1}^q\frac{1}{\sqrt{G_2}}\frac{\partial}{\partial\bar{x}^\ell}\left(\sqrt{G_2}g_2^{il}\right)dt + \frac{1}{2}\sum_{s=1}^n\mathrm{Tr}\left[D\psi\cdot\frac{\partial}{\partial\bar{x}^s}\left(D\psi^{-1}\right)\right]\cdot g_2^{is}dt \\ &+\frac{1}{2}\sum_{j=1}^q\sum_{l=1}^q\sum_{r=1}^q\frac{\partial}{\partial\bar{x}^\ell}\left[(D\psi)_j^i\left(D\psi^{-1}\right)_r^j\right]\cdot g_2^{rl}dt + \frac{1}{2}\sum_{l=1}^n\mathrm{Tr}\left[\frac{\partial}{\partial\bar{x}^\ell}(D\psi)\cdot\left(D\psi^{-1}\right)\right]g_2^{i\ell}dt \\ =&\frac{1}{2}\sum_{l=1}^q\frac{1}{\sqrt{G_2}}\frac{\partial}{\partial\bar{x}^\ell}\left(\sqrt{G_2}g_2^{il}\right)dt + \frac{1}{2}\sum_{s=1}^q\mathrm{Tr}\left[\frac{\partial}{\partial\bar{x}^s}\left(D\psi\cdot D\psi^{-1}\right)\right]\cdot g_2^{is}dt + 0 \\ =&\frac{1}{2}\sum_{\ell=1}^q\frac{1}{\sqrt{G_2}}\frac{\partial}{\partial\bar{x}^\ell}\left(\sqrt{G_2}g_2^{i\ell}\right)dt + 0. \end{aligned} \tag{13}$$

The dB term in Equation 12 can be written in vector form as $(D\psi)^\top\cdot g_1^{-\frac{1}{2}}\cdot dB$. Since $(D\psi)^\top\cdot g_1^{-\frac{1}{2}}\cdot dB \quad \left((D\psi)^\top g_1^{-\frac{1}{2}}dB\right)^\top = g_2^{-1}dt$, we have $(D\psi)^\top\cdot g_1^{-\frac{1}{2}}\cdot dB\sim\mathcal{N}\left(0,g_2^{-1}dt\right)$, i.e. the same distribution as $g_2^{-1/2}dB$.

Conclusion: ψ maps the process in $\mathbb{U}_1 \subset \mathbb{R}^n$ defined by

$$dx^i = \frac{1}{2}\frac{1}{\sqrt{G_1}}\sum_{j=1}^{q}\frac{\partial}{\partial x^j}\left(\sqrt{G_1}g_1^{ij}\right)dt + \left(g_1^{-\frac{1}{2}}dB\right)^i, \quad i = 1, \ldots, q, \tag{14}$$

to the process in $\mathbb{U}_2 \subset \mathbb{R}^n$ defined by

$$d\bar{x}^i = \frac{1}{2}\frac{1}{\sqrt{G_2}}\sum_{j=1}^{q}\frac{\partial}{\partial \bar{x}^j}\left(\sqrt{G_2}g_2^{ij}\right)dt + \left(g_2^{-\frac{1}{2}}dB\right)^i, \quad i = 1, \ldots, q. \tag{15}$$

This verifies that the process defined by the above formula is coordinate independent. As a result, with a given dt, simulating one step using any choice of local coordinates as above is equivalent as a step in $\mathbb{M}$. ■

A.2 Proof of Theorem 2 and Corollary 3 in the Main Paper

Theorem. *$\hat{h}^t$ in Equation 8 of the main paper is asymptotically unbiased and consistent.*

Proof Let $\mathbb{M}$ be a manifold parameterized by an atlas $\mathcal{A}$ consisting of n_v charts. Simulating Brownian Motion $\{S(t) : t > 0\}$ on $\mathbb{M}$ is equivalent to simulate a stochastic process governed by the Stochastic Differential Equations(SDEs) on the atlas $\mathcal{A}$. For each chart, the SDEs are derived as follows:

$$dx_k^i(t) = \frac{1}{2}{G_k}^{-1/2}\sum_{j=1}^{q}\frac{\partial}{\partial x_k^j}\left({\mathcal{G}_k}^{ij}{G_k}^{1/2}\right)dt + \left({\mathcal{G}_k}^{-1/2}dB_k(t)\right)_i \tag{16}$$

where $\mathcal{G}_k$ is the metric of chart k and G_k is the determinant of $\mathcal{G}_k$. x_k^i is the coordinate in the i_{th} dimension of chart k, and $B_k(t)$ is an independent BM in Euclidean space. Transition between different charts in the atlas is handled via the change of coordinates operations. Consider a Brownian motion $\{S(t) : t > 0\}$ on $\mathbb{M}$ with $S(0) = s_0$. For any $t > 0, s \in \mathbb{M}$, the true BM transition probability evaluated at $S(t)$ reaching a small neighborhood set $\mathbb{A}$ is given by: $p(S(t) \in \mathbb{A}|S(0) = s_0) = \int_{\mathbb{A}} h^t(s_0, s)ds$, where $h^t(s_0, s)$ is the heat kernel on $\mathbb{M}$. Given a point s with local coordinates $(r_1, \cdots, r_p)$, a window size w as the measurement of the volume for neighborhood set $\mathbb{A}$, and assuming $r_1(s) = \cdots = r_p(s) = 0$ for simplifying the calculation, the heat kernel can be approximated by:

$$h^{t'} = \frac{1}{Vol(\mathbb{A})}p[|r_i(S(t))| < w \quad \text{for } i = 1, \cdots, p] \tag{17}$$

where $Vol(\mathbb{A}) = \{|r_i| < w, \quad i = 1, \cdots, q\}$ is the Riemannian volume of $\mathbb{A}$. By applying the Taylor expansion at point s, the kernel estimator will be

$$h^{t'} = h^t + \mathcal{O}(w^2) \tag{18}$$

where the order of the magnitude of $h^{t'} - h^t$ is $\mathcal{O}(w^2)$. Increasing w thus increases the error term. By counting how many of these paths reach $\mathbb{A}$ at time t the transition probability can be evaluated. The transition density $h^{t'}$ of BM at s is approximated by $\hat{h}^t$:

$$\hat{h}^t = \frac{1}{Vol(\mathbb{A})} \cdot \frac{N_A}{N}, \quad \text{where } N_A \sim \text{Bin}(N, Vol(\mathbb{A})h^{t'}). \tag{19}$$

N_A is the number of sample paths with $\|S(t) - s\| < w$ (the number of paths reach the neighbourhood at time t). And follows a binomial distribution with N trails (the total number of simulated BM paths) and probability of success of $Vol(\mathbb{A})h^{t'}$. The expectation of $\hat{h}^t$ is:

$$E[\hat{h}^t] = E[\frac{1}{Vol(\mathbb{A})} \cdot \frac{N_A}{N}] = \frac{1}{Vol(\mathbb{A})} \cdot \frac{E[N_A]}{N} \tag{20}$$

Since N_A is binomially distributed, $E(N_A) = N \cdot Vol(\mathbb{A})h^{t'}$, which gives:

$$E[\hat{h}^t] = \frac{1}{Vol(\mathbb{A})} \cdot \frac{N \cdot Vol(\mathbb{A})h^{t'}}{N} = h^{t'} = h^t + \mathcal{O}(w^2) \tag{21}$$

Besides, the standard error of $\hat{h}^t$ is:

$$sd(\hat{h}^t) = \sqrt{(\frac{1}{Vol(\mathbb{A})N})^2 \cdot Var(N_A)} = \frac{1}{Vol(\mathbb{A})N} \cdot \sqrt{N \cdot Vol(\mathbb{A})h^{t'}(1 - Vol(\mathbb{A})h^{t'})} \tag{22}$$

$$\leq \sqrt{\frac{h^{t'}}{N \cdot Vol(\mathbb{A})}} = \mathcal{O}(N^{-1/2}w^{-d/2}) \tag{23}$$

The standard deviation decreases with w and N. As $w^2 \to 0$, we obtain $E(\hat{h}^t(s_0, s)) = h^t(s_0, s)$, and as $\frac{w^{-d}}{N} \to 0$, we have $\mathrm{Var}(\hat{h}^t(s_0, s)) \to 0$. Therefore, the estimator $\hat{h}^t(s_0, s)$ is asymptotically unbiased and consistent. ■

Corollary. Finite sample error bound: $\hat{h}_t(s_0, s) - h_t(s_0, s) = O_p(N^{-1/2}w^{d/2}) + O(w^2)$.
Proof From Equation 22, $sd(\widehat{h}_t) = O(N^{-1/2}w^{-d/2})$, which implies

$$\widehat{h}_t - E(\widehat{h}_t) = O_p(N^{-1/2}w^{-d/2}).$$

By Equation 20, $E(\widehat{h}_t) - h_t = O(w^2)$. Therefore,

$$\widehat{h}_t - h_t = (\widehat{h}_t - E(\widehat{h}_t)) + (E(\widehat{h}_t) - h_t) = O_p(N^{-1/2}w^{-d/2}) + O(w^2).$$

The argument uses the standard binomial variance calculation for the simulated transition probability; the stochastic order follows from Chebyshev's inequality applied to this variance bound (Van der Vaart, 2000) ■

A.3 Proof of Theorem 4 in the Main Paper

Proof Let $K_h = \big(h_t(s_i, s_j)\big)_{i,j=1}^{n_v}$ denote the population heat-kernel matrix evaluated at the chart representatives. Since the heat kernel $h_t(\cdot, \cdot)$ is a positive-semidefinite kernel on the manifold $\mathbb{M}$, the matrix K_h is positive semidefinite. The expanded heat-kernel matrix $\widetilde{K}_h$ is obtained by replacing each entry of K_h with a block of ones of the appropriate dimension. Equivalently, $\widetilde{K}_h = K_h \otimes B$, where B is a block matrix of ones and $\otimes$ denotes the Kronecker product. Since a matrix of ones is positive semidefinite, B is positive semidefinite. Therefore, by the closure property of the Kronecker product, $\widetilde{K}_h$ is postive semidefinite.

Furthermore, K_{rbf} is constructed from the RBF kernel and is therefore positive semidefinite. The population Riemannian-corrected kernel matrix is given by $K_{RC} = K_{rbf} \odot \widetilde{K}_h$, where $\odot$ denotes the Hadamard product. Since both K_{rbf} and $\widetilde{K}_h$ are positive semidefinite, the Schur product theorem implies that their Hadamard product is also positive semidefinite. Therefore, the population Riemannian-corrected kernel matrix is positive semidefinite. ■

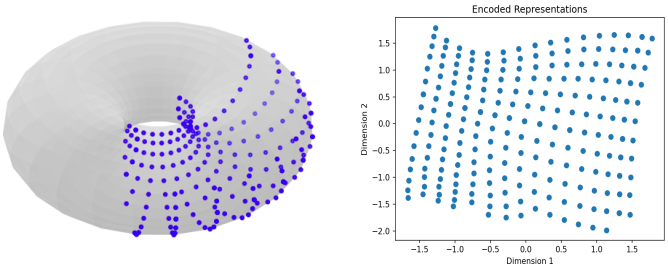
Appendix B. Autoencoder Latent Space of Torus.

The autoencoder (AE, Baldi 2012) can also be used to construct the latent space and learn the mapping function. It has been widely used as a neural network-based dimensionality reduction method. The encoder of the AE projects the original input into a lower-dimensional latent space, and the decoder of the AE maps the points in the latent space back to the original space. In fact, GPLVM can also be thought of as a GP decoder (Moreno-Muñoz et al., 2022). Unlike GPLVM, which learn the mapping as a probabilistic function, the AE-learned mapping function, which is a neural network, is deterministic. For each subset $\mathcal{S}_i = \{s_{ij} | j = 1, \cdots, ns_i\}$, we use AE to learn the mapping φ_i and construct latent space $\mathcal{X}_i$, where $s_{ij} = \varphi_i(x_{ij})$.

An autoencoder (AE) is a neural network composed of nodes (or artificial neurons) organized in multiple layers to match the input data. The aim of Autoencoders is to reconstruct the output as similar as possible to the original input. To achieve this, a reconstruction error (or loss function) is introduced, which measures the discrepancy between the input S_i and its reconstructions (the output) $\hat{S}_i$. It is important to note that separate AEs are modeled for each subset S_i. The goal of training an AE is to minimize this reconstruction error. In this context, we use mean squared error (MSE) as the reconstruction error. $L(S_i, \hat{S}_i) = \frac{1}{ns_i} \sum_{j=1}^{ns_i} L(s_{ij}, \hat{s}_{ij})$ where ns_i is the number of points in the ith subset S_i and s_{ij} is the jth observation of ith subset.

Once an AE model is constructed, it allows for straightforward transformation of points between the latent space and the original space. We denote an encoder as φ_i^{-1} that is a feature-extracting function in a pre-determined parametric closed form(e.g., Rectified Linear Activation Function). The latent variable are estimated as $X_i = \varphi_i^{-1}(S_i)$. In this context, the encoder can also be considered as the backward mapping. The Decoder is the forward mapping represented as φ_i, which maps from the latent space back to the original input space, generating a reconstruction $\hat{S}_i = \varphi_i(X_i)$. Figure 11(a) shows one subset of the Torus point cloud, and Figure 11(b) is the corresponding latent space generated by the AE decoder.

As the Riemannian metric defined in the main parper, the metirc tensor $\mathbf{g}_i$ for each subset S_i are $\mathbf{g}_i = \mathcal{J}_i^T \mathcal{J}_i$ where the $\mathcal{J}_i$(the Jacobian matrix of subset S_i) can be regarded as the partial derivative $\frac{\partial \varphi_i(x_*)}{\partial x}$. It can be calculated in AE decoder model by applying the chain rule of the partial derivative of the activate function and the inputs in each layers (Rifai et al., 2011).



(a) A subset of torus (b) AE Latent space

Figure 11: A subset latent spaces leaned from autoencoder of torus cloud.

Appendix C. Gaussian Process Latent Variable Model

The Gaussian Process Latent Variable Model (GPLVM, Lawrence 2005)is a probabilistic latent variable model. The probabilistic mapping links the latent variable in latent space $x \in \mathbb{R}^q$ and its corresponding observations in the original dimension $s \in \mathbb{R}^p$. For each subset $\mathcal{S}_i$, independent mappings are defined. We have $s_{ij}^l = \varphi_i^l(x_{ij}) + \epsilon_{ij}^l$ where s_{ij}^l is ith chart, lth dimension of jth observations and ϵ_{ij}^l is a i.i.d noise which is typically taken to be Gaussian $p(\epsilon) = \mathcal{N}(0, \beta^2)$. GPLVMs place a prior on the mapping, $\varphi_i^l(x_{ij}) \sim \mathcal{N}(0, \mathcal{K}_{xx})$ where $\mathcal{K}$ is a RBF kernel. Let X_i denotes the set of all latent variables x_{ij} correspondent to $\mathcal{S}_i$. With the assumption that the prior is independent across the dimension in the orginal data space, the marginal likelihood can be written as:

$$p(S_i|X_i) = \int \prod_{l=1}^{p} p(S_i|\varphi_i^l, \beta) p(\varphi_i^l|X_i) d\varphi_i = \prod_{l=1}^{p} \mathcal{N}(s_{i:}^l, \mathcal{K}_{XX} + \beta^2 I) \tag{24}$$

where $\mathcal{K}_{XX}$ is the covariance matrix generated by the RBF kernel. $s_{i:}^l$ represents the lth dimension of all points in $\mathcal{S}_i$. The latent variables and the kernel hyper-parameters can be estimated by maximum likelihood estimates. $X_i = \mathbf{arg}\max_{X_i} p(S_i|X_i)$. When simulating BM in latent space, we can observe a new location in each time step as testing data denoted as x_*. $\varphi_i(x_*)$ is the projection of x_* in the original high dimensional space. The joint distribution of $\varphi_i(X_i)$ and $\varphi_i(x_*)$ is:

$$\begin{bmatrix} \varphi_i(X_i) \\ \varphi_i(x_*) \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \mathcal{K}_{XX} & \mathcal{K}_{Xx^*} \\ \mathcal{K}_{x^*X} & \mathcal{K}_{x^*x^*} \end{bmatrix}) \tag{25}$$

where $\mathcal{K}_{Xx^*}$ is the covariance matrix between given points and new points. The predictive distribution is

$$p(\varphi_i(x_*)|S_i) \sim \mathcal{N}(\mathcal{K}_{x^*X}(\mathcal{K}_{XX} + \beta^2 I_n)^{-1} S_i,\ \mathcal{K}_{x^*x^*} - \mathcal{K}_{x^*X}(\mathcal{K}_{XX} + \beta^2 I_n)^{-1} \mathcal{K}_{x^*X}^T) \tag{26}$$

In addition to GPLVM, a variational method, the Bayesian Gaussian Process Latent Variable Model (Bayesian GPLVM), can also be applied. A key advantage of Bayesian GPLVM is its incorporation of uncertainty in the latent variables, which allows for a distribution over the latent space rather than a fixed point estimate (Titsias, 2009). This leads

to more robust handling of small data sets and improved generalization by accounting for uncertainty in the latent space. However, the increased number of hyperparameters introduces additional complexity to the optimization process, which can lead to convergence difficulties in some cases. In the following section, we will focus on GPLVM.

C.1 GPLVM Mappings

Since we know how to infer latent variables from the original manifold and generate new points in high dimensions from latent space points, we can construct both forward and backward mappings between the latent space and the original manifold.

Denote x_* as a point in the latent space, by applying the predictive distribution (forward mapping), the corresponding point s^* in the manifold can be generated.

$$s^*|\mathcal{S}_i \sim \mathcal{N}(\mathcal{K}_{x^*X}(\mathcal{K}_{XX}+\beta^2 I_n)^{-1}S_i,\ \mathcal{K}_{x^*x^*}-\mathcal{K}_{x^*X}(\mathcal{K}_{XX}+\beta^2 I_n)^{-1}\mathcal{K}_{x^*X}^T) \tag{27}$$

On the contrary, when a point s^* on the manifold is given in the high dimensional space, we can estimate the latent variables in a chart (x_* in the latent space) by maximising the likelihood(backword mapping).

$$x_* = \mathbf{arg}\max_{x_*} p(s_*, S_i|X_i, x_*)$$

C.2 GPLVM Metric

The Jacobian of ith chart, $\mathcal{J}_i$ can be computed as the partial derivative $\frac{\partial\varphi_i(x_*)}{\partial x^l}$ with respect to the lth dimension for any point x_* in the latent space.

$$\mathcal{J}_i = \frac{\partial\varphi_i(x_*)}{\partial x} = \begin{bmatrix} \frac{\partial\varphi_i^1(x_*)}{\partial x^1} & \cdots & \frac{\partial\varphi_i^1(x_*)}{\partial x^k} & \cdots & \frac{\partial\varphi_i^1(x_*)}{\partial x^q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial\varphi_i^l(x_*)}{\partial x^1} & \cdots & \frac{\partial\varphi_i^l(x_*)}{\partial x^k} & \cdots & \frac{\partial\varphi_i^l(x_*)}{\partial x^q} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial\varphi_i^p(x_*)}{\partial x^1} & \cdots & \frac{\partial\varphi_i^p(x_*)}{\partial x^k} & \cdots & \frac{\partial\varphi_i^p(x_*)}{\partial x^q} \end{bmatrix} \tag{28}$$

where the Jacobian is a $p \times q$ matrix. Recall that p represents the dimension of the original space and q denotes the dimension of the latent space. Since the RBF Kernel is differentiable, the derivative of a GP is also a GP. Therefore the distribution of Jacobian for chart i is $p(\mathcal{J}_i|X_i, S_i) = \prod_{l=1}^p \mathcal{N}(\mu^l_{\mathcal{J}_i}, \Sigma_{\mathcal{J}_i})$ The expression for $\mu^l_{J_i}, \Sigma_{J_i}$(the mean and covariance for the Jacobian distribution) are shown later. The expected metric tensor $\mathcal{G}_i$ for ith chart is defined as: $\mathcal{G}_i = \mathbb{E}(\mathcal{J}_i^T)\mathbb{E}(\mathcal{J}_i) + p\Sigma_{\mathcal{J}_i}$

The joint distribution of lth dimension of the mapping φ_i and the corresponding column of the Jacobian are

$$\mathcal{J}_i^{l,k} = \frac{\partial\varphi_i^l}{\partial x^k}, \quad \begin{bmatrix} \varphi_i^l(x) \\ \frac{\partial\varphi_i^l(x_*)}{\partial x_*} \end{bmatrix} \sim \mathcal{N}\left(0 \quad \begin{bmatrix} \mathcal{K}_{xx} & \partial\mathcal{K}_{x*} \\ \partial\mathcal{K}_{x*}^T & \partial^2\mathcal{K}_{**} \end{bmatrix}\right) \tag{29}$$

where the superscript indicates the lth dimension of the ith chart and kth dimension of the observation space. The covariance matrix is computed as $\mathcal{K}_{xx_*} = \gamma\mathbf{exp}(-\rho\|x_{ij} - x_*\|^2)$

where γ and ρ regarded as hyperparameters are length scale and variance for the RBF kernel. The derivatives of $\mathcal{K}$ are:

$$(\partial \mathcal{K}_{x*})_j^l = \frac{\partial \mathcal{K}_{xx_*}}{\partial x_*^l} = 2\rho(x_{ij}^l - x_*^l)\mathcal{K}_{xx_*} \quad j = 1, \cdots, n, \quad l = 1, \cdots, q$$

$$(\partial^2 \mathcal{K}_{**})^{r,l} = \frac{\partial^2 \mathcal{K}_{x'_* x_*}}{\partial x_*^r \partial x_*^l} = \begin{cases} -4\rho^2(x'^r_* - x_*^r)(x'^l_* - x_*^l)\mathcal{K}_{x'_* x_*} & r \neq l \\ 2\rho(1 - 2\rho(x'^r_* - x_*^r)^2)\mathcal{K}_{x'_* x_*} & r = l \end{cases}$$

$$(\partial^2 \mathcal{K}_{**})^{r,l} = \frac{\partial^2 \mathcal{K}_{x_* x_*}}{\partial x_*^r \partial x_*^l} = \begin{cases} 0 & r \neq l \\ 2\rho \mathcal{K}_{x_* x_*} = 2\rho\gamma & r = l \end{cases} \text{ When } x_* = x'_*$$

The conditional probability over the Jacobian also follows a Gaussian distribution. For any point x_* in the latent space, the distribution of the Jacobian can be expressed as

$$p(\mathcal{J}_i | S_i, \varphi_i) = \prod_{l=1}^{p} \mathcal{N}(\partial \mathcal{K}_{x*}^T \mathcal{K}_{xx}^{-1} s_{i,:}^l, \partial^2 \mathcal{K}_{**} - \partial \mathcal{K}_{x*}^T \mathcal{K}_{xx}^{-1} \partial \mathcal{K}_{x*}) \tag{30}$$

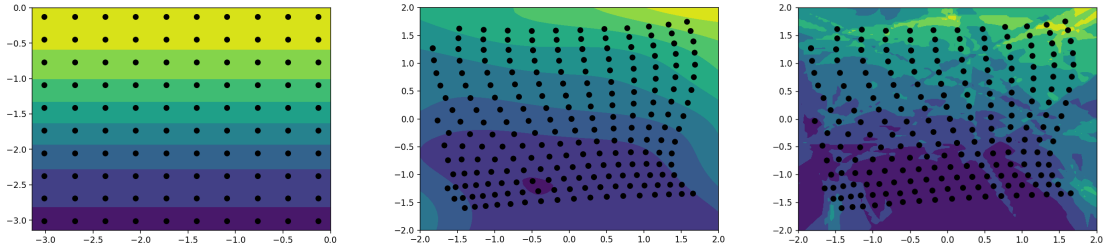
C.3 Magnification Factor

The magnification factor(MF) refers to the square root of the determinant of the Riemannian metric (Bishop et al., 1997). $\mathcal{MF} = \sqrt{det(\mathcal{G})}$ The magnification factor can express how much a part of latent space in $\mathbb{R}^q$ is stretched or bend when mapped to $\mathbb{M}$. In other words, it quantifies how distances in the latent space X_i are transformed when mapped to the high-dimensional observed space S_i. A higher magnification factor indicates more stretching in the latent space. For example, Figure 12 illustrates a latent space generated from S_i(Torus subset) using different methods. The background color of Figure 12 represents the magnification factor, where darker color indicate larger magnification. In Figure 12(a), the vertical axis corresponds to the angle of the inner circle, while the horizontal axis represents the angle of the outer circle. As the inner circle angle approaches $-\pi$, corresponding to the inner hole of the torus, the latent space becomes increasingly stretched when mapped back to the original manifold. The magnification factor computed from the analytical mapping function is plotted in Figure 12(a). In Figure 12(b), where the mapping function is generated by GPLVM, the magnification factor appears in a similar pattern, but the contours are curved. The the magnification factor generated from AE mapping is plotted in Figure 12(c). The density of the MF varies less smoothly, and the overall pattern deviates from the analytical result.

Appendix D. Kernel Estimates on Scatter Torus.

We tested the robustness of our Atlas Brownian Motion (BM) framework by varying the configuration of the torus, specifically the radius of the inner circle ϕ and the outer circle θ, and the distribution of points on the torus. A total of 625 points were sampled from a torus with an outer circle radius of $R = 2$ and an inner circle radius of $r = 1$, forming a point cloud with a non-uniform and irregular distribution, as illustrated in Figure 13(a).

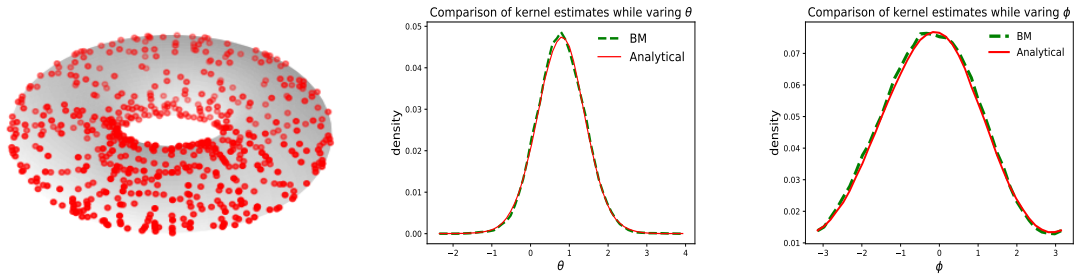
In this experiment, we implemented the Atlas Brownian Motion (BM) framework outlined in the main paper. The probabilistic atlas was constructed from the non-uniform



(a) MF with true mapping (b) GPLVM MF (c) AE MF

Figure 12: Plots of the magnification factor in one of the latent spaces of the torus, constructed using different methods: (a) MF with true mapping; The background color corresponds to the scale of MF. (b)MF with GPLVM mapping. (c) MF with Autoencoder mapping.

torus point cloud using latent variable models. Kernel density estimation was then performed based on BM simulations across the atlas.



(a) Non-uniform and irregular distributed Torus point cloud (b) Comparison of kernel density while varying θ (c) Comparison of kernel density while varying ϕ

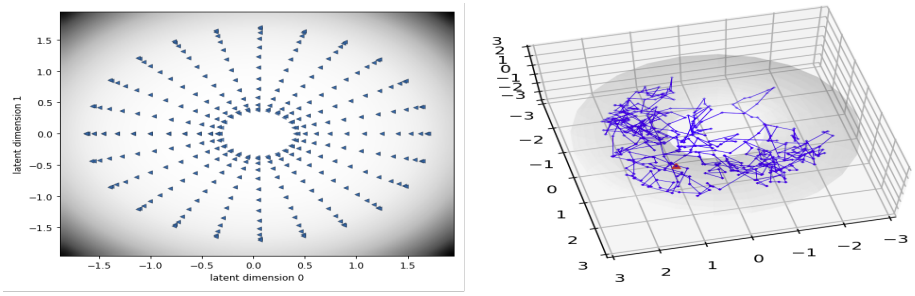
Figure 13: Kernel estimates of scatter torus. The horizontal axes represent the outer circle angle θ or inner circle angle ϕ, while the vertical axis shows the transition density. The truth is presented as the red line and the green dashed line represents the kernel estimates generated by atlas BM approach.

Figure 13(b) and 13(c) show the inner and outer circle angles of the scatter torus on the horizontal axes, while the vertical axis represents the transition density. The kernel estimates derived from the Atlas BM approach (the green dashed line) are closely aligned with the true values (the red solid line). This demonstrates the robustness and accuracy of the Atlas BM framework in estimating kernel densities, even with complex, irregular point clouds.

Appendix E. Problematic Single Chart for Torus Point Cloud.

A manifold cannot be covered by a single chart, unless it is homeomorphic to an open subset of an Euclidean space (Lee, 2013, Section 1). In this section we will show the problem of using a single chart to parameterise a torus point cloud which have non-trivial topology. As shown in Figure 14(a), we created a latent space for the torus point cloud using GPLVM.

The distribution of latent variables exhibits a pattern resembling a sunflower. This suggests that the latent space is generated by compressing the torus, leading to points stacking on top of each other.



(a) Single chart latent space

(b) One sample path on $\mathbb{M}$ generated by the single chart

Figure 14: (a)The latent space of the torus cloud generated by the single-chart approach; (b) The top view of the sample path generated by the single-chart approach.

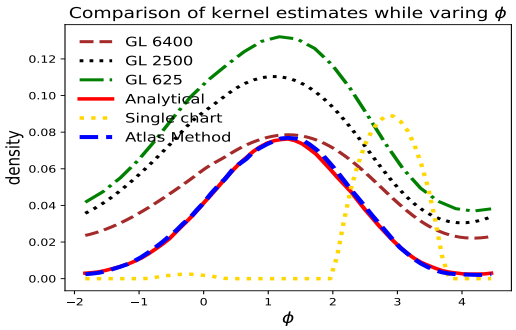


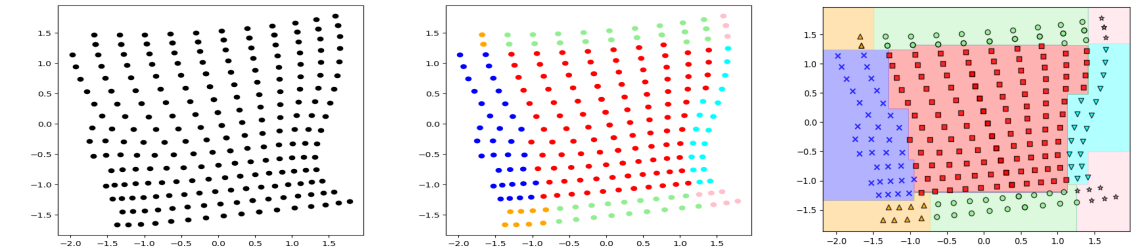
Figure 15: Kernel estimates comparison for different methods. The analytical kernel density is represented by the solid red line, while the kernel density generated from a single chart is depicted by the yellow dashed line. The kernel estimates obtained using the BM on Atlas approach are illustrated as a dashed blue line, with the atlas constructed from 625 points on the torus. The GL kernel estimates are plotted as a green dash-dotted line when the number of points is 625. Additionally, the GL estimates are represented as a black dotted line and a brown dashed line for increased point counts of 2,500 and 6,400, respectively.

A stochastic process has been simulated in the latent space and projected back to the observational space, illustrated in Figure 14(b). It is clear that the simulated path travels inside the Torus, treating the manifold as a disk. It indicates the stochastic process is not a valid BM on $\mathbb{M}$. Figure 15 presents the kernel density estimates from different methods. The horizontal axis represents the inner circle angle ϕ of the torus, while the vertical axis represents the transition density. The yellow dashed line represents the kernel estimates from the single-chart BM approach, which deviate notably from the analytical kernel, with a skewed peak compared to other estimates. The kernel density estimates using the Atlas BM method introduced in the main paper, is depicted as the blue dashed line in Figure

15. It closely aligns with the analytical kernel(the red solid line). Additionally, the kernel can also be estimated using the Graph Laplacian (GL) method. By adjusting the number of grid points, the GL kernel estimates shift from the green dash-dotted line to the black dotted and brown dashed lines, respectively. These results highlight the limitations of the single-chart approach in capturing the manifold's geometry, leading to unreliable kernel estimates.

Appendix F. Define the Overlapping Region in the Latent Space of Torus Subsets

Given the torus subsets generated from the Section 3 in main paper, we can identify overlapping points among different subsets. For example, points that belong to both subsets S_1 and S_2 will be classified as Category 12. In contrast, points exclusive to a single chart(either S_1 or S_2) will be labeled as either Category 1 or 2, depending on the respective chart. This labeling process can be extended to assign labels to points across all charts within the atlas. Once the shared points between different charts are identified, classifiers such as decision trees (Hastie et al., 2009), can be used to define overlapping regions in the latent spaces, facilitating the switching of SDEs between different charts.



(a) One of the latent space of Torus (b) Points labeled by categories (c) Points with labels and overlapping regions

Figure 16: Identify overlapping regions in the latent space: (a) Visualization of a latent space of Torus. (b) Data points labeled by colors, where each color denotes a category. (c) Overlapping regions in the latent space: The regions highlighted in yellow, pink, blue, green, and cyan represent areas that overlap with other subsets in the latent space. The red region does not overlap with any other subsets.

As shown in Figure 16, the decision tree method is applied to identify the overlapping regions in the latent space. Figure 16(a) illustrates one of the latent spaces of Torus, while Figure 16(b) presents the points labelled with six different colors, each representing a different category. The red points correspond to points belonging exclusively to a single chart, whereas the remaining colors (green, yellow, cyan, blue, and pink) indicate points shared across at least two charts. The decision tree method is applied and identifies the overlapping region in the latent space in Figure 16(c). The background colors represent the regions associated with each category. When simulating SDE steps in the latent space, if the simulated point enters an overlapping region, such as the left middle area (blue area), the point can be transferred to another latent space via the corresponding mapping function.

Appendix G. Sparse Atlas GP

As a comparison to the RC-AGP, the Sparse Atlas GP (S-AGP) are developed by combining the Atlas BM framework with sparse GP inference. In this approach, we first introduce a set of inducing points on $\mathbb{M}$, $\boldsymbol{s_z} = [s_{z1}, ..., s_{zm}]$, $s_{zi} \in \mathbb{M}$. zm is the number of inducing points. The realisation of the regression function on these inducing points can be represented as zm random variables $\boldsymbol{u} = [f(s_{z1}), ..., f(s_{zm})]$. The distribution of $\boldsymbol{u}$ is a multivariate Gaussian, $\boldsymbol{u} \sim \mathcal{N}(0, \Sigma_{\mathbf{uu}})$, with mean zero and covariance matrix $\Sigma_{\mathbf{uu}}$. Each element of $\Sigma_{\mathbf{uu}}$ can be computed as the heat kernel estimates in Section 4 of the main paper. To facilitate efficient inference, we employ the Subset of Regressors (SoR) approximation (Smola and Bartlett, 2000; Quiñonero-Candela et al., 2007). Let $\mathbf{f}$ represent the vector of function values at the labeled points, and $\mathbf{f}_*$ denote the function values at the unlabeled test points. The prior for the Sparse Atlas GP (S-AGP) can then be expressed as the following

$$q(\mathbf{f}, \mathbf{f}_*) = \mathbf{N}\left(0, \begin{bmatrix} Q_{\mathbf{ff}} & Q_{\mathbf{ff}_*} \\ Q_{\mathbf{f}_*\mathbf{f}} & Q_{\mathbf{f}_*\mathbf{f}_*} \end{bmatrix}\right)$$

where $Q_{\mathbf{a},\mathbf{b}} = \Sigma_{\mathbf{a},\mathbf{u}}\Sigma^{-1}_{\mathbf{u},\mathbf{u}}\Sigma_{\mathbf{u},\mathbf{b}}$. By applying Algorithm in Section 4, the matrices $\Sigma_{\mathbf{uu}}$, $\Sigma_{\mathbf{uf}}$, and $\Sigma_{\mathbf{uf}_*}$ are derived by estimating the transition densities of the BM simulation paths, using the inducing points as the initial points. With this approximation, the marginal distribution of the S-AGP can be expressed as: $p(\boldsymbol{y}|\mathbf{f}) \approx q(\boldsymbol{y}|\boldsymbol{u}) = \prod_{i=1}^{n} \mathbf{N}\left(y_i|\Sigma_{f_i\mathbf{u}}\Sigma^{-1}_{\mathbf{uu}}\mathbf{u}, \sigma^2_{noise}\mathbf{I}\right)$. The inducing points in the marginal likelihood can be integrated out by incorporating their prior distribution: $p(\boldsymbol{y}) = \int q(\boldsymbol{y}|\boldsymbol{u})p(\boldsymbol{u})d\boldsymbol{u} = \mathbf{N}\left(0, \Sigma_{\mathbf{fu}}\Sigma^{-1}_{\mathbf{uu}}\Sigma_{\mathbf{uf}} + \sigma^2_{noise}\mathbf{I}\right)$. The hyper-parameters such the diffusion time and rescaling parameter can be optimised by maximising the marginal likelihood. Using the proposed model, the predictive distribution can be derived as follows: $q(\mathbf{f}_*|\mathbf{y}) = \mathbf{N}\left(\mathbf{Q}_{\mathbf{f}_*\mathbf{f}}\left(\mathbf{Q}_{\mathbf{ff}} + \sigma^2\mathbf{I}\right)^{-1}\mathbf{y}, \mathbf{Q}_{\mathbf{f}_*\mathbf{f}_*} - \mathbf{Q}_{\mathbf{f}_*\mathbf{f}}(\mathbf{Q}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}\mathbf{Q}_{\mathbf{ff}_*}\right)$. By employing the SoR approximation, the number of required BM sample path simulations is significantly reduced from $n_d \times N_{bm}$ to $m \times N_{bm}$, where m denotes the number of inducing points, n_d represents the number of data points, and N_{bm} is the number of BM paths per starting point. Additionally, the computational complexity of inverting the covariance matrix is reduced from $O(n_d^3)$ to $O(n_d \times m^2)$.

Method	S-AGP ($m = 8$)	S-AGP ($m = 10$)	S-AGP ($m = 16$)	Euclidean GP	RC-AGP
RMSE	0.48 (0.05)	0.39 (0.03)	0.36 (0.05)	0.41 (0.07)	0.32 (0.07)

Table 5: Root-Mean-Squared Error (RMSE) of predictive means for Sparse-AGP with different numbers of inducing points (m) on the torus point cloud.

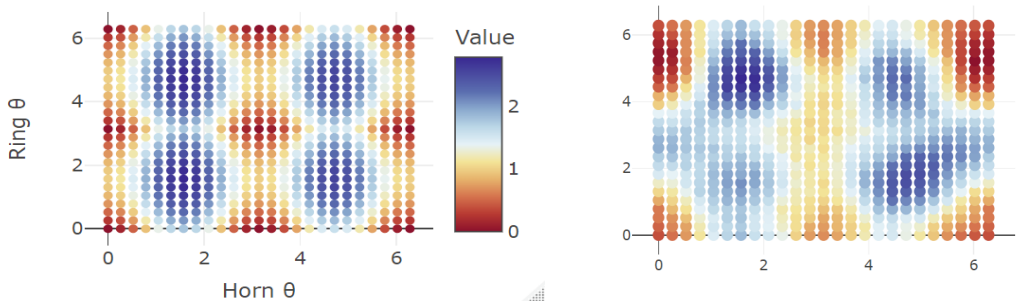
The number of inducing points in sparse GP can affect the regression performance (Galy-Fajou and Opper, 2021; Moss et al., 2023). Sparse GPs approximate the posterior using a smaller number of inducing points. However, if the number of inducing points is too small, the model may fail to capture the underlying trends in the data set, leading to reduced predictive performance. To illustrate the impact of inducing point quantity on model performance, we apply S-AGP on the torus point cloud example and vary the number of inducing points. The mean RMSE (calculated over 10 replicates) presented in Table 5. In each replicate, there are 30 labeled points and 625 grid points. The observed labeled points

are considered noise free. From Table 5, we can see that when the number of inducing points increases from 8 to 16, the Mean RMSE of S-AGP reduces from 0.48 to 0.36. While the Euclidean GP with RBF kerne has RMSE 0.41. And the RC-AGP achieve the minimum RMSE with 0.32.

Appendix H. Other Regression Results of Sparse Atlas GP

In the main paper, we have already discussed results generated by different methods across various applications. In this section, we will focus on the results produced by the S-AGP method for these applications.

Figure 17 presents the true function and S-AGP predictions on the torus point cloud. For a better visualization, the true function values are displayed in two-dimensional plot (Figure 17(a)), where the horizontal axis corresponds to the horn angle θ and the vertical axis to the ring angle θ. Figure 17(b) display the S-AGP predictions. The blue regions in Figure 17(b) appear more bracket-like rather than four circles in the Figure 6(b). Despite this discrepancy, S-AGP outperforms GL-GP and Euclidean $\mathbb{R}^3$GP, as the central region in the S-AGP prediction appears orange, rather than light blue from other methods.

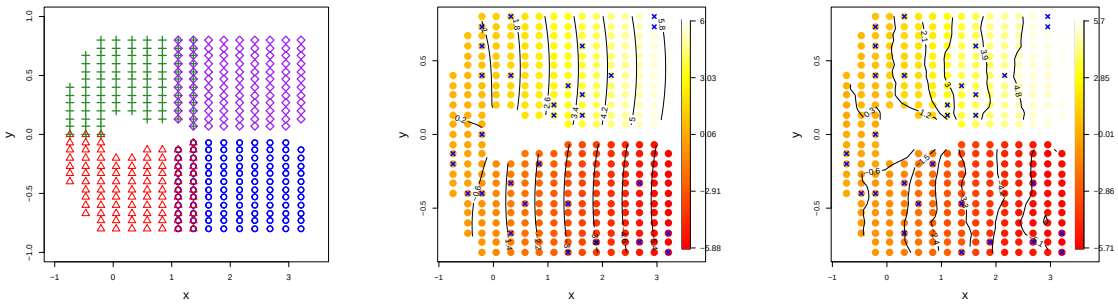


(a) True function on Torus in 2D (b) S-AGP prediction

Figure 17: True function and prediction on Torus: (a) True function on the point cloud; red-to-blue colors indicate function values on the grid points. (b) S-AGP prediction at the grid points.

Figure 18 compares the true function values and the S-AGP prediction for the U-shape point cloud. Four subsets are defined in Figure 18(a). The blue crosses indicate the 30 randomly selected observations used for the analysis. The true function values range from approximately -6 at the lower-right part to 6 at the upper-right part in Figure 18(b). The contours of the S-AGP predictive mean in Figure 18(c) closely match the true function, although the contour line is slightly irregular compared to the true function. The values in the lower-right and upper-right parts of the plot show no interaction, suggesting that the performance of S-AGP is better than that of both GL-GP and $\mathbb{R}^2$GP.

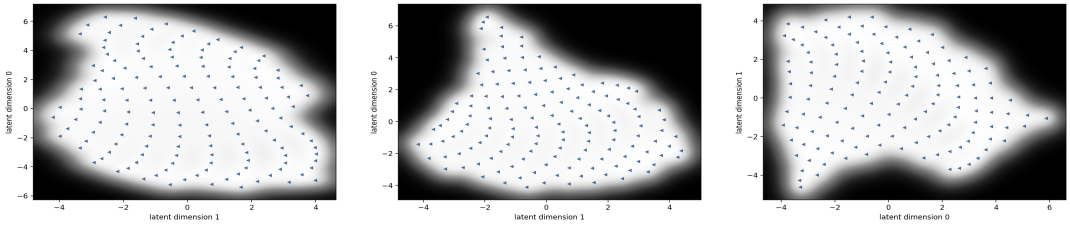
The Shark Prey LEGO image data set is divided into eight subsets, with each subset used to construct a latent space through GPLVM. Figure 19 visualizes three of these latent spaces. The background color reflects the variance of the mapping, with darker areas indicating higher uncertainty. The shapes of the bright regions differ across the three latent



(a) The U-shape subsets (b) True function in U-shape (c) S-AGP prediction

Figure 18: U-shape example: (a) The U-shape subsets. (b) True function on the point cloud (blue crosses represent the training observations). (c) S-AGP prediction.

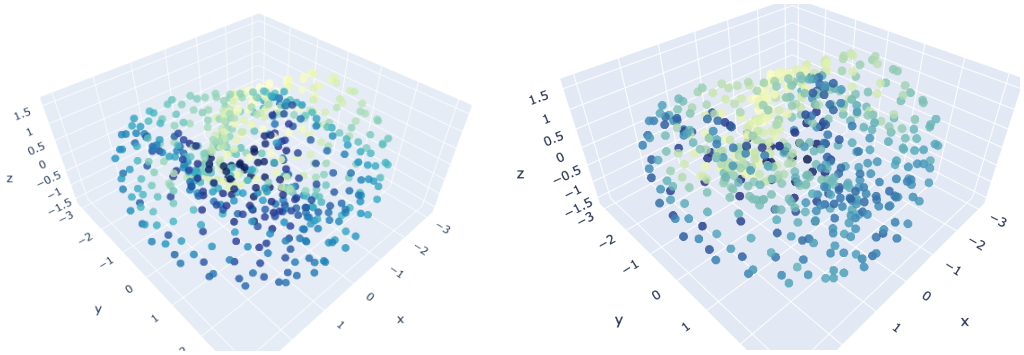
spaces: the points in Figure 19(a) and Figure 19(c) form quadrilateral patterns, while those in Figure 19(b) have a triangular shape.



(a) Latent Space of subset S_1 (b) Latent Space of subset S_2 (c) Latent Space of subset S_3

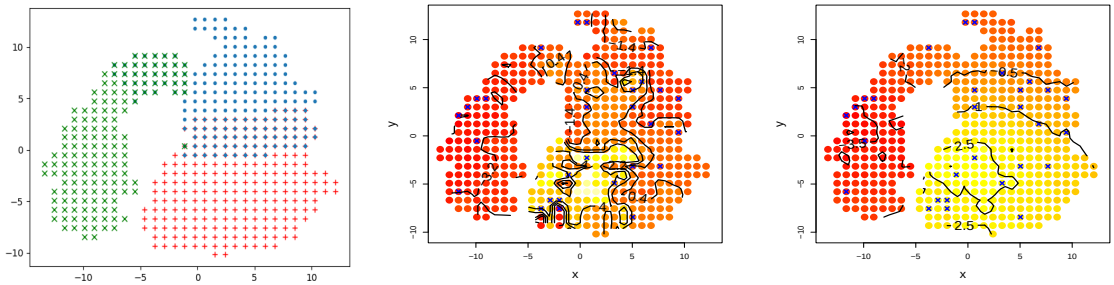
Figure 19: Visualization of three latent spaces among eight for Shark Prey data set; The background color indicates the variance of the mapping, with darker areas representing higher uncertainty.

For the Shark Prey image data set, the regression function is generated according to $y = \cos(\theta_i) + 2\sin(\phi_j)$, where θ_i denotes the rotation angle of the predator–prey pair, and ϕ_j denotes the angular position of the prey relative to the predator along the circular trajectory. For better visualization of the regression function on shark prey images, each point in Figure 20, projected in $\mathbb{R}^3$ using PCA, represents an image with the color scale corresponding to the function values. Although the S-AGP predictions shown in Figure 20(b) generally maintain the overall pattern of the true function displayed in Figure 20(a), there is a noticeable discrepancy where the lower central points are shown in green rather than dark blue, differing from that of the true function. The subsets of the Aral Sea point cloud are depicted in Figure21(a). A comparison between the S-AGP predictions and the true chlorophyll concentration levels in the Aral Sea is provided in Figure21(b). In Figure21(b), the color intensity represents the true chlorophyll concentration across 485 distinct locations, with blue crosses marking 30 randomly selected observations. The S-AGP prediction shown in Figure21(c) broadly reflects a similar pattern to the actual chlorophyll concentration, with the exception of the rightmost region, where the predictions exhibit over-smoothing.



(a) Image cloud in 3D (b) S-AGP prediction

Figure 20: Comparison of different methods in Shark prey LEGO image cloud:(a) true function in the point cloud in 3D; (b) S-AGP prediction.



(a) Aral sea subsets (b) Truth in the Aral Sea (c) S-AGP prediction

Figure 21: Aral Sea data set: (a) the Aral Sea subsets; (b) the true function on the point cloud; (c) the S-AGP prediction at the grid points, where the blue crosses denote the training observations.

Appendix I. Brownian Motion on Torus.

The three-dimensional coordinates of a Torus $\in \mathbb{R}^3$ can be parametrised by four variables: r radius of tube, R distance from centre of the tube to the centre of the torus, θ and ϕ are angles to make full circles while θ for angle of torus and ϕ for angle of tube. In our case, we fix $R = 2$ and $r = 1.3$ and let $\theta \in [-\pi, \pi]$ and $\phi \in [0, 2\pi]$. Consider the Torus parametrised by $\mathbf{X}(\theta, \phi) = ((R + r\cos\theta)\cos\phi, (R + r\cos\theta)\sin\phi, r\sin\theta)$

To find its metric tensor, we first compute the partial derivatives

$$\mathbf{X}_\phi = ((R + r\cos\theta)(-\sin\phi), (R + r\cos\theta)\cos\phi, 0)$$
$$\mathbf{X}_\theta = (r\cos\phi(-\sin\theta), r\sin\phi(-\sin\theta), r\cos\theta)$$

The metric tenor is given by

$$(\mathbf{X}_\theta \cdot \mathbf{X}_\theta)d\theta^2 + 2(\mathbf{X}_\theta \cdot \mathbf{X}_\phi)d\theta\, d\phi + (\mathbf{X}_\phi \cdot \mathbf{X}_\phi)d\phi^2 = r^2 d\theta^2 + (R + r\cos\theta)^2 d\phi^2$$

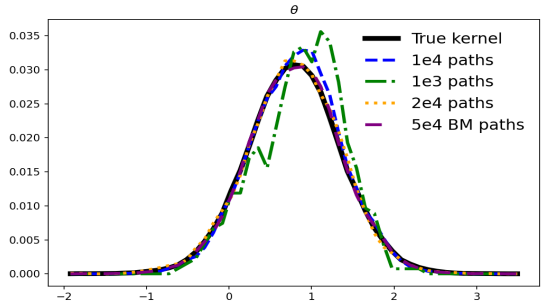


Figure 22: Comparison of BM transition density estimates and the analytical kernel on Torus. The black solid line represents the analytical kernel. Estimates obtained from 1,000 simulated paths are shown as the green dash-dotted line, from 10^4 paths as the blue dashed line, from 2×10^4 paths as the yellow dotted line, and from 5×10^4 paths as the purple dashed line.

or in matrix form

$$g = \begin{bmatrix} r^2 & 0 \\ 0 & (R + r\cos\theta)^2 \end{bmatrix}, \qquad \frac{\partial g}{\partial \theta} = \begin{bmatrix} 0 & 0 \\ 0 & -2(R + r\cos\theta) r \sin\theta \end{bmatrix}$$

By substituting each term into the general equation of BM on manifold, the BM on Torus can be written as

$$\begin{aligned} d\theta(t) &= \frac{1}{2}(-g^{-1}\frac{\partial g}{\partial \theta} g^{-1})_{11} dt + \frac{1}{4}(g^{-1})_{11} tr(g^{-1}\frac{\partial g}{\partial \theta}) dt + (g^{-1/2})_{11} dB_1(t) \qquad (31) \\ &= -\frac{1}{2} r^{-1} \sin\theta (R + r\cos\theta)^{-1} dt + r^{-1} dB_1(t) \\ d\phi(t) &= (g^{-1/2})_{22} dB_2(t) = |(R + r\cos\theta)^{-1}| dB_2(t) \qquad (32) \end{aligned}$$

The accuracy of the heat-kernel estimates also depends on the number of simulated BM paths. As the number of paths increases, the Monte Carlo error decreases, leading to smoother density estimates that more closely approximate the true heat kernel. With the aid of the analytical parameterization of the Torus introduced in this section, we plot the analytical heat kernel as the solid black line in Figure 22, the ground-truth heat kernel is evaluated at 50 uniformly spaced points with a diffusion time $t = 2$. When using 1,000 simulated paths, the estimated density exhibits noticeable fluctuations. Increasing the number of paths to 10,000 substantially improves the smoothness of the estimates and brings them closer to the analytical kernel, although minor discrepancies persist, particularly near the density peak around $\theta = 0.75$. When the number of simulated paths increases to 20,000 or more, the estimated kernel closely aligns with the ground truth across the entire domain.

Appendix J. Using TDA Mapper and Persistence Diagrams to Partition the Point Cloud

Topological Data Analysis (TDA) provides a collection of tools for identifying topological features of point-cloud data, such as connected components, loops, and voids (Carlsson,

2009). In our setting, the input is a point cloud sampled from an unknown lower-dimensional manifold $\mathbb{M}$, possibly with noise. The goal is to use topological summaries to guide the construction of an atlas: the full point cloud may have nontrivial topology, but each local chart should be learned from a subset with simple, or trivial topology.

Persistent homology (Carlsson, 2009; Edelsbrunner and Harer, 2010) is used to summarize stable topological features of a point cloud. The most common constructions are based on Vietoris–Rips or Cech filtrations (Tralie et al., 2018). These filtrations build a nested sequence of simplicial complexes over a scale parameter. As the scale changes, topological features appear and disappear. Persistent homology records each feature by its birth and death times, producing a persistence diagram. Points close to the diagonal have short lifetimes and are usually regarded as topological noise, whereas points far from the diagonal indicate persistent topological features. These features include connected components (H_0), loops or tunnels (H_1), and voids (H_2). For a more complete introduction to persistent homology, see Edelsbrunner and Harer (2008).

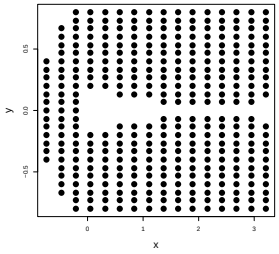
In this work, persistence diagrams are used as diagnostics for deciding whether a subset is simple enough to be represented by a single chart. We follow the statistical framework of Fasy et al. (2014), which provides hypothesis tests for distinguishing topological signal from topological noise based on the distance of persistence points from the diagonal. If the statistical test indicates that a subset contains significant persistent homology, the subset is further partitioned until no statistically significant topological features remain.

To construct subsets with trivial topology, we use TDA Mapper (Singh et al., 2007) together with persistent homology. Mapper first projects the data to a lower-dimensional filter space, using a filter such as ISOMAP (Tenenbaum et al., 2000), or an autoencoder (Baldi, 2012). The filter space is then divided into overlapping intervals or bins. Within each bin, the corresponding original observations are clustered in the original data space, for example using k-means (Macqueen, 1967) or DBSCAN (Ester et al., 1996). This produces a collection of overlapping subsets of the point cloud.

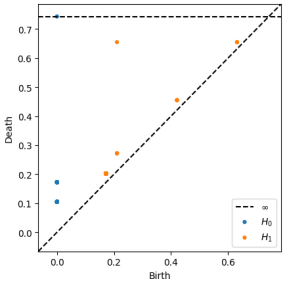
Algorithm 2 summarizes the procedure used in this work. We first apply Mapper to partition the point cloud into overlapping subsets. We then compute persistence diagrams for each subset. If a subset contains significant persistent homology, Mapper is applied again to refine that subset. This process is repeated until every subset has no significant persistent topological features. The resulting subsets form the empirical atlas cover.

Figure 23 illustrates this diagnostic for the U-shape, Aral Sea, Torus, and Shark Prey image data sets. The persistence diagrams of the full data sets show significant topological features, represented by points (orange and green) far from the diagonal. After partitioning, the persistence diagrams of the representative subsets show no significant persistent features; the remaining points(orange and green) lie close to the diagonal and are treated as topological noise. This supports the use of these subsets as local chart domains.

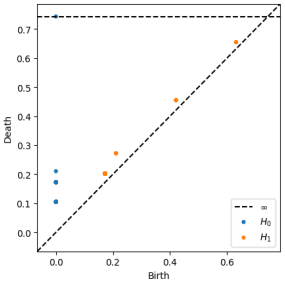
We also consider an alternative local-ball construction that does not use Mapper. This approach is summarized in Algorithm 3. Starting from a point s_i, we construct a local subset by intersecting the point cloud with an ambient Euclidean ball $\mathbf{B}_r^{\mathbb{R}^p}(s_i)$. We then compute the persistence diagrams of this subset. If significant persistent topology remains, the radius is decreased and the subset is recomputed. After one subset has been accepted, another point is selected from the uncovered part of the point cloud, and the process is



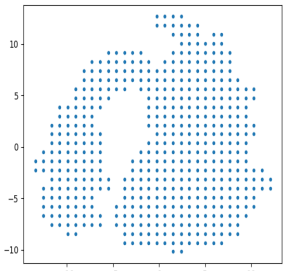
(a) U-shape Domain



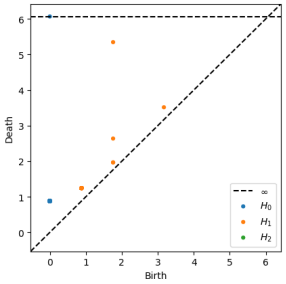
(b) U-shape Domain PD



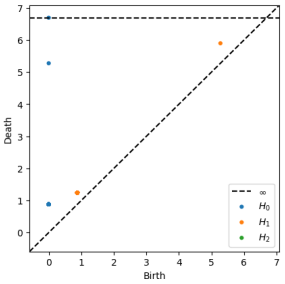
(c) U-shape Subset PD



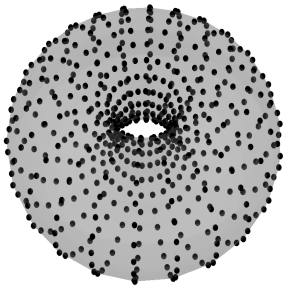
(d) Aral Sea Data Set



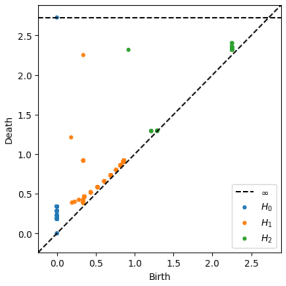
(e) Aral Sea PD



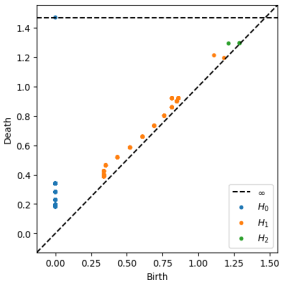
(f) Aral Sea Subset PD



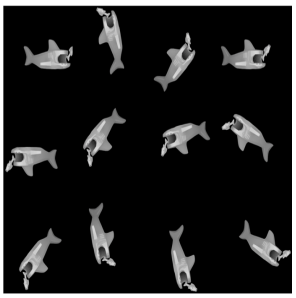
(g) Torus Point Cloud



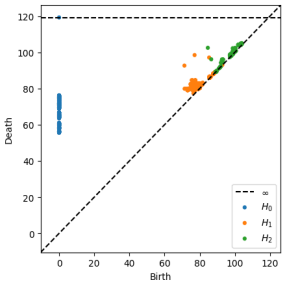
(h) Torus Point Cloud PD



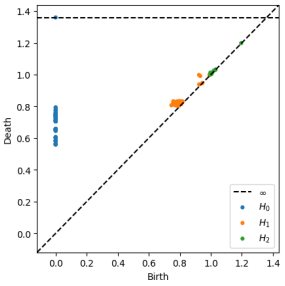
(i) Torus Subset PD



(j) Shark Prey Images



(k) Shark Prey PD



(l) Shark Prey Subset PD

Figure 23: Persistence diagrams for the data sets used in this paper. The left column shows the full point cloud for each example. The middle column shows the persistence diagram of the full point cloud, where persistent features indicate nontrivial topology (orange and green points locate far from diagonal). The right column shows the persistence diagram of one representative subset after partitioning. The subset diagrams have no significant persistent features (no points locate far from diagonal), supporting their use as local chart domains.

Algorithm 2 Construct overlapping subsets of the point cloud using Mapper and persistence diagrams.

1: Apply TDA Mapper to divide the point cloud $\mathcal{S}$ into overlapping subsets $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_c$.
2: Compute the persistence diagrams $PD_{\mathcal{S}_i}$ for each subset $\mathcal{S}_i$.
3: **while** some subset has significant persistent topological features **do**
4: Apply Mapper again to the subset with significant persistent homology.
5: Compute the persistence diagrams of the refined subsets.
6: Return the resulting subsets as local chart domains.

repeated until the full point cloud is covered. In practice, we found the Mapper-based procedure in Algorithm 2 to be more efficient and easier to tune.

Algorithm 3 Construct local subsets using ambient balls and persistence diagrams.

1: Select a point s_i from the point cloud $\mathcal{S}$.
2: Construct the ball $\mathbf{B}_r^{\mathbb{R}^p}(s_i)$ with radius r centered at s_i.
3: Define the local subset $\mathcal{S}_{s_i} = \mathbf{B}_r^{\mathbb{R}^p}(s_i) \cap \mathcal{S}$.
4: Compute the persistence diagrams $PD_{\mathcal{S}_{s_i}}$.
5: **while** $\mathcal{S}_{s_i}$ has significant persistent topological features **do**
6: Decrease the radius from r to r'.
7: Update the subset $\mathcal{S}'_{s_i} = \mathbf{B}_{r'}^{\mathbb{R}^p}(s_i) \cap \mathcal{S}$.
8: Compute the new persistence diagrams $PD_{\mathcal{S}'_{s_i}}$.
9: Repeat with points not yet covered. For each new subset, choose the center and radius so that $\mathcal{S}_{s_i} \setminus \bigcup_{j<i} \mathcal{S}_{s_j} \neq \emptyset$ and $\mathcal{S}_{s_i} \cap \mathcal{S}_{s_j} \neq \emptyset$ for at least one neighboring subset $\mathcal{S}_{s_j}$, except for the first subset in each connected component.

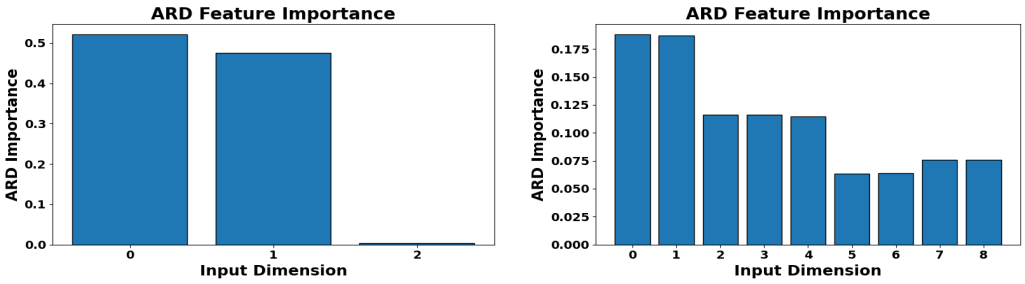
Appendix K. Dimension Estimation

In this section, we provide additional details on latent dimension selection. Our main diagnostic is automatic relevance determination (ARD) within the GPLVM, while cross-validation based on noise splitting provides an additional empirical check. In the examples considered here, both diagnostics support the same choice of latent dimension.

K.1 Automatic Relevance Determination

We apply Automatic Relevance Determination (ARD) (Lawrence, 2000; Zwiessele, 2017) to quantify the relative importance of each input dimension. In GPLVM, ARD is implemented by assigning a separate length-scale parameter to each input dimension in the kernel function (Rasmussen and Williams, 2006). Dimensions with small learned length-scales correspond to directions along which the function varies rapidly and are therefore more relevant, whereas large length-scales indicate weak influence. The hyperparameters are learned by maximising the marginal likelihood.

As shown in Figure 24, ARD assigns significantly higher relevance to the first two dimensions, with a noticeable drop thereafter. We therefore retain the two most relevant



(a) ARD of Torus

(b) ARD of Shark Prey images

Figure 24: ARD relevance scores learned on the Torus data set and Shark prey images. Clear drops after the first two dimensions suggest the dominant predictive structure is effectively two-dimensional.

dimensions as the latent space, as they capture the dominant predictive structure identified by the marginal likelihood optimisation.

K.2 Cross-validation using Data Augmentation Technique

The data sets considered in this work have limited sample sizes, standard validation procedures such as k-fold cross-validation or hold-out splitting may use the available data inefficiently. To address this issue, we use a noise-splitting data augmentation approach to construct two independent pseudo-data sets that share the same conditional mean given the latent variables (Rasines and Young, 2023). One split is used for latent-space fitting, while the other is used for evaluating candidate latent dimensions.

Let $\{s_i\}_{i=1}^n$, $s_i \in \mathbb{R}^p$, denote independent observations. For each observation s_i, we construct a pair of noise-split replicates. Given an arbitrary positive definite matrix $\Lambda \succ 0$, we sample $\xi_i \sim \mathcal{N}(0, \Lambda)$, and define $s_i^A := s_i + \xi_i \sim \mathcal{N}(\eta_i, I_p + \Lambda)$, and $s_i^B := s_i - \Lambda^{-1}\xi_i \sim \mathcal{N}(\eta_i, I_p + \Lambda^{-1})$. The two constructed observations s_i^A and s_i^B have the same conditional mean η_i. And they are not correlated, $\mathrm{Cov}(s_i^A, s_i^B) = \mathrm{Cov}(s_i + \xi_i, s_i - \Lambda^{-1}\xi_i) = \mathrm{Cov}(s_i, s_i) - \mathrm{Cov}(\xi_i, \xi_i)\Lambda^{-1} = I_p - \Lambda\Lambda^{-1} = 0$.

For the training split S^A, we fit the latent space using GPLVM or VAE. For the evaluation split S^B, we evaluate the learned latent representation using overall reconstruction error. We repeat this procedure and average the resulting errors to select the latent dimension. As shown in Tables 6, cross-validation attains its minimum error when the intrinsic dimension is set to two. This is consistent with the ARD results.

data set	$q=1$	$q=2$	$q=3$	$q=4$	$q=5$	$q=6$	$q=7$	$q=8$
Torus	0.5614	**0.4635**	0.4781	–	–	–	–	–
Shark	0.8124	**0.8085**	0.8087	0.8104	0.8125	0.8125	0.8140	0.8143

Table 6: Cross-validation mean squared error (MSE) for selecting the latent dimension q.

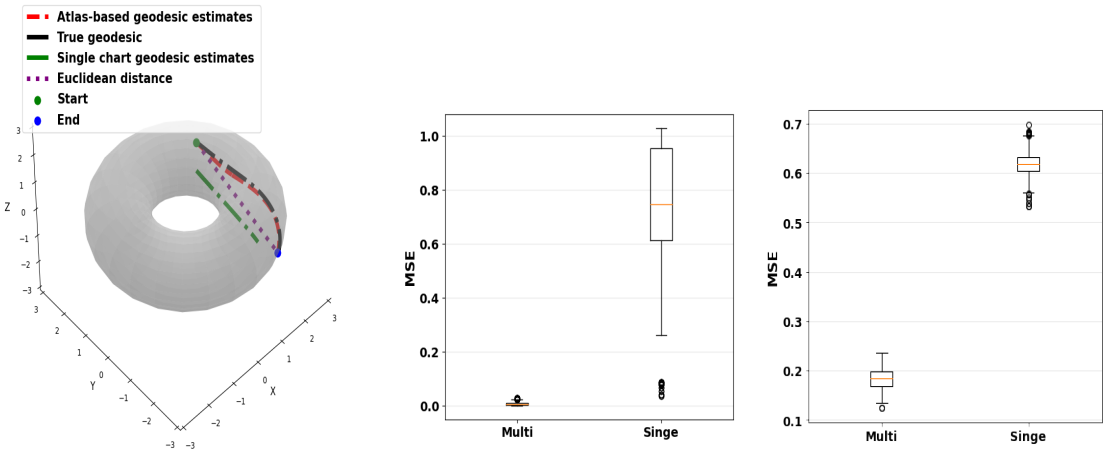
Appendix L. Diagnostics for the Learned Atlas

An atlas is not unique, and its coordinate representation may change under chart reparameterization. Therefore, we do not assess the learned atlas by direct coordinate-wise comparison with a particular "true" atlas or by entrywise comparison of metric tensors. Instead, we use two diagnostics that are directly relevant to the proposed method: intrinsic geometric accuracy, measured through geodesic distances when ground truth is available, and reconstruction accuracy in the observation space.

L.1 Geodesic-Distance Diagnostic

For the torus example, the ground-truth geometry is known, so we can compare geodesic distances computed from the learned atlas with the true geodesic distances. This diagnostic checks whether the learned atlas captures the intrinsic geometry of the torus.

Geodesics are curves that locally minimize length between two points on a manifold and satisfy the geodesic equation: $\ddot{x}^k(t) + \sum_{i,j=1}^{q} \Gamma_{ij}^k(x(t))\dot{x}^i(t)\dot{x}^j(t) = 0$, where Γ_{ij}^k are the Christoffel symbols associated with the estimated metric (Jost, 2005; Lee, 2018), $\Gamma_{kl}^m(\gamma) = \frac{1}{2}g^{mi}(\gamma)\left(g_{ik,l}(\gamma) + g_{il,k}(\gamma) - g_{kl,i}(\gamma)\right)$. Given a pair of points on the manifold, we compute the geodesic by solving the ODEs using the metric induced by the learned charts. Figure 25(a) shows an example in which the atlas-based geodesic closely follows the true geodesic on the torus and is more accurate than the Euclidean path and the single-chart path. Table 7 reports this comparison for five randomly selected pairs of points. RE represent relative errors. The atlas-based estimates are consistently close to the true geodesic, indicating the learned atlas captures the intrinsic geometry of torus point cloud.



(a) Geodesic on the torus (b) Torus reconstruction error (c) Shark prey reconstruction error

Figure 25: Atlas diagnostics. Panel (a) compares the atlas-based geodesic with the true torus geodesic and alternative paths. Panels (b) and (c) compare reconstruction errors for the multi-chart atlas and a single global chart.

Case	True	Atlas	Euclidean	SC	RE (Atlas) (%)	RE(Euc.) (%)	RE (SC) (%)
1	5.646	5.670	3.907	3.003	0.42	30.80	46.81
2	5.308	5.333	3.866	2.886	0.48	27.16	45.62
3	5.885	5.890	3.209	2.868	0.09	45.46	51.25
4	7.868	7.860	4.529	1.240	0.11	42.44	84.24
5	5.122	5.106	4.205	3.725	0.32	17.91	27.28

Table 7: Comparison of atlas-based geodesic lengths, Euclidean shortest path distances, and single-chart(SC) geodesic lengths against the true geodesic lengths.

L.2 Reconstruction Diagnostic

We also assess whether each learned chart accurately represents its assigned data in the observation space. For a point $s_{ij} \in \mathcal{S}_i$, we first map it to the latent coordinate $\widehat{x}_{ij} = \widehat{\phi}_i^{-1}(s_{ij})$, and then reconstruct it in the observation space using $\widehat{\phi}_i(\widehat{x}_{ij})$. The pointwise reconstruction error is

$$\mathrm{RE}_{ij} = \left\| s_{ij} - \widehat{\phi}_i(\widehat{x}_{ij}) \right\|^2, \qquad \widehat{x}_{ij} = \widehat{\phi}_i^{-1}(s_{ij}).$$

The forward and inverse maps are explained in Appendix C. We compare the multi-chart atlas with a single global chart learned by GPLVM using the same latent dimension. For a manifold with nontrivial topology, such as the torus, a single low-dimensional global chart is expected to be less effective. Therefore, lower reconstruction error from the multi-chart atlas supports the use of a local atlas representation.

For the Shark prey image data, the true manifold structure is unknown, so we focus on reconstruction error rather than geodesic comparison. The reconstruction-error boxplots are shown in Figures 25(b) and 25(c). In both cases, the multi-chart atlas gives smaller reconstruction errors than the single-chart model, supporting the quality of the learned atlas representation. To account for the scale of each data set, we also compute the normalized reconstruction error

$$\mathrm{NRE}_{\mathrm{atlas}} = \frac{\sum_i \sum_{j=1}^{n_i} \left\| s_{ij} - \widehat{\phi}_i(\widehat{x}_{ij}) \right\|^2}{\sum_i \sum_{j=1}^{n_i} \left\| s_{ij} - \bar{s}_i \right\|^2}.$$

Here $\bar{s}_i = n_i^{-1} \sum_{j=1}^{n_i} s_{ij}$, i indexes the subset, and j indexes the data point within chart i. A value close to zero indicates accurate reconstruction, while a value near one indicates that the reconstruction error is comparable to using the subset mean as a baseline.

The normalized reconstruction errors for the torus and Shark prey image data sets are given in Tables 8. For both data sets, the normalized reconstruction error of the atlas is substantially smaller than that of the single-chart approach.

data set	Atlas	Single chart
Torus	0.0001	0.12
Shark prey image	0.05	0.51

Table 8: Normalised reconstruction error (NRE) for atlas and single-chart representations.

Appendix M. Additive Riemannian-Corrected Kernels

Let $K_h^{\text{chart}} \in \mathbb{R}^{n_v \times n_v}$ denote the estimated heat-kernel matrix between the n_v charts. Let $c(s_i)$ denote the chart containing input s_i. We expand the chart-level matrix to the observation level $\widetilde{K}_h(s_i, s_j)$ as in Equation 7 in Section 5. $\widetilde{K}_h(s_i, s_j) = K_h^{\text{chart}}\big(c(s_i), c(s_j)\big)$. For n inputs, both $\widetilde{K}_h$ and the local covariance matrix K_{local} therefore have dimension $n \times n$.

For the additive caes, the two matrices are placed on comparable scales. We define

$$\overline{K}_h = \frac{\widetilde{K}_h}{\text{mean}\{\text{diag}(\widetilde{K}_h)\}}, \qquad \overline{K}_{\text{local}} = \frac{K_{\text{local}}}{\text{mean}\{\text{diag}(K_{\text{local}})\}}. \tag{33}$$

For the RBF correlation matrix used in our experiments, $\text{diag}(K_{\text{local}}) = 1$, so its normalization does not change its values. The additive covariance matrix is defined as

$$K_{\text{add}} = \sigma^2 \left\{\alpha \overline{K}_h + (1-\alpha)\overline{K}_{\text{local}}\right\} + \sigma_n^2 I, \qquad 0 \le \alpha \le 1. \tag{34}$$

The additive construction is positive semidefinite because it is a nonnegative weighted sum of positive-semidefinite matrices. The hyperparameters are optimized by maximum likelihood.

We compare the original Hadamard-product construction with the additive construction on the 30 dB torus and U-shape experiments. For each example, we use the same ten training sets, noisy responses, chart definitions, and Brownian-motion covariance candidates as in the corresponding experiments in the main paper. Table 9 presents the mean RMSE and standard deviation over ten replicates.

data set	Hadamard RC-AGP	Additive RC-AGP$_{\text{add}}$	Euclidean GP (RBF)
Torus	0.33 (0.06)	0.35 (0.06)	0.42 (0.09)
U-shape	0.17 (0.05)	0.74 (0.10)	1.73 (0.25)

Table 9: Comparison of the Hadamard-product and additive kernel constructions on the U-shape and Torus data sets. Values in parentheses are sample standard deviations.

For the torus, the additive model performs similarly to, but slightly worse than, the Hadamard construction. The difference is more pronounced for the U-shape, where the additive model has substantially larger RMSE than the Hadamard RC-AGP. Both Riemannian-corrected constructions outperform the Euclidean GP with an RBF kernel based on Euclidean distances.

These results clarify the different roles of the two kernel compositions. Consider two points located on opposite arms of the U-shape. They can be close in Euclidean distance but far apart according to the intrinsic geometry. Consequently, $K_{\text{local}}(s_i, s_j)$ may be large, while $\widetilde{K}_h(s_i, s_j)$ is small. Under the additive construction in Equation 34, the local term continues to contribute positive covariance whenever $\alpha < 1$. The riemannian component can reduce the relative importance of this correlation, but it cannot remove the contribution from the local kernel. In contrast, the Hadamard construction is

$$K_{\text{prod}} = \widetilde{K}_h \odot K_{\text{local}}, \to K_{\text{prod}}(s_i, s_j) = \widetilde{K}_h(s_i, s_j) K_{\text{local}}(s_i, s_j). \tag{35}$$

Thus, when the heat-kernel covariance is close to zero, the product suppresses an otherwise large Euclidean correlation. In this sense, the riemannian component acts as a geometry-

aware gate. The product construction requires both components to support a large covariance. This compatibility condition is particularly beneficial for domains containing gaps or barriers, as demonstrated by the U-shape experiment.

Appendix N. Additional Comparison Results

N.1 Comparison with Matérn Covariance

To examine whether the performance differences in the main experiments are attributable to the use of the RBF covariance in the Euclidean GP baseline, we conducted an additional comparison using the Matérn-5/2 covariance function. The Euclidean Matérn GP uses exactly the same training and test sets as the Euclidean RBF GP and RC-AGP. Its covariance parameters, including the marginal variance, length scale, and noise variance, are estimated by maximizing the marginal likelihood.

Table 10 reports the results for the 10 dB U-shape and torus experiments, as well as the shark and Aral Sea data sets. These examples cover inputs embedded in low-dimensional Euclidean spaces, a nonlinear manifold in $\mathbb{R}^3$, and a high-dimensional image space.

data set	Euclidean RBF GP	Euclidean Matérn GP	RC-AGP
U-shape,	1.83 (0.30)	1.82 (0.45)	0.68 (0.13)
Torus,	0.48 (0.09)	0.42 (0.07)	0.40 (0.05)
Shark	1.09 (0.06)	1.10 (0.06)	0.62 (0.04)
Aral Sea	2.65 (0.42)	2.49 (0.38)	2.31 (0.23)

Table 10: Comparison with a Euclidean GP using the Matérn covariance function. Values are mean RMSE, with sample standard deviations over ten replicates in parentheses.

The results show that changing the Euclidean covariance family can improve prediction in some settings, although the improvement is not uniform. For the U-shape and shark data sets, the Euclidean Matérn and RBF GPs have similar predictive performance. The Matérn GP improves upon the RBF GP for the torus and Aral Sea data sets. Nevertheless, RC-AGP remains substantially more accurate than both Euclidean GP baselines for the U-shape, shark, and Aral Sea data sets. For the 10 dB torus experiment, RC-AGP and the Euclidean Matérn GP perform similarly, with RC-AGP giving a slightly smaller mean RMSE. These comparisons demonstrate that replacing the RBF covariance with a less smooth Matérn covariance does not generally account for the improvements obtained by RC-AGP. In particular, changing the Euclidean covariance family does not resolve the problem that points can be close in the ambient Euclidean space while being widely separated according to the intrinsic manifold geometry.

N.2 Comparison with FLGP

We evaluated the Fast Graph Laplacian Gaussian Process (FLGP) implementation provided by He et al. (2026) on two representative examples: the Torus and U-shape data sets. In both cases, we used exactly the same training/test splits and noisy responses (30 dB) as those used in the corresponding experiments in the manuscript. The FLGP implementation requires several graph-approximation tuning parameters. In particular, s denotes the

number of subsampled anchor points used to construct the reduced graph, r denotes the number of nearest neighbours used in graph construction, and K denotes the number of graph-Laplacian eigenpairs retained in the heat-kernel approximation. We found that predictive performance depends noticeably on these tuning parameters. Therefore, in addition to the coarse tunning settings, we performed an extensive grid search over (s, r, K) to obtain the best-performing FLGP model.

The results are reported in Table 11. Using the coarse tuning settings, FLGP achieved RMSE values of approximately (0.93) on the U-shape data set and (0.55) on the Torus data set. After the extensive grid search tuning, the performance improved substantially, yielding RMSE values of (0.35(0.08)) on the U-shape and (0.43(0.09)) on the Torus. These results are considerably better than those obtained by the Euclidean RBF GP on the U-shape data set, and comparable to the Euclidean RBF GP on the Torus data sets. It partially supports the reviewer's observation that graph-based methods can outperform Euclidean GPs on some non-Euclidean domains. However, even after the extensive tuning, FLGP remained less accurate than the proposed RC-AGP, which achieved RMSE values of (0.16(0.05)) on the U-shape data set and (0.33(0.06)) on the Torus data set.

data set	Euclidean RBF GP	FLGP (coarse tuning)	FLGP (Extensive tuned)	RC-AGP
Torus	0.42(0.09)	0.55(0.06)	0.43(0.09)	0.33(0.06)
U-shape	1.73(0.25)	0.93(0.61)	0.35(0.08)	0.16(0.05)

Table 11: Comparison with the FLGP package of He et al. (2026) on the Torus and U-shape data sets. Values are mean RMSE with sample standard deviations over ten replicates.

N.3 Comparison with Intrinsic AGP

A direct intrinsic AGP, similar in spirit to Niu et al. (2019), uses the estimated heat kernel itself as the covariance function. In this construction, prediction depends entirely on Monte Carlo estimates of transition densities between training and prediction locations. Consequently, finite-sample prediction accuracy is directly affected by Monte Carlo estimation error and numerical asymmetry in the estimated heat kernel, particularly for prediction locations that are reached by relatively few simulated Brownian-motion paths. In contrast, RC-AGP does not replace the covariance function entirely by the estimated heat kernel. Instead, the estimated heat kernel is used as a Riemannian correction to a standard positive-definite base kernel, $K_{\mathrm{RC}}(s_i, s_j) = K_{\mathrm{RBF}}(s_i, s_j)\,\widetilde{K}_h(s_i, s_j)$, where K_{RBF} provides a stable local covariance structure and $\widetilde{K}_h$ acts as a geometry-aware correction. Thus, the Brownian-motion estimate is used more conservatively: rather than determining the covariance directly, it modulates an already well-conditioned covariance kernel by downweighting pairs of points that may be close in Euclidean space but geometrically distant on the manifold. This construction reduces the influence of finite-sample errors in the estimated heat kernel while preserving the intrinsic geometric information.

To compare the two constructions, we performed an additional experiment on the U-shape data set. The direct intrinsic AGP estimates the heat kernel by simulating BM paths from every training point (30 starting locations), whereas RC-AGP simulates Brownian

motion only from the four atlas representatives. Consequently, the intrinsic AGP requires more than an order of magnitude more Brownian-motion simulations than RC-AGP.

Table 12 reports the resulting prediction errors. Both intrinsic AGP and RC-AGP substantially outperform the Euclidean RBF GP by exploiting the intrinsic manifold geometry. Despite requiring substantially fewer BM simulations, RC-AGP also achieves the lowest prediction error.

Metric	Euclidean RBF GP	Intrinsic AGP / Direct Heat-kernel GP	RC-AGP
RMSE	1.20	0.24	0.16

Table 12: Comparison on the U-shape data set.

The improved predictive performance of RC-AGP is consistent with its finite-sample construction. Because the direct intrinsic AGP relies entirely on the Monte Carlo estimate of the heat kernel to define the covariance function, its predictions are more sensitive to finite-sample estimation error. In contrast, RC-AGP uses the estimated heat kernel only as a geometry-aware correction to a well-conditioned base kernel, making the resulting covariance construction more robust to inaccuracies in the estimated heat-kernel estimates while still capturing the intrinsic manifold geometry.

Appendix O. BM Running Time

In our framework, the dominant computational cost arises from the Brownian-motion (BM) simulations used to estimate the heat kernel. Atlas construction is performed only once as a preprocessing step, while subsequent regression and prediction use standard Gaussian-process operations.

The computational complexity of the BM simulation stage is approximately $O(n_v N_{\mathrm{BM}} T)$, where n_v denotes the number of chart-level representative points, N_{BM} is the number of simulated trajectories, and T is the number of time steps per trajectory. Importantly, BM simulations are performed only from chart-level representative points rather than from every observation in the point cloud. Consequently, the computational cost grows with the number of atlas representatives rather than directly with the total sample size. In practice, n_v is typically substantially smaller than the total number of observations, which helps mitigate the computational burden associated with large point clouds.

To provide a reproducible measure of computational cost across data sets, we report a standardized benchmark corresponding to the simulation of 10,000 Brownian-motion paths with 200 time steps from a single starting point. This benchmark was chosen because our study indicates that approximately 10,000 trajectories are sufficient to obtain stable transition-density estimates. We report a standardized unit benchmark rather than aggregate wall-clock times for the full experiments, since overall runtime depends strongly on the available computational resources, hardware configuration, and parallelization strategy. The standardized benchmark therefore provides a more reproducible measure of the intrinsic computational cost of the proposed Brownian-motion simulation procedure. All timings were obtained on a MacBook laptop equipped with an M3 CPU.

The differences in runtime across data sets are mainly due to differences in the cost of a single BM step. For the U-shape and Aral Sea examples, the simulations are essen-

data set	U-shape	Torus	Shark	Aral Sea
BM benchmark time (seconds)	6.5	134.2	196.1	6.3

Table 13: Brownian-motion simulation benchmark. Reported wall-clock times correspond to 10,000 trajectories, 200 time steps, and a single starting point on an M3 MacBook laptop.

tially Euclidean Brownian motions with boundary handling, so each step is inexpensive. In contrast, the Torus point cloud and Shark image-cloud examples use learned probabilistic Riemannian metrics. Therefore, each BM step requires evaluating a spatially varying metric tensor. Moreover, when a path crosses from one chart to another, the transition requires mapping the latent point back through the observation space and projecting it into the target chart. These metric-evaluation and chart-transition operations are more expensive than the trivial Euclidean BM updates used in the U-shape and Aral Sea examples. This explains why the Torus and Shark examples require longer runtimes.

Nevertheless, even for the most computationally demanding examples (Torus and Shark), the total cost remains practical (approximately 2–3 minutes for benchmark). And the BM simulation is performed only once to construct the Riemannian-corrected kernel, after which GP regression and prediction proceed without further Brownian-motion simulation. Moreover, the BM simulations are embarrassingly parallel since trajectories are independent across both starting points and simulated paths.

References

Theodore W Anderson. The non-central wishart distribution and certain problems of multivariate statistics. *The Annals of Mathematical Statistics*, pages 409–431, 1946.

Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012.

Christopher M Bishop, Markus Svensén, and Christopher KI Williams. Magnification factors for the gtm algorithm. In *Fifth International Conference on Artificial Neural Networks (Conf. Publ. No. 440)*, pages 64–69. IET, 1997.

David Bolin, Alexandre B Simas, and Jonas Wallin. Gaussian Whittle-Matérn fields on metric graphs. *Bernoulli*, 30(2):1611–1639, 2024.

Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46 (2):255–308, 2009.

I. Castillo, G. Kerkyacharian, and D. Picard. Thomas bayes' walk on manifolds. *Probab. Theory Relat. Fields*, page 665–710, 2014.

Isaac Chavel. *Eigenvalues in Riemannian geometry*. Academic press, 1984.

David Dunson, Hau-Tieng Wu, and Nan Wu. Diffusion based Gaussian processes on restricted domains. *arXiv preprint arXiv:2010.07242*, 2020.

H Edelsbrunner and J Harer. Persistent homology—a survey, in "surveys on discrete and computational geometry. twenty years later", 257-282, eds. je goodman, j. pach and r. pollack. *Contemporary Mathematics*, 453, 2008.

Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, Larry Wasserman, Sivaraman Balakrishnan, and Aarti Singh. Confidence sets for persistence diagrams. *The Annals of Statistics*, 42(6):2301–2339, 2014.

Daniel Floryan and Michael D Graham. Data-driven discovery of intrinsic dynamics. *Nature Machine Intelligence*, 4(12):1113–1120, 2022.

Théo Galy-Fajou and Manfred Opper. Adaptive inducing points selection for gaussian processes. *arXiv preprint arXiv:2107.10066*, 2021.

Nicolás García Trillos, Moritz Gerlach, Matthias Hein, and Dejan Slepčev. Error estimates for spectral convergence of the graph laplacian on random geometric graphs toward the laplace–beltrami operator. *Foundations of Computational Mathematics*, 20(4):827–887, 2020.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Junhui He, Guoxuan Ma, Jian Kang, and Ying Yang. Scalable bayesian inference for heat kernel gaussian processes on manifolds. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 88(2):516–539, 2026.

Elton P Hsu. A brief introduction to Brownian motion on a Riemannian manifold. *Lecture Notes*, 2008.

Pei Hsu. Brownian motion and Riemannian geometry. *Contemporary Mathematics*, 73: 95–104, 1988.

Jürgen Jost. *Riemannian geometry and geometric analysis*. Springer, 2005.

Neil Lawrence. *Variational inference in probabilistic models*. PhD thesis, University of Cambridge, 2000.

Neil Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of machine learning research*, 6(Nov):1783–1816, 2005.

John M. Lee. *Introduction to smooth manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer, New York, second edition, 2013. ISBN 978-1-4419-9981-8.

John M. Lee. *Introduction to Riemannian manifolds*, volume 176 of *Graduate Texts in Mathematics*. Springer, Cham, second edition, 2018. ISBN 978-3-319-91754-2; 978-3-319-91755-9.

Didong Li, Wenpin Tang, and Sudipto Banerjee. Inference for gaussian processes with matérn covariogram on compact riemannian manifolds. *Journal of Machine Learning Research*, 24(101):1–26, 2023.

Lizhen Lin, Mu Niu, Pokman Cheung, and David B. Dunson. Extrinsic gaussian processes for regression and classification on manifolds. *Bayesian Analysis*, 14(3):887–906, 2019. doi: 10.1214/18-BA1135.

J Macqueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*, 1967.

Pablo Moreno-Muñoz, Cilie Feldager, and Søren Hauberg. Revisiting active sets for gaussian process decoders. *Advances in Neural Information Processing Systems*, 35:6603–6614, 2022.

Henry B Moss, Sebastian W Ober, and Victor Picheny. Inducing point allocation for sparse gaussian processes in high-throughput bayesian optimisation. In *International Conference on Artificial Intelligence and Statistics*, pages 5213–5230. PMLR, 2023.

Mu Niu, Pokman Cheung, Lizhen Lin, Zhenwen Dai, Neil Lawrence, and David Dunson. Intrinsic Gaussian processes on complex constrained domains. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(3):603–627, 2019.

Mu Niu, Zhenwen Dai, Pokman Cheung, and Yizhu Wang. Intrinsic gaussian process on unknown manifolds with probabilistic metrics. *Journal of Machine Learning Research*, 24(104):1–42, 2023.

Taejin Paik, Jaemin Park, and Jung Ho Park. Atlas flow: compatible local structures on the manifold. *arXiv preprint arXiv:2210.14149*, 2022.

J. Quiñonero-Candela, CE. Rasmussen, and CKI. Williams. *Approximation Methods for Gaussian Process Regression*, pages 203–223. Neural Information Processing. MIT Press, September 2007.

Tim Ramsay. Spline smoothing over difficult regions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 64(2):307–319, 2002.

Daker García Rasines and G. Alastair Young. Splitting strategies for post-selection inference. *Biometrika*, 110(3):597–614, 2023. doi: 10.1093/biomet/asac070. URL `https://academic.oup.com/biomet/article/110/3/597/6955603`.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th international conference on machine learning*, pages 833–840, 2011.

Stefan Schonsheck, Jie Chen, and Rongjie Lai. Chart auto-encoders for manifold structured data. *arXiv preprint arXiv:1912.10094*, 2019.

Stefan C Schonsheck, Scott Mahan, Timo Klock, Alexander Cloninger, and Rongjie Lai. Semi-supervised manifold learning with complexity decoupled chart autoencoders. *arXiv preprint arXiv:2208.10570*, 2022.

Gurjeet Singh, Facundo Mémoli, Gunnar E Carlsson, et al. Topological methods for the analysis of high dimensional data sets and 3d object recognition. *PBG@ Eurographics*, 2: 091–100, 2007.

Alex Smola and Peter Bartlett. Sparse greedy gaussian process regression. *Advances in neural information processing systems*, 13, 2000.

Jakob Stolberg-Larsen and Stefan Sommer. Atlas generative models and geodesic interpolation. *Image and Vision Computing*, 122:104433, 2022.

Joshua B Tenenbaum, Vin de Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

Christopher Tralie, Nathaniel Saul, and Rann Bar-On. Ripser. py: A lean persistent homology library for python. *Journal of Open Source Software*, 3(29):925, 2018.

Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

Hendrik Jacob van Veen, Nathaniel Saul, David Eargle, and Sam W. Mangham. Kepler mapper: A flexible python implementation of the mapper algorithm. *Journal of Open Source Software*, 4(42):1315, 2019. doi: 10.21105/joss.01315. URL `https://doi.org/10.21105/joss.01315`.

Adam Watkins. *Getting Started in 3D with Maya: Create a Project from Start to Finish—Model, Texture, Rig, Animate, and Render in Maya*. Routledge, 2012.

Simon N Wood. Generalized additive models. *Annual Review of Statistics and Its Application*, 12(1):497–526, 2025.

Simon N Wood, Mark V Bravington, and Sharon L Hedley. Soap film smoothing. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(5):931–955, 2008.

Yun Yang and David Dunson. Bayesian manifold regression. *The Annals of Statistics*, 44 (2):876–905, 2016.

Max Zwiessele. *Bringing models to the domain: Deploying gaussian processes in the biological sciences*. PhD thesis, University of Sheffield, 2017.