Infinite-dimensional Diffusion Bridge Simulation via Operator Learning

Gefan Yang[1] **Elizabeth L. Baker**[1] **Michael L. Severinsen**[2] **Christy A. Hipsley**[3] **Stefan Sommer**[1]
[1]DIKU, UCPH [2]Globe Institute, UCPH [3]Department of Biology, UCPH

Abstract

The diffusion bridge, which is a diffusion process conditioned on hitting a specific state within a finite period, has found broad applications in various scientific and engineering fields. However, simulating diffusion bridges for modeling natural data can be challenging due to both the intractability of the drift term and continuous representations of the data. Although several methods are available to simulate finite-dimensional diffusion bridges, infinite-dimensional cases remain under explored. This paper presents a method that merges score matching techniques with operator learning, enabling a direct approach to learn the infinite-dimensional bridge and achieving a discretization equivariant bridge simulation. We conduct a series of experiments, ranging from synthetic examples with closed-form solutions to the stochastic nonlinear evolution of real-world biological shape data. Our method demonstrates high efficacy, particularly due to its ability to adapt to any resolution without extra training.

1 INTRODUCTION

Diffusion processes are commonly utilized in diverse scientific fields, including mathematics, physics, evolutionary biology, and finance, to model stochastic dynamics. Updating the posterior of the model through conditioning on observed data is a crucial element of the process, and there are several methods to achieve this. For example, Gaussian process regression can be used for finite-dimensional or infinite-dimensional linear models (Shi and Choi, 2011), and finite-dimensional Doob's h-transform is designed for finite nonlinear processes (Rogers and Williams, 2000, Chapter 6). On the other hand, many data types, such as images and sound signals, are naturally continuous and described by functions, thereby being infinite-dimensional. Moreover, the stochasticity is also nonlinear. One approach is to discretize them and represent them by vectors using finite-dimensional models. However, a more natural way is to work directly in the infinite-dimensional function space, performing only finite-dimensional projections during inference. This functional formulation offers resolution-invariant features, which are more memory-efficient and demonstrate greater generalization capacities. Simulating infinite-dimensional linear processes has been explored in several studies (Franzese et al., 2024; Lim et al., 2023; Pidstrigach et al., 2023). However, extending these methods to condition nonlinear processes has received relatively little exploration except Baker et al. (2024).

Interest in the study of linear diffusion processes has surged recently, largely driven by advancements in diffusion generative models. In diffusion generative models, the data undergoes perturbation via an unconditional linear diffusion process, resulting in noise that follows a simple distribution and is easy to sample. The sampled noise is then transformed back to the clean data through a reverse process, which is also linear and can be learned by the score matching techniques (Hyvärinen and Dayan, 2005; Vincent, 2011). Recently, the study of diffusion models using stochastic differential equations (SDEs) and their mathematical interpretations has gained significant attention (Song et al., 2020; Huang et al., 2021). Based on the rich research on the time-reversed diffusion processes and the design of neural network approximation, we show that similar techniques can be applied to simulating conditioned nonlinear infinite-dimensional diffusion processes.

Extending the finite-dimensional bridge simulation schemes into infinite dimensions is nontrivial, where the problems that need to be addressed are: 1) the theoretical establishment of the conditioning mechanism, 2) the theoretical soundness of the infinite-dimensional

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

time-reversed nonlinear diffusion bridge, and 3) an efficient and straightforward approach to learning the infinite-dimensional bridge. The first problem has been solved in Baker et al. (2024), through the *infinite-dimensional Doob's h-transform*. In this paper, we focus on the remaining two. We first derive the form of the time reversal of the infinite-dimensional diffusion bridge, which lifts the framework in Heng et al. (2021) to infinite dimensions. Furthermore, we propose a tractable optimization objective for learning the reverse bridge. We also design a suitable neural operator structure that can learn the bridge efficiently.

2 RELATED WORK AND CONTRIBUTIONS

2.1 Related work

Bridge simulations: The linear diffusion process always has a Gaussian transition density, which gives the additional drift term in the corresponding conditional process a closed form. The challenge lies in the nonlinear case, where the transition is no longer Gaussian. A mainstream approach is *guided proposals*, originally conceived in Clark (1990) and developed further by Delyon and Hu (2006); Schauer et al. (2017); Mider et al. (2021). The essential idea of a guided proposal is to use an approximated bridge with known transition density as the proposal. Then a Markov Chain Monte Carlo (MCMC) algorithm is used to correct the sampling by computing the likelihood ratio between the proposed and true bridges in closed form. However, such a method suffers from expensive covariance matrix inversion in high-dimensional cases, time-costly MCMC updating iterations, and redundancy for adjusting different levels of resolution. Another approach is to directly approximate the unknown drift in the true bridge using either score-based-learning (Heng et al., 2021; Baker et al., 2024) or kernel approximation (Chau et al., 2024). Heng et al. (2021) used denoising score matching to learn the score from a variational perspective for finite-dimensional diffusion bridges; Baker et al. (2024) derived an infinite-dimensional Doob's h-transform for conditioning infinite-dimensional nonlinear processes, and trained a neural network to learn the finite-dimensional base-projection of the infinite-dimensional bridge, extending the score-based bridge simulation scheme to infinite dimensions. Chau et al. (2024), on the other hand, proposed to use Gaussian kernels to approximate the intractable score for finite-dimensional bridges.

Infinite-dimensional diffusion models: (Score-based) diffusion generative models (SGMs) are initially developed to generate samples in Euclidean spaces (Sohl-Dickstein et al., 2015; Song et al., 2021). Various studies have been done to generalize SGMs to infinite-dimensional spaces with either Hilbert-space-defined score functions (Pidstrigach et al., 2023; Baldassari et al., 2024; Lim et al., 2023) or finite-dimensional score projections (Franzese et al., 2024; Hagemann et al., 2023). Being used for generative modelling, these methods are limited to linear SDEs, i.e. SDEs with state-independent diffusion coefficients. Such a linear setting simplifies the construction of bridges as the transition probability is Gaussian and tractable. In a more general nonlinear case, where the diffusion coefficients are state-dependent, simulating diffusion bridges is nontrivial due to inaccessible closed form of the score functions.

Diffusion Schrödinger bridge: The diffusion bridge simulation problem studied in this paper is distinct from the diffusion Schrödinger bridge (DSB) problem, despite their similar names. A diffusion bridge describes a stochastic process conditioned to start at a fixed point and reach another fixed point at a given terminal time. In contrast, the DSB problem seeks a stochastic process that transports an initial probability distribution to a target distribution in a way that minimizes a relative entropy functional, making it a form of entropy-regularized optimal transport. While DSB methods have gained popularity in generative modeling (De Bortoli et al., 2021; Shi et al., 2024; Tang et al., 2024; Thornton et al., 2022), it is crucial to distinguish between these two problems, as they involve fundamentally different formulations and objectives.

2.2 Contributions

We aim to incorporate score matching techniques into the diffusion bridge simulation methodology, especially in handling infinite-dimensional nonlinear bridges. Our contributions are outlined as follows:

- We derive the time reversal of the infinite-dimensional diffusion bridge, together with a computable optimization objective under finite discretization.
- We design a time-dependent neural operator structure that can learn the infinite-dimensional object through finite samples.
- We demonstrate our method with various continuous function-data-valued stochastic processes and compare them against other related methods.

3 PRELIMINARIES

3.1 Doob's h-transform

Given a *finite-dimensional* Euclidean diffusion process $\mathbf{X} = \{\mathbf{X}_t\}_{t\in[0,T]} \in \mathbb{R}^d$ governed by the SDE:

$$\mathrm{d}\mathbf{X}_t = \mathbf{f}(t, \mathbf{X}_t)\mathrm{d}t + \mathbf{g}(t, \mathbf{X}_t)\mathrm{d}\mathbf{W}_t \tag{1}$$

with $\mathbf{f} : [0,T] \times \mathbb{R}^d \to \mathbb{R}^d, \mathbf{g} : [0,T] \times \mathbb{R}^d \to \mathbb{R}^{d'}$ as finite-dimensional drift and diffusion terms and $\mathbf{W}$ as a Brownian motion in $\mathbb{R}^{d'}$. When the event $\mathbf{X}_T = \mathbf{v}$ is observed at $t = T$, the dynamics of $\mathbf{X}$ changes according to Doob's h-transform (Rogers and Williams, 2000) by reweighting the transition probabilities. Baker et al. (2024) showed that such a transformation can be generalized to *infinite dimensions* as well.

Let $(\mathcal{H}, \langle\cdot,\cdot\rangle_{\mathcal{H}})$ denote a separable Hilbert space with a chosen countable orthonormal basis $\{e_i\}_{i=1}^{\infty}$. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space with natural filtration $\{\mathcal{F}_t\}$. An $\mathcal{H}$-valued diffusion process $X = \{X_t\}_{t\in[0,T]}$ is defined as:

$$\mathrm{d}X_t = f(t, X_t)\mathrm{d}t + g(t, X_t)\mathrm{d}W_t^{\mathbb{P}}, \quad X_0 = u \in \mathcal{H} \tag{2}$$

where $W^{\mathbb{P}}$ is a $\mathbb{P}$-Brownian motion with a covariance operator Q in a Hilbert space $\mathcal{U}$ (where $\mathcal{U}$ can equal $\mathcal{H}$), $f : [0,T] \times \mathcal{H} \to \mathcal{H}$, $g : [0,T] \times \mathcal{H} \to \mathrm{HS}(Q^{1/2}(\mathcal{U}), \mathcal{H})$, where $\mathrm{HS}(Q^{1/2}(\mathcal{U}), \mathcal{H})$ denotes the Hilbert-Schmidt operator $Q^{1/2}(\mathcal{U}) \to \mathcal{H}$. The following theorem (Baker et al., 2024, Theorem 5.1) states a change of dynamics of X:

Theorem 1. *Let $h : [0,T] \times \mathcal{H} \to \mathbb{R}_{>0}$ be a continuous Fréchet differentiable function. Let $Z(t) \coloneqq h(t, X_t)$ be a strictly positive mean-one martingale. Define a new measure $\mathbb{P}^\star$ by:*

$$\mathrm{d}\mathbb{P}^\star \coloneqq Z(T)\mathrm{d}\mathbb{P}. \tag{3}$$

Then under the new defined measure $\mathbb{P}^\star$, X_t solves

$$\begin{aligned}\mathrm{d}X_t &= f^\star(t, X_t)\mathrm{d}t + g(t, X_t)\mathrm{d}W_t^{\mathbb{P}^\star}, \quad X_0 = u,\\ f^\star(s,x) &= f(s,x) + a(s,x)\nabla_x \log h(s,x),\end{aligned} \tag{4}$$

where $W^{\mathbb{P}^\star}$ is a $\mathbb{P}^\star$-Wiener process. Let $g^* : [0,T] \times \mathcal{H} \to \mathrm{HS}(\mathcal{H}, Q^{1/2}(\mathcal{U}))$ be the adjoint operator of g, $a = gg^* : [0,T] \times \mathcal{H} \to \mathrm{HS}(\mathcal{H}, \mathcal{H})$, and $\nabla_x \log h : [0,T] \times \mathcal{H} \to \mathcal{H}$ be the score function. We further denote $X^\star$ as X under the measure $\mathbb{P}^\star$.

Theorem 1 reveals a change of SDE by a change of measures. Moreover, Baker et al. (2024) demonstrated that when such a change is induced by conditioning on $X_T \in \Gamma \subseteq \mathcal{H}$, $h(s,x) = \mathbb{P}(X_T \in \Gamma \mid X_s = x)$. However, Equation (4) is not directly available to sample because 1) $\nabla_x \log h(s,x)$ is inaccessible; 2) it is infinite-dimensional. We will address these two issues in Section 4 by applying a score matching technique on the infinite-dimensional time-reversal process and a finite basis projection.

3.2 Time-reversed diffusion processes

Haussmann and Pardoux (1986) showed that under certain conditions (see (Haussmann and Pardoux, 1986, Assumption (A))), the finite-dimensional diffusion process $\mathbf{X}$ has a time-reversed process $\mathbf{Y}$, which solves the following SDE with:

$$\begin{aligned}\mathrm{d}\mathbf{Y}_t &= \bar{\mathbf{f}}(T-t, \mathbf{Y}_t)\mathrm{d}t + \mathbf{g}(T-t, \mathbf{Y}_t)\mathrm{d}\mathbf{W}_t, \quad \mathbf{Y}_0 = \mathbf{v},\\ \bar{\mathbf{f}}(T-s, x) &= -\mathbf{f}(s,x) + \frac{1}{p_s(x)}\nabla_x\left(\mathbf{a}(s,x)p_s(x)\right),\end{aligned} \tag{5}$$

where $\mathbf{a}(s,x) = \mathbf{g}(s,x)\mathbf{g}^\intercal(s,x)$, and $p_t(x)$ is the marginal density of $\mathbf{X}_t$.

When lifting to infinite dimensions, Millet et al. (1989) showed that the projections of X also have a system of similar time-reversed SDEs. To show that, let $\{k_j\}_{j=1}^{\infty}$ denote a countable orthonormal basis of $\mathcal{U}$ Write Equation (2) as an infinite system of $\mathbb{R}$-valued SDEs with respect to the basis $\{e_i\}_{i=1}^{\infty}$ of $\mathcal{H}$:

$$\mathrm{d}[X_t]_i = [f(t, X_t)]_i\mathrm{d}t + \sum_j^{\infty}[g(t, X_t)]_{ij}\mathrm{d}[W_t^{\mathbb{P}}]_j, \tag{6}$$

where $[X_t]_i, [W_t^{\mathbb{P}}]_j$ are the i-th, j-th components of $X_t, W_t^{\mathbb{P}}$ under the bases e_i, k_j, defined by the inner products $\langle\cdot,\cdot\rangle_{\mathcal{H}}, \langle\cdot,\cdot\rangle_{\mathcal{U}}$, $[f(s,x)]_i \coloneqq \langle f(s,x), e_i\rangle_{\mathcal{H}}$, $[g(s,x)]_{ij} \coloneqq \langle g(s,x)(k_j), e_i\rangle_{\mathcal{H}}$.

Millet et al. (1989) made an analogy to the finite-dimensional time reversal diffusion. Suppose that given $i \in \mathbb{Z}$, there exists a *finite* $I(i) \subset \mathbb{Z}$, such that for all s, $[a(s,x)]_{ij} \coloneqq \sum_\ell^{\infty}[g(s,x)]_{i\ell}[g(s,x)]_{j\ell} = 0$ if $j \notin I(i)$. Intuitively, $I(i)$ contains the indices that make $[a(s,x)]_{ij}$ diagonal. Denote $\mathbf{x}^{(i)} \coloneqq \{[X_t]_j; j \in I(i)\} \in \mathbb{R}^{\#I(i)}$ and $\hat{\mathbf{x}}^{(i)} \coloneqq \{[X_t]_j; j \notin I(i)\}$. For $t > 0$, and $\mathbf{z}^{(i)} \coloneqq \{z_j;\, z_j \in \mathbb{R}, j \notin I(i)\}$, the conditional law of $\mathbf{x}^{(i)}$ given $\hat{\mathbf{x}}^{(i)} = \mathbf{z}^{(i)}$ is assumed to have a density $p_t(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)})$ (see Millet et al. (1989) Theorem 5.3 for the restrictions on the coefficients that guarantee the existence of the density). Then Millet et al. (1989) Theorem 3.1 gives the time reversal of Equation (6) as:

$$\begin{aligned}\mathrm{d}[Y_t]_i &= [\bar{f}(T-t, Y_t)]_i\mathrm{d}t + \sum_j^{\infty}[g(T-t, Y_t)]_{ij}\mathrm{d}[W_t^{\mathbb{P}}]_j,\\ [\bar{f}(T-s,x)]_i &= -[f(s,x)]_i\\ &+ \frac{1}{p_s(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})}\sum_{j\in I(i)}\nabla_j\left([a(s,x)]_{ij}p_s(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})\right).\end{aligned} \tag{7}$$

This result is especially useful, as it shows that we can find a well-defined time reversal for an infinite-dimensional diffusion process in terms of the conditional density in the finite subspace of $\mathcal{H}$, which shall serve as the foundation of our infinite-dimensional time-reversed diffusion bridge.

3.3 Denoising score matching

Score matching (Hyvärinen and Dayan, 2005) is a technique that is used to estimate the score function $\nabla_x \log p_t(x)$ by a parameterized estimator $s_\theta(t,x)$. An important variant of score matching is denoising score matching (DSM) (Vincent, 2011), where the optimization problem is formulated as:

$$L_{\text{DSM}}(\theta) = \mathbb{E}_x \left[\|s_\theta(t,x) - \nabla_x \log p_{t|0}(x|x_0)\|^2 \right], \quad (8)$$

where $p_{t|0}(x|x_0)$ as the transition density from $(0,x_0)$ to (t,x). The DSM loss requires X to be linear with Gaussian transition density $p_{t|0}(x|x_0)$. For more general diffusion processes, inspired by DSM, Heng et al. (2021) introduced a variational formulation to learn the intractable $\nabla_x \log p_{t|0}(x \mid x_0)$, by defining the optimization problem:

$$L_{\text{t-reversed}}(\theta) = \frac{1}{2} \sum_{m=1}^{M} \int_{t_{m-1}}^{t_m} \mathbb{E}_x \left[\|s_\theta(t,x) - \nabla_x \log p_{t|t_{m-1}}(x|x_{t_{m-1}})\|^2_{\mathbf{a}(t,x)} \right] \mathrm{d}t. \quad (9)$$

They propose applying this optimization twice on forward and backward processes sequentially to recover the original forward-in-time conditional process.

4 METHODOLOGY

4.1 Infinite-dimensional time-reversed bridge

To extend the framework proposed by Heng et al. (2021) into infinite-dimensional settings, we need to establish the time reversal of the infinite-dimensional diffusion bridge, which is given by the following theorem:

Theorem 2. *Let f, g, a, and $W_t^{\mathbb{P}^\star}$ be as defined before. The conditional process $X^\star$ has a time reversal $Y^\star = \{Y_t^\star\}_{t\in[0,T]} = \{X_{T-t}^\star\}_{t\in[0,T]}$, with the chosen bases e_i and k_j of $\mathcal{H}$ and $\mathcal{U}$ respectively, which follows the SDEs:*

$$\mathrm{d}[Y_t^\star]_i = [\overline{f^\star}(t, Y_t^\star)]_i \mathrm{d}t + \sum_j^\infty [g(T-t, Y_t^\star)]_{ij} \mathrm{d}[W_t^{\mathbb{P}^\star}]_j \quad (10)$$

where,

$$[\overline{f^\star}(T-s,x)]_i = -[f(s,x)]_i + \sum_{j\in I(i)} \nabla_{[x]_j} [a(s,x)]_{ij} + \sum_{j\in I(i)} [a(s,x)]_{ij} \nabla_{[x]_j} \log p_s(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0)) \quad (11)$$

The proof of Theorem 2 can be found in Section 7.1. Note that Equation (11) involves $p_{T-s}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0))$, which is the conditional density of $\mathbf{x}^{(i)}$ given $\hat{\mathbf{x}}^{(i)} = \mathbf{z}^{(i)}$ and $X_0 = x_0 \in \mathcal{H}$. Compared with the h-function in Equation (4), we are able to design an objective function to learn this finite-dimensional density. Inspired by Heng et al. (2021), we use a variational approach to approximate the unconditioned time reversal Y with a parameterized operator $\mathcal{G}^{(\theta)}(s,x) : [0,T] \times \mathcal{H} \to \mathcal{H}$, such that when the object function is optimized, its projection under e_i, $[\mathcal{G}^{(\theta)}(s,x)]_i$, approximates $\sum_{j\in I(i)} [a(T-s,x)]_{ij} \nabla_{[x]_j} \log p_{T-s}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0))$, and therefore is available in Equation (10).

4.2 Learning the time reversal

To approximate Y, we define a diffusion process $Y^{(\theta)} = \{Y_t^{(\theta)}\}_{t\in[0,T]} \in \mathcal{H}$ that solves:

$$\mathrm{d}Y_t^{(\theta)} = f^{(\theta)}(T-t, Y_t^{(\theta)})\mathrm{d}t + g^{(\theta)}(T-t, Y_t^{(\theta)}) \mathrm{d}W_t^{\mathbb{P}^{(\theta)}}, \quad (12)$$

where $f^{(\theta)} : [0,T] \times \mathcal{H} \to \mathcal{H}$ is a mapping parameterized by $\theta \in \Theta$, and $W^{\mathbb{P}^{(\theta)}}$ is a $\mathbb{P}^{(\theta)}$-Brownian motion. The set of measures $\{\mathbb{P}^{(\theta)};\ \theta \in \Theta\}$ provides a variational class for approximating $\mathbb{P}$. The absolute continuity between $\mathbb{P}^{(\theta)}$ and $\mathbb{P}$ is guaranteed by choosing $g^{(\theta)}(T-s,x) = g(T-s,x)$. Furthermore, we formulate $f^{(\theta)}$ such that its projection under e_i satisfies:

$$[f^{(\theta)}(T-s,x)]_i = -[f(s,x)]_i + \sum_{j\in I(i)} \nabla_{[x]_j} [a(s,x)]_{ij} + [\mathcal{G}^{(\theta)}(s,x)]_i, \quad (13)$$

where $\mathcal{G}^{(\theta)}(t,x) : [0,T] \times \mathcal{H} \to \mathcal{H}$ is a bounded parameterized operator. The advantage of such a formulation is that the KL divergence $\mathrm{D}_{\text{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)})$ can be expressed by the difference between the drift coefficients explicitly, obtained by the following lemma:

Lemma 1. *Let $\{e_i\}$ be the basis for which $a(s,x)$ is diagonalizable for any $s \in [0,T], x \in \mathcal{H}$. Let $f^{(\theta)} : [0,T] \times \mathcal{H} \to \mathcal{H}$ be a Lipschitz function of linear growth, parameterized by θ, such that its decomposition under e_i follows Equation* (13)*. Define the difference of drifts as:*

$$[\Psi(s,x)]_i \coloneqq [\{\bar{f} - f^{(\theta)}\}(s,x)]_i. \quad (14)$$

Then the KL divergence can be expressed as:

$$\mathrm{D}_{\text{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)}) = \frac{1}{2} \mathbb{E}_{\mathbb{P}} \left[\int_0^T \sum_{i=1}^\infty \lambda_i [\Psi(s, X_s)]_i^2 \, \mathrm{d}s \right], \quad (15)$$

where $\lambda_i = \|g^*(e_i)\|^2_{Q^{1/2}(\mathcal{U})}$ is the i-th eigenvalue of a, and $\Psi(s,x) = \sum_{j\in I(i)} [a(s,x)]_{ij} \nabla_{[x]_j} \log p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}) - [\mathcal{G}^{(\theta)}(s,x)]_i$.

It turns out that matching the two measures via minimizing the KL divergence is equivalent to matching the two drifts by their components. The proof can be found in Section 7.2. However, Equation (15) still cannot be used as the optimization object due to 1) the intractable $p_s(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})$; 2) the infinite sum. Fortunately, since X is Markovian, it is possible to work with a smaller transition step $t_{n-1} \to t$ such that $t - t_{n-1} \ll T$. The following theorem gives an objective that shares the same gradient with respect to θ as $\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)})$. The proof of theorem is shown in Section 7.3.

Theorem 3. *For any time partition $\{t_n\}_{n=1}^N$ of the interval $[0,T]$ and arbitrarily chosen orthogonal basis e_i of $\mathcal{H}$, define*

$$[\Psi'(s,x;x_{t_{n-1}})]_i = [\mathcal{G}^{(\theta)}(s,x)]_i - [b_s(x;x_{t_{n-1}})]_i, \quad (16)$$

with $b_s(\cdot\ ;x_{t_{n-1}}) : \mathcal{H} \to \mathcal{H}$ for $x_{t_{n-1}} \in \mathcal{H}$, such that its component under e_i satisfies $[b_s(x;x_{t_{n-1}})]_i = \sum_{j\in I(i)}[a(s,x)]_{ij}\nabla_j \log p_{s|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_{t_{n-1}}))$. Then the objective function

$$L(\theta) = \frac{1}{2}\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n} \mathbb{E}_{\mathbb{P}}\left[\sum_{i=1}^{\infty}\lambda_i[\Psi'(s,X_s;x_{t_{n-1}})]_i^2\right]\mathrm{d}s \quad (17)$$

is equivalent to $\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)})$ up to a θ-independent constant C, where $p_{s|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_{t_{n-1}}))$ denotes the conditional density of $\mathbf{x}^{(i)}$ at time s given $\hat{\mathbf{x}} = \mathbf{z}^{(i)}, X_{t_{n-1}} = x_{t_{n-1}} \in \mathcal{H}$, with $\mathbf{x}^{(i)}, \hat{\mathbf{x}}^{(i)}, \mathbf{z}^{(i)}$ as defined before. X_s is sampled from Equation (2).

4.3 Sampling the infinite-dimensional object

To practically address the infinite sum, we truncate the bases e_i and k_j up to a finite M, and set $\mathbf{z}^{(i)} = \{[X_t]_j; j > M\} = 0$. In such a setting, $b_s(x;x_{t_{n-1}})$ can be approximated by:

$$[\hat{b}_s(x;x_{t_{n-1}})]_i \approx \sum_{j=1}^{M}[a(s,\tilde{x})]_{ij}\nabla_{[x]_j}\log p_{s|t_{n-1}}(\tilde{x} \mid \tilde{x}_{t_{n-1}}), \quad (18)$$

where $\tilde{x} = \{[x]_1, \cdots, [x]_M\} \in \mathbb{R}^M$. Although $p_{s|t_{n-1}}(\tilde{x} \mid \tilde{x}_{t_{n-1}})$ is generally intractable, we can approximate it by a Gaussian transition when using a discrete numerical solving scheme like Euler-Maruyama, as long as the time step $\Delta t = s - t_{n-1}, s \in [t_{n-1}, t_n]$ is small enough. Specifically, we can simulate a finite system of Equation (6) containing M $\mathbb{R}$-valued SDEs:

$$\mathrm{d}[X_t]_i = [f(t,X_t)]_i\mathrm{d}t + \sum_{j}^{M}[g(t,X_t)]_{ij}\mathrm{d}[W_t^{\mathbb{P}}]_j, \quad (19)$$

and then,

$$[\hat{b}_s(x;x_{t_{n-1}})]_i = \sum_{j=1}^{M}[a(s,\tilde{x})]_{ij}\nabla_j \log p(\tilde{x} \mid \tilde{x}_{t_{n-1}})$$
$$\approx -(\Delta t)^{-1}\cdot([\tilde{x}]_i - [\tilde{x}_{t_{n-1}}]_i - \Delta t\cdot[f(s,x)]_i). \quad (20)$$

Finally, we set both bases to be Fourier bases and compute $[X_t]_i$ by applying the Fast Fourier Transform (FFT) on the finite evaluation of X_t on the grid $(\xi_1, \ldots, \xi_M)$. In the following section, we shall discuss the design of $\mathcal{G}^{(\theta)}$ such that it incorporates the FFT within its structure and, therefore, only accepts the evaluation of functions as input. We define our optimization objective as:

$$L(\theta) = \frac{1}{2}\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n}\mathbb{E}_{\mathbb{P}}\left[\sum_{i=1}^{M}\lambda_i\Big(\mathcal{G}^{(\theta)}(s,X_s[\xi_i])\right.$$
$$\left.+(\Delta t)^{-1}\cdot\{X_s[\xi_i] - x_{t_{n-1}}[\xi_i] - \Delta t\cdot f(s,X_s[\xi_i])\}\Big)^2\right]\mathrm{d}t. \quad (21)$$

Under the finite truncation, a can be represented as a $M\times M$ matrix $a_{M\times M}$, and since a must be diagonalizable under e_i, λ_i can be simply chosen as the diagonal entries of $a_{M\times M}$.

4.4 Time-dependent Fourier neural operators

4.4.1 Fourier neural operator

As described in the previous section, $\mathcal{G}^{\theta}(t,x) : [0,T]\times\mathcal{H} \to \mathcal{H}$ is a time-dependent operator. Franzese et al. (2024) proposed two different approaches to model it: implicit neural representations (Saharia et al., 2022) and Transformers (Vaswani et al., 2017). Another solution suggested by Baker et al. (2024) is to choose a certain truncated basis and use a multi-layer perceptron (MLP) to learn the coefficients under the projection. We found that compared with these tools, neural operators (Li et al., 2020a,b; Lu et al., 2021; Kovachki et al., 2023) offer a more natural representation of $\mathcal{G}^{\theta}(t,x)$, as it directly parameterizes a nonlinear operator in a learnable fashion. Specifically,

$$\mathcal{G}^{(\theta)} := \mathcal{Q}\circ\mathcal{L}_L\circ\cdots\circ\mathcal{L}_1\circ\mathcal{P}, \quad x \mapsto y \quad (22)$$

where $x \in \mathcal{H}(\Omega;\mathbb{R}^{d_x}), y \in \mathcal{H}(\Omega;\mathbb{R}^{d_y})$, $\mathcal{P} : \mathbb{R}^{d_x} \to \mathbb{R}^{d_1}, \mathcal{Q} : \mathbb{R}^{d_L} \to \mathbb{R}^{d_y}$ are the local lifting and projection mappings respectively. The time-independent Fourier layer (Li et al., 2020a) $\mathcal{L}_\ell : \mathcal{H}(\Omega;\mathbb{R}^{d_x}) \to \mathcal{H}(\Omega;\mathbb{R}^{d_\ell})$ is defined as:

$$\mathcal{L}_\ell(x)(\xi) := \sigma\left(W_\ell(x)(\xi) + \mathcal{F}^{-1}\{R_\ell\cdot(\mathcal{F}\{x\})\}(\xi)\right), \quad (23)$$

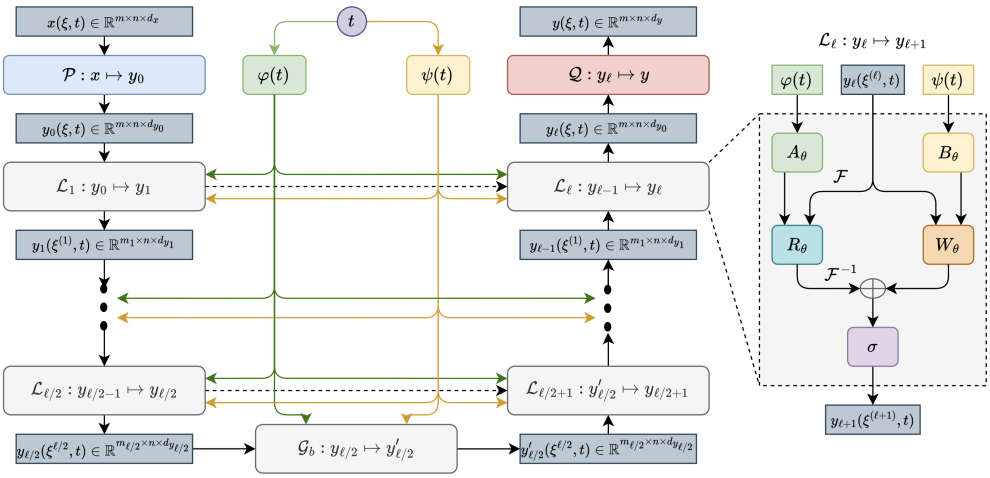


Figure 1: Continuous-time U-shaped FNO architecture, acting as a parameterized operator $\mathcal{G}^{(\theta)} : (x,t) \mapsto y$ for $x, y \in \mathcal{H}$. Both x, y are evaluated on a discrete spatial-temporal grid (ξ, t) with $\xi \in \mathbb{R}^{m^d \times d}$ and $t \in \mathbb{R}^n$. The figure only shows the case when $d = 1$, but the application to $d > 1$ can be achieved through similar architecture, e.g. $x(\xi,t) \in \mathbb{R}^{m\times m\times n\times d_x}$ for $d = 2$ and so on.

where $\xi = (\xi_1, \ldots, \xi_m) \in \mathbb{R}^m$ is the finite grid discretization of the domain Ω, W_ℓ is a linear transform $\mathbb{R}^{d_x} \to \mathbb{R}^{d_\ell}$, and $\mathcal{F}\{\cdot\}, \mathcal{F}^{-1}\{\cdot\}$ are the FFT and inverse FFT (iFFT) respectively, $R_\ell = \mathcal{F}\{\kappa_\ell\} : \mathbb{R}^m \to \mathbb{C}^{d_\ell \times d_\ell}$ for kernel $\kappa_\ell : \mathbb{R}^m \to \mathbb{R}^{d_\ell \times d_\ell}$, and $\sigma : \mathbb{R} \to \mathbb{R}$ is the nonlinear activation function applied component-wise.

4.4.2 Continuous-time modulation

In Li et al. (2020a), the designed $\mathcal{G}^{(\theta)}$ is time-independent, even though the domain Ω can be the product of $[0,T] \subset \mathbb{R}$ with some coordinate domain. While in our case, $\mathcal{G}^{(\theta)}$ depends on t explicitly. A natural idea is to introduce additional time embeddings into the weight matrix $W_\ell(x)$ at each layer, i.e., $W'_\ell(t,s) = \alpha(t)W_\ell(x) + \beta(t)$, where α, β are learnable MLPs to embed t. However, such a setting ignores the contribution from the frequency component $\mathcal{F}(x)$. To address this, Park et al. (2023) proposed a continuous-time-modulated Fourier layer formulation to integrate such a time dependency into both physical and frequency domains:

$$\mathcal{L}_\ell(x)(t,\xi) = \sigma\left(W_\ell \psi_\ell(t) x(\xi) + \mathcal{F}^{-1}\{\varphi_\ell(t) R_\ell \cdot (\mathcal{F}\{x\})(\xi)\}\right) \quad (24)$$

where $\psi_\ell(t) \in \mathbb{R}^{d_\ell \times d_\ell}$, $\varphi(t) : \mathbb{R}^m \to \mathbb{C}$ are time modulations. Essentially, the time information is not only modulated into the physical domain through $\psi(t)$, but also into the Fourier domain through $\varphi(t)$.

In practice, $\psi(t)$ and $\varphi(t)$ are implemented with the same structure, as suggested in Park et al. (2023), by using the sinusoidal embeddings (Vaswani et al., 2017). We find introducing the time modulation in both the physical and frequency domains shows better performance than only acting on the physical domain using transformed time embeddings to shift and scale the output from the Fourier block.

4.4.3 U-shaped FNO

The operator $\mathcal{G}^{(\theta)}(t,\cdot)$ maps the element $x \in \mathcal{H}$ into the same space $\mathcal{H}$, preserving the dimensionality of the input. This structure-preserving property motivates the adoption of a U-shaped architecture for the Fourier Neural Operator (FNO), as proposed by Ashiqur Rahman et al. (2022). The U-shape design is particularly advantageous for: 1) extracting and refining multi-level Fourier features across successive scales, enabling the operator to capture both local and global dependencies efficiently; 2) reducing redundant parameterization by reusing feature maps at corresponding resolutions, thereby lowering memory costs while maintaining expressive power.

We combine the continuous-time modulation and U-shape skeleton, and propose the *continuous-time U-shaped FNO*, as shown in Figure 1. This operator is used to map the pair $(x \in \mathcal{H}, t \in [0,T])$ onto $y \in \mathcal{H}$. We assume that the input and output functions share the same spatial and temporal domain, and the spatial domain is usually set as a d-dimensional manifold $\mathcal{M}$. We choose the spatial domain as periodic subsets of $\mathbb{R}$ or $\mathbb{R}^2$. In the experiment section, we shall detail the choice of such bounded domains for specific tasks.

To evaluate the functions, the spatial domain is discretized into $m \times \cdots \times m = m^d$ grid points in $\mathbb{R}^d$, and the temporal domain $[0,T]$ is also discretized

into n intervals. Then the evaluation of x, $x(\xi, t) \in \mathbb{R}^{m\times\cdots\times m\times n\times d_x}$, where d_x is the dimension of the codomain of x. For example, for a function $x \in \mathcal{H}([0,T]\times\Omega,\mathbb{R}), \Omega \subset \mathbb{R}$, its evalutation can be represented as the tensor of shape $[m, n, 1]$.

The lifting operator $\mathcal{P}$ acts on the codomain which lifts its dimension to d_{y_0}. The lifted representation together with the $\psi(t), \varphi(t)$ embeddings are fed into one of the continuous-time Fourier layers $\mathcal{L}_\ell$, where inside the layer, $\varphi(t)$ and $\psi(t)$ are transformed by independent learnable linear layers A_θ and B_θ individually to fit the size of the Fourier and physical domain features. After the modulation, features from both domains are added and activated by the nonlinear activation function σ.

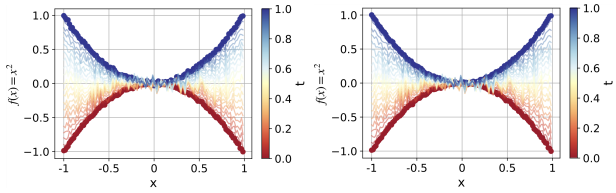
The FFT's inherent flexibility enables dynamic adjustment of spatial resolution through downsampling or upsampling, mirroring the multi-scale feature extraction in standard UNet architectures. This contrasts with conventional FNO implementations that maintain fixed intermediate resolution, as our U-shaped design facilitates simultaneous capture of low- and high-frequency features through multi-resolution processing. We preserve UNet-style skip connections by concatenating physical-domain features from each downsampling stage with corresponding upsampling outputs along the channel dimension, thereby doubling the codomain dimension for subsequent operations. This modified architecture outperforms baseline FNO implementations through three key advantages: (1) superior performance with fewer parameters, (2) faster convergence rates, and (3) enhanced suitability for operator approximation in diffusion bridge problems.

5 EXPERIMENTS

5.1 Functional Brownian bridges

Although the main motivation of our proposed method is for nonlinear processes, we spend some time evaluating cylindrical Brownian motion in various settings. This is because in this case we also know the true bridge processes, and therefore can fully evaluate the method against the truth. For a Hilbert space $\mathcal{U}$ with basis k_j, the cylindrical Brownian motion can be defined as $\sum_{i=1}^{\infty} w_i(t)e_i(t)$, where w_i are independent 1-dimensional Brownian motions. This sum does not converge in $\mathcal{U}$, but can be shown to converge in another Hilbert space (Da Prato and Zabczyk, 2014).

Quadratic functions: We first study the cylindrical Brownian motion of quadratic function evolutions, whose score has a closed form. For the start/target functions, we choose $f(x) = ax^2+\varepsilon$, where $a = \{1, -1\}$ and $\varepsilon \sim \mathcal{N}(0, 10^{-4})$ is used to equip the set with nonzero measure, as used in Phillips et al. (2022). We choose the bounded grid of $[-1, 1]$ to evaluate the function, and the grid is uniformly discretized into a finite number of points. Such a discretization is equivalent to choosing a finite basis and projecting it into finite dimensions. However, the projected process should be consistent under arbitrary discretizations. We show it by training the neural operator under a low-resolution scheme (8 points distributed within $[-1, 1]$) and evaluate it in a high-resolution case (128 points). Note that the learned bridge (Figure 2a) shows high consistency with the true Brownian bridge (Figure 2b), even on a much finer grid.



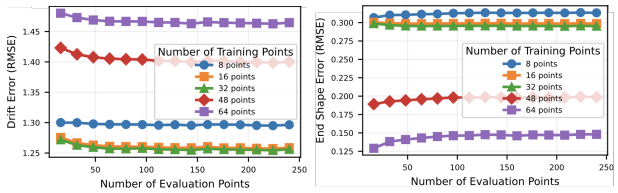
(a) Learned bridge (b) Ground truth

Figure 2: Qualitative results for Brownian bridges between two quadratic functions; (a) One sample from the learned reversed Brownian bridge, evaluated at 128 evenly distributed points; (b) One sample from the true reversed Brownian bridge, simulated with the same random seed as (a) for comparison.

Ellipses: We further test our method by simulating the Brownian bridge between 2D ellipses, which acts as the simplest 2D stochastic shape matching task whilst still holding a known form to which we can compare. An ellipse can be treated as a function from $[0, 1] \subset \mathbb{R}$ to $\mathcal{S}^1 \subset \mathbb{R}^2$, and can be characterized by finite points along the outlines. We are trying to bridge two ellipses with different axes. As we expect our method to show consistency under different levels of discretization, we train the same model with different numbers of training sample points and evaluate it on more points. Then we compare the model output at all discrete time steps against the true drift term of the Brownian bridge. Figure 3a and Figure 3b show the results. The model demonstrates consistent generalization from finite training samples to infinite-dimensional processes, with errors converging under finer discretization. Fewer training points yield lower drift error but higher end shape discrepancies, as reduced dimensionality facilitates drift estimation while neglecting higher Fourier modes. This truncation implicitly zeros unlearned basis elements, producing over-smoothed trajectories that amplify terminal errors despite accurate drift recovery. Figure 4 visualizes these truncation effects in Brownian bridges between ellipses.

The neural operator demonstrates superior accuracy and memory efficiency for drift estimation compared to traditional coefficient-based approaches. To our

knowledge, Baker et al. (2024) represents the sole prior work addressing general infinite-dimensional diffusion bridges through truncated Fourier coefficient learning via MLPs. We establish a Brownian motion benchmark where closed-form solutions exist, unlike nonlinear scenarios where analytical solutions become intractable. Table 1 compares the mean squared error (MSE) of drift estimates against ground truth across all time steps, contrasting our neural operator with the Fourier network in Baker et al. (2024). Notably, our operator achieves consistent error levels even with increased evaluation points (i.e., larger basis sets), whereas the Fourier network exhibits error escalation with higher spatial resolution. Furthermore, our architecture achieves comparable accuracy with substantially fewer parameters. Additional implementation details and results are provided in Section 8.2.



(a) Drift error (b) End shape error

Figure 3: Quantive evaluation of the model on different levels of discretization; (a) root MSE (RMSE) between the model's output and the ground true drift term evaluated on the whole trajectory; (b) RMSE between the end of estimated trajectories and the true target shape; All the statistics are done for 64 independent samplings.

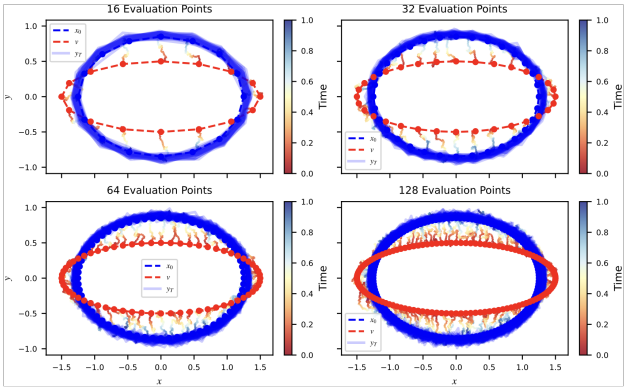


Figure 4: Visualization of Brownian bridges between ellipse shapes evaluated under different levels of discretizations, the training is done on 16 points. The blue shade represents 64 independent samples of end shape, and only one colored sample of the trajectories is shown with different colors indicating time steps.

Spheres: Since the construction of our proposed neural operator does not put limitations on the dimension of the domain of the functions, we demonstrate it by modeling the bridges between 3D spheres, where the sphere is considered as a 2D manifold parameterized by a function from $[0, \pi] \times [0, 2\pi] \subset \mathbb{R}^2$ to $\mathcal{S}^2 \subset \mathbb{R}^3$. We discretize the domain evenly into a $m \times m$ grid and thus represent the sphere function evaluation by a $[m, m, 3]$-shaped array. During inference, we sample a denser $m' \times m'$ grid where $m' > m$, which represents a function evaluation with a higher resolution. Figure 5 presents a comparison between the estimated bridge and the true bridge, demonstrating an almost indistinguishable difference. Note that the evaluated grid is three times as dense as the training grid, which requires eight times more evaluation points, but our model can still produce a consistent estimation.

5.2 Nonlinear stochastic landmark matching

We finally evaluate our method on real data. Specifically, we are interested in modeling the stochastic change in butterfly morphometry characterized by discrete landmarks over time. In the phylogenetic analysis, such bridge processes model the transitions along the edge between nodes in a phylogenetic tree, where the nodes store the observed data coming from the specimens. Developing the fast bridge simulation approach facilitates large-scale phylogenetic tree simulation and biological inference. However, linear processes are not suitable for modeling the landmark dynamics since they may cause the unexpected collapse of landmarks, corresponding to the intersections and overlapping of different parts of the shape. To address this, Sommer et al. (2021); Arnaudon et al. (2022) used the *stochastic flow of landmarks* defined by the SDE:

$$\mathrm{d}X_t = Q(X_t)\mathrm{d}W_t \tag{25a}$$

$$Q(X_t)(f(\xi)) = \int_{\mathbb{R}^2} k(X_t(\xi), \zeta) f(\zeta)\mathrm{d}\zeta \tag{25b}$$

where $X_t \in \mathcal{H}$, $Q(X_t) : \mathcal{H} \to \mathcal{H}$ is a Hilbert-Schmidt operator, and $k : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2$ is a smooth kernel (normally chosen as a Gaussian kernel). Equation (25a) is a nonlinear stochastic process that strengthens the correlation between adjacent landmarks to avoid collapse, and X_t is a diffeomorphism for all t (Kunita, 1990). We examine our method of modeling the conditional process for Equation (25a) on real butterfly shapes that are obtained by processing the raw images from gbif.org (GBIF.Org User, 2024). Images are segmented with the Python package *Segment Anything* (Kirillov et al., 2023) and *Gounding Dino* (Liu et al., 2023). The assigned landmarks are interpolated to be evenly distributed along the outlines, then aligned by the R package *Geomorph v.4.06* (Adams and Otárola-Castillo, 2013) and normalized. Figure 6 shows the nonlinear shape bridges between two selected species, Papilio polytes (red) and Parnassius honrathi (blue).

	Neural operator (#modes in 1st Fourier layer)			Fourier coeffs net (#truncated modes)		
#eval pts	8	16	32	8	16	32
32	1.5966	1.5894	1.5389	3.1133	5.7329	9.7032
64	1.5959	1.5815	1.5271	3.1092	5.7279	9.7080
128	1.5937	1.5779	1.5198	3.1085	5.7313	9.7389
256	1.5984	1.5768	1.5142	3.1091	5.7317	9.7273
#params	131,314	185,586	294,130	220,608	875,392	3,487,488

Table 1: Comparison between the neural operator and the MLP for Fourier coefficients, tested on Brownian bridges between ellipses.

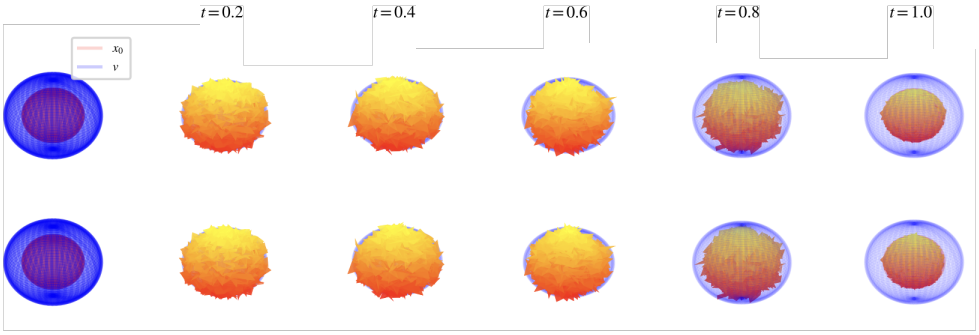
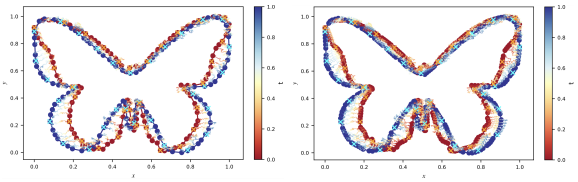


Figure 5: Visualization of the Brownian bridges between two nested spheres with different radii. The top row is the estimated bridge, and the bottom row is the ground true path with the same seed. The intermediate shapes are plotted in light orange. The training is done on 16×16 grid size and evaluated on 48×48 grid size.

We treat the closed outline shape of butterflies as functions $([0, 1] \subset \mathbb{R}) \to \mathbb{R}^2$. The model is trained on 32 evenly spaced landmarks and evaluated on more landmarks. We can see from the visualization of trajectories that no landmarks collide during the evolution, and the relative positions between landmarks remain. As the number of evaluation points increase, the model can give reasonable interpolations between landmarks. We also test the model on more butterfly species that are close on the phylogenetic tree in Section 8.2.4, where the simulated trajectory represents the morphological evaluation.



(a) Evaluate on 128 points (b) Evaluate on 256 points

Figure 6: Visualization of the nonlinear shape bridges between two butterfly shapes. The cyan crosses mark the landmarks used for training.

6 CONCLUSION

This paper introduces a novel approach for simulating nonlinear diffusion bridges in infinite-dimensional spaces via operator learning, where we first identify the infinite-dimensional time-reversed diffusion bridge, and propose an objective and time-dependent neural operator architecture to learn the time reversal. We validate our approach through several functional diffusion bridge test cases, where our method showcases its effectiveness of high-resolution zero-shot training and generalization to various shapes, together with consistency under different levels of discretization. Future work will be dedicated to developing a direct forward bridge learning scheme without reversing the bridge and incorporating the endpoints to avoid redundant training for simulating bridges on trees along edges.

Acknowledgements

The authors thank the anonymous reviewers for their valuable comments and suggestions, which have helped improve the clarity and quality of this work. This research was supported by Villum Foundation Grant 40582, the Novo Nordisk Foundation Grant NNF18OC0052000, and Danmarks Frie Forskningsfond Grant 10.46540/3103-00296B.

References

Jian Qing Shi and Taeryon Choi. *Gaussian process regression analysis for functional data*. CRC press, 2011.

Leonard CG Rogers and David Williams. *Diffusions, markov processes, and martingales: Volume 1, foundations*, volume 1. Cambridge university press, 2000.

Giulio Franzese, Giulio Corallo, Simone Rossi, Markus Heinonen, Maurizio Filippone, and Pietro Michiardi. Continuous-time functional diffusion processes. *Advances in Neural Information Processing Systems*, 36, 2024.

Jae Hyun Lim, Nikola B Kovachki, Ricardo Baptista, Christopher Beckham, Kamyar Azizzadenesheli, Jean Kossaifi, Vikram Voleti, Jiaming Song, Karsten Kreis, Jan Kautz, et al. Score-based diffusion models in function space. *arXiv preprint arXiv:2302.07400*, 2023.

Jakiw Pidstrigach, Youssef Marzouk, Sebastian Reich, and Sven Wang. Infinite-dimensional diffusion models for function spaces. *arXiv e-prints*, pages arXiv–2302, 2023.

Elizabeth Louise Baker, Gefan Yang, Michael L Severinsen, Christy Anna Hipsley, and Stefan Sommer. Conditioning non-linear and infinite-dimensional diffusion processes. *arXiv preprint arXiv:2402.01434*, 2024.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23 (7):1661–1674, 2011.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

Chin-Wei Huang, Jae Hyun Lim, and Aaron C Courville. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34:22863–22876, 2021.

Jeremy Heng, Valentin De Bortoli, Arnaud Doucet, and James Thornton. Simulating diffusion bridges with score matching. *arXiv preprint arXiv:2111.07243*, 2021.

JMC Clark. The simulation of pinned diffusions. In *29th IEEE conference on decision and control*, pages 1418–1420. IEEE, 1990.

Bernard Delyon and Ying Hu. Simulation of conditioned diffusion and application to parameter estimation. *Stochastic Processes and their Applications*, 116(11):1660–1675, 2006.

Moritz Schauer, Frank Van Der Meulen, and Harry Van Zanten. Guided proposals for simulating multi-dimensional diffusion bridges, 2017.

Marcin Mider, Moritz Schauer, and Frank Van der Meulen. Continuous-discrete smoothing of diffusions. *Electronic Journal of Statistics*, 15(2):4295–4342, 2021.

H. Chau, J. L. Kirkby, D. H. Nguyen, D. Nguyen, N. Nguyen, and T. Nguyen. An efficient method to simulate diffusion bridges. *Statistics and Computing*, 34(4):131, August 2024. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-024-10439-z. URL `https://link.springer.com/10.1007/s11222-024-10439-z`.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=PxTIG12RRHS`.

Lorenzo Baldassari, Ali Siahkoohi, Josselin Garnier, Knut Solna, and Maarten V de Hoop. Conditional score-based diffusion models for bayesian inference in infinite dimensions. *Advances in Neural Information Processing Systems*, 36, 2024.

Paul Hagemann, Sophie Mildenberger, Lars Ruthotto, Gabriele Steidl, and Nicole Tianjiao Yang. Multilevel diffusion: Infinite dimensional score-based diffusion models for image generation. *arXiv preprint arXiv:2303.04772*, 2023.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021.

Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion schrödinger bridge matching. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhicong Tang, Tiankai Hang, Shuyang Gu, Dong Chen, and Baining Guo. Simplified diffusion schrödinger bridge. *arXiv preprint arXiv:2403.14623*, 2024.

James Thornton, Michael Hutchinson, Emile Mathieu, Valentin De Bortoli, Yee Whye Teh, and Arnaud Doucet. Riemannian diffusion schrödinger bridge. *arXiv preprint arXiv:2207.03024*, 2022.

Ulrich G Haussmann and Etienne Pardoux. Time reversal of diffusions. *The Annals of Probability*, pages 1188–1205, 1986.

Annie Millet, David Nualart, and Marta Sanz. Time reversal for infinite-dimensional diffusions. *Probability theory and related fields*, 82(3):315–347, 1989.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar, et al. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2020a.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020b.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Yesom Park, Jaemoo Choi, Changyeon Yoon, Myungjoo Kang, et al. Learning pde solution operator for continuous modeling of time-series. *arXiv preprint arXiv:2302.00854*, 2023.

Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators. *arXiv e-prints*, pages arXiv–2204, 2022.

Giuseppe Da Prato and Jerzy Zabczyk. *Stochastic equations in infinite dimensions*. Cambridge university press, 2014.

Angus Phillips, Thomas Seror, Michael Hutchinson, Valentin De Bortoli, Arnaud Doucet, and Emile Mathieu. Spectral diffusion processes. *arXiv preprint arXiv:2209.14125*, 2022.

Stefan Horst Sommer, Moritz Schauer, and Frank van der Meulen. Stochastic flows and shape bridges. In *Statistics of Stochastic Differential Equations on Manifolds and Stratified Spaces (hybrid meeting)*, number 48 in Oberwolfach Reports, pages 18–21. Mathematisches Forschungsinstitut Oberwolfach, 2021. doi: 10.4171/OWR/2021/48. null ; Conference date: 03-10-2021 Through 09-10-2021.

Alexis Arnaudon, Frank van der Meulen, Moritz Schauer, and Stefan Sommer. Diffusion bridges for stochastic hamiltonian systems and shape evolutions. *SIAM Journal on Imaging Sciences*, 15(1):293–323, 2022.

Hiroshi Kunita. *Stochastic flows and stochastic differential equations*, volume 24. Cambridge university press, 1990.

GBIF.Org User. Occurrence download, 2024. URL `https://www.gbif.org/occurrence/download/0075323-231120084113126`.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

Dean C Adams and Erik Otárola-Castillo. geomorph: an r package for the collection and analysis of geometric morphometric shape data. *Methods in ecology and evolution*, 4(4):393–399, 2013.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2024. URL `http://github.com/google/flax`.

Checklist

1. For all models and algorithms presented, check if you include:

(a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]: We describe the mathematical settings in detail in the methodology section.

(b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]: We evaluate the performance of our model under different levels of discretization for both training and inference stages.

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]: The link to the code repository is provided in the supplementary materials

2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. [No]: We do not explicitly state all of the assumptions we've made for the theoretical demonstrations.

(b) Complete proofs of all theoretical results. [Yes]: We attach the detailed proofs of two theorems in the supplementary materials.

(c) Clear explanations of any assumptions. [Yes]: We made explanations for the assumptions we have explicitly made.

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

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]: The results can be fully reproduced using the provided code in the repository.

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]: We detailed the training procedure in the supplementary materials

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]: As explained in the main text and supplementary materials

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]: As mentioned in the supplement materials.

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

(a) Citations of the creator If your work uses existing assets. [Yes]: We've cited all the used open-sourced resources and tools.

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

(c) New assets either in the supplemental material or as a URL, if applicable. [No]: We do not create new assets.

(d) Information about consent from data providers/curators. [Not Applicable]: All the resources are open-sourced

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]: We do not have sensible content.

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

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

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

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

Infinite-dimensional Diffusion Bridge Simulation via Operator Learning: Supplementary Materials

7 PROOFS

In this section, we detail the proofs of the two theorems and one lemma stated in the paper.

7.1 Proof of Theorem 2

Proof. We follow the setup in (Millet et al., 1989). Let $\{e_i\}_{i=1}^{\infty}$ be an orthonormal basis for $\mathcal{H} \ni X_t^\star$, and $\{k_j\}_{j=1}^{\infty}$ be an orthonormal basis for $\mathcal{U} \ni W_t$. Then suppose that for each $i \in \mathbb{Z}$, there exists a finite set $I(i)$, such that for all s, $[a(s,x)]_{ij} \coloneqq \sum_{\ell}^{\infty}[g(s,x)]_{i\ell}[g(s,x)]_{j\ell} = 0$ if $j \notin I(i)$. Then we split $X_t^\star$ into an infinite number of finite vectors; specifically, we denote $\mathbf{x}^{\star(i)} = \{[X_t^\star]_j; j \in I(i)\} \in \mathbb{R}^{\#I(i)}$ to be the vector consisting of all the components of $X_t^\star$ indexed by the entries in $I(i)$, with $[X_t^\star]_i = \langle X_t^\star, e_j\rangle_{\mathcal{H}}$, and its complement to be $\hat{\mathbf{x}}^{\star(i)} = \{[X_t^\star]_j; j \notin I(i)\}$, and let $\mathbf{z}^{\star(i)} = \{z_j \in \mathbb{R};\ j \notin I(i)\}$. Therefore, $X_t^\star = \{\mathbf{x}^{\star(i)}, \hat{\mathbf{x}}^{\star(i)}\}$. We assume that the conditional law of $\mathbf{x}^{\star(i)}$ given $\hat{\mathbf{x}}^{\star(i)} = \mathbf{z}^{\star(i)}$ has a density $\tilde{p}_t(\mathbf{x}^{\star(i)} \mid \mathbf{z}^{\star(i)})$ with respect to the Lebesgue measure in the vector space spanned by $\{e_j, j \in I(i)\}$. We refer to (Millet et al., 1989, Section 5) for the existence of such a density. Consider the infinite-dimensional bridge process $X^\star$ defined by Equation (4), written in terms of the bases e_i and k_j:

$$\mathrm{d}X_t^\star = [f(t, X_t^\star) + a(t, X_t^\star)\nabla \log h(t, X_t^\star)]_i \mathrm{d}t + \sum_{j=1}^{\infty}[g(t, X_t^\star)]_{ij}\mathrm{d}[W_t^{\mathbb{P}^\star}]_j. \tag{26}$$

Then apply the time reversal theorem (Millet et al., 1989, Theorem 4.3) to obtain the reversed bridge $Y^\star$ in terms of basis coefficients:

$$\mathrm{d}Y_t^\star = [\overline{f^\star}(t, Y_t^\star)]_i \mathrm{d}t + \sum_{j=1}^{\infty}[\overline{g^\star}(t, Y_t^\star)]_{ij}\mathrm{d}[W_t^{\mathbb{P}^\star}]_j, \tag{27}$$

where,

$$\begin{aligned}[\overline{f^\star}(s,x)]_i = &-[f(T-s,x) + a(T-s,x)\nabla \log h(T-s,x)]_i \\ &+ \tilde{p}_{T-s}^{-1}(\mathbf{x}^{\star(i)} \mid \mathbf{z}^{\star(i)}) \sum_{j\in I(i)} \nabla_{[x]_j}\{[a(T-s,x)]_{ij}\tilde{p}_{T-s}(\mathbf{x}^{\star(i)}|\mathbf{z}^{\star(i)})\},\end{aligned} \tag{28a}$$

$$[\overline{g^\star}(s,x)]_{ij} = [g(T-s,x)]_{ij}. \tag{28b}$$

Using $p^{-1}\nabla p = \nabla \log p$, we can rewrite $[\overline{f^\star}(s,x)]_i$ as

$$\begin{aligned}[\overline{f^\star}(s,x)]_i = &-[f(T-s,x) + a(T-s,x)\nabla \log h(T-s,x)]_i \\ &+ \sum_{j\in I(i)} \nabla_{[x]_j}[a(T-s,x)]_{ij} + \sum_{j\in I(i)}[a(T-s,x)]_{ij}\nabla_{[x]_j}\log \tilde{p}_{T-t}(\mathbf{x}^{\star(i)}|\mathbf{z}^{\star(i)}).\end{aligned} \tag{29}$$

We then focus on the sum

$$-[a(T-s,x)\nabla \log h(T-s,x)]_i + \sum_{j\in I(i)}[a(T-s,x)]_{ij}\nabla_{[x]_j}\log \tilde{p}_{T-s}(\mathbf{x}^{\star(i)}|\mathbf{z}^{\star(i)}). \tag{30}$$

Noting that $[a(t,x)]_{ij} = 0$ for any $j \notin I(i)$, we expand the first term into basis elements:

$$[a(T-s,x)\nabla \log h(T-s,x)]_i = \sum_{j\in I(i)}[a(T-s,x)]_{ij}\nabla_{[x]_j}\log h(T-s,x). \tag{31}$$

Hence, Equation (30) becomes

$$\sum_{j\in I(i)} [a(T-t,x)]_{ij}\nabla_{[x]_j} \log\left[\frac{\tilde{p}_{T-t}(\mathbf{x}^{\star(i)} \mid \mathbf{z}^{\star(i)})}{h(T-t,x)}\right]. \tag{32}$$

Finally, we note that $\tilde{p}_{T-s}(\mathbf{x}^{\star(i)} \mid \mathbf{z}^{\star(i)})$ is the h-transformation of $p_{T-s|0}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0))$, where $p_{T-s|0}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0))$ is the conditional law of $\mathbf{x}^{(i)}$ given $\hat{\mathbf{x}}^{(i)} = \mathbf{z}^{(i)}$ and $X_0 = x_0$. Hence,

$$\tilde{p}_{T-s}(\mathbf{x}^{\star(i)} \mid \mathbf{z}^{\star(i)}) = p_{T-s}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0))h(T-s,x). \tag{33}$$

Substituting it back gives the desired form stated in the theorem

$$[\bar{f}(s,x)]_i = -[f(T-s,x)]_i + \sum_{j\in I(i)} \nabla_{[x]_j}[a(T-s,x)]_{ij} + \sum_{j\in I(i)} [a(T-s,x)]_{ij}\nabla_{[x]_j} \log p_{T-s|0}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_0)). \tag{34}$$

□

7.2 Proof of Lemma 1

Proof. By definition, the KL divergence between $\mathbb{P}$ and $\mathbb{P}^{(\theta)}$ can be expressed as the expectation:

$$\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)}) = \mathbb{E}_{\mathbb{P}}\left[\frac{\mathrm{d}\mathcal{L}^{\mathbb{P}}}{\mathrm{d}\mathcal{L}^{\mathbb{P}^{(\theta)}}}(Y)\right] = \mathbb{E}_{\mathbb{P}}\left[\log\frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{P}^{(\theta)}}\right], \tag{35}$$

where $\mathcal{L}^{\mathbb{P}}, \mathcal{L}^{\mathbb{P}^{(\theta)}}$ denote the laws of Y under $\mathbb{P}, \mathbb{P}^{(\theta)}$ respectively. With a change of variable notation from y to x with reversing the time direction from $T-s$ to s , the Radon-Nikodym derivative $\frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{P}^{(\theta)}}$ can be computed by Girsanov's theorem (see e.g., Da Prato and Zabczyk (2014)) as:

$$\frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{P}^{(\theta)}} = \exp\left\{\int_0^T \left\{g^*(\bar{f}-f^{(\theta)})\right\}(s,x)\,\mathrm{d}W_s^{\mathbb{P}} + \frac{1}{2}\int_0^T \left\|\left\{g^*(\bar{f}-f^{(\theta)})\right\}(s,x)\right\|^2_{Q^{1/2}(\mathcal{U})}\,\mathrm{d}s\right\}. \tag{36}$$

The first Itô integral is a zero-mean martingale, therefore, it immediately reads:

$$\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)}) = \frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\int_0^T \left\|\left\{g^*(\bar{f}-f^{(\theta)})\right\}(s,x)\right\|^2_{Q^{1/2}(\mathcal{U})}\,\mathrm{d}s\right], \tag{37}$$

where $\|\cdot\|_{Q^{1/2}(\mathcal{U})}$ denotes the norm in $Q^{1/2}(\mathcal{U})$. Introducing the basis element $g^*(s,x)e_i \in Q^{1/2}(\mathcal{U})$ and using the equivalence $\langle g^*e_i, g^*e_j\rangle_{Q^{1/2}(\mathcal{U})} = \langle gg^*e_i, e_j\rangle_{\mathcal{H}}$, we see:

$$\begin{aligned}
\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)}) &= \frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\int_0^T \sum_{i=1}^{\infty}\left\langle\left\{g^*(\bar{f}-f^{(\theta)})\right\}(s,x), g^*(s,x)e_i\right\rangle^2_{Q^{1/2}(\mathcal{U})}\,\mathrm{d}s\right] && \text{(38a)}\\
&= \frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\int_0^T \sum_{i=1}^{\infty}\left\langle\{\bar{f}-f^{(\theta)}\}(s,x), gg^*(s,x)e_i\right\rangle^2_{\mathcal{H}}\,\mathrm{d}s\right] && \text{(38b)}\\
&= \frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\int_0^T \sum_{i=1}^{\infty}\lambda_i[\{\bar{f}-f\}(s,x)]_i^2\,\mathrm{d}s\right] && \text{(38c)}\\
&= \frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\int_0^T \sum_{i=1}^{\infty}\lambda_i[\Psi(s,x)]_i^2\,\mathrm{d}s\right]. && \text{(38d)}
\end{aligned}$$

The second last equivalence comes from $a(s,x)$ being diagonal in $\{e_i\}$, i.e., $a(e_i) = \lambda_i e_i$ for some scalar λ_i. To derive $\Psi(s,x)$, consider the definitions of $\bar{f}$ and $f^{(\theta)}$:

$$[\bar{f}(T-s,x)]_i = -[f(s,x)]_i + \frac{1}{p_s(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})}\sum_{j\in I(i)} \nabla_j\left([a(s,x)]_{ij}p_s(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})\right) \tag{39a}$$

$$[f^{(\theta)}(T-s,x)]_i = -[f(s,x)]_i + \sum_{j\in I(i)} \nabla_{[x]_j}[a(s,x)]_{ij} + [\mathcal{G}^{(\theta)}(s,x)]_i. \tag{39b}$$

We focus on the product in the RHS of Equation (39a):

$$\frac{1}{p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)})} \sum_{j \in I(i)} \nabla_j \left([a(s,x)]_{ij} p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}) \right) \tag{40a}$$

$$= \frac{1}{p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)})} \left\{ \sum_{j \in I(i)} p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}) \nabla_{[x]_j} [a(s,x)]_{ij} + \sum_{j \in I(i)} [a(s,x)]_{ij} \nabla_{[x]_j} p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}) \right\} \tag{40b}$$

$$= \sum_{j \in I(i)} \nabla_{[x]_j} [a(s,x)]_{ij} + \sum_{j \in I(i)} [a(s,x)]_{ij} \nabla_{[x]_j} \log p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}). \tag{40c}$$

Then it turns out that:

$$\Psi(s,x) = \sum_{j \in I(i)} [a(s,x)]_{ij} \nabla_{[x]_j} \log p_s(\mathbf{x}^{(i)} \mid \mathbf{z}^{(i)}) - [\mathcal{G}^{(\theta)}(s,x)]_i \tag{41}$$

□

7.3 Proof of Theorem 3

Proof. We follow the derived expression of $\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)})$ in Lemma 1, and denote $\nabla_j := \nabla_{[x]_j} = \partial/\partial[x]_j$ for simplicity:

$$\mathrm{D_{KL}}(\mathbb{P}||\mathbb{P}^{(\theta)}) = \frac{1}{2}\mathbb{E}_{\mathbb{P}} \left[\int_0^T \sum_{i=1}^{\infty} \lambda_i \left\{ \sum_{j \in I(i)} [a(s,x)]_{ij} \nabla_{[x]_j} \log p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) - [\mathcal{G}^{(\theta)}(s,x)]_i \right\}^2 \mathrm{d}t \right] \tag{42a}$$

$$= \underbrace{\frac{1}{2}\mathbb{E}_{\mathbb{P}} \left[\int_0^T \sum_{i=1}^{\infty} \lambda_i [\mathcal{G}^{(\theta)}(t,x)]_i^2 \mathrm{d}t \right]}_{C_1}$$

$$+ \underbrace{\frac{1}{2}\mathbb{E}_{\mathbb{P}} \left[\int_0^T \sum_{i=1}^{\infty} \lambda_i \left\{ \sum_{j \in I(i)} [a(t,x)]_{ij} \nabla_j \log p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) \right\}^2 \mathrm{d}t \right]}_{C_2}$$

$$- \underbrace{\mathbb{E}_{\mathbb{P}} \left[\int_0^T \sum_{i=1}^{\infty} \lambda_i \left\langle \sum_{j \in I(i)} [a(t,x)]_{ij} \nabla_j \log p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}t \right]}_{C_3}. \tag{42b}$$

We then focus on C_3: for any time partition $(t_n)_{n=1}^N$, we define $\mathbf{x}_{t_n}^{(i)}, \hat{\mathbf{x}}_{t_n}^{(i)}$, and $\mathbf{z}_{t_n}^{(i)}$ for X_{t_n} in the same manner as $\mathbf{x}^{(i)}, \hat{\mathbf{x}}^{(i)}$, and $\mathbf{z}^{(i)}$. Then we can write C_3 as

$$C_3 = \mathbb{E}_{\mathbb{P}} \left[\sum_{n=1}^{N} \int_{t_{n-1}}^{t_n} \sum_{i=1}^{\infty} \lambda_i \left\langle \sum_{j \in I(i)} [a(t,x)]_{ij} \nabla_j \log p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}t \right] \tag{43a}$$

$$= \sum_{n=1}^{N} \sum_{i=1}^{\infty} \lambda_i \int_{\Omega = \mathrm{span}(\{e_j; j \in I(i)\})} p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) \int_{t_{n-1}}^{t_n} \left\langle \sum_{j \in I(i)} [a(t,x)]_{ij} \nabla_j \log p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}\mathbf{x}^{(i)} \mathrm{d}t \tag{43b}$$

$$= \sum_{n=1}^{N} \sum_{i=1}^{\infty} \lambda_i \int_{\Omega} \int_{t_{n-1}}^{t_n} \left\langle \sum_{j \in I(i)} [a(t,x)]_{ij} \nabla_j p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}\mathbf{x}^{(i)} \mathrm{d}t. \tag{43c}$$

By using the Chapman-Kolmogorov equation (Da Prato and Zabczyk, 2014, Corollary 9.15), one can write $p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})$ as

$$p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) = \int_\Omega p_0(\mathbf{x}_0^{(i)}|\mathbf{z}_0^{(i)})\mathrm{d}\mathbf{x}_0^{(i)} \int_\Omega p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)}))p_{t_{n-1}|0}(\mathbf{x}_{t_{n-1}}^{(i)} \mid (\mathbf{z}_{t_{n-1}}^{(i)}, \mathbf{x}_0^{(i)}, \mathbf{z}_0^{(i)}))\mathrm{d}\mathbf{x}_{t_{n-1}}^{(i)}. \quad (44)$$

We can then differentiate Equation (44):

$$\begin{aligned}
&\nabla_j p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) \\
&= \int_\Omega p_0(\mathbf{x}_0^{(i)} \mid \mathbf{z}_0^{(i)})\mathrm{d}\mathbf{x}_0^{(i)} \int_\Omega \nabla_j p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)}))p_{t_{n-1}|0}(\mathbf{x}_{t_{n-1}}^{(i)} \mid (\mathbf{z}_{t_{n-1}}^{(i)}, \mathbf{x}_0^{(i)}, \mathbf{z}_0^{(i)}))\mathrm{d}\mathbf{x}_{t_{n-1}}^{(i)} && (45a)\\
&= \int_\Omega p_0(\mathbf{x}_0^{(i)} \mid \mathbf{z}_0^{(i)})\mathrm{d}\mathbf{x}_0^{(i)} \int_\Omega \nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)})) \\
&\quad p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)}))p_{t_{n-1}|0}(\mathbf{x}_{t_{n-1}}^{(i)} \mid (\mathbf{z}_{t_{n-1}}^{(i)}, \mathbf{x}_0^{(i)}, \mathbf{z}_0^{(i)}))\mathrm{d}\mathbf{x}_{t_{n-1}}^{(i)} && (45b)\\
&= \int_\Omega p_0(\mathbf{x}_0^{(i)} \mid \mathbf{z}_0^{(i)})\mathrm{d}\mathbf{x}_0^{(i)} \int_\Omega \nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)})) \\
&\quad p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)}))p_{t_{n-1}|0}(\mathbf{x}_{t_{n-1}}^{(i)} \mid (\mathbf{z}_{t_{n-1}}^{(i)}, \mathbf{x}_0^{(i)}, \mathbf{z}_0^{(i)}))\mathrm{d}\mathbf{x}_{t_{n-1}}^{(i)}. && (45c)
\end{aligned}$$

Substituting it back into Equation (43c) gives

$$\begin{aligned}
C_3 &= \sum_{n=1}^{N}\sum_{i=1}^{\infty}\lambda_i \int_\Omega p_t(\mathbf{x}^{(i)}|\mathbf{z}^{(i)}) \\
&\qquad \int_{t_{n-1}}^{t_n} \left\langle \sum_{j\in I(i)} [a(t,x)]_{ij}\nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, \mathbf{x}_{t_{n-1}}^{(i)}, \mathbf{z}_{t_{n-1}}^{(i)})), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}\mathbf{x}^{(i)}\mathrm{d}t && (46a)\\
&= \mathbb{E}_{\mathbb{P}}\left[\sum_{n=1}^{N}\sum_{i=1}^{\infty}\lambda_i \left\langle \sum_{j\in I(i)} [a(t,x)]_{ij}\nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, X_{t_{n-1}})), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}t\right]. && (46b)
\end{aligned}$$

We then claim that:

$$L(\theta) = \frac{1}{2}\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n} \mathbb{E}_{\mathbb{P}}\left[\sum_{i=1}^{\infty}\lambda_i \left\{[\mathcal{G}^{(\theta)}(t,X_t)]_i - b_t(X_t, x_{t_{n-1}})\right\}^2\right]\mathrm{d}t = C_1 + C_4 - C_3, \quad (47)$$

where,

$$C_4 = \frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n}\sum_{i=1}^{\infty}\lambda_i \left\{\sum_{j\in I(i)} [a(t,x)]_{ij}\nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_{t_{n-1}}))\right\}^2 \mathrm{d}t\right]. \quad (48)$$

To show this, we expand Equation (47):

$$\begin{aligned}
L(\theta) = &\underbrace{\frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n}\sum_{i=1}^{\infty}\lambda_i [\mathcal{G}^{(\theta)}(t,x)]_i^2 \mathrm{d}t\right]}_{C_1} \\
&+ \underbrace{\frac{1}{2}\mathbb{E}_{\mathbb{P}}\left[\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n}\sum_{i=1}^{\infty}\lambda_i \left\{\sum_{j\in I(i)} [a(t,x)]_{ij}\nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_{t_{n-1}}))\right\}^2 \mathrm{d}t\right]}_{C_4} \\
&- \underbrace{\mathbb{E}_{\mathbb{P}^\star}\left[\sum_{n=1}^{N}\int_{t_{n-1}}^{t_n}\sum_{i=1}^{\infty}\lambda_i \left\langle \sum_{j\in I(i)} [a(t,x)]_{ij}\nabla_j \log p_{t|t_{n-1}}(\mathbf{x}^{(i)} \mid (\mathbf{z}^{(i)}, x_{t_{n-1}})), [\mathcal{G}^{(\theta)}(t,x)]_i \right\rangle \mathrm{d}t\right]}_{C_3}. \quad (49)
\end{aligned}$$

Then take $C = C_4 - C_2$, which is θ-independent, and we prove the statement in the theorem.

□

8 EXPERIMENT DETAILS

8.1 Numerical implementations

When sampling from Equation (2), we first choose a grid $\xi = \{\xi_i\}_{i=1}^M$ as a M-discretized domain of X_t. Such a discretization varies according to different X_t. We also sample M independent 1-dimensional Brownian motions $W_t^{(1)}, W_t^{(2)}, \ldots, W_t^{(M)}$ on a discrete time partition $\{t_n\}_{n=1}^N$, such that for any j-th Brownian path, $\Delta W^{(j)} = W_{t_n}^{(j)} - W_{t_{n-1}}^{(j)} \sim \mathcal{N}(0, \delta t)$. We sample from a system of finite-dimensional SDEs:

$$\mathrm{d}X_t[\xi_i] = f(t, X_t[\xi_i])\mathrm{d}t + \sum_{j=1}^{M} [g(t, X_t[\xi_i])]_j \mathrm{d}W_t^{(j)}, \quad i = 1, 2, \ldots, M \tag{50}$$

Equation (50) can be used as the approximation of the original SDE in Equation (2), and can be solved using different numerical approaches. We choose Euler-Maruyama, which updates $X_t[\xi_i]$ according to:

$$X_{t_n}[\xi_i] = X_{t_{n-1}}[\xi_i] + f(t, X_{t_{n-1}}[\xi_i])\delta t + \sum_{j=1}^{M} [g(t, X_{t_{n-1}}[\xi_i])]_j \Delta W^{(j)}, \quad X_{t_0}[\xi_i] = X_0[\xi_i]. \tag{51}$$

The objective function in Equation (21) can be then be approximated by:

$$\hat{L}(\theta) = \frac{1}{2B} \sum_{n=1}^{N} \sum_{b=1}^{B} \sum_{i=1}^{M} \left[\lambda_i \left(\mathcal{G}^{(\theta)}(t_n, X_{t_n}^{(b)}[\xi_i]) + (\Delta t)^{-1} \cdot \{ X_{t_n}^{(b)}[\xi_i] - X_{t_{n-1}}^{(b)}[\xi_i] - \Delta t \cdot f(t_{n-1}, X_{t_{n-1}}^{(b)}[\xi_i] \} \right)^2 \right] \cdot \Delta t, \tag{52}$$

where $X_{t_n}^{(b)}[\xi_i]$ is a sample of $X_{t_n}[\xi_i]$ indexed by b, and $\Delta t = t_n - t_{n-1}$. We show the training and inference algorithms in Algorithm 1 and Algorithm 2.

All the experiments conducted in the main paper and supplementary materials are done on the platform with one NVIDIA RTX A6000 GPU and Intel(R) Xeon(R) Gold 5420+ @ 2.0GHz with 256GB memory and Ubuntu 22.04.4 LTS OS. The code for reproducing the results is available on Github.

Algorithm 1 Training

Require: Domain discretization ξ, initial state $X_0[\xi]$
Require: Total time length T, time step Δt
Require: Batch size B
Require: Initialized $\mathcal{G}_\theta$
Require: Drift f, diffusion g
Form discrete time grid $\{t_0 = 0, t_1, \ldots, t_{N-1}, t_N = T\}, n = \lfloor T/\Delta t \rfloor$
while Not converged **do**
  for do $n \leftarrow 1, \ldots, N$
    for do $b \leftarrow 1, \ldots, B$
      Sample $\varepsilon_{t_n}^{(b)} \sim \mathcal{N}(0, \Delta t \cdot \mathbf{I}_M)$, i.i.d
      Update $X_t[\xi]$: $X_{t_n}^{(b)}[\xi_i] \leftarrow X_{t_{n-1}}^{(b)} + f(t_{n-1}, X_{t_{n-1}}^{(b)}[\xi_i])\Delta t + \sum_{j=1}^{M} [g(t_{n-1}, X_{t_{n-1}}^{(b)}[\xi_i])]_j [\varepsilon_{t_n}^{(b)}]_j$
      Compute $a(t_n, X_{t_n}[\xi_i])$ and extract the diagonal entries λ_i.
    end for
  end for
  Compute $\hat{L}(\theta)$ by Equation (52)
  Perform gradient descent step on $\hat{L}(\theta)$
end while

Algorithm 2 Inference

Require: Domain discretization ξ', initial state $Y_0[\xi']$
Require: Total time length T, time step Δt
Require: Pretrained $\mathcal{G}^{(\theta)}$
Require: Drift f, diffusion g
Form discrete time grid $\{t_0 = 0, t_1, \ldots, t_{N-1}, t_N = T\}, N = \lfloor T/\Delta t \rfloor$
for do $n \leftarrow 1, \ldots, N$
    Sample $\varepsilon_{t_n} \sim \mathcal{N}(0, \Delta t \cdot \mathbf{I}_M)$, i.i.d
    Update $Y_t^\star[\xi']$: $Y_{t_n}^\star[\xi_i'] \leftarrow Y_{t_{n-1}}^\star[\xi_i'] + \{f(t_{n-1}, Y_{t_{n-1}}^\star[\xi_i']) + \mathcal{G}^{(\theta)}(t_{n-1}, Y_{t_{n-1}}^\star[\xi_i'])\}\Delta t$
        $+ \sum_{j=1}^{M} \left\{ \nabla_j [(gg^\intercal)(t_{n-1}, Y_{t_{n-1}}^\star[\xi_i'])]_j \Delta t + [g(t_{n-1}, Y_{t_{n-1}}^\star[\xi_i'])]_j [\varepsilon_{t_n}]_j \right\}$
end for
$X_{t_n}^\star[\xi'] \leftarrow Y_{t_{N-n}}^\star[\xi']$

8.2 Additional experiment results

8.2.1 Quadratic functions

We model the stochastic dynamics in the function space as the Brownian motion:

$$\mathrm{d}X_t = \sigma \mathrm{d}W_t \tag{53}$$

with $\sigma = 0.1$, which has a closed-form time reversed bridge:

$$\mathrm{d}Y_t^\star = \frac{x_0 - Y_t^\star}{t}\mathrm{d}t + \sigma \mathrm{d}W_t. \tag{54}$$

We designed our operator to have 6 Fourier layers. The details are shown in Table 2. We train with 10,000 i.i.d. batched samples and with a batch size of 16. We chose the Adam optimizer with an initial learning rate of 0.001 and cosine decay to 1e-5 as 80% of the training is finished. Figure 7 provides more samples of Brownian bridges between two quadratic functions.

Layer	Input	Output	Grid size	Fourier modes	Activation
Lifting	$u : \mathbb{R} \to \mathbb{R}$	$v_0 : \mathbb{R} \to \mathbb{R}^{16}$	8	-	-
Down1	$v_0 : \mathbb{R} \to \mathbb{R}^{16}$	$v_1 : \mathbb{R} \to \mathbb{R}^{16}$	8	6	GeLU
Down2	$v_1 : \mathbb{R} \to \mathbb{R}^{16}$	$v_2 : \mathbb{R} \to \mathbb{R}^{32}$	4	4	GeLU
Down3	$v_2 : \mathbb{R} \to \mathbb{R}^{32}$	$v_3 : \mathbb{R} \to \mathbb{R}^{64}$	2	2	GeLU
Up1	$v_3 : \mathbb{R} \to \mathbb{R}^{64}$	$v_4 : \mathbb{R} \to \mathbb{R}^{32}$	2	2	GeLU
Up2	$v_4 : \mathbb{R} \to \mathbb{R}^{32}$	$v_5 : \mathbb{R} \to \mathbb{R}^{16}$	4	4	GeLU
Up3	$v_5 : \mathbb{R} \to \mathbb{R}^{16}$	$v_6 : \mathbb{R} \to \mathbb{R}^{16}$	8	6	GeLU
Projection	$v_6 : \mathbb{R} \to \mathbb{R}^{16}$	$v : \mathbb{R} \to \mathbb{R}$	8	-	-

Table 2: Neural operator structure for quadratic function experiments

8.2.2 Ellipses

The SDE we used for modeling the stochastic development of ellipses follows Equation (53) as well. The starting ellipse (blue) with parameters $a = 1.25, b = 0.85$, and the target ellipse with parameters $a = 1.5, b = 0.5$, where a, b stand for the lengths of semi-major and semi-minor axes of the ellipse. For the neural operator design, it is shown in Table 3. We train with 10,000 i.i.d. batched samples and with a batch size of 16. We chose the Adam optimizer with an initial learning rate of 5e-4 and cosine decay to 5e-6 as 80% of the training is finished. We also compare the performance of our proposed neural operator architecture against the MLP structures used in Baker et al. (2024). Mainly, we compare the MSE between the estimated score and the true score under different levels of discretizations as shown in Table 1, while keeping the number of trained bases fixed. In Baker et al. (2024), such a hyperparameter is directly reflected on the input dimensions of the network, since the Fourier transformation has been done before the neural network, while in our implementation, it depends on the kept

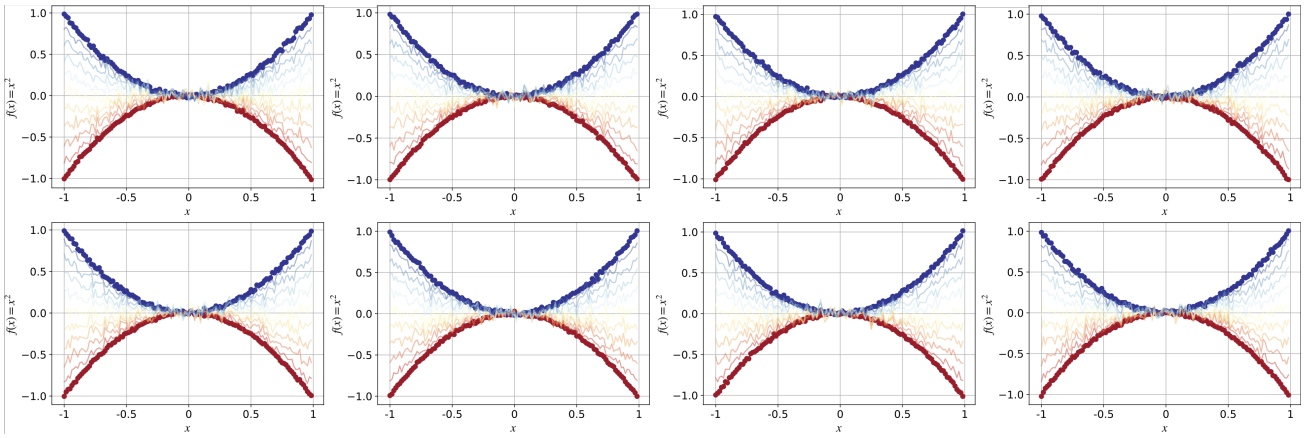


Figure 7: More samples of estimated Brownian bridges between two quadratic functions

number of Fourier modes in the first Fourier layer after the lifting layer, because all the rest of the Fourier layers are operating on the outputs from the first layer, which determines how much information is fed into the network. All the benchmarks are conducted on the same platform with the same *FLAX* (Heek et al., 2024) framework to avoid potential unfairness. The code implementation of the method in Baker et al. (2024) is slightly modified from the public repository https://github.com/libbylbaker/infsdebridge. We note that our proposed neural operator structures have much lower error with smaller model sizes. We use the same Adam optimizer with the same learning rate of 2e-4 and cosine learning rate decay scheduler. The batch sizes for both models are set to be 32, and the models are both trained for 100,000 iterations. Figure 8 shows more samples of Brownian bridges.

Layer	Input	Output	Grid size	Fourier modes	Activation
Lifting	$u : \mathbb{R} \to \mathbb{R}^2$	$v_0 : \mathbb{R} \to \mathbb{R}^{16}$	16	-	-
Down1	$v_0 : \mathbb{R} \to \mathbb{R}^{16}$	$v_1 : \mathbb{R} \to \mathbb{R}^{16}$	16	8	GeLU
Down2	$v_1 : \mathbb{R} \to \mathbb{R}^{16}$	$v_2 : \mathbb{R} \to \mathbb{R}^{32}$	8	6	GeLU
Down3	$v_2 : \mathbb{R} \to \mathbb{R}^{32}$	$v_3 : \mathbb{R} \to \mathbb{R}^{64}$	4	4	GeLU
Up1	$v_3 : \mathbb{R} \to \mathbb{R}^{64}$	$v_4 : \mathbb{R} \to \mathbb{R}^{32}$	4	4	GeLU
Up2	$v_4 : \mathbb{R} \to \mathbb{R}^{32}$	$v_5 : \mathbb{R} \to \mathbb{R}^{16}$	8	6	GeLU
Up3	$v_5 : \mathbb{R} \to \mathbb{R}^{16}$	$v_6 : \mathbb{R} \to \mathbb{R}^{16}$	16	8	GeLU
Projection	$v_6 : \mathbb{R} \to \mathbb{R}^{16}$	$v : \mathbb{R} \to \mathbb{R}^2$	16	-	-

Table 3: Neural operator structure for ellipse experiments

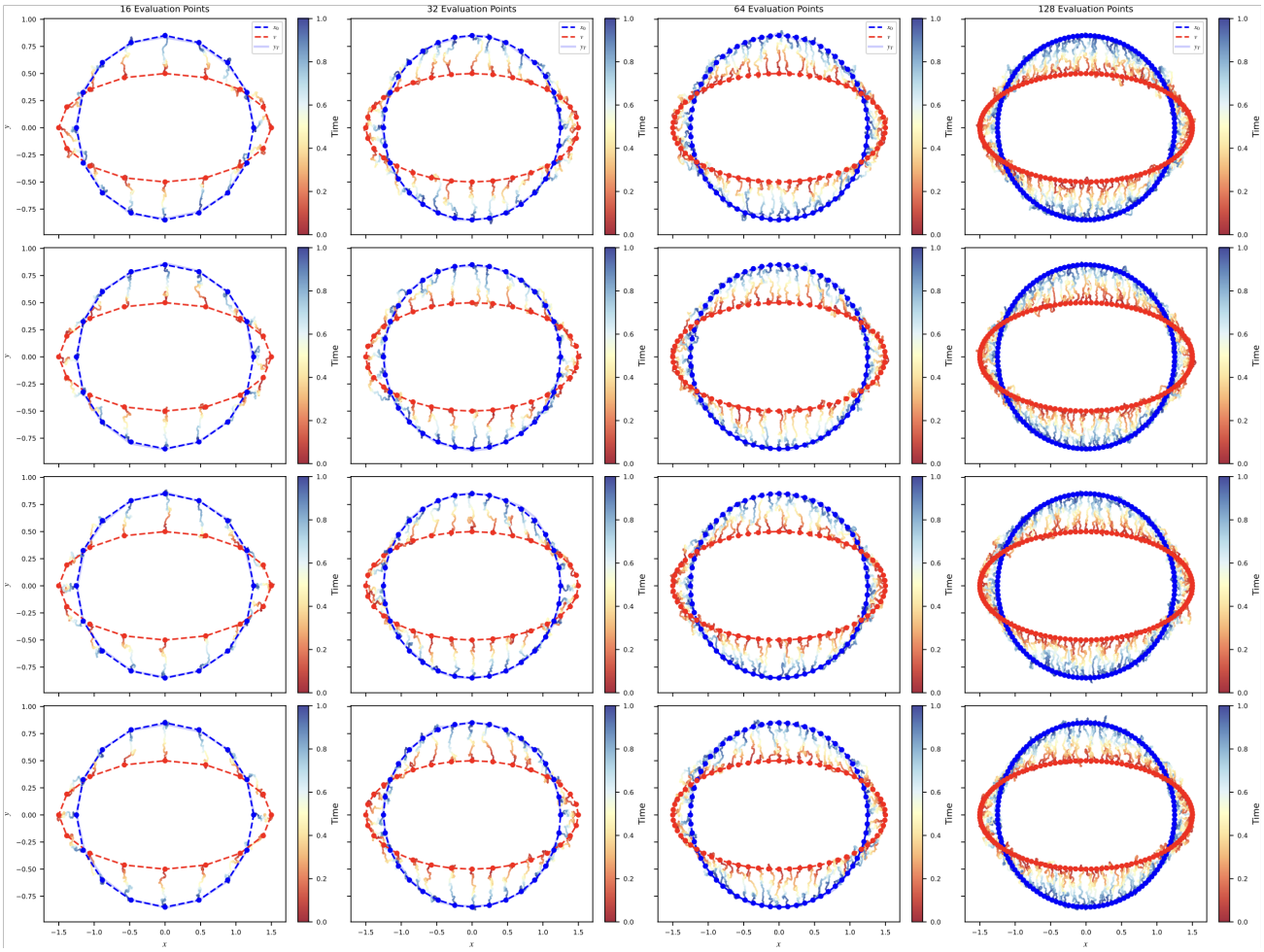


Figure 8: More samples of estimated Brownian bridges between two ellipses, evaluated under different levels of discretization

8.2.3 Spheres

Unlike the previous two experiments, spheres should be treated as functions $\mathbb{R}^2 \to (\mathcal{S}^2 \subset \mathbb{R}^3)$. The samples then have the shape of $(m, m, 3)$ and we modify the neural operator structure to fit in it. Table 4 shows the detailed structure and Figure 9 shows more samples.

Layer	Input	Output	Grid size	Fourier modes	Activation
Lifting	$u : \mathbb{R}^2 \to \mathbb{R}^3$	$v_0 : \mathbb{R}^2 \to \mathbb{R}^{32}$	(16, 16)	-	-
Down1	$v_0 : \mathbb{R}^2 \to \mathbb{R}^{32}$	$v_1 : \mathbb{R}^2 \to \mathbb{R}^{32}$	(16, 16)	(12, 12)	GeLU
Down2	$v_1 : \mathbb{R}^2 \to \mathbb{R}^{32}$	$v_2 : \mathbb{R}^2 \to \mathbb{R}^{64}$	(16, 16)	(8, 8)	GeLU
Down3	$v_2 : \mathbb{R}^2 \to \mathbb{R}^{64}$	$v_3 : \mathbb{R}^2 \to \mathbb{R}^{128}$	(8, 8)	(4, 4)	GeLU
Up1	$v_3 : \mathbb{R}^2 \to \mathbb{R}^{128}$	$v_4 : \mathbb{R}^2 \to \mathbb{R}^{64}$	(8, 8)	(4, 4)	GeLU
Up2	$v_4 : \mathbb{R}^2 \to \mathbb{R}^{64}$	$v_5 : \mathbb{R}^2 \to \mathbb{R}^{32}$	(16, 16)	(8, 8)	GeLU
Up3	$v_5 : \mathbb{R}^2 \to \mathbb{R}^{64}$	$v_6 : \mathbb{R}^2 \to \mathbb{R}^{32}$	(16, 16)	(12, 12)	GeLU
Projection	$v_6 : \mathbb{R}^2 \to \mathbb{R}^{32}$	$v : \mathbb{R}^2 \to \mathbb{R}^3$	(16, 16)	-	-

Table 4: Neural operator structure for sphere experiments

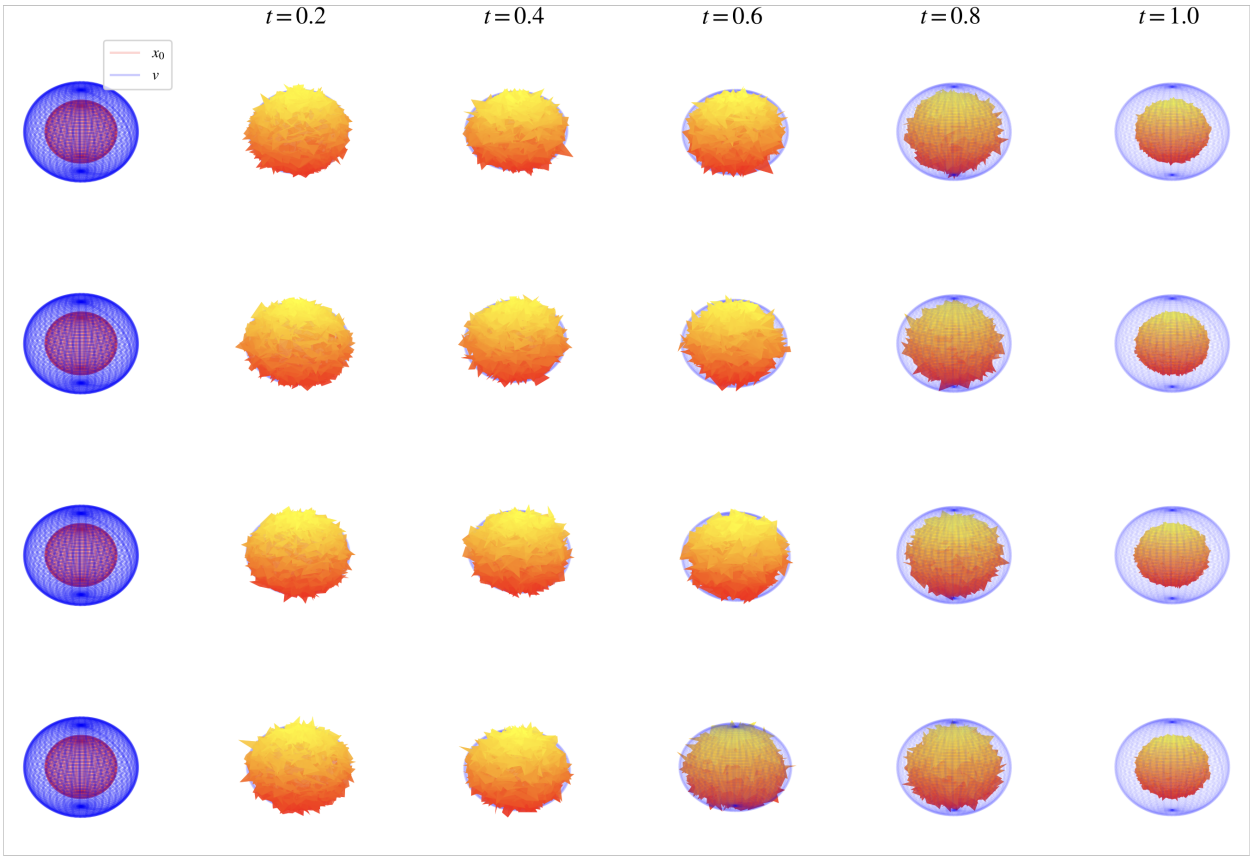


Figure 9: More samples of estimated Brownian bridges between two spheres

8.2.4 Butterflies

As introduced in Equation (25a), we let the butterfly shapes be subsets of $\mathbb{R}^2$, and choose the kernel k to be the 2D-Gaussian kernel, i.e., $k(x, y) = \sigma \exp(-\|x - y\|^2/\kappa)$. Specifically, we set the domain in $\mathbb{R}^2$ to be finite as $[-0.5, 1.5]^2$ and discretize it with the resolution of 50×50. We then choose the kernel with $\sigma = 0.04$ and $\kappa = 0.02$, which allows correlations only within small areas. We then design the operator as Table 5. We train with 20,000 i.i.d. batched samples and with a batch size of 16. We chose the Adam optimizer with an initial learning rate of 0.001 and cosine decay to 1e-5 as 80% of the training is finished. Figure 10 and Figure 11 show the nonlinear shape bridges between five chosen butterfly species and their specimen photos.

Layer	Input	Output	Grid size	Fourier modes	Activation
Lifting	$u : \mathbb{R} \to \mathbb{R}^2$	$v_0 : \mathbb{R} \to \mathbb{R}^{16}$	32	-	-
Down1	$v_0 : \mathbb{R} \to \mathbb{R}^{16}$	$v_1 : \mathbb{R} \to \mathbb{R}^{16}$	32	16	GeLU
Down2	$v_1 : \mathbb{R} \to \mathbb{R}^{16}$	$v_2 : \mathbb{R} \to \mathbb{R}^{32}$	16	8	GeLU
Down3	$v_2 : \mathbb{R} \to \mathbb{R}^{32}$	$v_3 : \mathbb{R} \to \mathbb{R}^{64}$	8	6	GeLU
Down4	$v_3 : \mathbb{R} \to \mathbb{R}^{64}$	$v_4 : \mathbb{R} \to \mathbb{R}^{64}$	8	6	GeLU
Up1	$v_4 : \mathbb{R} \to \mathbb{R}^{64}$	$v_5 : \mathbb{R} \to \mathbb{R}^{64}$	8	6	GeLU
Up2	$v_5 : \mathbb{R} \to \mathbb{R}^{64}$	$v_6 : \mathbb{R} \to \mathbb{R}^{32}$	8	6	GeLU
Up3	$v_6 : \mathbb{R} \to \mathbb{R}^{32}$	$v_7 : \mathbb{R} \to \mathbb{R}^{16}$	16	8	GeLU
Up4	$v_7 : \mathbb{R} \to \mathbb{R}^{16}$	$v_8 : \mathbb{R} \to \mathbb{R}^{16}$	32	16	GeLU
Projection	$v_8 : \mathbb{R} \to \mathbb{R}^{16}$	$v : \mathbb{R} \to \mathbb{R}^2$	32	-	-

Table 5: Neural operator structure for butterfly experiments

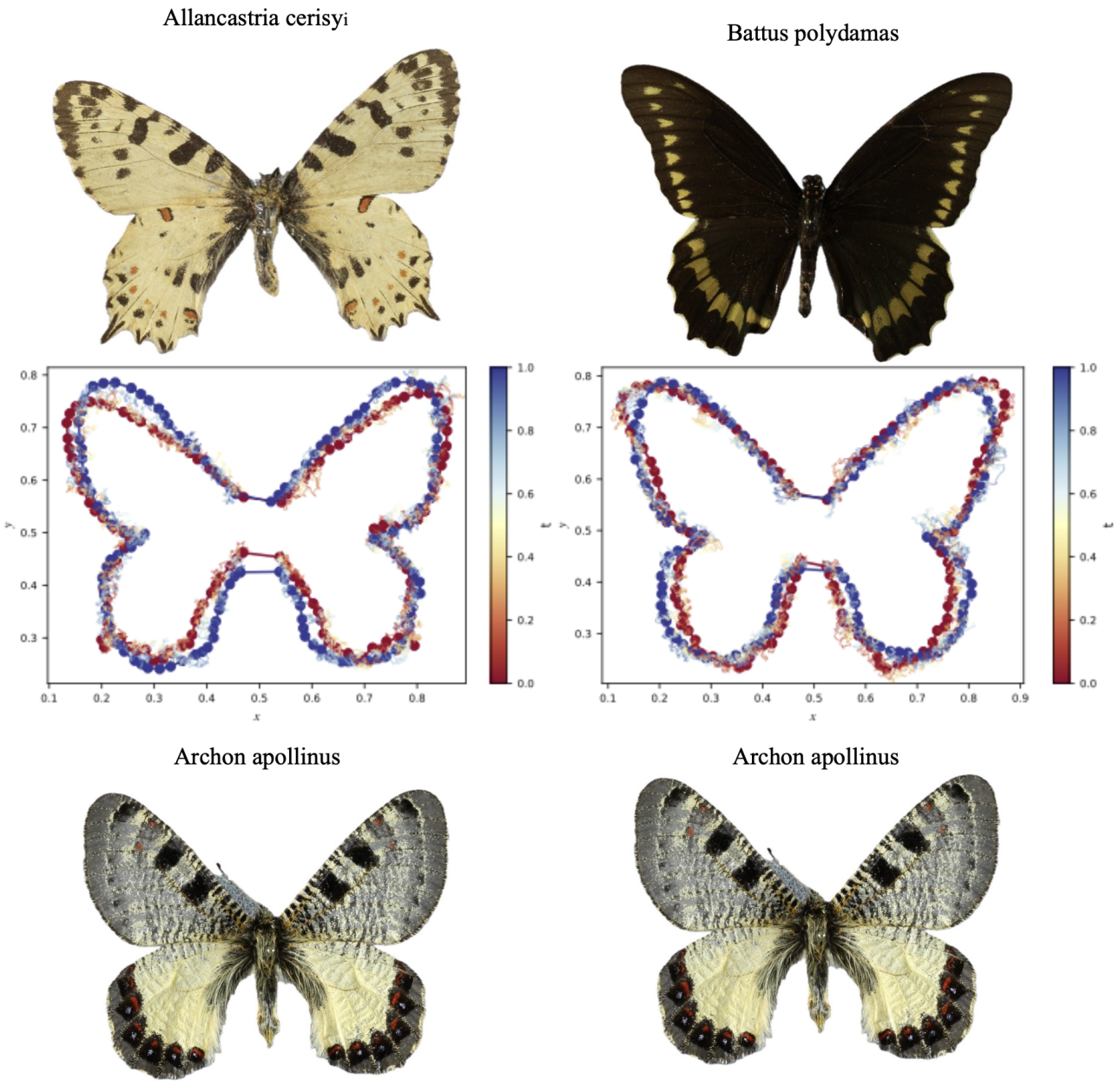


Figure 10: The additional bridge simulations between different species, the bridges are constructed between *Archon apollinus* (blue) and *Allancastria cerisyi/Battus polydamas* (red).

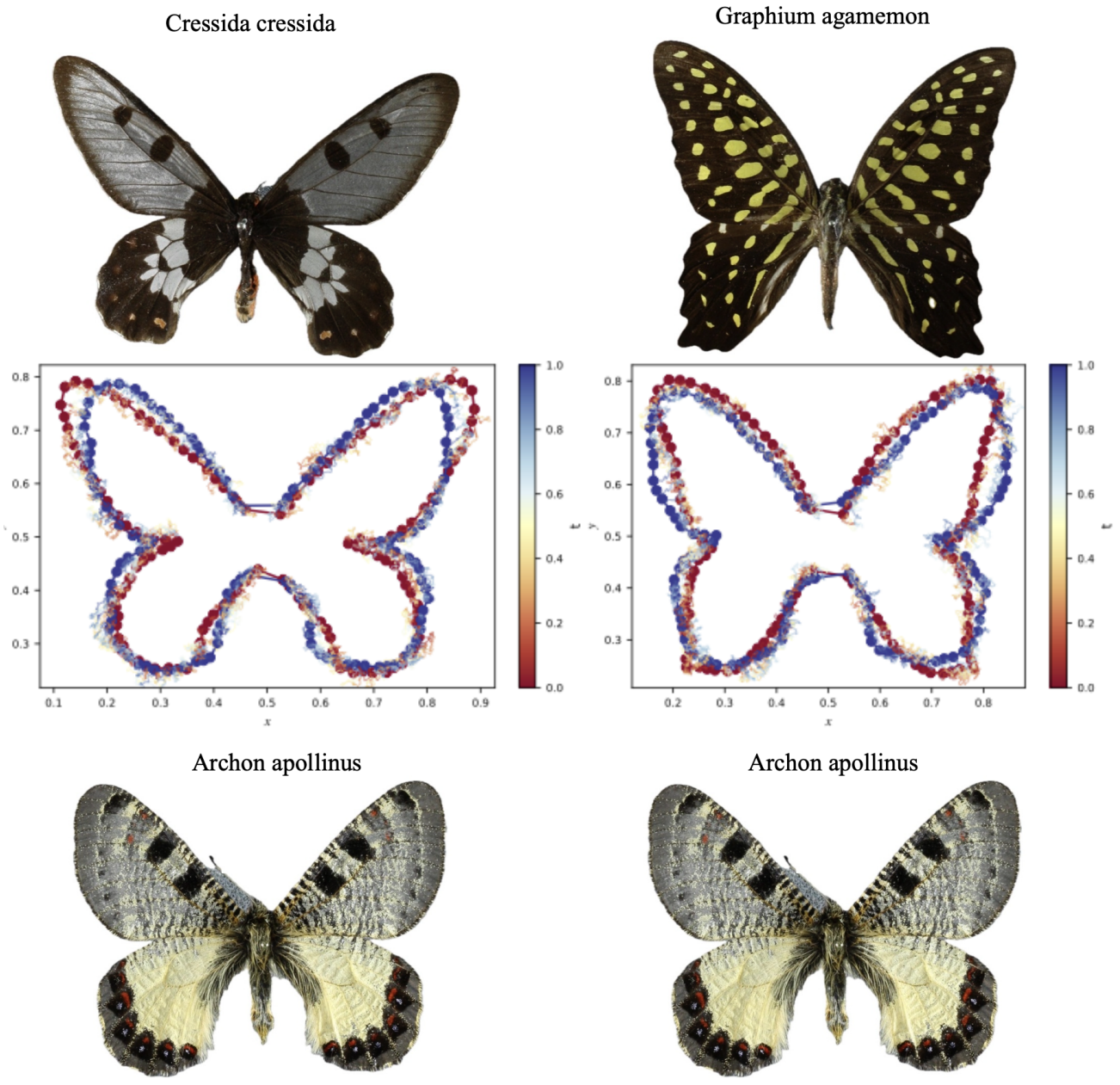


Figure 11: The additional bridge simulations between different species, the bridges are constructed between *Archon apollinus* (blue) and *Cressida cressida/Graphium agamemon* (red).