Generative Modeling on Manifolds Through Mixture of Riemannian Diffusion Processes

Jaehyeong Jo [1] **Sung Ju Hwang** [1 2]

Abstract

Learning the distribution of data on Riemannian manifolds is crucial for modeling data from non-Euclidean space, which is required by many applications in diverse scientific fields. Yet, existing generative models on manifolds suffer from expensive divergence computation or rely on approximations of heat kernel. These limitations restrict their applicability to simple geometries and hinder scalability to high dimensions. In this work, we introduce the Riemannian Diffusion Mixture, a principled framework for building a generative diffusion process on manifolds. Instead of following the denoising approach of previous diffusion models, we construct a diffusion process using a mixture of bridge processes derived on general manifolds without requiring heat kernel estimations. We develop a geometric understanding of the mixture process, deriving the drift as a weighted mean of tangent directions to the data points that guides the process toward the data distribution. We further propose a scalable training objective for learning the mixture process that readily applies to general manifolds. Our method achieves superior performance on diverse manifolds with dramatically reduced number of in-training simulation steps for general manifolds.[1]

1. Introduction

Deep generative models have shown great success in learning the distributions of the data represented in Euclidean space, e.g., images and text. While the focus of the previous works has been biased toward data in the Euclidean space, modeling the distribution of the data that naturally resides in manifolds with specific geometry has been underexplored, while they are required for wide application: For example, the earth and climate science data (Karpatne et al., 2018; Mathieu & Nickel, 2020) lives in the sphere, whereas protein structures (Jumper et al., 2021; Watson et al., 2022) and robotic movements (Simeonov et al., 2022) are best represented by the group SE(3), and 3D computer graphics shapes (Hoppe et al., 1992) can be identified as a general closed manifold. However, previous generative methods are ill-suited for modeling these data as they do not take into consideration the specific geometry describing the data space and may assign a non-zero probability to regions outside the desired space.

Recent works (Bortoli et al., 2022; Huang et al., 2022) extend the diffusion generative framework to the Riemannian manifolds that learn to reverse the noising process, similar to Euclidean diffusion models. Although diffusion models have been shown to successfully model the distribution on simple manifolds, e.g., sphere and torus, they have difficulty in training since the score matching objective either relies on an imprecise approximation of the intractable heat kernel that degrades the performance or requires the computation of the divergence which is computationally expensive and scales poorly to high dimensions. In addition, previous diffusion models are geometrically not intuitive as their generative processes are parameterized by the score function which does not provide explicit geometric interpretation.

On the other hand, continuous normalizing flow (CNF) models on manifold (Mathieu & Nickel, 2020; Rozen et al., 2021; Ben-Hamu et al., 2022; Chen & Lipman, 2024) aim to learn the continuous-time flow by parameterizing the vector field. While CNF models leverage deterministic processes and alleviate the challenges of leveraging Brownian motion, most of the CNF models require computation of divergence during training that cannot even scale to moderately high dimensions and further cannot be readily adapted to general geometries. Even though several works (Rozen et al., 2021; Ben-Hamu et al., 2022; Chen & Lipman, 2024) proposed simulation-free methods on simple manifolds, they still require in-training simulation for general manifolds which necessitates a large number of steps to obtain accurate trajectories for the deterministic process.

[1]Korea Advanced Institute of Science and Technology (KAIST) [2]DeepAuto.ai. Correspondence to: Jaehyeong Jo <harryjo97@kaist.ac.kr>, Sung Ju Hwang <sjhwang82@kaist.ac.kr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]Code: github.com/harryjo97/riemannian-diffusion-mixture

In this work, we present Riemannian Diffusion Mixture, a novel generation framework for learning a diffusion process on Riemannian manifolds based on a geometric perspective. We build upon the diffusion mixture representation (Peluchetti, 2021; Liu et al., 2023), constructing a diffusion process directly on the manifold as a mixture of bridge processes, i.e., diffusion process conditioned to endpoints, without the need for heat kernel estimation. We show that by designing the drift of a diffusion process as a weighted mean of tangent vectors to the data points, the resulting process is guided to the data distribution and yields a prediction of the final result as the most probable endpoint. We further derive a scalable training objective, namely the two-way bridge matching, based on simple regression on the drifts of the diffusion processes that do not require computing divergence. We establish a theoretical background for the diffusion mixture framework on the Riemannian setting that is readily applicable to general manifolds and show that the previous CNF model is a special case of our framework.

We experimentally validate our approach on diverse manifolds of both real-world and synthetic datasets, on which our method outperforms or is on par with the state-of-the-art baselines. We demonstrate that ours can scale to high dimensions while allowing significantly faster training compared to the previous diffusion models relying on score matching. Especially on general manifolds, our method shows superior performance with dramatically reduced in-training simulation steps, using only 5% of the steps compared to CNF model. We summarize our main contributions as follows:

- We propose a principled framework for building generation processes on general manifolds as a mixture of bridge processes that does not require estimation of heat kernel.
- We present a geometric design for the drift of the diffusion process as a weighted mean of tangent directions on manifolds that guides the process to the target distribution, and introduce an efficient training objective readily applicable to general manifolds.
- Our method achieves superior performance on diverse manifolds, and we empirically show that ours can scale to higher dimensions with significantly faster training compared to previous diffusion models.
- Especially on general manifolds, our approach outperforms CNF model with greatly reduced in-training simulation steps, demonstrating the necessity of stochasticity.

2. Background

In this section, we introduce basic concepts of Riemannian manifolds and diffusion processes defined on manifolds.

Riemannian Manifold We consider complete, orientable, connected, and boundaryless Riemannian manifolds $\mathcal{M}$ equipped with Riemannian metric g that defines the inner product of tangent vectors. $T_x\mathcal{M}$ denotes the tangent space at point $x \in \mathcal{M}$ and $\|\eta\|_{\mathcal{M}}$ denotes the norm of the tangent vector $\eta \in T_x\mathcal{M}$. For smooth function $f : \mathcal{M} \to \mathbb{R}$, $\nabla f(x) \in T_x\mathcal{M}$ denotes the Riemannian gradient, $\mathrm{div}(v)$ denotes the Riemannian divergence for the smooth vector field $v : \mathcal{M} \to T_x\mathcal{M}$, and $\Delta_{\mathcal{M}}$ denotes the Laplace-Beltrami operator defined by $\Delta f = \mathrm{div}(\nabla f)$. $\exp_x : T_x\mathcal{M} \to \mathcal{M}$ and $\exp_x^{-1} : \mathcal{M} \to T_x\mathcal{M}$ denotes the Riemannian exponential and logarithm map, respectively. Lastly, dvol_x denotes the volume form on the manifold, and $\int f(x)\mathrm{dvol}_x$ is the integration of function f on the manifold.

Diffusion Process on Riemannian Manifold Brownian motion on a Riemannian Manifold $\mathcal{M}$ is a diffusion process generated by $\Delta_{\mathcal{M}}/2$ (Hsu, 2002) which is a generalization of the Euclidean Brownian motion. The transition distribution of the Brownian motion corresponds to the heat kernel, i.e. the solution to the heat equation, which coincides with the Gaussian distribution when $\mathcal{M}$ is a Euclidean space. One can construct a diffusion process that converges to a stationary distribution described by the Langevin dynamics:

$$\mathrm{d}\boldsymbol{X}_t = -\frac{1}{2}\nabla_{\boldsymbol{X}_t} U(\boldsymbol{X}_t)\mathrm{d}t + \mathrm{d}\mathbf{B}_t^{\mathcal{M}}, \tag{1}$$

where $\mathbf{B}_t^{\mathcal{M}}$ denotes the Brownian motion defined on $\mathcal{M}$, such that the terminal distribution satisfies $\mathrm{d}p(x)/\mathrm{dvol}_x \propto e^{-U(x)}$ (Durmus, 2016) for a potential function U which we describe in detail in Appendix A.1. A diffusion process on the manifold can be simulated using the Geodesic Random Walk (Jørgensen, 1975; Bortoli et al., 2022) which corresponds to taking a small step on the tangent space in the direction of the drift.

3. Riemannian Diffusion Mixture

We now present Riemannian Diffusion Mixture, a new framework for learning a generative diffusion process on Riemannian manifolds using a mixture of bridge processes.

3.1. Bridge Processes on Manifold

The first step of constructing the generative process is designing a diffusion process conditioned to fixed endpoints, i.e. the bridge process. In contrast to the Euclidean space which is equipped with simple families of bridge processes derived from the Brownian motion or the Ornstein-Uhlenbeck process (Peluchetti, 2023; Jo et al., 2024), designing a bridge process on general manifolds is challenging since the transition density of the Brownian motion is intractable in general.

To achieve a simple bridge process that can be used for building a generative model, we start with the Brownian bridge on the manifold $\mathcal{M}$ with fixed endpoints, modeled

by the following SDE (see Appendix A.1 for details of the Brownian bridge):

$$\mathrm{d}\boldsymbol{X}_t = \nabla_{\boldsymbol{X}_t} \log p_{\mathcal{M}}(\boldsymbol{X}_t, z, T-t)\mathrm{d}t + \mathrm{d}\mathbf{B}_t^{\mathcal{M}}, \quad (2)$$

for $\boldsymbol{X}_0 = z_0$ where $p_{\mathcal{M}}$ denotes the heat kernel on $\mathcal{M}$ and z denotes the fixed endpoint. We cannot directly use this Brownian bridge process as the heat kernel is known in very limited cases and even on a simple manifold such as a sphere, the heat kernel is represented as an infinite sum (Tulovsky & Papiez, 2001). Thereby we explore a new family of bridge processes that do not require the heat kernel.

Intuitively, a diffusion process that takes each step in the direction of the endpoint should carry the process toward the desired endpoint regardless of the prior distribution. The natural choice for this direction would be following the shortest path between the current state and the endpoint on the manifold, which corresponds to the inverse of the exponential map[2], i.e., the logarithm map. The logarithm map provides a simple approach to represent a tangent vector that heads toward the desired endpoint, as illustrated in Figure 1 by the blue vectors pointing to the endpoints.

From this observation, we introduce a simple family of bridge processes on manifolds derived from the logarithm map, namely the *Logarithm Bridge Process* $\mathbb{Q}_{log}^{x,z}$:

$$\mathrm{d}\boldsymbol{X}_t = \frac{\sigma_t^2}{\tau_T - \tau_t} \exp_{\boldsymbol{X}_t}^{-1}(z)\mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \quad (3)$$

for $\boldsymbol{X}_0 = x$, where z is the fixed endpoint, $\exp^{-1}$ is the logarithm map, σ_t is the time-dependent noise schedule that uniquely determines the process, and $\tau(t) := \int_0^t \sigma_s^2 \mathrm{d}s$ denotes the rescaled time with respect to σ_t. A key observation is that the logarithm map $\exp^{-1}$ represents the direction of the shortest path between the current state $\boldsymbol{X}_t$ and the endpoint z. As the magnitude of the drift increases to infinity with a rate $\sigma_t^2/(\tau_T - \tau_t)$ as $t \to T$, the process is forced to converge to z by the direction of the drift.

By leveraging the short-time asymptotic behavior of the Brownian motion and the Girsanov theorem, we theoretically derive in Appendix A.2 that the Logarithm bridge exhibits a similar convergence behavior as the Brownian bridge, regardless of the prior distribution Γ. In particular, when $\mathcal{M}$ is a Euclidean space $\mathbb{R}^d$, the logarithm bridge process reduces to the well-known Euclidean Brownian bridge process. But for general manifolds, our Logarithm bridge process differs from the Brownian bridge process of Eq. (2) due to the difference in the drifts.

Although the Logarithm bridge provides a simple solution for constructing the generative process on manifolds, the logarithm map of general manifolds may not be given in closed form and could be costly to compute on the fly. We can bypass the difficulty by taking a different perspective for defining the direction toward the endpoint on the manifold. Specifically, inspired by Chen & Lipman (2024), we consider a path on the manifold that minimizes the spectral distance $d_w(\cdot,\cdot)$, which is defined by the eigenvalues λ_i and eigenfunctions ϕ_i of the Laplace-Beltrami operator $\Delta_{\mathcal{M}}$:

$$d_w(x,y)^2 = \sum_{i=1}^{\infty} w(\lambda_i)\big(\phi_i(x) - \phi_i(y)\big)^2, \quad (4)$$

where w is a monotonically decreasing function. From the fact that $\nabla d_w(\cdot, z)^2$ describes the tangent vector at the current state with the direction that minimizes the spectral distance to the endpoint z, we introduce a new family of bridge processes, namely *Spectral Bridge Process* $\mathbb{Q}_{spec}^{x,z}$:

$$\mathrm{d}\boldsymbol{X}_t = -\frac{1}{2}\frac{\sigma_t^2}{\tau_T - \tau_t}\frac{\nabla_{\boldsymbol{X}_t} d_w(\boldsymbol{X}_t, z)^2}{\|\nabla_{\boldsymbol{X}_t} d_w(\boldsymbol{X}_t, z)\|_{\mathcal{M}}^2}\mathrm{d}t + \sigma_t \mathbf{B}_t^{\mathcal{M}} \quad (5)$$

for $\boldsymbol{X}_0 = x$ where the norm of the gradient ∇d_w normalizes the magnitude of the drift. The Spectral bridge process in Eq. (5) is designed so that substituting the spectral distance d_w with the geodesic distance d_g results in the Logarithm bridge process in Eq. (3). Note that the eigenvalues and the eigenfunctions are computed only once, in advance of training our generative model, and do not require computing eigenfunctions during training.

Other choices of bridge processes on manifolds are possible, such as semi-classical Brownian bridge (Elworthy & Truman, 1981) or Fermi bridge (Thompson, 2015) which we describe in Appendix A.3. Yet we focus on the Logarithm bridge and Spectral bridge due to their simplicity and practicality for real-world problems where the most relevant manifolds either have known geodesics or eigenfunctions of the Laplace-Beltrami operator, for example, $SE(3)^N$ for protein modeling (Jumper et al., 2021), $SU(N)$ for high energy physics (Boyda et al., 2021), and product of tori for

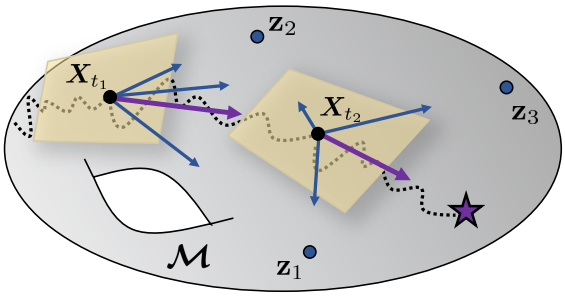


Figure 1: We construct a generative process on general manifolds as a mixture of bridge processes (Eq. (6)). The drift of the mixture process (**purple vector**) corresponds to the weighted mean of the tangent vectors pointing to the directions of the endpoints (**blue vector**), guiding the diffusion process (black dotted) to the data distribution.

[2]Here we assume that the endpoint is not in the cut locus of the current state for the inverse to be well-defined.

molecular conformed generation (Jing et al., 2022). We further note that one can consider using the Brownian bridge (Eq. (2)) with appropriate estimation of the heat kernel, e.g., Varadhan approximation (Bortoli et al., 2022) or eigenfunction expansion on maximum torus (Lou et al., 2023).

3.2. Generative Process on Riemannian Manifold

Riemannian Diffusion Mixture Having the bridge processes at hand, we now build a diffusion process on manifolds that transports a prior distribution Γ to the data distribution Π. Extending the diffusion mixture framework (Peluchetti, 2021; Liu et al., 2023) to the Riemannian setting, we construct a generative process on the manifold $\mathcal{M}$ by mixing a collection of bridge processes on $\mathcal{M}$, denoted as $\{\mathbb{Q}^{x,z} : x \sim \Gamma, z \sim \Pi\}$.

We derive a diffusion process that admits a marginal density p_t which is equal to the mixture of marginal densities $p_t^{x,z}$ of $\mathbb{Q}^{x,z}$, that is modeled by the following SDE (we provide a formal definition and a detailed derivation in Appendix A.4):

$$\mathrm{d}\boldsymbol{X}_t = \left[\int \eta^z(\boldsymbol{X}_t, t) \frac{p_t^z(\boldsymbol{X}_t)}{p_t(\boldsymbol{X}_t)} \Pi(\mathrm{dvol}_z)\right] \mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \quad (6)$$

for $X_0 \sim \Gamma$ where η^z denotes the drift of $\mathbb{Q}^{x,z}$, $p_t^z(\cdot) = \int p_t^{x,z}(\cdot)\Gamma(\mathrm{dvol}_x)$, and $p_t(\cdot) = \int p_t^z(\cdot)\Pi(\mathrm{dvol}_z)$.

From the geometrical viewpoint, the drift of the mixture process (Figure 1 purple) corresponds to the weighted mean of tangent vectors (Figure 1 blue) heading in the direction of the points in the data distribution. Therefore, simulating the mixture process can be interpreted as taking a small step in the direction of the most likely endpoint of the process. From this perspective, we can derive an explicit prediction from the mixture process, in particular, from the mixture of Logarithm bridges by projecting the drift onto the manifold along the geodesic using the exponential map as follows:

$$\hat{\boldsymbol{X}}_t := \exp_{\boldsymbol{X}_t}\left(\int \exp_{\boldsymbol{X}_t}^{-1}(z) \frac{p_t^z(\boldsymbol{X}_t)}{p_t(\boldsymbol{X}_t)} \Pi(\mathrm{dvol}_z)\right), \quad (7)$$

which corresponds to the most probable endpoint of the mixture process given the current state. This differentiates our method from the previous diffusion models that do not admit straightforward predictions for their denoising process on non-Euclidean manifolds.

Notably, by the construction of the mixture process, its terminal distribution is guaranteed to be equal to the data distribution Π regardless of the initial distribution Γ, and thereby our framework can be trivially applied to an arbitrary initial distribution. This is not the case for previous diffusion models (Bortoli et al., 2022; Huang et al., 2022) which require careful design of the potential $U(\cdot)$ in the noising process (Eq. (1)), since they rely on the denoising diffusion framework, in contrast to our bridge mixture construction. Our framework yields freedom for the choice of the noise schedule σ_t in Eq. (6) where σ_t need not be decreasing or be large for small t.

When the mixture consists of the Logarithm bridges or the Spectral bridges, we refer to the mixture processes as the *Logarithm Bridge Mixture* (LogBM) and *Spectral Bridge Mixture* (SpecBM), respectively. Note that our LogBM generalizes previous diffusion mixture framework (Peluchetti, 2021; Liu et al., 2023) since the Logarithm bridge recovers the Brownian bridge for the Euclidean space.

Probability Flow ODE For a mixture process $\mathbb{Q}_f$, there exists a deterministic process that admits the same marginal densities, i.e., the probability flow (Maoutsa et al., 2020; Song et al., 2021). The time-reversed process $\mathbb{Q}_b$ of $\mathbb{Q}_f$ is also a mixture process built from the collection of time-reversed bridge processes, which can be derived from $\mathbb{Q}_f$ in terms of the score function (Eq. (43)). As a result, the probability flow associated with $\mathbb{Q}_f$ satisfies the following ODE (see Appendix A.5 for detailed derivation):

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{Y}_t = \frac{1}{2}\Big(\eta_f(\boldsymbol{Y}_t, t) - \eta_b(\boldsymbol{Y}_t, T-t)\Big), \quad \boldsymbol{Y}_0 \sim \Gamma, \quad (8)$$

where η_f and η_b denote the drift of $\mathbb{Q}_f$ and $\mathbb{Q}_b$, respectively, and the likelihood of the probability flow as follows:

$$\log p_T(\boldsymbol{Y}_T) - \log p_0(\boldsymbol{Y}_0) = \frac{1}{2}\int_0^T \mathrm{div}\Big(\eta_f(\boldsymbol{Y}_t, t) - \eta_b(\boldsymbol{Y}_t, T-t)\Big)\mathrm{d}t. \quad (9)$$

We further discuss in Appendix A.5 that the probability flow derived from our mixture process is different from the continuous flows used in previous CNF models.

3.3. Two-way Bridge Matching

Now we show how to train a generative model that approximates the mixture process in Eq. (6). We parameterize the drifts of the mixture process and its time-reversed process with neural networks, i.e., $\boldsymbol{s}_f^\theta(z,t) \approx \eta_f(z,t)$ and $\boldsymbol{s}_b^\phi(z,t) \approx \eta_b(z,t)$. However, the drifts of the mixture processes cannot be directly approximated since we do not have access to the integration in Eq. (6). In what follows, we derive a simple and efficient training objective that is applicable to general manifolds without computing the Riemannian divergence.

For a diffusion process $\mathbb{Q} : \mathrm{d}\boldsymbol{Z}_t = \eta(\boldsymbol{Z}_t, t)\mathrm{d}t + \nu_t \mathrm{d}B_t^{\mathcal{M}}$ and its parameterized process $\mathbb{P}^\psi : \mathrm{d}\boldsymbol{Z}_t = s^\psi(\boldsymbol{Z}_t, t)\mathrm{d}t + \nu_t \mathrm{d}B_t^{\mathcal{M}}$, the KL divergence between two processes can be obtained from the Girsanov theorem as follows (we provide detailed derivation in Appendix A.6):

$$D_{KL}(\mathbb{Q}_T \| \mathbb{P}_T^\psi) \leq D_{KL}(\mathbb{Q} \| \mathbb{P}^\psi) \quad (10)$$
$$= \mathbb{E}_{\substack{z\sim\mathbb{Q}_T \\ \boldsymbol{Z}\sim\mathbb{Q}^z}}\left[\frac{1}{2}\int_0^T \left\|\nu_t^{-1}\Big(\boldsymbol{s}^\psi(\boldsymbol{Z}_t, t) - \eta^z(\boldsymbol{Z}_t, t)\Big)\right\|_{\mathcal{M}}^2 \mathrm{d}t\right] + C$$

where $\mathbb{Q}_T$ and $\mathbb{P}_T^{\psi}$ denotes the terminal distributions of $\mathbb{Q}$ and $\mathbb{P}^{\psi}$ respectively, $\mathbb{Q}^z$ and η^z denotes the process $\mathbb{Q}(\cdot|\boldsymbol{Z}_T = z)$ and its drift, and C is a constant.

Although we can directly use Eq. (10) to train s_f^{θ} and s_b^{ϕ}, it is computationally expensive as the samples $\boldsymbol{Z}_t$ should be obtained through a simulation of bridge processes. This is because the transition density of the Brownian motion is not accessible for general manifolds. Especially, simulating the bridge process requires a large number of discretized steps due to the exploding magnitude of the drift of the bridge process near the terminal time, i.e., magnitude of the drift explodes with a rate $\sigma_t^2/(\tau_T - \tau_t)$ as $t \to T$.

We address this issue by proposing an efficient training scheme, which we refer to as the *two-way bridge matching*. The main idea is to exploit the fact that (1) the simulation of the bridge process can be performed from both forward and backward directions, which can bypass the explosion of drift that allows larger step size, and (2) $\boldsymbol{Z}_t$ can be obtained from a single bridge process with fixed endpoints instead of simulating two different bridge processes, reducing the computational cost in half. Altogether, the two-way bridge matching is formalized as follows:

$$\begin{aligned} &\mathbb{E}_{\substack{(x,y)\sim(\Pi,\Gamma),\\ t\sim[0,T]}} \mathbb{E}_{\boldsymbol{Z}_t\sim\mathbb{Q}^{x,y}}\ \sigma_t^{-2}\left[F_f^{\theta,x}(\boldsymbol{Z}_t,t) + F_b^{\phi,y}(\boldsymbol{Z}_t,t)\right] \\ &F_f^{\theta,x}(\boldsymbol{Z}_t,t) = \left\|\boldsymbol{s}_f^{\theta}(\boldsymbol{Z}_t,t) - \eta_f^x(\boldsymbol{Z}_t,t)\right\|_{\mathcal{M}}^2, \\ &F_b^{\phi,y}(\boldsymbol{Z}_t,t) = \left\|\boldsymbol{s}_b^{\phi}(\boldsymbol{Z}_t,T-t) - \eta_b^y(\boldsymbol{Z}_t,T-t)\right\|_{\mathcal{M}}^2, \end{aligned} \tag{11}$$

where $\mathbb{Q}^{x,y}$ denotes the bridge process with fixed starting point x and endpoint y, and $\boldsymbol{Z}_t$ is obtained in a two-way approach, i.e., simulating $\mathbb{Q}^{x,y}$ from time 0 to t if $t < T/2$, and otherwise simulating from time T to t. Notably, from the result of Eq. (10), minimizing Eq. (11) guarantees to minimize the KL divergence between data distribution and terminal distribution of our parameterized mixture process.

However, we empirically observe that using Eq. (11) introduces high variance during training due to the coefficient σ_t^{-2}. Therefore, we present an equivalent objective that enables stable training by leveraging importance sampling, where we use a proposal distribution $q(t) \propto \sigma_t^{-2}$ to adjust the weighting as follows:

$$\mathbb{E}_{\substack{(x,y)\sim(\Pi,\Gamma),\\ t\sim q}} \mathbb{E}_{\boldsymbol{Z}_t\sim\mathbb{Q}^{x,y}}\left[F_f^{\theta,x}(\boldsymbol{Z}_t,t) + F_b^{\phi,y}(\boldsymbol{Z}_t,t)\right], \tag{12}$$

which we refer to as the *time-scaled* two-way bridge matching. During training with Eq. (12), we first sample $t \sim q$ and $(x, y) \sim (\Pi, \Gamma)$, then simulate $\boldsymbol{Z}_t \sim \mathbb{Q}^{x,y}$ using the two-way approach, where different triplets (t, x, y) are sampled to compute the expectation. We summarize the training process of our two-way bridge matching in Algorithm 1.

Algorithm 1 Two-way bridge matching

Input: Training set $\mathcal{D}$, prior distribution Γ, trained neural networks $\boldsymbol{s}_f^{\theta}$ and $\boldsymbol{s}_b^{\phi}$, terminal time T, number of in-training simulation step N

For each epoch:

Sample $x \sim \mathcal{D}$, $y \sim \Gamma$, and $t \sim q$
$(z_0, z_f, \eta) \leftarrow (y, x, \eta_f)$ if $t < T/2$ else (x, y, η_b)
$\mathrm{d}t \leftarrow t/N$
$z \leftarrow z_0$, $s \leftarrow 0$, $\mathrm{d}t \leftarrow t/N$
for $n = 1$ **to** N **do** ▷ In-training simulation of $\boldsymbol{Z}_t$
  $W \sim \mathcal{N}(0, \mathrm{Id})$ ▷ Random normal in $T_z\mathcal{M}$
  $v \leftarrow \eta^{z_f}(z, s)\mathrm{d}t + \sigma_t W\sqrt{\mathrm{d}t}$
  $z \leftarrow \exp_z v$ ▷ Geodesic step in direction of v
  $s \leftarrow s + \mathrm{d}t$
end for
$\mathcal{L}_{\theta,\phi} \leftarrow F_f^{\theta,x}(z,t) + F_b^{\phi,y}(z,t)$ ▷ Eq. (12)
Update θ, ϕ using $\mathcal{L}_{\theta,\phi}$

In particular, our two-way bridge matching consists of a simple regression on the drifts of bridge processes, i.e., $F_f^{\theta,x}$ and $F_b^{\phi,y}$, without computation of divergence or any approximation. Thereby, on manifolds for modeling real-world problems, our framework can scale to high dimensions, which previous generative models cannot scale to.

We note that our time-scaled two-way bridge matching differs from the Flow Matching objective which regresses the conditional vector field over uniformly distributed time. We experimentally validate the importance of the time distribution q in Section 5.5, where using a uniform time distribution results in a significant drop in performance compared to using $t \sim q$. This is because Eq. (12) guarantees to maximize the likelihood of our generative model, whereas it is not true for a simple regression over uniformly distributed time.

We empirically validate that our two-way approach can obtain accurate trajectories with significantly reduced simulation steps compared to the one-way simulation, resulting in up to $\times 34.9$ speed up for training. Furthermore, we show in Section 5 that the in-training simulation is not a significant overhead during training since the two-way approach greatly reduces the number of simulation steps without sacrificing the accuracy, which is significantly faster than the implicit score matching used for previous diffusion models.

Connection with Riemannian Flow Matching Especially, in the case when the noise schedule is set to be very small, i.e., $\sigma_t \to 0$, we can recover the deterministic flow of Riemannian Flow Matching (RFM) (Chen & Lipman, 2024) from our mixture process, where the bridge processes with $\sigma_t \to 0$ correspond to the conditional vector fields of Flow Matching. Thereby RFM can be considered a special case of our framework when the randomness is removed.

However, stochasticity is crucial for learning the density on manifolds with non-trivial curvature. While obtaining a trajectory of the probability path for RFM during training requires a large number of simulation steps, we can obtain trajectories from the mixture process with only a few simulation steps thanks to its stochastic nature, which we empirically show in Section 5.5. The existence of stochasticity dramatically reduces the number of in-training simulation steps compared to RFM, achieving $\times 12.8$ speed up in training without sacrificing the performance. We demonstrate in Figures 3 and 9 that our method is able to model complex distribution on the manifold with only a few in-training simulation steps, whereas RFM completely fails in such a setting. Furthermore, we can leverage the Girsanov theorem to derive that our training objective in Eq. (12) is guaranteed to minimize the KL divergence between the data distribution and the terminal distribution of our parameterized process, which does not apply to RFM as it is based on a deterministic process.

Sampling Generating samples with the Riemannian diffusion mixture can be achieved in two different ways: (1) simulating the approximation of the mixture process (Eq. (6)) and (2) simulating the probability flow (Eq. (8)). First, the mixture process can be approximated by using the drift estimation $\boldsymbol{s}_f^\theta$ as follows:

$$\mathrm{d}\boldsymbol{X}_t = \boldsymbol{s}_f^\theta(\boldsymbol{X}_t, t)\mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}}. \tag{13}$$

which can be simulated using the Geodesic Random Walk (Jørgensen, 1975; Bortoli et al., 2022). Alternatively, the probability flow can be modeled by the ODE using the drift estimations $\boldsymbol{s}_f^\theta$ and $\boldsymbol{s}_b^\phi$ as follows:

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X}_t = \frac{1}{2}\left(\boldsymbol{s}_f^\theta(\boldsymbol{X}_t, t) - \boldsymbol{s}_b^\phi(\boldsymbol{X}_t, T-t)\right), \tag{14}$$

which can be solved using integrators on Riemannian manifolds (Hairer, 2011).

4. Related Work

Euclidean Diffusion Models Diffusion models (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021) model the generative process via the denoising diffusion process derived from the time-reversal of the noising process. Recent works (Peluchetti, 2021; Liu et al., 2023; Peluchetti, 2023) introduce an alternative approach for modeling the generative process without using the time-reversal, namely diffusion mixture, by building bridge processes between the initial and the terminal distributions. However, these methods are limited to Euclidean space and sub-optimal for modeling the data living on manifolds, for example, sphere for climate data and tori for biological data such as proteins. Our work shows how to extend the diffusion mixture framework to manifolds that generalize the Euclidean case.

Generative Models on Riemannian Manifolds Previous generative models (Gemici et al., 2016; Rezende et al., 2020; Bose et al., 2020) relied on projecting a Euclidean space to manifolds which is problematic since such mapping cannot be bijective, resulting in numerical instabilities. Recent works address this problem by constructing a mapping on the manifold that describes the transport from a prior distribution to the data distribution, namely the diffusion models and the CNF models.

The seminal work of Bortoli et al. (2022) extends the score-based model to the manifold, while Huang et al. (2022) introduces a variational framework for diffusion models on manifolds. However, both rely on score matching that either needs to be approximated or scales poorly to higher dimensions. Specifically, denoising score matching requires the conditional score function to be approximated which obstructs exact training. Further, implicit score matching requires the computation of the Riemannian divergence which scales poorly to high dimensions, and using the Hutchinson estimator (Hutchinson, 1989) introduces high variance in training. In contrast, our framework provides efficient and scalable training that does not require divergence and does not rely on approximations of the heat kernel. Since the construction of the mixture process guarantees convergence to the data distribution regardless of the prior distribution, our method can be readily extended for arbitrary prior distribution. We provide discussion on Diffusion Schrödinger Bridge (Thornton et al., 2022), improvement of Riemannian diffusion models (Lou et al., 2023), and recent works focusing on specific geometries in Appendix A.7.

On the other hand, CNF models build a continuous-time flow (Chen et al., 2018; Grathwohl et al., 2019) on the manifold by parameterizing the vector field. However, previous CNF models (Lou et al., 2020; Mathieu & Nickel, 2020; Falorsi & Forré, 2020) rely on simulation-based maximum likelihood training which is computationally expensive. Recent works (Rozen et al., 2021; Ben-Hamu et al., 2022) introduce simulation-free training methods on simple geometries, but they scale poorly to high-dimension due to the computation of the divergence and further cannot be adapted to non-simple geometries. Chen & Lipman (2024) extends the Flow Matching framework (Lipman et al., 2023) to manifolds which learns the probability path by regressing the conditional vector fields. Instead of the deterministic flow, our work constructs a diffusion-based generative process for which stochasticity is crucial for learning on general geometries, as it achieves superior performance with greatly reduced in-training simulation steps.

5. Experiments

We experimentally validate our method on diverse datasets including real-world benchmarks as well as synthetic dis-

Table 1: **Test NLL results on earth and climate science datasets**. We report the mean of 5 different runs with different data splits. Best performance and its comparable results ($p > 0.05$) from the t-test are highlighted.

Dataset size	Volcano 827	Earthquake 6120	Flood 4875	Fire 12809
RCNF (Mathieu & Nickel, 2020)	-6.05 ± 0.61	0.14 ± 0.23	1.11 ± 0.19	-0.80 ± 0.54
Moser Flow (Rozen et al., 2021)	-4.21 ± 0.17	-0.16 ± 0.06	0.57 ± 0.10	-1.28 ± 0.05
CNFM (Ben-Hamu et al., 2022)	-2.38 ± 0.17	**-0.38** ± 0.01	**0.25** ± 0.02	-1.40 ± 0.02
RFM (Chen & Lipman, 2024)	-7.93 ± 1.67	-0.28 ± 0.08	0.42 ± 0.05	-1.86 ± 0.11
StereoSGM (Bortoli et al., 2022)	-3.80 ± 0.27	-0.19 ± 0.05	0.59 ± 0.07	-1.28 ± 0.12
RSGM (Bortoli et al., 2022)	-4.92 ± 0.25	-0.19 ± 0.07	0.45 ± 0.17	-1.33 ± 0.06
RDM (Huang et al., 2022)	-6.61 ± 0.96	**-0.40** ± 0.05	0.43 ± 0.07	-1.38 ± 0.05
RSGM-improved (Lou et al., 2023)	-4.69 ± 0.29	-0.27 ± 0.05	0.44 ± 0.03	-1.51 ± 0.13
Ours (LogBM)	**-9.52** ± 0.87	-0.30 ± 0.06	0.42 ± 0.08	**-2.47** ± 0.11

Dataset size	Glycine (2D) 13283	Proline (2D) 7634	RNA (7D) 9478
MoPS (De Cao & Aziz, 2020)	2.08 ± 0.009	0.27 ± 0.008	4.08 ± 0.368
RDM (Huang et al., 2022)	1.97 ± 0.012	**0.12** ± 0.011	-3.70 ± 0.592
RFM (Chen & Lipman, 2024)	**1.90** ± 0.055	0.15 ± 0.027	**-5.20** ± 0.067
Ours (LogBM)	**1.89** ± 0.056	**0.14** ± 0.027	**-5.27** ± 0.090

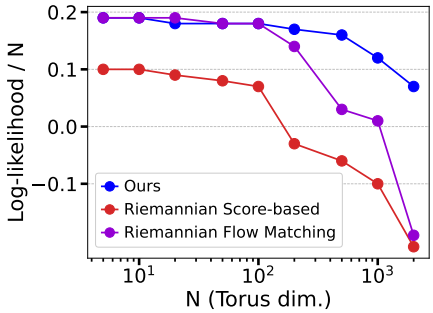


Figure 2: **(Left) Test NLL results on protein datasets**. Best performance and its comparable results ($p > 0.05$) from the t-test are highlighted in bold. **(Right) Comparison on high-dimensional tori**. We compare the log-likelihood in bits against RSGM and RFM where the results are obtained by running the open-source codes.

tributions. We follow the experimental settings of previous works (Bortoli et al., 2022; Chen & Lipman, 2024) where we provide the details of the training setup in Appendix B. We compare our method against generative models on manifolds: **RCNF** (Mathieu & Nickel, 2020), **Moser Flow** (Rozen et al., 2021), **CNFM** (Ben-Hamu et al., 2022) and **RFM** (Chen & Lipman, 2024) are CNF models, **StereoSGM** (Bortoli et al., 2022) is a Euclidean score-based model using stereographic projection, **RSGM** (Bortoli et al., 2022) is a Riemannian score-based model, **RDM** (Huang et al., 2022) is a Riemannian diffusion model based on a variational framework, and **RSGM-improved** (Lou et al., 2023) uses improved heat kernel estimator for RSGM.

5.1. Earth and Climate Science Datasets

We first evaluate the generative models on real-world datasets living on the 2-dimensional sphere, which consists of earth and climate science events including volcanic eruptions (NOAA, 2020b), earthquakes (NOAA, 2020a), floods (Brakenridge, 2017), and wild fires (EOSDIS, 2020). We use the LogBM, i.e. a mixture of Logarithm bridges, as the logarithm map is easy to compute on the sphere.

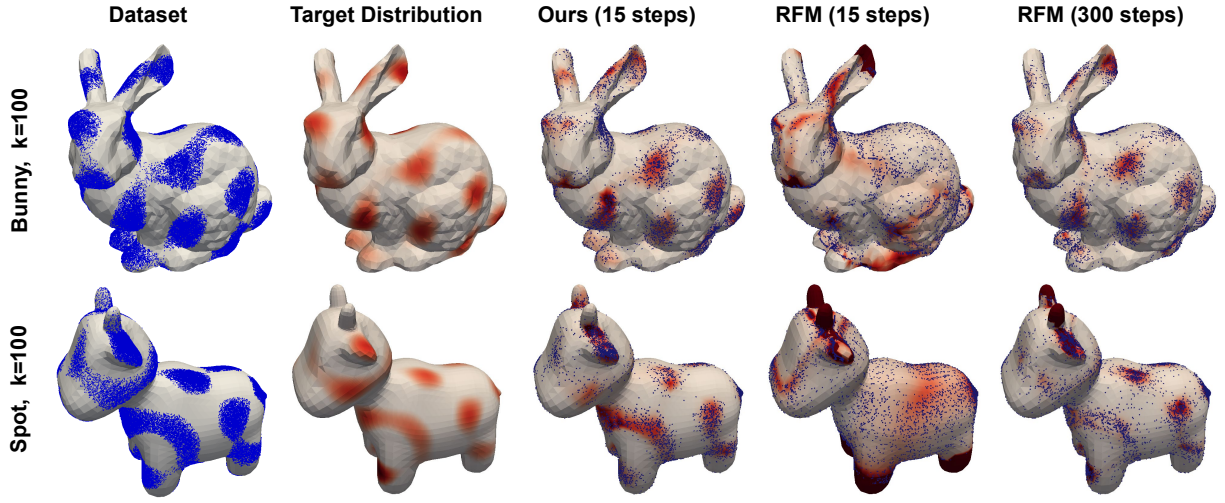
Table 1 demonstrates that our method significantly outperforms the baselines on the Volcano and the Fire datasets which require high fidelity as the data are concentrated in specific regions. Ours constantly outperforms the Riemannian score-based model while matching the performance of RFM on the Earthquake and the Flood dataset. We visualize the generated samples and learned densities of our model in Figure 4 showing that our method is capable of capturing the distribution on the sphere.

We further compare the convergence of the generative processes measured by the geodesic distance in Figure 5 where our generative process converges faster than that of the baselines. We observe that the prediction from our model (Eq. (7)) converges faster than that of RFM, verifying that ours is able to make more accurate predictions throughout the generation process.

5.2. Protein Datasets

We further experiment on protein datasets represented on n-dimensional torus from the torsion angles, consisting of 500 high-resolution proteins (Lovell et al., 2003) and 113 selected RNA sequences (Murray et al., 2003) preprocessed by Huang et al. (2022) (details in Appendix B.3). We additionally compare our method against the Mixture of Power Spherical (MoPS) (De Cao & Aziz, 2020) which models the distribution as a mixture of power spherical distributions.

Figure 3: **Visualization of the generated samples and the learned density** of our method and RFM on the mesh datasets. Blue dots represent the generated samples and darker red colors indicate higher likelihood. The numbers in the parentheses denote the number of in-training simulation steps used to train the model.



	Steps	Stanford Bunny $k=50$	Stanford Bunny $k=100$	Spot the Cow $k=50$	Spot the Cow $k=100$
RFM w/ Diff. Dist.	300	1.48 ± 0.01	1.53 ± 0.01	**0.95** ± 0.05	1.08 ± 0.05
RFM w/ Bihar. Dist.	300	1.55 ± 0.01	1.49 ± 0.01	**1.08** ± 0.05	1.29 ± 0.05
Ours w/ Diff. Dist.	15	**1.42** ± 0.01	**1.41** ± 0.00	**0.99** ± 0.03	**0.97** ± 0.03
Ours w/ Bihar. Dist.	15	1.55 ± 0.02	1.45 ± 0.01	**1.09** ± 0.06	**0.97** ± 0.02

	Torus	Spot
RSGM (ISM)	4.97	-
RSGM (SSM)	1.20	-
RFM	0.97	12.83
Ours	1.00	1.00

Table 2: **(Left) Test NLL results on mesh datasets**. We report the mean of 5 different runs. Best performance and its comparable results ($p > 0.05$) from the t-test are highlighted. **(Right) Comparison of the training time**. We report the relative training time of the baselines with respect to ours on high-dimensional torus and Spot.

We use the LogBM as the logarithm map is easy to compute on the torus. Table of Figure 2 demonstrates that ours outperforms or is on par with RDM while making marginal improvements over RFM, where the baselines are likely to be close to optimal. We provide the results of the other two protein datasets (General, Pre-Pro) in Table 3 showing comparable results with RFM. We visualize the learned density in Figure 8 using Ramachandran plots where ours models the data distribution almost perfectly.

5.3. High-Dimensional Tori

We validate the scalability of our method using the synthetic data on high-dimensional tori. We follow Bortoli et al. (2022) by creating a wrapped Gaussian distribution with a random mean and variance of 0.2 on n-dimensional tori where we compare the performance with RFM and RSGM trained via implicit score matching. To make a fair comparison, we use the same model architecture for all methods, where the total number of parameters for our models match that of the baselines. We describe the detailed setting in Appendix B.4. As shown in Figure 2 (Right), ours constantly outperforms RSGM, especially in high dimensions. RSGM scales poorly with the dimensions due to the high variance in computing the stochastic divergence of the training objective. RFM also shows a significant drop in high dimensions which implies that the vector field could not be well-approximated with a limited number of parameters. On the other hand, ours is able to scale fairly well even for high dimensions as our training objective of Eq. (12) does not require computation of divergence or any approximation. We observe that our method shows consistent performance without degradation for higher dimensions when using more parameters, scaling fairly well even to dimension 10^4.

In particular, as shown in Table 2 (Right), we achieve up to $\times 5$ speedup in training compared to RSGM that uses implicit score matching (ISM), and also significantly faster than RSGM using ISM with the stochastic estimator (SSM). Our training time is comparable to Flow Matching which is simulation-free on tori.

5.4. General Closed Manifolds

To validate our framework on general manifolds with non-trivial curvature, we evaluate modeling synthetic distribu-

tions on triangular meshes. Following Chen & Lipman (2024), we construct the target distribution on Stanford Bunny (Turk & Levoy, 1994) and Spot the Cow (Crane et al., 2013) from the k-th eigenfunction of the mesh, which we provide detail in Appendix B.5. We use SpecBM, i.e. a mixture of Spectral bridges, with using the diffusion distance (Coifman & Lafon, 2006) or the biharmonic distance (Lipman et al., 2010) for d_w. As visualized in Figure 3 and Figure 9, our method is able to fit the complex distributions using only 15 steps for the in-training simulation, while RFM completely fails when using a small number of in-training simulation steps. We show in Table 2 that our method outperforms RFM while using only 5% of in-training simulation steps, achieving $\times 12.8$ speed up in training compared to RFM.

5.5. Further Analysis

Non-Compact Manifold We further validate that our framework can be applied to non-compact manifolds, in particular for spaces of negative curvature. We experiment on the synthetic distributions on a 2-dimensional hyperboloid modeled by a mixture of wrapped Gaussian distributions. Figure 10 demonstrates that our method is capable of modeling the target distributions.

Time-scaled Training Objective We experimentally validate that the time-scaled objective of Eq. (12) is crucial for learning the distribution, by comparing ours with a variant trained with uniform time distribution similar to Flow Matching. Table 6 shows that the variant using uniform time distribution results in a significant drop in performance. This is because our time-scaled objective guarantees maximizing the likelihood of the generative model, whereas the variant using uniform time distribution does not.

Number of In-Training Simulation Steps We empirically demonstrate that our method can be trained using only 15 steps for the in-training simulation in Figures 11 and 12: We show in (a) and (c) that the trajectories of the mixture process simulated with 15 steps result in an almost similar distribution to the exact trajectories, which cannot be achieved with the one-way simulation as shown in (b). Especially, (d) demonstrates that using a small noise scale, resembling a deterministic process, requires a large number of simulation steps to obtain accurate trajectories, explaining the reason for the failure of RFM in Figure 3 and 9.

6. Conclusion

In this work, we present Riemannian Diffusion Mixture, a new approach for learning generative diffusion processes on general manifolds. We build the generation process using a mixture of bridge processes by designing the drift to be a weighted mean of tangent directions to the data distribution, which does not require approximating the heat kernel. We develop a highly scalable training scheme on general manifolds based on simple regression of the drifts which enables significantly faster training compared to previous diffusion models. Our approach shows superior performance on diverse manifolds with a dramatically reduced number of in-training simulation steps. We believe our work provides a promising direction for manifold diffusion models which could be applied to various scientific fields, for example, the design of proteins.

Impact Statement This paper presents work whose goal is to advance the field of deep generative models for data on non-Euclidean spaces. We believe that our work can enhance our understanding of diverse scientific fields including protein modeling and high-energy physics.

Acknowledgement This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075 Artificial Intelligence Graduate School Program(KAIST)), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-00713) and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2023-00256259).

References

Ben-Hamu, H., Cohen, S., Bose, J., Amos, B., Nickel, M., Grover, A., Chen, R. T. Q., and Lipman, Y. Matching normalizing flows and probability paths on manifolds. In *International Conference on Machine Learning*, 2022.

Bortoli, V. D., Mathieu, E., Hutchinson, M., Thornton, J., Teh, Y. W., and Doucet, A. Riemannian score-based generative modeling. In *Advances in Neural Information Processing Systems*, 2022.

Bose, A. J., Smofsky, A., Liao, R., Panangaden, P., and Hamilton, W. L. Latent variable modelling with hyperbolic normalizing flows. In *International Conference on Machine Learning*, 2020.

Bose, A. J., Akhound-Sadegh, T., Fatras, K., Huguet, G., Rector-Brooks, J., Liu, C.-H., Nica, A. C., Korablyov, M., Bronstein, M., and Tong, A. Se (3)-stochastic flow matching for protein backbone generation. *arXiv:2310.02391*, 2023.

Boyda, D., Kanwar, G., Racanière, S., Rezende, D. J., Albergo, M. S., Cranmer, K., Hackett, D. C., and Shanahan, P. E. Sampling using su (n) gauge equivariant flows. *Physical Review D*, 103(7):074504, 2021.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., et al. Jax: composable transformations of python+ numpy programs. 2018.

Brakenridge, G. Global active archive of large flood events. http://floodobservatory.colorado.edu/Archives/index.html, 2017. Dartmouth Flood Observatory, University of Colorado,.

Chen, R. T. and Lipman, Y. Flow matching on general geometries. In *International Conference on Learning Representations*, 2024.

Chen, T., Liu, G., and Theodorou, E. A. Likelihood training of schrödinger bridge using forward-backward sdes theory. In *International Conference on Learning Representations*, 2022.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

Coifman, R. R. and Lafon, S. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

Crane, K., Pinkall, U., and Schröder, P. Robust fairing via conformal curvature flow. *ACM Transactions on Graphics*, 32(4):1–10, 2013.

De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. Diffusion schr\"odinger bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems*, 2021.

De Cao, N. and Aziz, W. The power spherical distribution. *arXiv preprint arXiv:2006.04437*, 2020.

Dormand, J. R. and Prince, P. J. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

Driver, B. K. A cameron-martin type quasi-invariance theorem for pinned brownian motion on a compact riemannian manifold. *Transactions of the American Mathematical Society*, 342(1):375–395, 1994.

Durmus, A. *High dimensional Markov chain Monte Carlo methods: theory, methods and applications*. PhD thesis, Université Paris-Saclay (ComUE), 2016.

Elworthy, D. and Truman, A. Classical mechanics, the diffusion (heat) equation and the schrödinger equation on a riemannian manifold. *Journal of mathematical physics*, 22(10):2144–2166, 1981.

EOSDIS. Active fire data. https://earthdata.nasa.gov/earth-observation-data/near-real-time/firms/active-fire-data, 2020. Land, Atmosphere Near real-time Capability for EOS (LANCE) system operated by NASA's Earth Science Data and Information System (ESDIS).

Falorsi, L. and Forré, P. Neural ordinary differential equations on manifolds. *arXiv:2006.06663*, 2020.

Fishman, N., Klarner, L., Bortoli, V. D., Mathieu, E., and Hutchinson, M. Diffusion models for constrained domains. *arXiv:2304.05364*, 2023.

Gemici, M. C., Rezende, D., and Mohamed, S. Normalizing flows on riemannian manifolds. *arXiv:1611.02304*, 2016.

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., and Duvenaud, D. FFJORD: free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2019.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Hairer, E. Solving differential equations on manifolds. *Lecture notes*, 2011.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. Surface reconstruction from unorganized points. In *Annual conference on computer graphics and interactive techniques*, 1992.

Hsu, E. P. *Stochastic analysis on manifolds*. American Mathematical Society, 2002.

Hsu, P. Brownian bridges on riemannian manifolds. *Probability theory and related fields*, 84:103–118, 1990.

Huang, C., Aghajohari, M., Bose, J., Panangaden, P., and Courville, A. C. Riemannian diffusion models. In *Advances in Neural Information Processing Systems*, 2022.

Hutchinson, M. F. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Jing, B., Corso, G., Chang, J., Barzilay, R., and Jaakkola, T. S. Torsional diffusion for molecular conformer generation. In *Advances in Neural Information Processing Systems*, 2022.

Jo, J., Kim, D., and Hwang, S. J. Graph generation with diffusion mixture. In *International Conference on Machine Learning*, 2024.

Jørgensen, E. The central limit problem for geodesic random walks. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 32(1-2):1–64, 1975.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Karpatne, A., Ebert-Uphoff, I., Ravela, S., Babaie, H. A., and Kumar, V. Machine learning for the geosciences: Challenges and opportunities. *IEEE Transactions on Knowledge and Data Engineering*, 31(8):1544–1554, 2018.

Leach, A., Schmon, S. M., Degiacomi, M. T., and Willcocks, C. G. Denoising diffusion probabilistic models on so (3) for rotational alignment. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

Lipman, Y., Rustamov, R. M., and Funkhouser, T. A. Biharmonic distance. *ACM Transactions on Graphics*, 29(3): 1–11, 2010.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Liu, X., Wu, L., Ye, M., and Liu, Q. Learning diffusion bridges on constrained domains. In *International Conference on Learning Representations*, 2023.

Lou, A., Lim, D., Katsman, I., Huang, L., Jiang, Q., Lim, S., and Sa, C. D. Neural manifold ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2020.

Lou, A., Xu, M., Farris, A., and Ermon, S. Scaling riemannian diffusion models. In *Advances in Neural Information Processing Systems*, 2023.

Lovell, S. C., Davis, I. W., Arendall III, W. B., De Bakker, P. I., Word, J. M., Prisant, M. G., Richardson, J. S., and Richardson, D. C. Structure validation by cα geometry: ϕ, ψ and cβ deviation. *Proteins: Structure, Function, and Bioinformatics*, 50(3):437–450, 2003.

Maoutsa, D., Reich, S., and Opper, M. Interacting particle solutions of fokker–planck equations through gradient–log–density estimation. *Entropy*, 22(8):802, 2020.

Mathieu, E. and Nickel, M. Riemannian continuous normalizing flows. In *Advances in Neural Information Processing Systems*, 2020.

McCann, R. J. Polar factorization of maps on riemannian manifolds. *Geometric & Functional Analysis GAFA*, 11 (3):589–608, 2001.

Murray, L. J., Arendall III, W. B., Richardson, D. C., and Richardson, J. S. Rna backbone is rotameric. *Proceedings of the National Academy of Sciences*, 100(24):13904–13909, 2003.

NOAA. Global significant earthquake database. https://data.nodc.noaa.gov/cgi-bin/iso?id=gov.noaa.ngdc.mgg.hazards:G012153, 2020a. National Geophysical Data Center / World Data Service (NGDC/WDS): NCEI/WDS Global Significant Earthquake Database. NOAA National Centers for Environmental Information.

NOAA. Global significant volcanic eruptions database. https://data.nodc.noaa.gov/cgi-bin/iso?id=gov.noaa.ngdc.mgg.hazards:G10147, 2020b. National Geophysical Data Center / World Data Service (NGDC/WDS): NCEI/WDS Global Significant Volcanic Eruptions Database. NOAA National Centers for Environmental Information.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 2019.

Peluchetti, S. Non-denoising forward-time diffusions. *Openreview*, 2021.

Peluchetti, S. Diffusion bridge mixture transports, schrödinger bridge problems and generative modeling. *arXiv:2304.00917*, 2023.

Polyak, B. T. and Juditsky, A. B. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

Rezende, D. J., Papamakarios, G., Racanière, S., Albergo, M. S., Kanwar, G., Shanahan, P. E., and Cranmer, K. Normalizing flows on tori and spheres. In *International Conference on Machine Learning*, 2020.

Rozen, N., Grover, A., Nickel, M., and Lipman, Y. Moser flow: Divergence-based generative modeling on manifolds. *Advances in Neural Information Processing Systems*, 2021.

Shi, Y., Bortoli, V. D., Campbell, A., and Doucet, A. Diffusion schrödinger bridge matching. In *Advances in Neural Information Processing Systems*, 2023.

Simeonov, A., Du, Y., Tagliasacchi, A., Tenenbaum, J. B., Rodriguez, A., Agrawal, P., and Sitzmann, V. Neural

descriptor fields: Se (3)-equivariant object representations for manipulation. In *International Conference on Robotics and Automation*, 2022.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Thompson, J. *Submanifold bridge processes*. PhD thesis, University of Warwick, 2015.

Thompson, J. Brownian bridges to submanifolds. *Potential Analysis*, 49:555–581, 2018.

Thornton, J., Hutchinson, M., Mathieu, E., Bortoli, V. D., Teh, Y. W., and Doucet, A. Riemannian diffusion schrödinger bridge. *arXiv:2207.03024*, 2022.

Tulovsky, V. and Papiez, L. Formula for the fundamental solution of the heat equation on the sphere. *Applied mathematics letters*, 14(7):881–884, 2001.

Turk, G. and Levoy, M. Zippered polygon meshes from range images. In *Annual conference on Computer graphics and interactive techniques*, 1994.

Urain, J., Funk, N., Peters, J., and Chalvatzaki, G. Se (3)-diffusionfields: Learning smooth cost functions for joint grasp and motion optimization through diffusion. In *IEEE International Conference on Robotics and Automation*, 2023.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models. *BioRxiv*, pp. 2022–12, 2022.

Wu, L., Gong, C., Liu, X., Ye, M., and Liu, Q. Diffusion-based molecule generation with informative prior bridges. In *Advances in Neural Information Processing Systems*, 2022.

Yim, J., Trippe, B. L., Bortoli, V. D., Mathieu, E., Doucet, A., Barzilay, R., and Jaakkola, T. S. SE(3) diffusion model with application to protein backbone generation. In *International Conference on Machine Learning*, 2023.

Øksendal, B. *Stochastic Differential Equations*. Universitext. Springer Berlin Heidelberg, 2003.

Appendix

A. Derivations

A.1. Diffusion Process on Riemannian Manifold

Brownian bridge process is a diffusion process described by the Brownian motion conditioned to fixed endpoints, which is induced by the following infinitesimal generator:

$$\frac{1}{2}\Delta_{\mathcal{M}} + \nabla \log p(\cdot, z, T-t), \tag{15}$$

where p denotes the transition density of the Brownian motion, i.e. the heat kernel defined on $\mathcal{M}$, and z denotes the fixed endpoint. Thus Brownian bridge process can be modeled by the following SDE:

$$\mathbb{Q}^z_{bb} : \mathrm{d}\boldsymbol{X}_t = \nabla_{\boldsymbol{X}_t} \log p_{\mathcal{M}}(\boldsymbol{X}_t, z, T-t)\mathrm{d}t + \mathrm{d}\mathbf{B}^{\mathcal{M}}_t. \tag{16}$$

We refer the readers to Hsu (2002) for a formal definition of the Brownian bridge process. The theoretical properties of Brownian bridge have been studied in previous works (Hsu, 1990; Driver, 1994).

One can construct a diffusion process on $\mathcal{M}$ that converges to a stationary distribution described by the Langevin dynamics as follows:

$$\mathrm{d}\boldsymbol{X}_t = -\frac{1}{2}\nabla_{\boldsymbol{X}_t} U(\boldsymbol{X}_t)\mathrm{d}t + \mathrm{d}\mathbf{B}^{\mathcal{M}}_t, \tag{17}$$

where the terminal distribution satisfies $\mathrm{d}p(x)/\mathrm{dvol}_x \propto e^{-U(x)}$ (Durmus, 2016) for a potential function U. For example, $U(x) = d_g(x,\mu)^2/(2\gamma^2) + \log \|D \exp_\mu^{-1}(x)\|$ results in a stationary distribution equal to the wrapped Gaussian distribution with an arbitrary mean location $mu \in \mathcal{M}$, where d_g denotes the geodesic distance.

A.2. Logarithm Bridge Process

Here we show that the Logarithm bridge of Eq. (3) describes a diffusion process that converges to an endpoint. For the notational simplicity, we omit the subscript $\mathcal{M}$ for the heat kernel on $\mathcal{M}$. First, we derive that the following simplified process is a bridge process with an endpoint z:

$$\mathbb{Q}^z_{log} : \mathrm{d}\boldsymbol{X}_t = \frac{1}{T-t}\exp^{-1}_{\boldsymbol{X}_t}(x)\mathrm{d}t + \mathrm{d}\mathbf{B}^{\mathcal{M}}_t. \tag{18}$$

For any pair of points x and y on $\mathcal{M}$, the following short-time asymptotic of the heat kernel holds (Theorem 5.2.1 of (Hsu, 2002)):

$$\lim_{t\to 0} t \log p(x,y,t) = -\frac{d_g(x,y)^2}{2}, \tag{19}$$

where $d_g(\cdot,\cdot)$ is a geodesic distance defined on $\mathcal{M}$. Further leveraging the identity from Proposition 6 of McCann (2001), we obtain the following result:

$$\lim_{t\to 0} t\nabla_x \log p(x,y,t) = -\frac{1}{2}\nabla_x d_g(x,y)^2 = \exp^{-1}_x(y). \tag{20}$$

Furthermore, we have an upper bound for the gradient of the logarithmic heat kernel for any pair of points $x, y \in \mathcal{M}$ and $t \in [0,T]$ as follws (Theorem 5.5.3 of (Hsu, 2002)):

$$\left\|\nabla_x \log p(x,y,t)\right\|_{\mathcal{M}} \leq C\left[\frac{d_g(x,y)}{t} + \frac{1}{\sqrt{t}}\right], \tag{21}$$

where C is a constant. From Eq. (21), we have the following bound:

$$\left\| \frac{\exp_x^{-1}(z)}{T-t} - \nabla_x \log p\left(x, z, T-t\right) \right\|_{\mathcal{M}} \tag{22}$$

$$= \mathbb{1}_{\{t>T-\epsilon\}} \cdot \left\| \frac{\exp_x^{-1}(z)}{T-t} - \nabla_x \log p\left(x, z, T-t\right) \right\|_{\mathcal{M}} + \mathbb{1}_{\{t\leq T-\epsilon\}} \cdot \left\| \frac{\exp_x^{-1}(z)}{T-t} - \nabla_x \log p\left(x, z, T-t\right) \right\|_{\mathcal{M}} \tag{23}$$

$$\leq \delta(\epsilon) + \frac{1}{\epsilon} \left\| \exp_x^{-1}(z) \right\|_{\mathcal{M}} + \left\| \nabla_x \log p(x, z, T-t) \right\|_{\mathcal{M}} \tag{24}$$

$$\leq \delta(\epsilon) + \frac{1}{\epsilon} d_g(x, z) + C \left[\frac{d_g(x, z)}{\epsilon} + \frac{1}{\sqrt{\epsilon}} \right], \tag{25}$$

which holds for every $\epsilon > 0$ where $\delta(t)$ denotes the first term of Eq. (23). Since $\delta(\epsilon) \to 0$ as $\epsilon \to 0$ from the results of Eq. (20), we obtain the following bound:

$$\left\| \frac{\exp_x^{-1}(z)}{T-t} - \nabla_x \log p\left(x, z, T-t\right) \right\|_{\mathcal{M}} < \infty. \tag{26}$$

Finally, using the Girsanov theorem for $\mathbb{Q}_{bb}^z$ and $\mathbb{Q}_{log}^z$, we obtain the following result:

$$D_{KL}\left(\mathbb{Q}_{bb}^z \| \mathbb{Q}_{log}^z\right) = \frac{1}{2} \mathbb{E}_{\boldsymbol{X} \sim \mathbb{Q}_{bb}^z} \left[\int_0^T \left\| \frac{\exp_{\boldsymbol{X}_t}^{-1}(z)}{T-t} - \nabla_x \log p\left(\boldsymbol{X}_t, z, T-t\right) \right\|_{\mathcal{M}}^2 \mathrm{d}t \right] < \infty, \tag{27}$$

which implies that the Brownian bridge and the Logarithm bridge have the same support. Since the Brownian bridge converges to z by definition, we can conclude that the Logarithm bridge also converges to a fixed endpoint z.

The Logarithm bridge process presented in Eq. (3) is the result of using the change of time (Øksendal, 2003) on Eq. (18) with respect to the rescaled time $\tau(t) := \int_0^t \sigma_s^2 \mathrm{d}s$. Additionally, similar to the Euclidean bridge processes introduced in Wu et al. (2022), the derivation of our Logarithm bridge process provides a more general form of bridge processes on the manifold:

$$\mathrm{d}\boldsymbol{X}_t = \left[\frac{\sigma_t^2}{\tau_T - \tau_t} \exp_{\boldsymbol{X}_t}^{-1}(z) + \sigma_t \nabla_{\boldsymbol{X}_t} U(\boldsymbol{X}_t, t) \right] \mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}}, \tag{28}$$

for a scalar function U that satisfies $\mathbb{E}_{\boldsymbol{X} \sim \mathbb{Q}^{z,bb}} \int_0^T \|\nabla U(\boldsymbol{X}_t, t)\|_{\mathcal{M}}^2 < \infty$ which is sufficient for bounded functions U.

A.3. Other Bridge Processes

Semi-classical Brownian bridge A semi-classical Brownian bridge introduced by (Elworthy & Truman, 1981), also known as Brownian Riemannian bridge, is a bridge process defined by the infinitesimal generator

$$\frac{1}{2} \Delta_{\mathcal{M}} + \nabla \log k(\cdot, z, T-t), \tag{29}$$

where k is given by the Jacobian determinant of the exponential map at y as follows:

$$k(x, y, t) = \frac{1}{(2\pi t)^{n/2}} e^{-\frac{d_g^2(x,y)}{2t}} |\det D_{\exp_y^{-1}(x)} \exp_y|^{-1/2}. \tag{30}$$

Fermi bridge Previous works (Thompson, 2015; 2018) introduce a general family of bridge processes, namely the Fermi bridge, that describes diffusion processes conditioned to a submanifold N, which can be defined by the infinitesimal generator as follows:

$$\frac{1}{2} \Delta_{\mathcal{M}} - \frac{r_N}{T-t} \frac{\partial}{\partial r_N}, \tag{31}$$

where $r_N(\cdot) := d(\cdot, N)$ is the distance function to the submanifold N and $\partial / \partial r_N$ denotes differentiation in radial direction. Although the Logarithm bridge can be derived from the Fermi bridge by constraining N to a single point, to the best of our knowledge, the Logarithm bridge was not studied in previous literature, and further leveraging the Logarithm bridge process in the context of generative modeling is our novel contribution.

A.4. Diffusion Mixture on Riemannian Manifold

Diffusion Mixture Representation We extend the diffusion mixture representation (Peluchetti, 2021) to the Riemannian setting. We start with the statement of the diffusion mixture representation: Let $\{\mathbb{Q}^\lambda : \lambda \in \Lambda\}$ be a collection of diffusion processes on $\mathcal{M}$ modeled by the following SDEs:

$$\mathbb{Q}^\lambda : \mathrm{d}\boldsymbol{X}_t^\lambda = \eta^\lambda(\boldsymbol{X}_t^\lambda, t)\mathrm{d}t + \sigma_t^\lambda \mathrm{d}\mathbf{B}_t^\lambda \ , \ \ \boldsymbol{X}_0^\lambda \sim p_0^\lambda, \tag{32}$$

where $\mathbf{B}_t^\lambda$ are independent Brownian motions on $\mathcal{M}$ and p_0^λ denotes the initial distributions of $\mathbb{Q}^\lambda$. Denoting the marginal distribution of $\mathbb{Q}^\lambda$ as p_t^λ and a mixing distribution $\mathcal{L}$ on Λ, consider the following density and the distribution defined as the mixture of p_t^λ and p_0^λ, respectively, as follows:

$$p_t(x) = \int p_t^\lambda(x)\mathcal{L}(\mathrm{dvol}_x) \ , \ \ p_0(x) = \int p_0^\lambda(x)\mathcal{L}(\mathrm{dvol}_x). \tag{33}$$

Then there exists a diffusion process on $\mathcal{M}$ such that its marginal density is equal to p_t with the initial distribution given as p_0, described by the following SDE:

$$\mathbb{Q}^* : \mathrm{d}\boldsymbol{X}_t = \eta(\boldsymbol{X}_t, t)\mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \ , \ \ \boldsymbol{X}_0 \sim p_0, \tag{34}$$

where the drift η_t and the diffusion coefficient σ_t satisfy the following:

$$\eta(x,t) = \int \eta^\lambda(x,t)\frac{p_t^\lambda(x)}{p_t(x)}\mathcal{L}(\mathrm{dvol}_x) \ , \ \ \sigma_t^2 = \int (\sigma_t^\lambda)^2 \frac{p_t^\lambda(x)}{p_t(x)}\mathcal{L}(\mathrm{dvol}_x). \tag{35}$$

The proof for the Riemannian setting extends that of the Euclidean case, where we leverage the Fokker-Planck equation to characterize the marginal density. From the condition of Eq. (35), we can derive the following:

$$\frac{\partial p_t(x)}{\partial t} = \int \frac{\partial}{\partial t} p_t^\lambda(x)\mathcal{L}(\mathrm{dvol}_x) \tag{36}$$

$$= \int \left[-\mathrm{div}\Big(p_t^\lambda(x)\eta^\lambda(x,t)\Big) + \frac{1}{2}(\sigma_t^\lambda)^2 \Delta_{\mathcal{M}} p_t^\lambda(x) \right] \mathcal{L}(\mathrm{dvol}_x) \tag{37}$$

$$= -\mathrm{div}\left(p_t(x) \int \eta^\lambda(x,t)\frac{p_t^\lambda(x)}{p_t(x)}\mathcal{L}(\mathrm{dvol}_x) \right) + \frac{1}{2}\Delta_{\mathcal{M}} \left(p_t(x) \int (\sigma_t^\lambda)^2 \frac{p_t^\lambda(x)}{p_t(x)}\mathcal{L}(\mathrm{dvol}_x) \right) \tag{38}$$

$$= -\mathrm{div}\Big(p_t(x)\eta(x,t)\Big) + \frac{1}{2}\sigma_t^2 \Delta_{\mathcal{M}} p_t(x), \tag{39}$$

where the second equality is derived from the Fokker-Planck equation with respect to the process $\mathbb{Q}^\lambda$. Since Eq. (39) corresponds to the Fokker-Planck equation with respect to the mixture process, we can conclude that p_t is the marginal density of the mixture process.

Generative Process Now, we are ready to derive our Riemannian Diffusion Mixture in Eq. (6). Using the diffusion mixture representation, we can derive the mixture of a collection of the bridge processes $\{\mathbb{Q}^{x,z} : x \sim \Gamma, z \sim \Pi\}$ as follows:

$$\mathbb{Q}^\Pi : \mathrm{d}\boldsymbol{X}_t = \left[\int \int \eta^z(\boldsymbol{X}_t, t)\frac{p^{x,z}(\boldsymbol{X}_t)}{p_t(\boldsymbol{X}_t)}\Gamma(\mathrm{dvol}_x)\Pi(\mathrm{dvol}_z) \right] \mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \tag{40}$$

$$= \left[\int \eta^z(\boldsymbol{X}_t, t)\frac{p^z(\boldsymbol{X}_t)}{p_t(\boldsymbol{X}_t)}\Pi(\mathrm{dvol}_z) \right] \mathrm{d}t + \sigma_t \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \ , \ \boldsymbol{X}_0 \sim \Gamma, \tag{41}$$

where η^z denotes the drift of the bridge processes $\mathbb{Q}^{x,z}$ for $x \sim \Gamma$ and $p_t^z(\cdot) = \int p_t^{x,z}(\cdot)\Gamma(\mathrm{dvol}_x)$.

A.5. Probability Flow ODE

Here we provide the derivation of the probability flow of the mixture process $\mathbb{Q}_f$ by considering its time-reversed process $\mathbb{Q}_b$ in two different perspectives. First, the time-reversed process $\mathbb{Q}_b$ corresponds to a mixture process built from the collection

of time-reversed bridge processes from Π to Γ, which can be modeled by the following SDE:

$$\mathrm{d}\overline{\boldsymbol{X}}_t = \frac{\sigma_{T-t}^2}{\tau_T - \tau_{T-t}} \left[\int \eta_b^z(\overline{\boldsymbol{X}}_t, t) \frac{p_t^z(\overline{\boldsymbol{X}}_t)}{p_t(\overline{\boldsymbol{X}}_t)} \Gamma(\mathrm{dvol}_z) \right] \mathrm{d}t + \sigma_{T-t} \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \ , \quad \overline{\boldsymbol{X}}_0 \sim \Pi, \tag{42}$$

where $\tau(t) := \int_0^t \sigma_s^2 \mathrm{d}s$ as defined in Eq. (3) and η_b^z denotes the drift of the bridge process $\mathbb{Q}_b(\cdot|\overline{\boldsymbol{X}}_T = z)$. On the other hand, $\mathbb{Q}_b$ can be derived in terms of the score function from Theorem 3.1 of Bortoli et al. (2022) as follows:

$$\mathrm{d}\overline{\boldsymbol{X}}_t = \Big[-\eta_f(\overline{\boldsymbol{X}}_t, t) + \sigma_{T-t}^2 \nabla_{\overline{\boldsymbol{X}}_t} \log p_t(\overline{\boldsymbol{X}}_t) \Big] \mathrm{d}t + \sigma_{T-t} \mathrm{d}\mathbf{B}_t^{\mathcal{M}}. \tag{43}$$

Therefore, we can obtain the score function in terms of the drifts of $\mathbb{Q}_f$ and $\mathbb{Q}_b$ as follows:

$$\nabla \log p_t(\boldsymbol{X}_t) = \left(\eta_f(\boldsymbol{X}_t, t) + \eta_b(\boldsymbol{X}_t, T-t)\right) / \sigma_t^2, \tag{44}$$

and as a result, the probability flow associated with the mixture process $\mathbb{Q}_f$ is obtained by the following ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{Y}_t = \Big(\eta_f(\boldsymbol{Y}_t, t) - \frac{1}{2} \nabla \log p_t(\boldsymbol{Y}_t) \Big) = \frac{1}{2} \Big(\eta_f(\boldsymbol{Y}_t, t) - \eta_b(\boldsymbol{Y}_t, T-t) \Big) \ , \quad \boldsymbol{Y}_0 \sim \Gamma. \tag{45}$$

When $\mathcal{M}$ is a Euclidean space, our derived probability flow of Eq. (45) corresponds to the probability flow for Schrödinger bridges (De Bortoli et al., 2021; Chen et al., 2022; Shi et al., 2023).

Finally, using Proposition 2 of Mathieu & Nickel (2020), we can compute the likelihood of the probability flow as follows:

$$\log p_T(\boldsymbol{Y}_T) = \log p_0(\boldsymbol{Y}_0) + \frac{1}{2} \int_0^T \mathrm{div} \Big(\eta_f(\boldsymbol{Y}_t, t) - \eta_b(\boldsymbol{Y}_t, T-t) \Big) \mathrm{d}t. \tag{46}$$

It is worth noting that the probability flow of the mixture process is different from the continuous flows used in previous works. This is due to the difference in the marginal densities that are characterized by different laws: By construction, the marginal density of the probability flow is equal to the marginal density of the associated mixture process which is described by the Fokker-Planck equation:

$$\frac{\partial p_t(z)}{\partial t} = -\mathrm{div} \Big(\eta_f(z, t) p_t(z) \Big) + \frac{1}{2} \sigma_t^2 \Delta_{\mathcal{M}} p_t(z), \tag{47}$$

whereas the marginal density of a deterministic process is described by the transportation equation:

$$\frac{\partial \tilde{p}_t(z)}{\partial t} = -\mathrm{div} \Big(\eta_{\mathrm{CNF}}(z, t) \tilde{p}_t \Big). \tag{48}$$

A.6. Bridge Matching on Riemannian Manifold

We first derive the KL divergence between a diffusion process $\mathbb{Q} : \mathrm{d}\boldsymbol{Z}_t = \eta(\boldsymbol{Z}_t, t)\mathrm{d}t + \nu_t \mathrm{d}B_t^{\mathcal{M}}$ with terminal distribution $\mathbb{Q}_T$ and its parameterized process $\mathbb{P}^\psi : \mathrm{d}\boldsymbol{Z}_t = s^\psi(\boldsymbol{Z}_t, t)\mathrm{d}t + \nu_t \mathrm{d}B_t^{\mathcal{M}}$ by leveraging the Girsanov theorem as follows:

$$D_{KL}(\mathbb{Q}_T \| \mathbb{P}_T^\psi) \leq D_{KL}(\mathbb{Q} \| \mathbb{P}^\psi) = \mathbb{E}_{\substack{z \sim \mathbb{Q}_T, \\ \boldsymbol{Z} \sim \mathbb{Q}^z}} \left[\log \frac{\mathrm{d}\mathbb{Q}^z}{\mathrm{d}\mathbb{P}^\psi}(\boldsymbol{Z}) + \log \frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{Q}^z}(\boldsymbol{Z}) \right] \tag{49}$$

$$= \mathbb{E}_{z \sim \mathbb{Q}_T} \Big[D_{KL}(\mathbb{Q}^z \| \mathbb{P}^\psi) \Big] + C_1 \tag{50}$$

$$= \mathbb{E}_{\substack{z \sim \mathbb{Q}_T, \\ \boldsymbol{Z} \sim \mathbb{Q}^z}} \left[\frac{1}{2} \int_0^T \Big\| \nu_t^{-1} \Big(\boldsymbol{s}^\psi(\boldsymbol{Z}_t, t) - \eta^z(\boldsymbol{Z}_t, t) \Big) \Big\|_{\mathcal{M}}^2 \mathrm{d}t \right] + C_2, \tag{51}$$

where $\mathbb{Q}_T$ and $\mathbb{P}_T^\psi$ denotes the terminal distributions of $\mathbb{Q}$ and $\mathbb{P}^\psi$ respectively, $\mathbb{Q}^z$ and η^z denotes the process $\mathbb{Q}(\cdot|\boldsymbol{Z}_T = z)$ and its drift, and C is a constant. The first inequality is from the data processing inequality.

Using the result of Eq. (51) and leveraging the fact that the time-reversed process of the mixture process $\mathbb{Q}_f$ is also a mixture process of time-reversed bridge processes (Eq. (42)), the models $\boldsymbol{s}_f^\theta$ and $\boldsymbol{s}_b^\phi$ can be trained to approximate the drifts η_f and η_b, respectively, with the following objectives:

$$\mathcal{L}_f(\theta) = \mathbb{E}_{\substack{x\sim\Pi,\\ \boldsymbol{X}\sim\mathbb{Q}_f^x}}\left[\frac{1}{2}\int_0^T \left\|\sigma_t^{-1}\Big(\boldsymbol{s}_f^\theta(\boldsymbol{X}_t,t) - \eta_f^x(\boldsymbol{X}_t,t)\Big)\right\|_{\mathcal{M}}^2 \mathrm{d}t\right], \tag{52}$$

$$\mathcal{L}_b(\phi) = \mathbb{E}_{\substack{y\sim\Gamma,\\ \overline{\boldsymbol{X}}\sim\mathbb{Q}_b^y}}\left[\frac{1}{2}\int_0^T \left\|\sigma_{T-t}^{-1}\Big(\boldsymbol{s}_b^\phi(\overline{\boldsymbol{X}}_t,t) - \eta_b^y(\overline{\boldsymbol{X}}_t,t)\Big)\right\|_{\mathcal{M}}^2 \mathrm{d}t\right]. \tag{53}$$

Instead of separately training the models by simulating two different bridge processes $\mathbb{Q}_f^x$ and $\mathbb{Q}_b^y$, we can train the models simultaneously by simulating a single bridge process $\mathbb{Q}^{x,y}$ with a fixed starting point x and endpoint y, i.e., $\mathbb{Q}_f(\cdot|\boldsymbol{X}_0 = x, \boldsymbol{X}_T = y)$, reducing the computational cost for the simulation in half. Furthermore, when simulating $\mathbb{Q}^{x,y}$, we introduce a two-way approach to obtaining $\boldsymbol{Z}_t \sim \mathbb{Q}^{x,y}$: simulating $\mathbb{Q}^{x,y}$ from time 0 to t if $t < T/2$, and otherwise simulating from time T to t. Altogether, we obtain the following loss as presented in Eq. (11):

$$\mathbb{E}_{\substack{(x,y)\sim(\Pi,\Gamma),\\ \boldsymbol{Z}\sim\mathbb{Q}^{x,y}}} \frac{1}{2}\int_0^T \sigma_t^{-2}\left[\left\|\boldsymbol{s}_f^\theta(\boldsymbol{Z}_t,t) - \eta_f^x(\boldsymbol{Z}_t,t)\right\|_{\mathcal{M}}^2 + \left\|\boldsymbol{s}_b^\phi(\boldsymbol{Z}_t,T-t) - \eta_b^y(\boldsymbol{Z}_t,T-t)\right\|_{\mathcal{M}}^2\right]\mathrm{d}t. \tag{54}$$

Furthermore, by leveraging an importance sampling with a proposal distribution $q(t) \propto \sigma_t^{-2}$, we obtain the time-scaled two-way bridge matching as follows:

$$\mathbb{E}_{\substack{(x,y)\sim(\Pi,\Gamma),\\ t\sim q}} \mathbb{E}_{\boldsymbol{Z}_t\sim\mathbb{Q}^{x,y}}\left[\left\|\boldsymbol{s}_f^\theta(\boldsymbol{Z}_t,t) - \eta_f^x(\boldsymbol{Z}_t,t)\right\|_{\mathcal{M}}^2 + \left\|\boldsymbol{s}_b^\phi(\boldsymbol{Z}_t,T-t) - \eta_b^y(\boldsymbol{Z}_t,T-t)\right\|_{\mathcal{M}}^2\right]. \tag{55}$$

We note that while the idea of the importance sampling for the time distribution was also used in Huang et al. (2022), our approach leverages a simple and easy-to-sample proposal distribution q, which is effective in stabilizing the training and improving the generation quality, without the need for additional computation or training time.

A.7. Other Relevant Works

Here we further discuss relevant works on manifold diffusion models, extending Section 4. Thornton et al. (2022) extends Diffusion Schrödinger Bridge to the manifold setting, which aims to find the forward and backward processes between distributions that minimize the KL divergence to the Brownian motion. However, Thornton et al. (2022) uses Iterative Proportional Fitting to alternatively train the models that require computing the divergence for numerous iterations, which is computationally expensive compared to our divergence-free two-way bridge matching which can train the models simultaneously. Recently, Lou et al. (2023) introduced practical improvements for Riemannian diffusion models based on a refined estimator for the heat kernel on Riemannian symmetric spaces. For our framework, improved heat kernel estimators can be used to construct a mixture of Brownian bridges, instead of Logarithm bridges or Spectral bridges, but we leave it as future work. Furthermore, Bose et al. (2023) introduces a stochastic version of Flow Matching (Lipman et al., 2023) in the SO(3) group, which can be considered a special case of our Logarithm bridge applied to SO(3). Other line of works focus on specific geometries such as SO(3) (Leach et al., 2022), SE(3) (Yim et al., 2023; Urain et al., 2023), and product of tori (Jing et al., 2022), or constrained manifolds (Fishman et al., 2023) defined by set of inequality constraints.

B. Experimental Details

B.1. Implementation Details

We follow the experimental settings of previous works (Bortoli et al., 2022; Chen & Lipman, 2024) including the data splits with the same seed values of 0-4 for five different runs. We split the datasets into training, validation, and test sets with (0.8, 0.1, 0.1) proportions. Following Chen & Lipman (2024), we use the validation NLL for early stopping and the test NLL is computed from the checkpoint that achieved the best validation NLL.

We parameterize the drifts of the mixture processes with multilayer perceptrons where we concatenate the time to the input, following the previous works. For all experiments except the high dimensional tori, we use 512 hidden units and select the

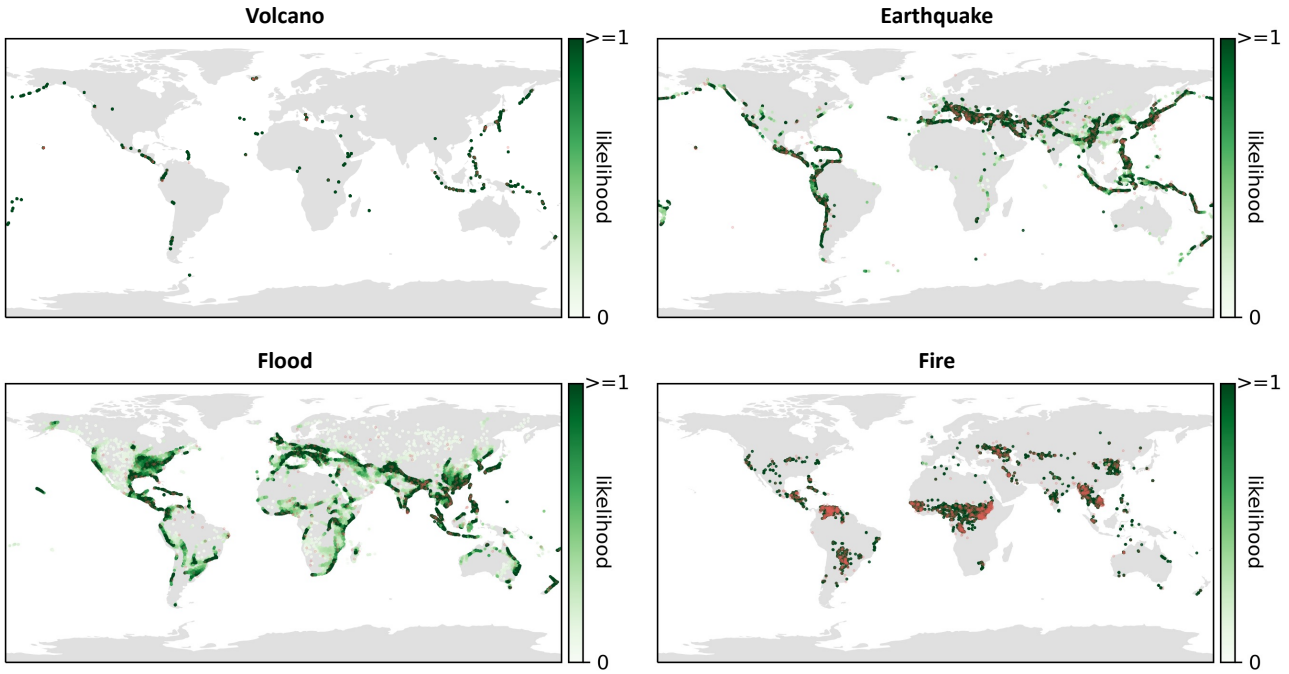


Figure 4: **Visualization of the generated samples and learned density** of our model on earth and climate science datasets. Red dots denote samples from the test set and green dots denote the generated samples. Darker green colors denote a higher likelihood modeled by our approach.

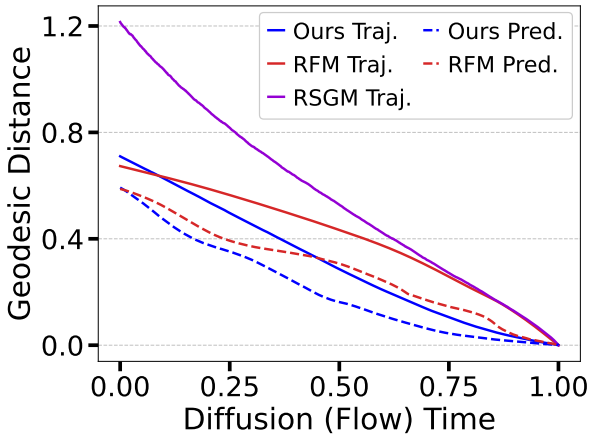


Figure 5: **Convergence of the generative processes** on the Volcano dataset. We compare the geodesic distance between the samples of the trajectory (Traj.) and the final results. We further compare the convergence of predictions (Pred.) made by ours and that of RFM.

Ours	Volcano	Flood
Uniform	-5.37 $\pm$ 0.67	0.67 $\pm$ 0.14
Time-scaled	**-9.52** $\pm$ 0.87	**0.42** $\pm$ 0.08

Figure 6: **Ablation study on the time-scaled training objective**. We report the test NLL of our method (Time-scaled) against a variant trained with uniformly distributed time (Uniform), instead of the time-scaled distribution q in Eq. (12).

Ours	Earthquake	Proline
Different samples	-0.29 $\pm$ 0.08	0.14 $\pm$ 0.025
Same samples	-0.30 $\pm$ 0.06	0.14 $\pm$ 0.027

Figure 7: **Ablation study on the two-way bridge matching**. We show that sampling $\boldsymbol{Z}_t$ from a single bridge process (denoted as Same samples) as proposed in our work does not introduce additional variance by comparing the performance with the variant of our method which samples different $\boldsymbol{Z}_t$ for each forward and backward direction of the bridge process (denoted as Different samples).

number of layers from 6 to 13, using either the sinusoidal or swish activation function. All models are trained with Adam optimizer and we either do not use a learning rate scheduler or use the scheduler with the learning rate annealed by a linear map which then applies cosine scheduler, as introduced in Bortoli et al. (2022). We also use the exponential moving average for the model weights (Polyak & Juditsky, 1992) with decay 0.999.

Table 3: **Test NLL results on protein datasets**. We report the mean of 5 different runs with different data splits. Best performance and its comparable results ($p > 0.05$) from the t-test are highlighted in bold.

Dataset size	General (2D) 138208	Glycine (2D) 13283	Proline (2D) 7634	Pre-Pro (2D) 6910	RNA (7D) 9478
MoPS (De Cao & Aziz, 2020)	1.15 $\pm$ 0.002	2.08 $\pm$ 0.009	0.27 $\pm$ 0.008	1.34 $\pm$ 0.019	4.08 $\pm$ 0.368
RDM (Huang et al., 2022)	1.04 $\pm$ 0.012	1.97 $\pm$ 0.012	**0.12** $\pm$ 0.011	1.24 $\pm$ 0.004	-3.70 $\pm$ 0.592
RFM (Chen & Lipman, 2024)	**1.01** $\pm$ 0.025	**1.90** $\pm$ 0.055	0.15 $\pm$ 0.027	**1.18** $\pm$ 0.055	**-5.20** $\pm$ 0.067
Ours (LogBM)	**1.01** $\pm$ 0.026	**1.89** $\pm$ 0.056	**0.14** $\pm$ 0.027	**1.18** $\pm$ 0.059	**-5.27** $\pm$ 0.090

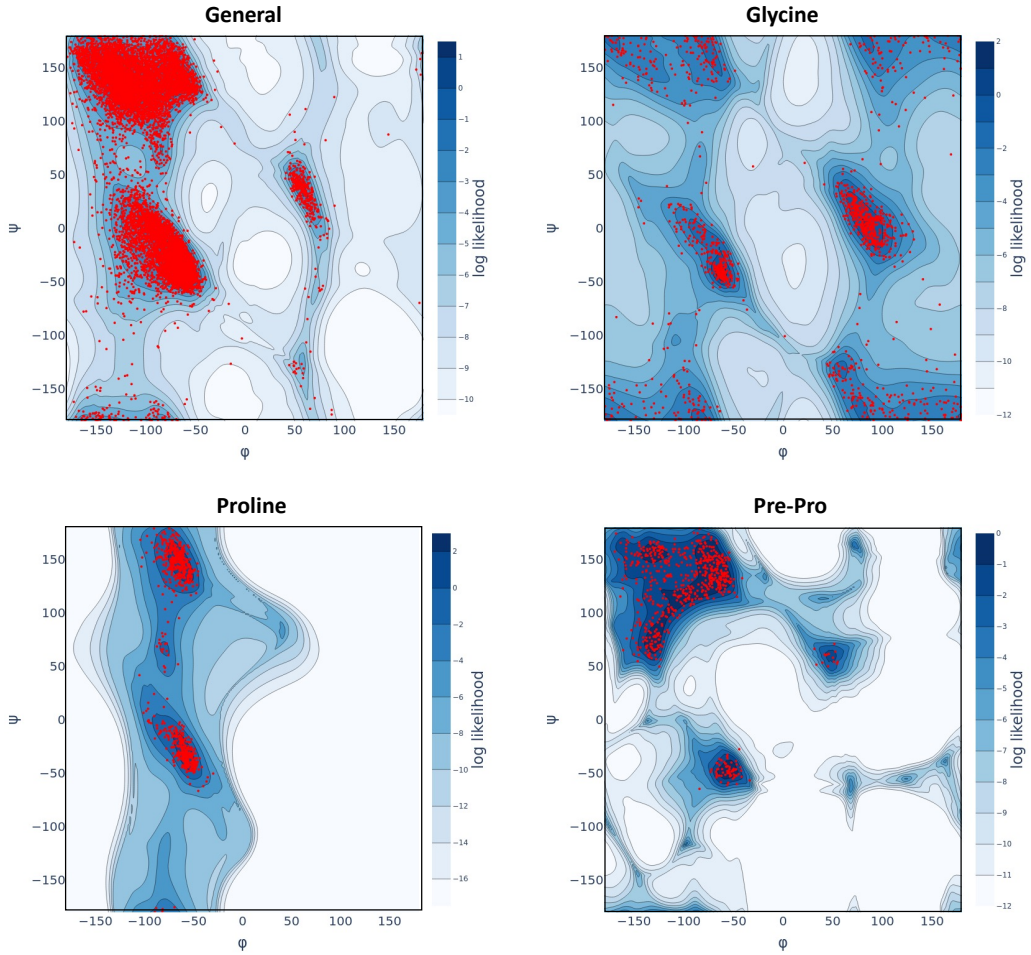


Figure 8: **Visualization of the learned density** of our model on protein datasets using the Ramachandran contour plots. The red dots denote the samples from the test set. The blue color denotes the log-likelihood computed from our model where the darker colors indicate a higher likelihood.

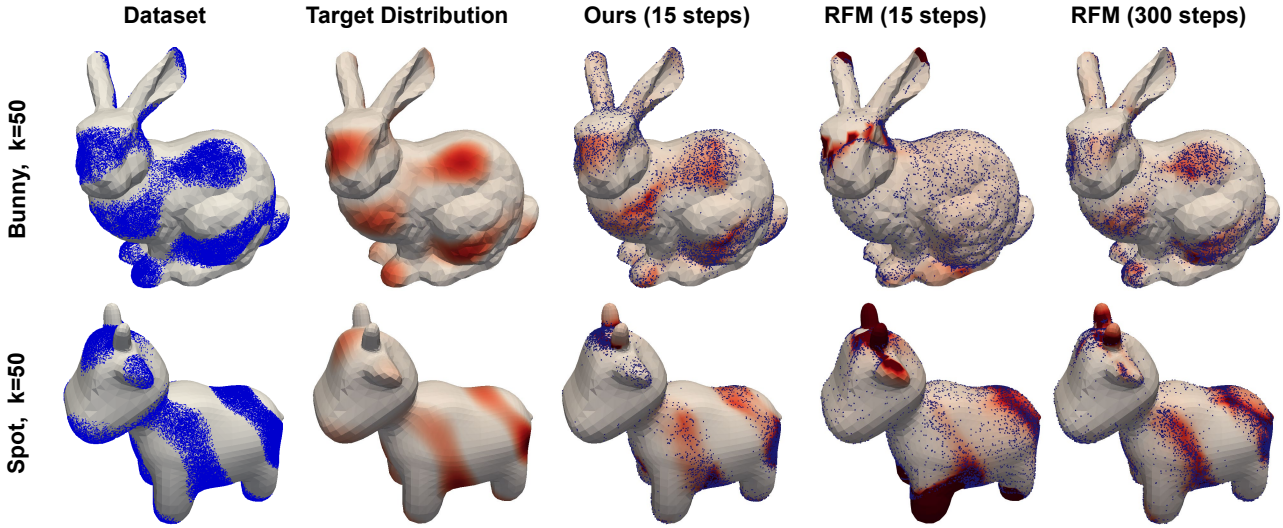
The drifts of the mixture processes are parameterized in the ambient space with projection onto the tangent space as follows:

$$\boldsymbol{s}^{\theta}(x,t) = \text{proj}_x(\tilde{\boldsymbol{s}}^{\theta}(x,t)). \tag{56}$$

where proj_x is a orthogonal projection onto the tangent space at x. For all experiments, we train our models using the time-scaled two-way bridge matching in Eq. (12), where we use 15 steps for the in-training simulation carried out by Geodesic Random Walk (Jørgensen, 1975; Bortoli et al., 2022).

Except for the mesh experiments, we compute the likelihood of our parameterized probability flow ODE using Dormand-Prince solver (Dormand & Prince, 1980) with absolute and relative tolerance of $1e-5$, following the previous works (Bortoli

Figure 9: **Visualization of the generated samples and the learned density** of our method and RFM on the mesh datasets. Blue dots represent the generated samples and darker red colors indicate higher likelihood. The numbers in the parentheses denote the number of in-training simulation steps used to train the model.



et al., 2022; Chen & Lipman, 2024). For the mesh experiments, we compute the likelihood with 1000 Euler steps with projection after every step as done in Chen & Lipman (2024). For all experiments, we use NVIDIA GeForce RTX 3090 and 2080 Ti and implement the source code with PyTorch (Paszke et al., 2019) and JAX.

B.2. Earth and Climate Science Datasets

We follow the data splits of previous works (Bortoli et al., 2022; Chen & Lipman, 2024), reporting an average of five different runs with different data splits using the same seed values of 0-4. For a fair comparison with baselines, we set the prior distribution to be a uniform distribution on the sphere. The convergence analysis demonstrated in Figure 5 was conducted on the models trained on the Volcano dataset, where we measure the geodesic distance between the final sample and the trajectory $\boldsymbol{Z}_t$ of each method for discretized time steps t. The convergence of the predictions was also measured similarly, where we use the parameterized prediction of Eq. (7).

B.3. Protein Datasets

We follow the experimental setup of Huang et al. (2022) and Chen & Lipman (2024) where we use the dataset compiled by Huang et al. (2022) that consists of 500 high-resolution proteins (Lovell et al., 2003) and 113 selected RNA sequences (Murray et al., 2003). The proteins and the RNAs are divided into monomers where the joint structures are removed and use the backbone conformation of the monomer. For proteins, this results in 3 torsion angles of the amino acid where the 180° angle is removed and can be represented on the 2D torus. For RNAs, the 7 torsion angles are represented on the 7D torus. We follow the data splits of Chen & Lipman (2024) and report an average of five different runs with different data splits using the same seed values of 0-4. For a fair comparison with the baselines, we also set the prior distribution to be a uniform distribution on the 2D and 7D tori.

B.4. High-Dimensional Tori

We follow the experimental setup of Bortoli et al. (2022) where we create the dataset as a wrapped Gaussian distribution on a high dimensional tori with uniformly sampled mean and scale of 0.2. Since we evaluate on higher dimensions, up to 2000 dimensions, we use 2048 hidden units for all methods. Specifically, we use MLP with 3 hidden layers and 2048 hidden units for RSGM (Bortoli et al., 2022) and RFM (Chen & Lipman, 2024). To make a fair comparison with the baselines, we match the number of model parameters by using MLP with 2 hidden layers and 2048 hidden units for the model estimating the mixture process, i.e. $\boldsymbol{s}_f^\theta$, and use MLP with 1 hidden layer and 512 hidden units for the model estimating the time-reversed mixture process, i.e. $\boldsymbol{s}_b^\phi$. We train all methods for 50k iterations with a batch size of 512 without early stopping and evaluate

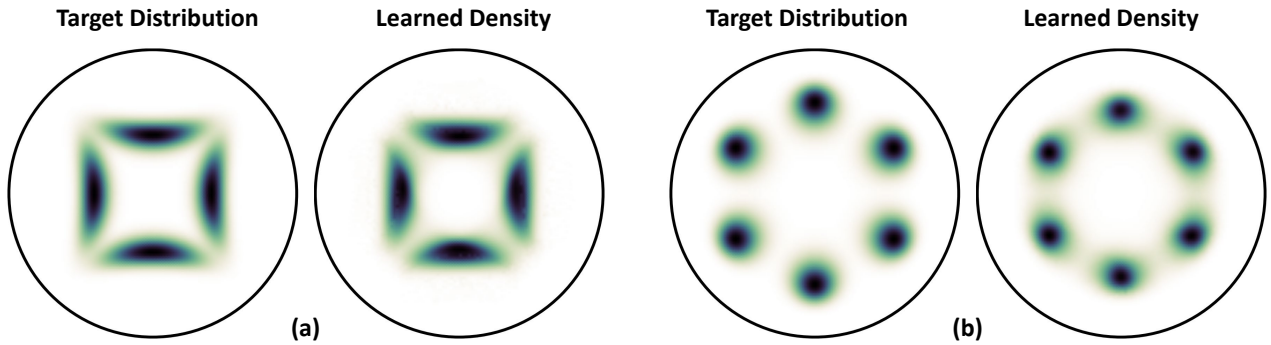


Figure 10: **Visualization of the learned density** of our model on the hyperboloid. We visualize the synthetic distributions and the learned density on the hyperboloid via projection onto a Poincare disk. (a) visualizes a mixture of four wrapped Gaussian distributions and (b) visualizes a mixture of six wrapped Gaussian distributions.

the log-likelihood per dimension for 20k generated samples. We also set the prior distribution to be a uniform distribution on the high-dimensional tori. We measure the training time of each method implemented by JAX (Bradbury et al., 2018) for a fair comparison.

B.5. General Closed Manifolds

We use the triangular meshes provided by Chen & Lipman (2024): An open-source mesh is used for Spot the Cow and a downsampled mesh with 5000 triangles is used for Stanford Bunny, where the 3D coordinates of the meshes are normalized so that the points lie in $(0, 1)$. Following Chen & Lipman (2024), the target distributions on the mesh are created by first computing the k-th eigenfunction (associated with non-zero eigenvalue) on three times upsampled mesh, thresholding at zero, and then normalizing the resulting function. The visualization of generated samples and learned density of RFM in Figure 3 and 9 are obtained by running the open source code. For a fair comparison with RFM, we set the prior distribution to be a uniform distribution on the mesh. We measure the training time of our model and RFM which are all implemented in JAX for a fair comparison.

B.6. Further Analysis

Time-scaled Training Objective We compare our framework trained with Eq. (12) against a variant trained with uniformly distributed time instead of time-scaled distribution q in Eq. (12), on the Volcano dataset. We follow the experimental setup of earth and climate science experiments.

Number of In-Training Simulation Steps For t in $[0, 1]$, let $\boldsymbol{X}_t^{(N)}$ be the sample $\boldsymbol{X}_t$ obtained by simulating LogBM with N discretized steps either from time 0 to t or T to t. We measure the maximum mean discrepancy (MMD) (Gretton et al., 2012) and Wasserstein distance between $\boldsymbol{X}_t^{(500)}$ and $\boldsymbol{X}_t^{(N)}$ for $N \leq 500$, where $\boldsymbol{X}_t^{(500)}$. For Figures 11 and 12 (d), we use a very small noise scale for the mixture process to mimic a deterministic process. Note that the absolute scales of the MMD among Figure 11 (a)-(d) are not directly comparable, as the MMD are measured for different reference distributions. The MMD results should be interpreted as how much they deviate from the MMD result of the 'almost exact' trajectories.

Non-Compact Manifold We create the synthetic distributions on a 2-dimensional hyperboloid using a mixture of wrapped Gaussian distributions. We use MLP with 4 layers with 512 hidden units and trained for 100k iterations without early stopping. We visualize the learned density in Figure 10 by projecting onto a Poincare disk.

C. Limitations

While our method shows superior performance on diverse manifolds, the theoretical guarantee for constructing the mixture process on non-compact manifolds is not sufficient. We experimentally show that our framework is capable of modeling distributions on non-compact manifold with negative curvature. Also, in our work, we validate that our approach can scale to higher dimensions on manifolds that are considered for real-world tasks, e.g., hypersphere and high-dimensional torus. Yet our method may have difficulty when scaling to complex manifolds for which computing the logarithm map or the eigenfunctions are expensive.

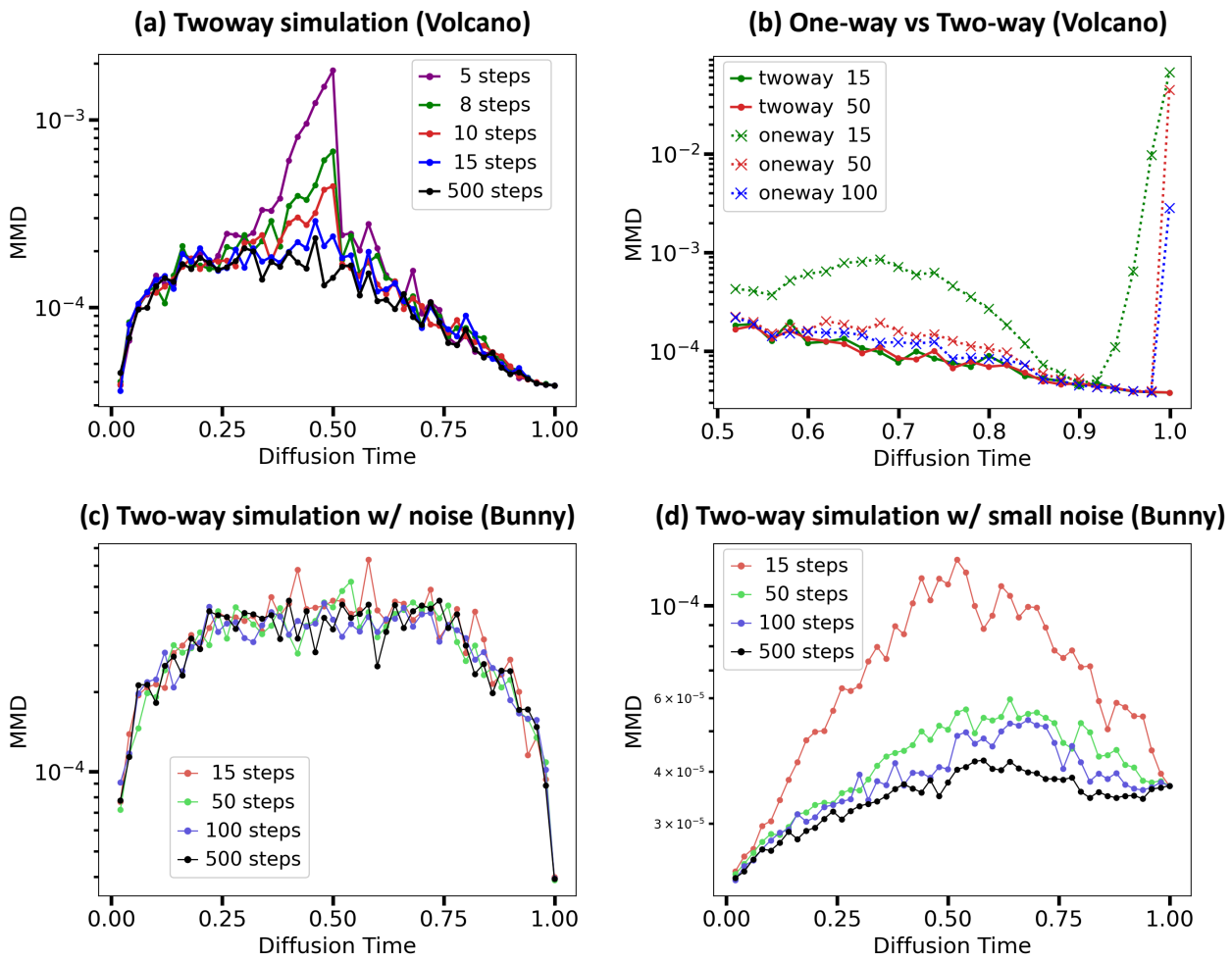


Figure 11: Denoting $\boldsymbol{X}_t^{(N)}$ as $\boldsymbol{X}_t$ of LogBM simulated with N steps, we measure the **MMD distance** between $\boldsymbol{X}_t^{(500)}$, i.e., almost exact sample, and $\boldsymbol{X}_t^{(N)}$ for $N \leq 500$. **(a) Results by differing the number of steps for the two-way approach** where we observe that the MMD results of 15 steps are almost the same as the exact simulation. **(b) Results with the one-way approach** where we can see that the one-way approach requires a significantly large number of steps to obtain accurate trajectories. **(c) Results for simulating the mixture process** on the Stanford Bunny dataset where we observe that 15 steps are enough to obtain accurate trajectories. **(d) Results for simulating the mixture process with small noise scale** where using 15 steps produces highly inaccurate trajectories, and we can see that it requires more than 100 steps to obtain accurate trajectories.

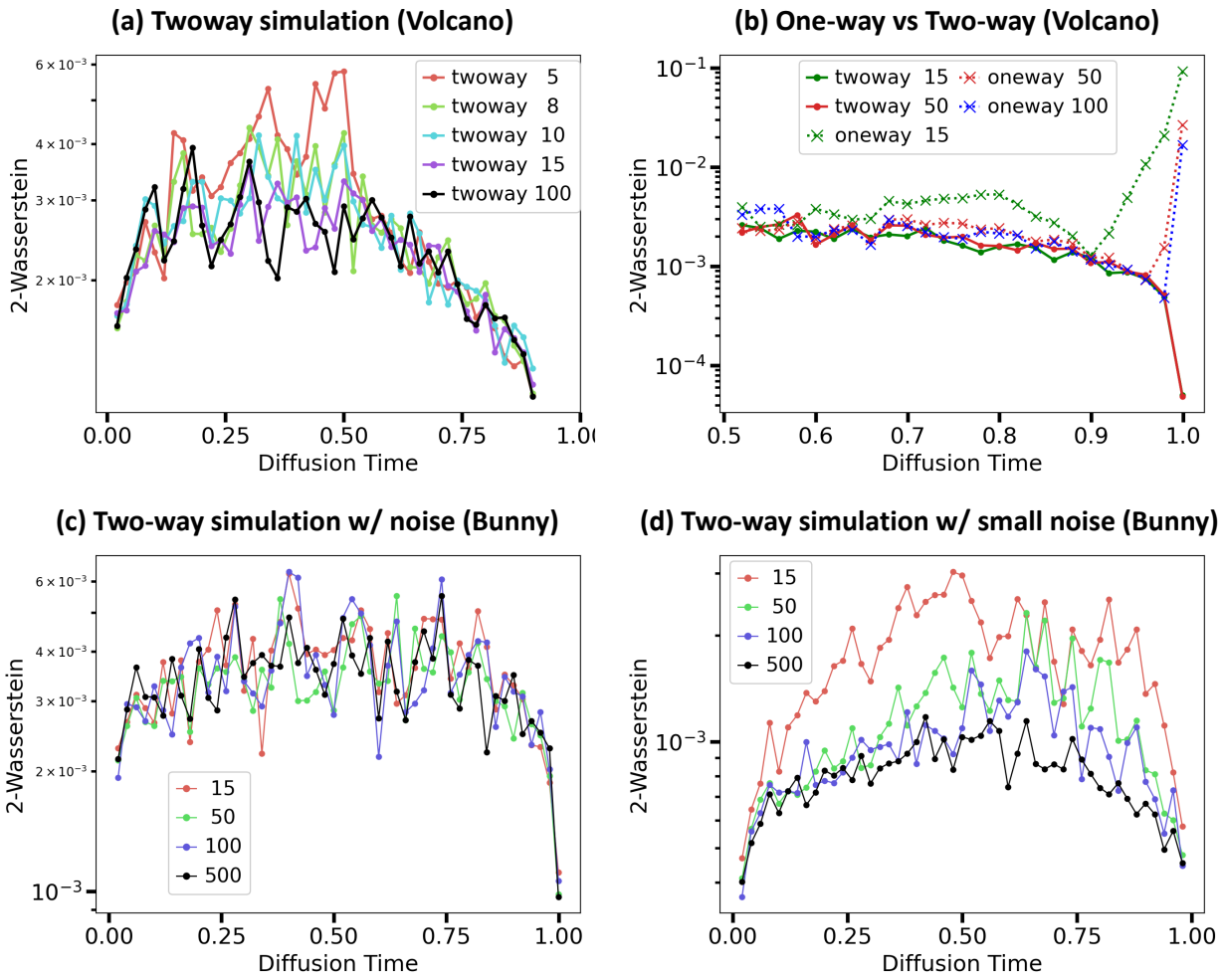


Figure 12: Denoting $\boldsymbol{X}_t^{(N)}$ as $\boldsymbol{X}_t$ of LogBM simulated with N steps, we measure the **Wasserstein distance** between $\boldsymbol{X}_t^{(500)}$, i.e., almost exact sample, and $\boldsymbol{X}_t^{(N)}$ for $N \leq 500$. **(a) Results by differing the number of steps for the two-way approach** where we observe that the MMD results of 15 steps are almost the same as the exact simulation. **(b) Results with the one-way approach** where we can see that the one-way approach requires a significantly large number of steps to obtain accurate trajectories. **(c) Results for simulating the mixture process** on the Stanford Bunny dataset where we observe that 15 steps are enough to obtain accurate trajectories. **(d) Results for simulating the mixture process with small noise scale** where using 15 steps produces highly inaccurate trajectories, and we can see that it requires more than 100 steps to obtain accurate trajectories.