Beyond Schrödinger Bridges: A Least-Squares Approach for Learning Stochastic Dynamics with Unknown Volatility

Renato Berlinghieri [*1] **Yunyi Shen** [*1] **Tamara Broderick** [1]

Abstract

Scientists are often interested in inferring the underlying stochastic dynamics of systems from population-level snapshot data, where individual trajectories are unavailable. For example, in biological studies of immune cell activation, mRNA concentrations are measured at a single time point per cell because the measurement process kills the cells. Existing methods based on Schrödinger bridge techniques rely on Kullback–Leibler divergence and assume known, constant volatility, limiting their applicability in realistic settings where volatility may be unknown or varying. In this work, we propose a new framework that directly matches the joint distribution of the state (e.g., mRNA levels) and the time of observation using maximum mean discrepancy. This approach reduces to a least-squares formulation in distributional space and motivates an R^2-type goodness-of-fit measure for model inspection and comparison. We show in our experiments that the proposed method outperforms existing Schrödinger bridge–based baselines in forecasting and is robust to unknown volatility and missing observations.

1. Introduction

Scientists are often interested in learning stochastic dynamics from population-level snapshots rather than complete individual trajectories. That is, instead of tracking each individual's dynamics over time, they infer the underlying dynamics by analyzing aggregated observations of the entire population at different points in time. For example, biologists often consider a population of inactive immune cells and are interested in understanding how gene expression — as measured by mRNA levels — changes when these cells transform into active cells capable of killing cancer. This is an important problem, as gaining deeper insights in this process could potentially aid in developing methods to prevent or treat cancer. The issue with this cell data is that measuring mRNA concentration typically requires killing the cells. As a result, scientists can only obtain mRNA data for each cell at a single time snapshot, preventing the tracking of any single cell over multiple time points.

Recently, *Schrödinger bridge* (SB) methods (Pavon, Trigila, and Tabak, 2021; De Bortoli, Thornton, Heng, and Doucet, 2021; Vargas, Thodoroff, Lamacraft, and Lawrence, 2021; Koshizuka and Sato, 2022; Wang, Jennings, and Gong, 2023) and their multi-marginal extensions (Shen et al., 2025; Zhang, 2024; Guan et al., 2024; Chen et al., 2024; Lavenant et al., 2021) have shown promise for reconstructing distributions of trajectories from these limited snapshot data. However, because SB-based approaches rely on Kullback–Leibler divergence to measure agreement between data and a nominal model, they cannot handle unknown or changing volatility. In many practical scenarios, volatility is difficult to determine in advance, which limits the applicability of these methods.

In this work, we provide a new method to learn stochastic dynamics that relaxes the strong assumption of a known, constant volatility. We observe that under typical "population dynamics" settings (Lavenant et al., 2021) — where each cell has a latent trajectory evaluated at only a single time point — the data can be treated as if they were sampled i.i.d. from the joint distribution of the *state* (mRNA level) and the *time* of evaluation. Consequently, we propose a method to match this joint distribution directly, which does not rely on observing all states; notably, our approach also remains robust even when some states are missing. Furthermore, we show that when maximum mean discrepancy (MMD, Gretton et al., 2012) with a specific choice of kernel is used as the distance metric for our matching problem, the estimation reduces to a least-squares formulation in distributional space. This perspective motivates an R^2-type goodness-of-fit measure that facilitates model inspection and comparison. Our experiments show that the proposed

[*]Equal contribution [1]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. Correspondence to: Renato Berlinghieri <renb@mit.edu>, Yunyi Shen <yshen99@mit.edu>.

Accepted at 7th Symposium on Advances in Approximate Bayesian Inference (AABI 2025).

method outperforms existing SB-based baselines, and can handle both missing observations and unknown volatility.

2. Setup and Background

Data Setup. We consider single-cell mRNA measurements collected at I time points $t_1 < t_2 < \cdots < t_I$, with $t_1 = 0$ and possibly uneven spacing. At each t_i, we observe N_{t_i} cells, each providing a single mRNA concentration measurement in $\mathbb{R}^d$, denoted $\boldsymbol{Y}_{t_i}^{n_i}$. Since a cell dies after the mRNA measurement is taken, each cell appears only once in the dataset. The total number of observations is $N = \sum_{i=1}^{I} N_{t_i}$, and we write $\boldsymbol{Y}_{t_i}^{\text{all}}$ for all measurements at t_i.

Goal. If unmeasured, a cell's mRNA concentration would evolve continuously. Our goal is to infer a probabilistic model for these unobserved trajectories. Let $\boldsymbol{X}_t^{(i,n_i)}$ denote the latent trajectory of the n_ith cell observed at t_i, with its observed state being $\boldsymbol{Y}_{t_i}^{n_i} = \boldsymbol{X}_{t=t_i}^{(i,n_i)}$. We assume these trajectories are independent samples from an underlying latent distribution, which in turn implies that the observations are independent as well. We seek the best model within a given family to describe these trajectories.

Consequence of having only one observation per cell. Since each cell is only observed at a single time point, our dataset consists of independent but not identically distributed observations. This is because different cells are measured at different times, leading to variations in the distributions of their observed states. However, if we consider time itself as a random variable sampled from a distribution over all possible times, $h(t)$, we arrive at a key insight: the pairs $(\boldsymbol{Y}_{t_i}^{n_i}, t_i)$ can be viewed as independent and identically distributed (iid) samples from a joint distribution.

We remark that this formulation aligns with the way biological experiments are typically conducted. Instead of sequencing each cell immediately after collection, experimenters often tag each cell with a unique identifier that encodes the time it was sampled. Then, all cells are sequenced together in a single batch, with the time information retrieved from the tags. As a result, the dataset effectively consists of iid samples from the joint distribution of $(\boldsymbol{Y}_{t_i}^{n_i}, t_i)$.

Model. We model the latent trajectory of the (i, n_i) particle with an SDE driven by a d-dimensional Brownian motion $\boldsymbol{W}_t^{(i,n_i)}$, independent across particles:

$$\mathrm{d}\boldsymbol{X}_t^{(i,n_i)} = \boldsymbol{b}_0(\boldsymbol{X}_t^{(i,n_i)}, t)\mathrm{d}t + \boldsymbol{g}_0(\boldsymbol{X}_t^{(i,n_i)}, t)\mathrm{d}\boldsymbol{W}_t \quad (1)$$

with $\boldsymbol{X}_{t=0}^{n_i} \sim \pi_0$. We assume that the drift $\boldsymbol{b}_0(\cdot,\cdot) : \mathbb{R}^d \times [0, t_I] \to \mathbb{R}^d$ and initial marginal distribution π_0 are unknown. Unlike previous Schrödinger bridge works assuming known constant volatility (e.g. De Bortoli et al., 2021; Vargas et al., 2021; Koshizuka & Sato, 2022; Wang et al., 2023; Shen et al., 2025; Zhang, 2024; Guan et al., 2024; Chen et al., 2024), we also assume that the volatility $\boldsymbol{g}_0$ is unknown and can vary with $\boldsymbol{X}_t^{(i,n_i)}$ and t. Allowing for unknown volatility is important in many scientific applications because it represents the intrinsic level of noise that researchers often cannot predict or measure directly in advance. For example, in single-cell RNA sequencing, gene expression levels can vary significantly due to a combination of technical factors (e.g., amplification biases, dropout events) and biological factors (e.g., stochastic gene transcription). And in such scenarios, precisely quantifying the overall noise — i.e., the volatility — is very challenging.

Finally, we assume standard SDE regularity conditions. The first assumption below ensures a strong solution to the SDE exists; see Pavliotis (2016, Chapter 3, Theorem 3.1). The second ensures that the process does not exhibit unbounded variability.

Assumption 2.1 *The drifts and volatility are L and L'-Lipschitz respectively; i.e., for all $t \in [0, t_I]$, $\|\boldsymbol{b}_0(x,t) - \boldsymbol{b}_0(y,t)\| \le L\|x-y\|$ and $|\boldsymbol{g}_0(x,t) - \boldsymbol{g}_0(y,t)| \le L'\|x-y\|$, where $\|\cdot\|$ denotes the usual L^2 norm of a vector. And we have at most linear growth; i.e., there exist $K, K' < \infty$ and constant c such that $\|\boldsymbol{b}_0(y,t)\| < K\|y\| + c$ and $\|\boldsymbol{g}_0(y,t)\| < K'\|y\| + c'$.*

Assumption 2.2 *At each time step t_i, the distribution of the N_{t_i} particles has bounded second moments. Moreover, the initial distribution π_0 also has bounded second moments.*

3. Our method

Our method allows us to learn the optimal model within a given family to represent the distribution of latent cell trajectories, even when the volatility is unknown and may vary over time. In what follows, we describe our optimization problem and its solution, outline criteria for assessing model fit, and finally discuss how the method can be extended to accommodate missing data.

State distribution at snapshots. At each snapshot time t_i the ground truth dynamic eq. (1) determines a distribution of states (e.g., mRNA level) specific to that time. Unfortunately, we do not have access to this ground truth distribution of states. However, we observe N_{t_i} independent samples $\boldsymbol{Y}_{t_i}^1, \ldots, \boldsymbol{Y}_{t_i}^{N_{t_i}}$ from that distribution at time t_i. With these samples, we can approximate the true state distribution with the empirical measure,

$$\hat{f}_{t_i}(\cdot) = \frac{1}{N_{t_i}} \sum_{j=1}^{N_{t_i}} \delta_{\boldsymbol{Y}_{t_i}^{(j)}}(\cdot), \quad (2)$$

where $\delta_{\boldsymbol{Y}_{t_i}^{(j)}}$ denotes the Dirac delta measure centered at $\boldsymbol{Y}_{t_i}^{(j)}$. This empirical measure is an approximation of the true marginal distribution of cell states at time t_i.

Treating snapshot time as random. We consider observations collected at I distinct time points $\{t_I, \ldots, t_I\}$. In our modeling framework, we posit that there exists an underlying time distribution, $\hat{h}(\cdot)$, which reflects how snapshot times are sampled. We assume that this distribution is the empirical distribution over time — constructed directly from the data — that may assign different weights to different time points, depending on the number of observations collected. Letting N_{t_i} denote the number of cell measurements at time t_i, the empirical time distribution is given by

$$\hat{h}(\cdot) = \frac{1}{\sum_{i=1}^{I} N_{t_i}} \sum_{i=1}^{I} N_{t_i}\, \delta_{t_i}(\cdot) = \sum_{i=1}^{I} \hat{h}_{t_i}\, \delta_{t_i}(\cdot), \quad (3)$$

with weights

$$\hat{h}_{t_i} = \frac{N_{t_i}}{\sum_{i=1}^{I} N_{t_i}}.$$

Joint distribution of state and time. Instead of considering distributions of $\boldsymbol{Y}_{t_i}$ separately at each snapshot, we consider the joint distribution of state-time observation $(\boldsymbol{Y}_{t_i}, t_i)$. Under a candidate SDE model parameterized by $\boldsymbol{\theta}$, let $f_{\boldsymbol{\theta},t}(\cdot)$ denote the predicted marginal distribution of cell states at time t. By pairing this with the empirical time distribution $\hat{h}$, we obtain the predicted joint distribution under the candidate model is

$$f_{\boldsymbol{\theta}} = \hat{h} \cdot f_{\boldsymbol{\theta},t}. \quad (4)$$

This construction implies that the time-augmented observations $(\boldsymbol{Y}_{t_i}^{(j)}, t_i)$ are modeled as independent samples from $f_{\boldsymbol{\theta}}$. In contrast, the empirical joint distribution, which reflects the observed data, is formed by combining the empirical time measure with the state distributions at each time:

$$\hat{f} = \hat{h} \cdot \hat{f}_t. \quad (5)$$

Our Approach: Directly Matching the State–Time Joint Distributions. Our primary objective is to estimate the parameters $\boldsymbol{\theta}$ of the candidate SDE model by ensuring that the model-implied joint distribution, $f_{\boldsymbol{\theta}}$, aligns with the empirical joint distribution, $\hat{f}$, derived from the data. In other words, we wish to have the distribution predicted by our model accurately approximate the true (but unknown) joint distribution of state and time. To quantify the discrepancy between $f_{\boldsymbol{\theta}}$ and $\hat{f}$, we adopt the maximum mean discrepancy (MMD) (Gretton et al., 2012). The MMD measures the L^2 distance between the kernel mean embeddings of the two distributions in a reproducing kernel Hilbert space (RKHS).

A key insight is that matching the joint distribution in state–time space can be interpreted as a form of least squares regression in the kernel embedding space with respect to time. This connection is formalized in the following proposition, which decomposes the MMD between joint distributions into a weighted average of the MMDs computed for their conditional distributions:

Proposition 3.1 *Let $f(\boldsymbol{x}, \boldsymbol{t}) = f(\boldsymbol{x} \mid \boldsymbol{t})\, h(\boldsymbol{t})$ and $g(\boldsymbol{x}, \boldsymbol{t}) = g(\boldsymbol{x} \mid \boldsymbol{t})\, h(\boldsymbol{t})$ be joint distributions on $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{t} \in \mathcal{T}$, where $\mathcal{T}$ is a discrete set and $h(\boldsymbol{t})$ is a probability mass function. Suppose we define the kernel $K\big((\boldsymbol{x}, \boldsymbol{t}), (\boldsymbol{x}', \boldsymbol{t}')\big) = K_x(\boldsymbol{x}, \boldsymbol{x}')\, \delta(\boldsymbol{t} - \boldsymbol{t}')$, which factorizes as a positive definite kernel K_x on the state space and a delta kernel on the time space. Denote by MMD_K the MMD computed with kernel K. Then, the squared MMD between f and g can be decomposed as*

$$\mathrm{MMD}_K^2\left(f(\boldsymbol{x}, \boldsymbol{t}), g(\boldsymbol{x}, \boldsymbol{t})\right) = \sum_{\boldsymbol{t} \in \mathcal{T}} h^2(\boldsymbol{t})\ \mathrm{MMD}_{K_x}^2\Big(f(\cdot \mid \boldsymbol{t}), g(\cdot \mid \boldsymbol{t})\Big).$$

This proposition shows that the overall MMD between two joint distributions can be understood as a time-weighted average of the MMDs between the corresponding conditional state distributions. We provide a complete proof of proposition 3.1 in appendix A.

Choosing a delta function as the time kernel is essential for this result. With this choice, the joint MMD reduces to an average over discrete time steps. Specifically, the MMD squared between $f_{\boldsymbol{\theta}}$ and $\hat{f}$ can be expressed as

$$\mathrm{MMD}^2\left(f_{\boldsymbol{\theta}}, \hat{f}\right) = \sum_{i=1}^{I} \left(\frac{N_{t_i}}{\sum_{i=1}^{I} N_{t_i}}\right)^2 \mathrm{MMD}^2\left(f_{\boldsymbol{\theta},t_i}, \hat{f}_{t_i}\right) \quad (6)$$

Our method in practice. In practice, at each time step we approximate the MMD squared between $f_{\boldsymbol{\theta}}$ and $\hat{f}$ using its U-statistic estimator (Lemma 6, (Gretton et al., 2012)). To do so, we simulate M trajectories from the candidate model and record the state snapshots at time t_i, denoted by $\boldsymbol{Z}_{t_i}^m$ for $m = 1, \ldots, M$. The U-statistic estimator is then given by

$$\begin{aligned}\mathrm{MMD}_{\mathrm{U,U}}^2\left(f_{\boldsymbol{\theta},t_i}, \hat{f}_{t_i}\right) = &\frac{1}{N_{t_i}(N_{t_i}-1)} \sum_{n_i \neq n_i'} K(\boldsymbol{Y}_{t_i}^{n_i}, \boldsymbol{Y}_{t_i}^{n_i'}) \\ &- \frac{2}{N_{t_i} M} \sum_{n_i, m} K(\boldsymbol{Y}_{t_i}^{n_i}, \boldsymbol{Z}_{t_i}^{m}) \\ &+ \frac{1}{M(M-1)} \sum_{m \neq m'} K(\boldsymbol{Z}_{t_i}^{m}, \boldsymbol{Z}_{t_i}^{m'}).\end{aligned}$$

This estimator is both unbiased and consistent (Hall, 2004). Finally, we estimate our model parameters by minimizing

the overall loss function:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{I} \left(\frac{N_{t_i}}{\sum_{i=1}^{I} N_{t_i}} \right)^2 \mathrm{MMD}^2_{\mathrm{U,U}} \left(f_{\boldsymbol{\theta},t_i}, \hat{f}_{t_i} \right). \tag{7}$$

In our implementation, we use the radial basis function (RBF) kernel for the state space, where we determined the length scale by the median heuristic (Garreau et al., 2017) applied to pairwise distances in the observed data. Additionally, we normalize time to the interval $[0, 1]$. For optimization, we compute the gradients with respect to the parameters using the stochastic adjoint method (Li et al., 2020). We use the Adam optimizer to perform the parameter updates.

Picking $f_{\boldsymbol{\theta},t_i}$ in practice. In practice, we choose the candidate marginal distributions $f_{\boldsymbol{\theta},t_i}$ based on domain knowledge about the underlying process. In particular, we select a candidate SDE model that captures the key dynamics of the system under study. This SDE can either follow a parametric form, in which drift and volatility are governed by functions with finitely many parameters (as in the Lotka–Volterra experiment; see section 4.1), or a more flexible design that incorporates neural network architectures for the drift and/or volatility terms. For example, in the repressilator experiment (section 4.2), we compare a purely parametric model to a semiparametric approach in which we used a multilayer perceptron to approximate part of the drift in the system (see eq. (16) for more details). This neural network component allows the model to capture complex, nonlinear effects that would be difficult to represent in a strictly parametric setting.

Handling missing dimensions. In many practical applications — such as our mRNA sequencing example — it is common to encounter scenarios where only a subset of the variables is observed. For instance, while mRNA concentrations are routinely measured, corresponding protein levels (which are also important for modeling the underlying dynamical system) are often unavailable. Our framework is well suited to such settings because it relies on matching the joint distribution of time and the observed dimensions, rather than requiring all dimensions to be measured. More precisely, since our loss (eq. (7)) is defined over the observed state variables (together with time), the model is trained to match the marginal distribution of the observed variables along with time, without making any additional assumptions or imputations for the missing dimensions. We discuss an experiment in this missing-data scenarios in section 4 (fig. 7).

3.1. Evaluating model fit with an R^2-type metric

In this subsection, we define a new metric that can be used to assess the model fit of our method.

Least squares for distributional data. The loss function in eq. (7) defines a least squares scheme in the RKHS associated with our kernel mean embeddings. In the special case where every model predicts a Dirac delta measure (i.e., a point mass) and we choose the kernel to be linear, this least squares criterion reduces to the standard Euclidean least squares. This correspondence motivates the introduction of an R^2 type metric in our setting to quantify the goodness-of-fit of a fitted model.

The "center" of a collection of distributions. In standard regression, we define the coefficient of determination, R^2, as one minus the ratio of the residual sum of squares (error) to the total sum of squares (total variability). We measure the total variability with respect to the sample mean, which serves as the "center" of the data because it minimizes the sum of squared deviations. When we work with distributions, the *barycenter* of the collection of distributions provides an analogous notion of center. In the context of kernel mean embeddings and MMD, Cohen et al. (2020) showed that the barycenter of a set of distributions corresponds to their mixture. More precisely, let $\{\hat{f}_{t_1}, \ldots, \hat{f}_{t_I}\}$ denote the empirical distributions observed at discrete time points $t_1, \ldots, t_I$. If we weight these distributions by the relative frequency of observations, where the weight at time t_i is given by $w_{t_i} = \left(\frac{N_{t_i}}{\sum_{i=1}^{I} N_{t_i}} \right)^2$, then the barycenter f_{bary} is defined as

$$\begin{aligned} f_{\text{bary}} &= \arg\min_{f} \sum_{i=1}^{I} w_{t_i} \ \mathrm{MMD}^2(f, \hat{f}_{t_i}) \\ &= \frac{1}{\sum_{i=1}^{I} N_{t_i}} \sum_{i=1}^{I} \sum_{n_{t_i}=1}^{N_{t_i}} \delta_{\boldsymbol{Y}_{t_i}^{n_{t_i}}}(\cdot). \end{aligned} \tag{8}$$

In this expression, the second equality shows that the barycenter is exactly the mixture (or the time-average) of the empirical distributions.

The coefficient of determination for distributional data. Following the analogy with classical regression, we can then define an R^2 metric in the RKHS as follows.

Definition 3.1 (RKHS-based R^2 Metric) *Let $\hat{f}_{t_i}$, $f_{\boldsymbol{\theta},t_i}$, f_{bary}, and w_{t_i} defined as above. The goodness-of-fit metric R^2 in the RKHS is then defined by*

$$R^2 = 1 - \frac{\sum_{i=1}^{I} w_{t_i} \ \mathrm{MMD}^2\Big(f_{\boldsymbol{\theta},t_i}, \hat{f}_{t_i} \Big)}{\sum_{i=1}^{I} w_{t_i} \ \mathrm{MMD}^2\Big(f_{bary}, \hat{f}_{t_i} \Big)}. \tag{9}$$

In definition 3.1, the numerator represents the weighted aggregate discrepancy (in terms of MMD squared) between

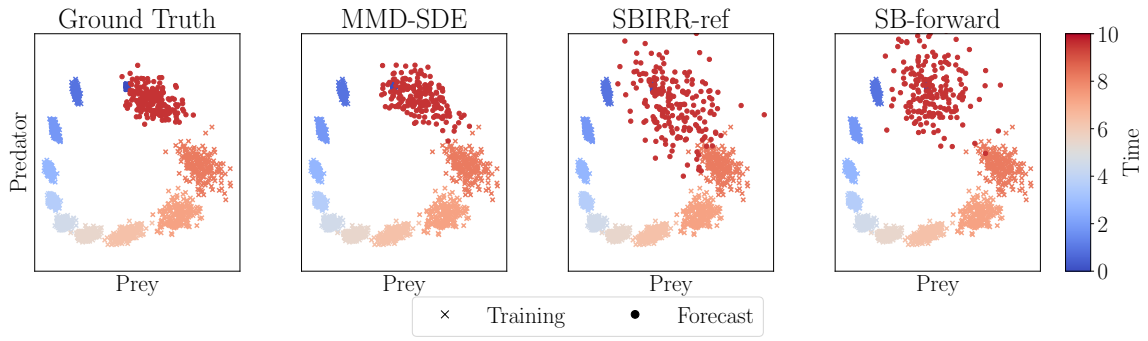

Figure 1. Experimental results for the Lotka-Volterra system. Our method performs better than the two baselines at the forecasting task. Note that the forecast points (red) overlap with the training points at time 0 (blue).

the model-predicted marginal distributions $f_{\boldsymbol{\theta},t_i}$ and the empirical marginals $\hat{f}_{t_i}$ across time. And the denominator quantifies the total variability of the empirical distributions with respect to the barycenter f_{bary}. Thus, the metric R^2 measures the proportion of the total variability that is explained by the model, similar to the traditional R^2 in regression. Besides being seen as classic R^2, we provide an alternative view of this metric in analogy to mutual information in appendix B. Finally, observe that since the MMD^2 is non-negative, the ratio in eq. (9) is bounded below by 0. Consequently, R^2 is upper bounded by 1. When the barycenter is within the model family, R^2 is also lower bound by 0 for a fitted model within the family that minimizes MMD. However, similar to regression models without an intercept, the R^2 value may be less than 0, indicating the model did not provide better fit than the barycenter.

4. Experiments

In this section, we first discuss how we evaluate if a method has been successful in our experiments. We next introduce the two baselines we consider, and then present the two simulated experiments, one for a Lotka-Volterra predator-prey system, and one for a repressilator system.

Metric of success. We consider two tasks in evaluating each method for fitting a model: (1) one-step-ahead forecasting performance and (2) underlying vector field reconstruction. For the forecasting task, since each model starts from the initial time point and propagates particles forward in time, we retain a validation time point to evaluate forecast accuracy. In particular, we assess the forecast accuracy by (i) computing the MMD between observation at the validation time point and forecast obtained by simulating particles forward from the initial time point using the learnt drift and volatility, and (ii) assessing visually how close the forecasted particles match the ground truth. We include results for the forecast task in the main text. For the vector field reconstruction task, instead, when the ground truth drift function is available, we evaluate how well the learned vector field matches the ground truth both visually and by computing the mean squared error (MSE) between the learnt field and the ground truth on a grid spanning the range of the observed data. We present these vector field reconstruction results in appendix C.

Baselines. We compare against two baselines: (1) a Schrödinger bridge with reference fitting (Shen et al., 2025; Zhang, 2024; Guan et al., 2024), that we denote by `SBIRR-ref`, and (2) a multimarginal Schrödinger bridge with shared forward drift across all consecutive pairs of snapshots, denoted by `SB-forward` (Shen et al., 2024). We discuss these baselines in details in appendix C.1.

4.1. Lotka-Volterra system

We simulated data from a two-dimensional Lotka–Volterra predator–prey system, where each coordinate's volatility scales proportionally with its own state variable. Specifically, we set the volatility for the prey population X to be σX, and for the predator population Y to be σY, with σ being the same constant across both dimensions. We consider 10 time points, each with 200 samples. A parametric Lotka-Volterra model was fitted to the data. See appendix C.2 for additional details. We present the forecasting results in fig. 1. Our method demonstrates superior forecasting accuracy both visually and in terms of MMD compared to the baselines. In particular, visually we see how the baselines exhibit higher noise levels in their forecast. We include MMD metrics in table 1. In the same table, we also include MSE results for the vector field reconstruction task, and in fig. 3 we provide a visualization for the learnt vector fields with the various methods. We further discuss results for this experiment in appendix C.2.3.

4.2. Repressilator

The repressilator system consists of a network of three genes that inhibit each other in a cyclic manner: each gene

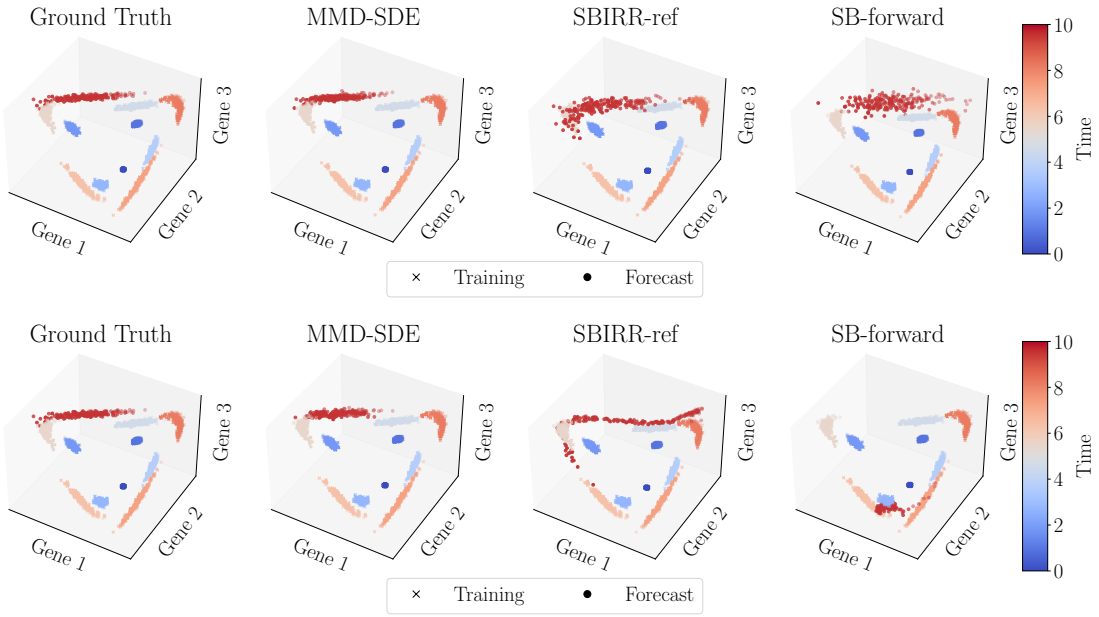

Figure 2. Results for repressilator. Upper: use of parametric repressilator model. Lower: use a semi-parametric model with an MLP-based activation function (see appendix C.3.2 for details). Our method better forecasts the last time point (red) compared to the baselines for both parametric and semiparametric models.

produces a protein that represses the next gene's expression, with the last one repressing the first, forming a feedback loop. The dynamics of the repressilator system can be modeled either by using only mRNA concentration levels (eq. (15)) or with both mRNA and protein levels (eq. (17)). In practice, we only measure mRNA concentrations. In this section, below we discuss an experiment where we generate data from an mRNA-only repressilator system and try to learn the underlying dynamics. In appendix C.4, we also discuss an example where we generate data from an mRNA-protein repressilator system and learn the dynamics having access only to the mRNA concentration levels.

In this experiment, we fit a stochastic (mRNA-only) repressilator system, where — as for the Lotka-Volterra experiment — each coordinate's volatility scales proportionally with its own state variable. We consider 10 time points with 200 samples per time point. We performed two experiments: one using a parametric model from the same family as the data generating process and another using a semiparametric model with a multilayer perceptron modeling production rate of mRNA of each gene as a function of all mRNA levels. See appendix C.3 for more details.

We present the visual results for the forecast task in fig. 2. As in the Lotka-Volterra experiment, our method demonstrates superior forecasting performance for both repressilator experiments. In particular, the one-step-ahead forecast particles with our method have a very similar shape compared to the ground truth validation points. The difference is particularly clear when the using the semiparametric model (bottom row), and is confirmed by the MMD metrics (see table 2 and table 3). We include MSE results, as well as a visualization of the reconstructed vector field, in table 2 and fig. 4 (for the parametric model), and table 3 and fig. 6 (for the semiparametric model). We further discuss these results in appendix C.3.3.

5. Discussion

In this work, we introduced a new method for learning SDEs from population-level snapshot data. Our approach is based on matching the state-space distributions using a least squares scheme in a distributional space, which allows us to infer the underlying dynamics of the system effectively. The proposed method handles challenging aspects such as unknown and non-constant volatility, and it is robust to missing or unobserved dimensions. Overall, our experiments indicate that the proposed framework outperforms existing methods in various scenarios. One of the key remaining challenges is related to the identifiability of the underlying model parameters. In practice, even if one had access to the complete time series of the marginal distributions, uniquely determining the drift and volatility functions of the SDE remains problematic. This issue arises because the evolution of the marginal distributions can be generated by multiple different combinations of drift and volatility. We discuss these identifiability issues in more detail in Appendix D.

Acknowledgments

This work was supported in part by an ONR Early Career Grant and by the NSF TRIPODS program (award DMS-2022448).

References

Tianrong Chen, Guan-Horng Liu, Molei Tao, and Evangelos Theodorou. Deep momentum multi-marginal Schrödinger bridge. *Advances in Neural Information Processing Systems*, 36, 2024.

Samuel Cohen, Michael Arbel, and Marc Peter Deisenroth. Estimating barycenters of measures in high dimensions. *arXiv preprint arXiv:2007.07105*, 2020.

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.

Damien Garreau, Wittawat Jitkrittum, and Motonobu Kanagawa. Large sample analysis of the median heuristic. *arXiv preprint arXiv:1707.07269*, 2017.

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

Vincent Guan, Joseph Janssen, Hossein Rahmani, Andrew Warren, Stephen Zhang, Elina Robeva, and Geoffrey Schiebinger. Identifying drift, diffusion, and causal structure from temporal snapshots. *arXiv preprint arXiv:2410.22729*, 2024.

Alastair Hall. *Generalized method of moments*. Wiley Online Library, 2004.

Takeshi Koshizuka and Issei Sato. Neural Lagrangian Schrödinger bridge: Diffusion modeling for population dynamics. *13th International Conference on Learning Representations*, 2022.

Hugo Lavenant, Stephen Zhang, Young-Heon Kim, and Geoffrey Schiebinger. Towards a mathematical theory of trajectory inference. pp. 1–76, 2021. URL `http://arxiv.org/abs/2102.09204`.

Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, pp. 3870–3882. PMLR, 2020.

Grigorios A Pavliotis. *Stochastic Processes and Applications*. Springer, 2016.

Michele Pavon, Giulio Trigila, and Esteban G Tabak. The data-driven Schrödinger bridge. *Communications on Pure and Applied Mathematics*, 74(7):1545–1573, 2021.

Yunyi Shen, Renato Berlinghieri, and Tamara Broderick. Learning a vector field from snapshots of unidentified particles rather than particle trajectories. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024.

Yunyi Shen, Renato Berlinghieri, and Tamara Broderick. Multi-marginal Schrödinger bridges with iterative reference refinement. *To appear at AISTATS*, 2025.

Francisco Vargas, Pierre Thodoroff, Austen Lamacraft, and Neil Lawrence. Solving Schrödinger bridges via maximum likelihood. *Entropy*, 23(9):1–30, 2021. ISSN 10994300. doi: 10.3390/e23091134.

Benjie Wang, Joel Jennings, and Wenbo Gong. Neural Structure Learning with Stochastic Differential Equations. *arXiv preprint*, (ii):1–29, 2023. URL `http://arxiv.org/abs/2311.03309`.

Gefei Wang, Yuling Jiao, Qian Xu, Yang Wang, and Can Yang. Deep generative learning via Schrödinger bridge. In *International Conference on Machine Learning*, pp. 10794–10804. PMLR, 2021.

Stephen Y Zhang. Joint trajectory and network inference via reference fitting. *arXiv preprint arXiv:2409.06879*, 2024.

Appendix

A. Proofs

Proof. [Proof of proposition 3.1] We start with the definition of the MMD squared between the joint distributions:

$$\begin{aligned}\mathrm{MMD}_K^2(f(\boldsymbol{x},\boldsymbol{t}), g(\boldsymbol{x},\boldsymbol{t})) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim f,(\boldsymbol{x}',\boldsymbol{t}')\sim f}\left[K((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}',\boldsymbol{t}'))\right] \\ - 2\mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim f,(\boldsymbol{x}',\boldsymbol{t}')\sim g}\left[K((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}',\boldsymbol{t}'))\right] \\ + \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim g,(\boldsymbol{x}',\boldsymbol{t}')\sim g}\left[K((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}',\boldsymbol{t}'))\right]\end{aligned} \tag{10}$$

Then we can rewrite the first term in right-hand side as follows:

$$\begin{aligned}\mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim f,(\boldsymbol{x}',\boldsymbol{t}')\sim f}\left[K((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}',\boldsymbol{t}'))\right] &= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim f,\ (\boldsymbol{x}',\boldsymbol{t}')\sim f}[K_x(\boldsymbol{x},\boldsymbol{x}')\,\delta(\boldsymbol{t}-\boldsymbol{t}')] \\ &= \mathbb{E}_{\boldsymbol{t}\sim h(\boldsymbol{t}),\ \boldsymbol{t}'\sim h(\boldsymbol{t}')}\left[\delta(\boldsymbol{t}-\boldsymbol{t}')\,\mathbb{E}_{\substack{\boldsymbol{x}\sim f(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim f(\boldsymbol{x}|\boldsymbol{t}')}}[K_x(\boldsymbol{x},\boldsymbol{x}')]\right] \\ &= \sum_{\boldsymbol{t},\boldsymbol{t}'\in\mathcal{T}}\left[\delta(\boldsymbol{t}-\boldsymbol{t}')\,\mathbb{E}_{\substack{\boldsymbol{x}\sim f(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim f(\boldsymbol{x}|\boldsymbol{t}')}}[K_x(\boldsymbol{x},\boldsymbol{x}')]\right] h(\boldsymbol{t})h(\boldsymbol{t}') \\ &= \sum_{\boldsymbol{t}\in\mathcal{T}}\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'\sim f(\boldsymbol{x}|\boldsymbol{t})}[K_x(\boldsymbol{x},\boldsymbol{x}')]\,h^2(\boldsymbol{t})\end{aligned} \tag{11}$$

where the first equality uses the factorized form of the kernel, the second equality is by the the law of iterated expectation conditioning on the time components.

Similarly the second term:

$$\begin{aligned}-2\mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim f,(\boldsymbol{x}',\boldsymbol{t}')\sim g}\left[K((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}',\boldsymbol{t}'))\right] &= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim f,\ (\boldsymbol{x}',\boldsymbol{t}')\sim g}[K_x(\boldsymbol{x},\boldsymbol{x}')\,\delta(\boldsymbol{t}-\boldsymbol{t}')] \\ &= \mathbb{E}_{\boldsymbol{t}\sim h(\boldsymbol{t}),\ \boldsymbol{t}'\sim h(\boldsymbol{t}')}\left[\delta(\boldsymbol{t}-\boldsymbol{t}')\,(-2\mathbb{E}_{\substack{\boldsymbol{x}\sim f(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t}')}}[K_x(\boldsymbol{x},\boldsymbol{x}')])\right] \\ &= \sum_{\boldsymbol{t},\boldsymbol{t}'\in\mathcal{T}}\left[\delta(\boldsymbol{t}-\boldsymbol{t}')\,(-2\mathbb{E}_{\substack{\boldsymbol{x}\sim f(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t}')}}[K_x(\boldsymbol{x},\boldsymbol{x}')])\right] h(\boldsymbol{t})h(\boldsymbol{t}') \\ &= \sum_{\boldsymbol{t}\in\mathcal{T}}-2\mathbb{E}_{\substack{\boldsymbol{x}\sim f(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t})}}[K_x(\boldsymbol{x},\boldsymbol{x}')]\,h^2(\boldsymbol{t})\end{aligned} \tag{12}$$

The third term

$$\begin{aligned}\mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim g,(\boldsymbol{x}',\boldsymbol{t}')\sim f}\left[K((\boldsymbol{x},\boldsymbol{t}),(\boldsymbol{x}',\boldsymbol{t}'))\right] &= \mathbb{E}_{(\boldsymbol{x},\boldsymbol{t})\sim g,\ (\boldsymbol{x}',\boldsymbol{t}')\sim g}[K_x(\boldsymbol{x},\boldsymbol{x}')\,\delta(\boldsymbol{t}-\boldsymbol{t}')] \\ &= \mathbb{E}_{\boldsymbol{t}\sim h(\boldsymbol{t}),\ \boldsymbol{t}'\sim h(\boldsymbol{t}')}\left[\delta(\boldsymbol{t}-\boldsymbol{t}')\,\mathbb{E}_{\substack{\boldsymbol{x}\sim g(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t}')}}[K_x(\boldsymbol{x},\boldsymbol{x}')]\right] \\ &= \sum_{\boldsymbol{t},\boldsymbol{t}'\in\mathcal{T}}\left[\delta(\boldsymbol{t}-\boldsymbol{t}')\,\mathbb{E}_{\substack{\boldsymbol{x}\sim g(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t}')}}[K_x(\boldsymbol{x},\boldsymbol{x}')]\right] h(\boldsymbol{t})h(\boldsymbol{t}') \\ &= \sum_{\boldsymbol{t}\in\mathcal{T}}\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t})}[K_x(\boldsymbol{x},\boldsymbol{x}')]\,h^2(\boldsymbol{t})\end{aligned} \tag{13}$$

Collect terms, we have

$$\begin{aligned}&\mathrm{MMD}_K^2(f(\boldsymbol{x},\boldsymbol{t}), g(\boldsymbol{x},\boldsymbol{t})) \\ &= \sum_{\boldsymbol{t}\in\mathcal{T}}h^2(\boldsymbol{t})\left[\mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'\sim f(\boldsymbol{x}|\boldsymbol{t})}[K_x(\boldsymbol{x},\boldsymbol{x}')] - 2\mathbb{E}_{\substack{\boldsymbol{x}\sim f(\boldsymbol{x}|\boldsymbol{t})\\ \boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t})}}[K_x(\boldsymbol{x},\boldsymbol{x}')] + \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'\sim g(\boldsymbol{x}|\boldsymbol{t})}[K_x(\boldsymbol{x},\boldsymbol{x}')]\right] \\ &= \sum_{\boldsymbol{t}\in\mathcal{T}}h^2(\boldsymbol{t})\,\mathrm{MMD}^2(f(\cdot\mid\boldsymbol{t}), g(\cdot\mid\boldsymbol{t}))\end{aligned}$$

□

In our application, we used empirical distribution for time.

B. Alternative view of R^2

An alternative way to view this metric is through the lens of comparing joint distributions to the product of their marginals. In information theory, mutual information quantifies the dependence between two random variables by measuring the divergence (typically via the Kullback–Leibler divergence) between the joint distribution and the product of the marginal distributions. Analogously, by reapplying proposition 3.1, we can interpret our R^2 metric as comparing the joint distribution of state and time as predicted by the model with the distribution obtained by taking the product of the marginal (state and time) distributions. In this view, the denominator in eq. (9) (which uses the barycenter) reflects the total variation or "spread" in the observed data. And the numerator captures the remaining error when the model-predicted joint distribution is compared to the empirical joint distribution. Thus, a higher R^2 indicates that the model captures more of the dependence structure between state and time—just as in regression a higher R^2 means the model explains a larger fraction of the variability in the data. This analogy to mutual information provides an intuitive understanding of how our metric not only assesses goodness-of-fit but also the degree to which the model captures the temporal structure of the data.

C. Further experimental details

C.1. Baselines

We compare against two baselines: (1) a Schrödinger bridge with reference fitting (Shen et al., 2025; Zhang, 2024; Guan et al., 2024), that we denote by `SBIRR-ref`, and (2) a multimarginal Schrödinger bridge with shared forward drift across all consecutive pairs of snapshots, denoted by `SB-forward` (Shen et al., 2024). `SBIRR-ref` iteratively imputes unseen trajectories using Schrödinger bridges and fits a model based on the imputed trajectories (the *reference refinement* step). In practice, we use the implementation from Shen et al. (2025) and use their best reference to evaluate vector field prediction and to propagate particles from the initial time step beyond the observed horizon for the forecasting task. Notably, this task differs from the main task in Shen et al. (2025), which focused on interpolation between time snapshots. Their approach employs the Schrödinger bridge solution rather than a fitted reference, making it unsuitable for forecasting. Zhang (2024) and Guan et al. (2024)'s method are based on the same reference fitting idea and have similar algorithms to Shen et al. (2025). However, these methods are focused on linear models[1], whereas Shen et al. (2025) allows for a general model family. For `SB-forward`, the authors — instead of allowing a flexible forward drift between consecutive time snapshots — constrain the forward drift to be drawn from a shared model family across all time steps. We use their best-fitted forward model to predict the vector field and generate forecasts. Both baselines cannot handle unknown or non-constant volatility due to their reliance on a Kullback–Leibler divergence-based formulation. In the experiments that follow, we set the fixed volatility values to 0.1, as done in Vargas et al. (2021); Wang et al. (2021); Shen et al. (2025).

C.2. Lotka-Volterra

C.2.1. Experiment setup

In this experiment, we study the stochastic Lotka-Volterra model, which describes predator-prey interactions over time. The population dynamics are governed by the following system of SDEs:

$$\begin{aligned} dX &= \alpha X - \beta XY + \sigma X dW_x, \\ dY &= \gamma XY - \delta Y + \sigma Y dW_y, \end{aligned} \tag{14}$$

where $[dW_x, dW_y]$ denotes a two-dimensional Brownian motion. We define the true parameter values as $\alpha = 1.0$, $\beta = 0.4$, $\gamma = 0.4$, $\delta = 0.1$, and $\sigma = 0.02$. The initial population sizes are sampled from uniform distributions, with $X_0 \sim U(5, 5.1)$ and $Y_0 \sim U(4, 4.1)$. To simulate the system, we numerically integrate the SDEs over 10 discrete time points, using a step size of 1, with the Euler-Maruyama scheme(implemented via the `torchsde` Python package).

[1]Zhang (2024) also included an L^1 regularization to obtain sparse solutions.

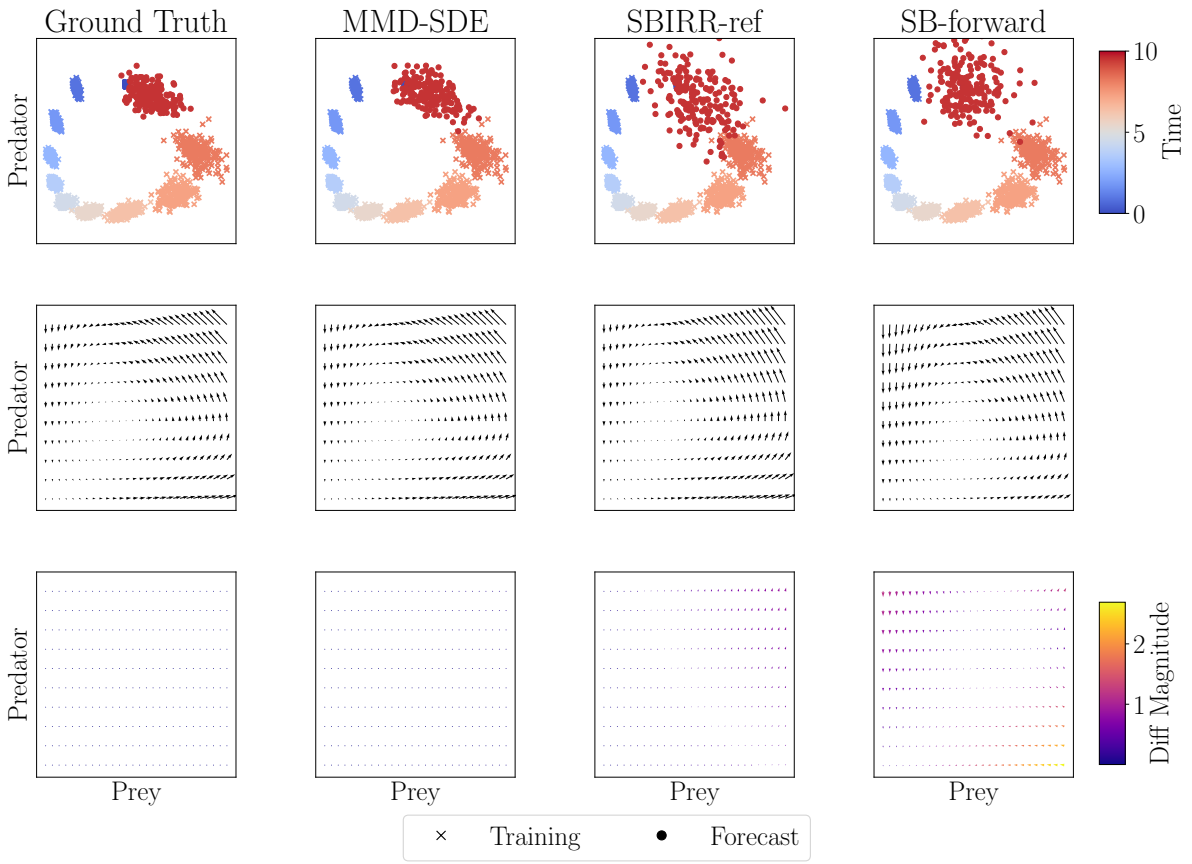

Figure 3. Experimental results for the Lotka-Volterra system. *Top row*: forecast prediction task. A method is successful if the forecast predicted points (in red) match the red points in the ground truth figure. *Middle row:* ground truth vector field (left) and reconstructed vector fields with the three methods. *Bottom row:* Difference between reconstructed vector fields and ground truth. For each point of interest on the grid, we represent the difference between the two vectors with an arrow and color it according to the magnitude of the difference (colorbar to the right).

C.2.2. Model family choice

For this experiment, we have access to the data-generating process, as described in eq. (14). Therefore, we select the model family to be the set of SDEs that satisfy this system of equations, eq. (14). The learning process involves optimizing the parameters using gradient descent, with a learning rate of 0.05 over 300 epochs, by observing R^2 metric dynamics.

C.2.3. Additional Results Discussion

In this section we further discuss results for the Lotka-Volterra experiment. In particular, we analyze the MMD and MSE metrics from table 1, and the visualization of the reconstructed fields from fig. 3.

In the first row of table 1 we show the MMD results for the forecast task. The MMD is computed using a Radial Basis Function (RBF) kernel with length scale 1. In each cell, the first number represent the MMD averaged across 10 different seeds, and the second number (in parenthesis) is the standard deviation over the same 10 seeds. We color in green the cell corresponding to the method with lowest MMD. We also highlight in green any other methods whose one-standard deviation confidence interval overlaps the mean of the best method. From the first row, we can see how our method is (by far) the best method at the forecasting task. In the second row we compare the MSEs for the vector reconstruction task. Also for this task we observe that our method is — from an MMD perspective — much better than the baselines. If we

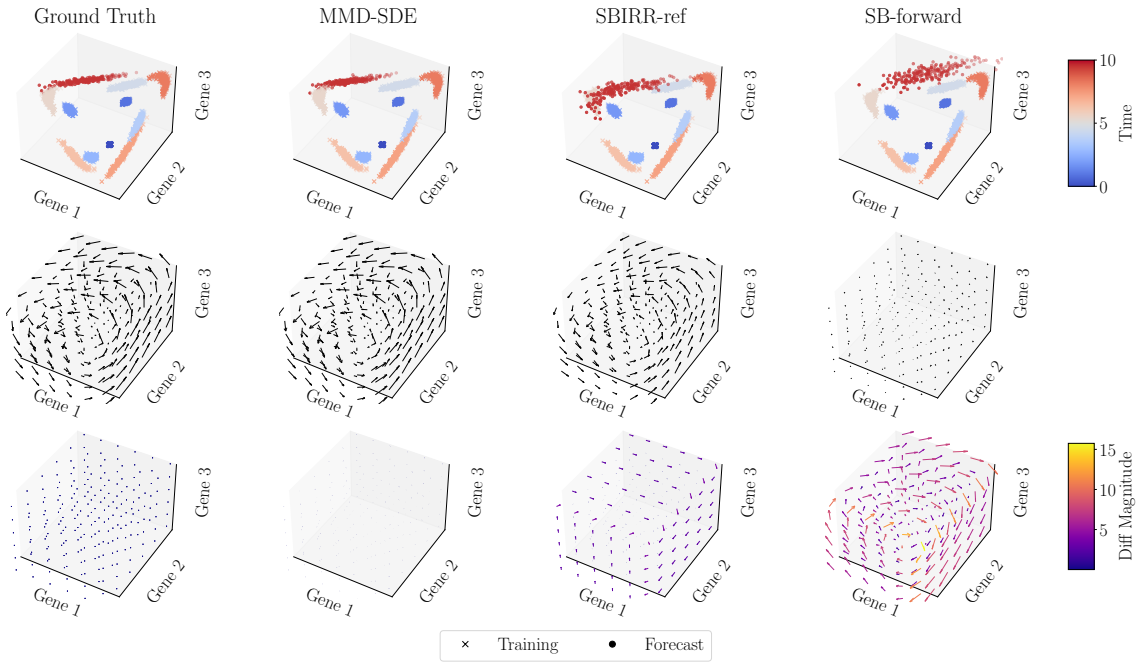

Figure 4. Experimental results for the repressilator system using parametric model as model family.*Top row*: forecast prediction task. A method is successful if the forecast predicted points (in red) match the red points in the ground truth figure. *Middle row:* ground truth vector field (left) and reconstructed vector fields with the three methods. *Bottom row:* Difference between reconstructed vector fields and ground truth. For each point of interest on the grid, we represent the difference between the two vectors with an arrow and color it according to the magnitude of the difference (colorbar to the right).

look at the middle row of fig. 3, we see that from a visual perspective the reconstructed vector fields are very similar to the ground truth for all the three methods. Also from the bottom row of the same figure, we can see that the difference between the reconstructed fields and ground truth for our method and `SBIRR-ref` is very similar, whereas for `SB-forward` it is a bit worse.

	LV		
Metric	`MMD-SDE`	`SBIRR-ref`	`SB-forward`
Forecast	**0.012 (0.0067)**	0.14 (0.023)	0.71 (0.49)
Drift	**0.00071 (0.000027)**	0.079 (0.0080)	0.59 (0.13)

Table 1. Evaluation metric for Lotka-Volterra (mean (sd)). Drift was evaluated using MSE on a grid, while the forecast was evaluated using MMD with RBF kernel and length scale 1.

C.3. Repressilator

C.3.1. Experiment setup

The repressilator is a synthetic genetic circuit designed to function as a biological oscillator, producing sustained periodic fluctuations in the concentrations of its components. It consists of a network of three genes arranged in a cyclic inhibitory loop: each gene encodes a protein that suppresses the expression of the next, with the last gene repressing the first, completing the feedback cycle.

The system's dynamics can be described by the following stochastic differential equations (SDEs):

$$\begin{aligned} dX_1 &= \frac{\beta}{1+(X_3/k)^n} - \gamma X_1 + \sigma X_1 dW_1, \\ dX_2 &= \frac{\beta}{1+(X_1/k)^n} - \gamma X_2 + \sigma X_2 dW_2, \\ dX_3 &= \frac{\beta}{1+(X_2/k)^n} - \gamma X_3 + \sigma X_3 dW_3, \end{aligned} \tag{15}$$

where $[dW_1, dW_2, dW_3]$ represents a three-dimensional Brownian motion. The inhibitory structure of the system is evident from the drift terms, which describe how each gene's expression is repressed by another in the cycle. For our simulations, we set the parameters to $\beta = 10$, $n = 3$, $k = 1$, $\gamma = 1$, and $\sigma = 0.02$. The initial conditions are sampled from uniform distributions: $X_1, X_2 \sim U(1, 1.1)$ and $X_3 \sim U(2, 2.1)$. We simulate the SDEs over 10 discrete time points, with 200 samples collected at each step.

C.3.2. Model family choice

Parametric model. For this experiment, we have access to the data-generating process, as described in eq. (15). Therefore, we pick as a model family the set of SDEs that satisfy this system of equations, eq. (15). The learning process involves optimizing the parameters using gradient descent, with a learning rate of 0.05 over 500 epochs, as determined by R^2 metric dynamics.

Semiparametric model. We assume we know the functional form up until that

$$d\boldsymbol{X}_t = \boldsymbol{M} f_{\boldsymbol{\theta}}(\boldsymbol{X}_t) - \boldsymbol{L}\boldsymbol{X}_t + \boldsymbol{G}\,\mathrm{diag}(\boldsymbol{X}_t) d\boldsymbol{W}_t \tag{16}$$

where $\boldsymbol{M}$ is a diagonal matrix of (positive) maximum production rate, $\boldsymbol{L}$ is a diagonal matrix of (positive) degradation rate, $\boldsymbol{G}$ is a diagonal matrix of (positive) volatility all unknown (parameterized by their logarithm). We parameterize the so-called activation function $f_{\boldsymbol{\theta}} : \mathbb{R}^3_+ \to [0,1]^3$, encoding regulation among genes, using an MLP with three hidden layers of [32, 64, 32] hidden neurons each, and one final sigmoid function layer.

C.3.3. Further experimental results

Parametric model. In this section we further discuss results for the repressilator experiment with parametric model family. In particular, we analyze the MMD and MSE metrics from table 2, and the visualization of the reconstructed fields from fig. 4.

	Repressilator (parametric)		
Metric	`MMD-SDE`	`SBIRR-ref`	`SB-forward`
Forecast	**0.016 (0.016)**	0.47 (0.34)	0.42 (0.16)
Drift	**0.027 (0.063)**	1.71 (0.20)	12.9 (0.21)

Table 2. Evaluation metric for repressilator when using the parametric model (mean(sd)). Drift was evaluated using MSE on a grid, while the forecast was evaluated using MMD with RBF kernel and length scale 1.

In the first row of table 2, we see that for the forecasting task, our method achieves a much lower MMD compared to the two baselines. This quantitatively supports the visual intuition from fig. 2 in the main text, where our approach more accurately captures the underlying distribution of the data. In the second row, we observe that the MSE for the vector field reconstruction task is significantly lower for our method, indicating superior performance in recovering the true dynamics. This is further corroborated by the visualizations in fig. 4: in the middle row, our reconstructed vector field closely resembles the ground truth, whereas `SBIRR-ref` exhibits small but notable deviations, and `SB-forward` fails both in magnitude and direction. The bottom row further reinforces this conclusion, showing that the magnitude of the differences between the reconstructed and true vector fields is substantially larger for the two baselines compared to our method (for which is very close to 0 everywhere on the grid).

Semiparametric model. In this section we further discuss results for the repressilator experiment with the semiparametric model family. In particular, we analyze the MMD and MSE metrics from table 3, and the visualization of the reconstructed fields from fig. 6.

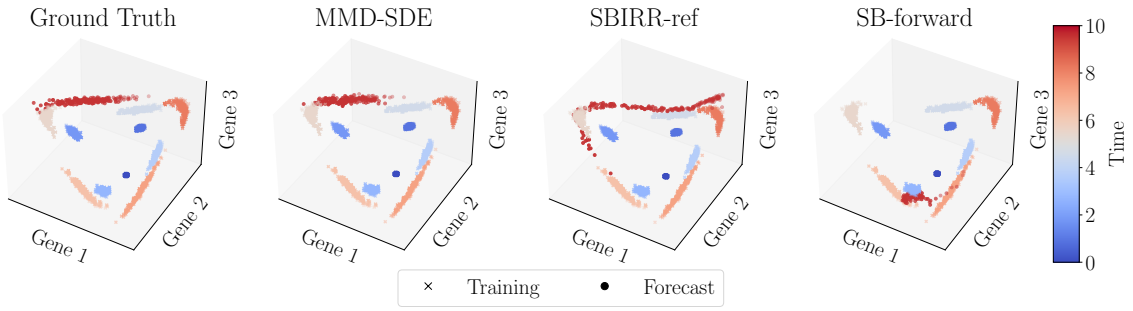

Figure 5. Results for repressilator with semiparametric model. Our method better forecasts the last time point (red) compared to the baselines.

In the first row of table 3, we see that also for this forecasting task, our method achieves a substantially lower MMD compared to the two baselines. This aligns with the visual evidence from fig. 2 in the main text (bottom row), where our method's predicted points (in red) more closely match the ground truth. In the second row of table 3, we see that for this model choice our method and `SBIRR-ref` achieve similar results, whereas `SB-forward` exhibits much higher MSE. Figure 6 confirms this intuition: our reconstructed vector field and the one for `SBIRR-ref` are quite similar and not too different from the ground truth, whereas `SB-forward` performs particularly poorly, failing to recover both the direction and magnitude of the vector field.

	Repressilator (semiparametric)		
Metric	`MMD-SDE`	`SBIRR-ref`	`SB-forward`
Forecast	**0.077 (0.031)**	0.29 (0.11)	1.15 (0.33)
Drift	**6.25 (0.37)**	**7.85 (1.85)**	12.00 (0.74)

Table 3. Evaluation metric for Repressilator using MLP activation function (mean(sd)). Drift was evaluated using MSE on a grid, while forecast was evaluated using MMD with RBF kernel and length scale 1.

C.4. Repressilator with missing protein

In this experiment, we focus on the situation where we generate data following the more complete biochemical model including both mRNA and protein eq. (17) and then fit this parametric (mRNA-protein) repressilator model (appendix C.4.2) using only mRNA concentration data. The experimental settings are the same as for the mRNA-only experiment in the main text. We show the results for our method on the forecasting task in fig. 7. Our method accurately forecasts the concentrations even when protein concentration is not observed. For the two baselines, since they cannot account for missing data, we fit a model that takes into account only the mRNA levels. In fig. 7 we can see that the forecasts for the baselines are much worse than for our method. The key difference here is that in our method we do not observe protein levels but the method is aware that protein levels are also driving the underlying dynamics. Whereas for the baselines there is no way of encoding this information in the methods without observing the protein levels. The visual difference is also confirmed by the metric results in table 4.

C.4.1. Experiment setup

In appendix C.3 we introduced the repressilator system as a system of SDEs governing changes in mRNA concentration. A more complete model for this system takes also into account protein levels. Indeed, each gene produces a protein that represses the next gene's expression, with the last one repressing the first. So proteins play a big role in the repressilator feedback loop. And this is why scientists often consider a more complex version of this system, that evolves according to

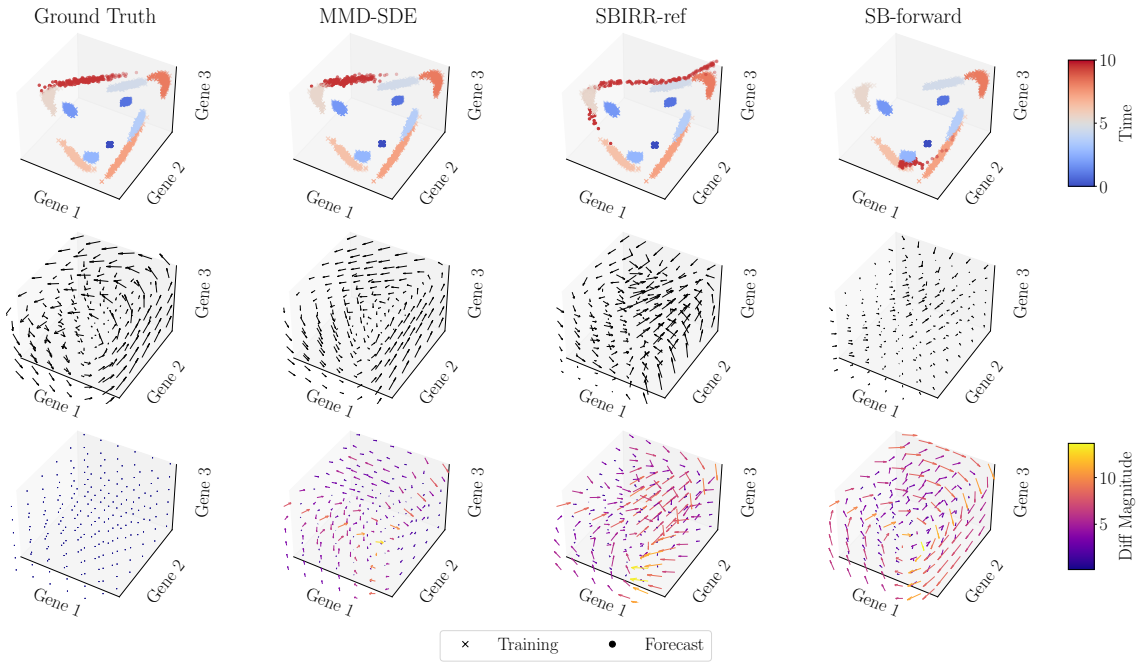

Figure 6. Experimental results for the repressilator system using semiparametric model as model family. *Top row*: forecast prediction task. A method is successful if the forecast predicted points (in red) match the red points in the ground truth figure. *Middle row:* ground truth vector field (left) and reconstructed vector fields with the three methods. *Bottom row:* Difference between reconstructed vector fields and ground truth. For each point of interest on the grid, we represent the difference between the two vectors with an arrow and color it according to the magnitude of the difference (colorbar to the right).

the following SDEs:

$$
\begin{aligned}
dX_1 &= \alpha + \frac{\beta}{1+(Y_3/k)^n} - \gamma X_1 + \sigma X_1 dW_1 \\
dX_2 &= \alpha + \frac{\beta}{1+(Y_1/k)^n} - \gamma X_2 + \sigma X_2 dW_2 \\
dX_3 &= \alpha + \frac{\beta}{1+(Y_2/k)^n} - \gamma X_3 + \sigma X_2 dW_3 \\
dY_1 &= \beta_p X_1 - \gamma_p Y_1 + \sigma Y_1 dW_4 \\
dY_2 &= \beta_p X_2 - \gamma_p Y_2 + \sigma Y_2 dW_5 \\
dY_3 &= \beta_p X_3 - \gamma_p Y_3 + \sigma Y_3 dW_6
\end{aligned}
\tag{17}
$$

where $[dW_1, dW_2, dW_3, dW_4, dW_5, dW_6]$ is a 6D Brownian motion. X_1, X_2, X_3 represents the mRNA levels while Y_1, Y_2, Y_3 are the corresponding proteins. As explained above, the actual system regulation is now mediated by proteins rather than mRNA themselves.

To obtain data, we fix the following parameters: $\alpha = 10^{-5}, \beta = 10, n = 3, k = 1, \gamma = 1, \beta_p = 1, \gamma_p = 1, \sigma = 0.02$. We start the dynamics with initial distribution $X_1, X_2 \sim U(1, 1.1)$ and $X_3 \sim U(2, 2.1)$, while $Y_i \sim U(0, 0.1)$. We simulate the SDEs for 10 instants of time. At each time step, we take 200 samples and only took X_i as observations.

C.4.2. Model family choice

Our method. For this experiment, we have access to the data-generating process, as described in eq. (17). Therefore, we select our model family to be the set of SDEs that satisfy this system of equations, eq. (17). The learning process involves

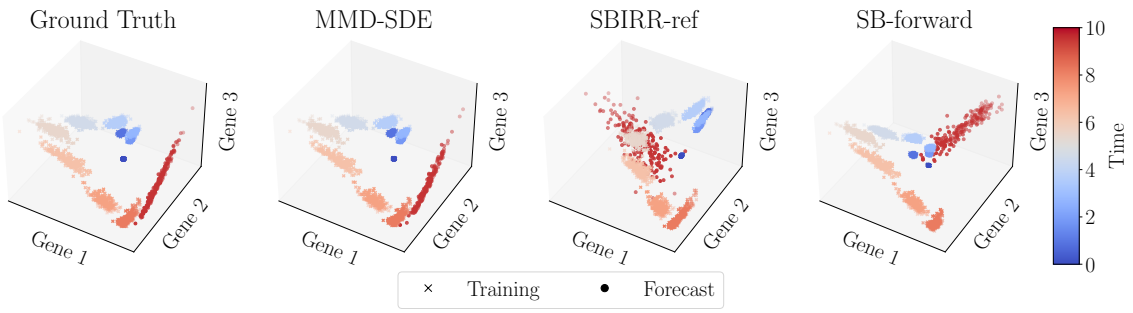

Figure 7. Forecasts for the repressilator experiment when protein concentration was not observed.

optimizing the parameters using gradient descent, with a learning rate of 0.05 over 500 epochs, as determined by R^2 metric dynamics.

Baselines. Since the two baseline methods that we consider cannot handle missing data we cannot use them to fit eq. (17). Instead, we fit a simpler mRNA-only model as described in eq. (15).

	Repressilator (with missing protein)		
Metric	`MMD-SDE`	`SBIRR-ref`	`SB-forward`
Forecast	**0.51 (0.17)**	1.26 (0.06)	1.22 (0.09)

Table 4. Evaluation metric for Repressilator forecasting with missing protein observations. Forecast was evaluated using MMD with RBF kernel and length scale 1.

D. Identifiability Analysis

In this appendix, we provide further details on the identifiability problem from the the main text discussion.

Why drift and volatility are not identified in general. Even with complete access to the marginal distributions π_t over time, the pair $(\boldsymbol{b}_0, \boldsymbol{g}_0)$ is not uniquely determined by the Fokker–Planck equation

$$\frac{\partial \pi_t}{\partial t} = \nabla \cdot \left[-\boldsymbol{b}_0 \, \pi_t + \frac{1}{2} \, \boldsymbol{g}_0 \, \boldsymbol{g}_0^\top \nabla \pi_t \right]$$

For example, suppose $(\boldsymbol{b}_0, \boldsymbol{g}_0)$ satisfies the equation for a given π_t. Then, for any vector field $\boldsymbol{h}$ that satisfies the continuity condition $\nabla \cdot (\boldsymbol{h} \, \pi_t) = 0$, the modified drift $\boldsymbol{b}_0' = \boldsymbol{b}_0 + \boldsymbol{h}$ with the same volatility $\boldsymbol{g}_0$ also satisfies the Fokker–Planck equation. This observation indicates that an infinite family of drift functions can generate the same evolution of the marginal distribution if no further constraints are imposed. Furthermore, let $\boldsymbol{A}$ be any orthogonal matrix (i.e., $\boldsymbol{A}\boldsymbol{A}^\top = \boldsymbol{I}$). Then, the pair $(\boldsymbol{b}_0, \boldsymbol{g}_0 \, \boldsymbol{A})$ also satisfies the Fokker–Planck equation. These examples illustrate the inherent non-uniqueness (or non-identifiability) of the drift and volatility functions based solely on the evolution of the marginal distributions.

In practice, to achieve identifiability, one must restrict the candidate function classes for $\boldsymbol{b}_0$ and $\boldsymbol{g}_0$. For instance, assuming that $\boldsymbol{b}_0$ is a gradient field (i.e., $\boldsymbol{b}_0 = \nabla \Phi$ for some potential Φ and that $\boldsymbol{g}_0$ is constant is known to yield identifiability under suitable conditions (Lavenant et al., 2021; Guan et al., 2024). A complete characterization of identifiability in more general settings is beyond the scope of this work and constitutes an important direction for future research.