entropy

MDPI

Article

A Score-Based Approach for Training Schrödinger Bridges for Data Modelling

Ludwig Winkler [1,*], Cesar Ojeda [2,3] and Manfred Opper [2,3,4]

[1] Machine Learning Group, Technische Universität Berlin, 10623 Berlin, Germany
[2] Artificial Intelligence Group, Technische Universität Berlin, 10623 Berlin, Germany
[3] Institute of Mathematics, University of Potsdam, 14469 Potsdam, Germany
[4] Centre for Systems Modelling and Quantitative Biomedicine, University of Birmingham, Birmingham B15 2TT, UK
* Correspondence: winkler@tu-berlin.de

Abstract: A Schrödinger bridge is a stochastic process connecting two given probability distributions over time. It has been recently applied as an approach for generative data modelling. The computational training of such bridges requires the repeated estimation of the drift function for a time-reversed stochastic process using samples generated by the corresponding forward process. We introduce a modified score- function-based method for computing such reverse drifts, which can be efficiently implemented by a feed-forward neural network. We applied our approach to artificial datasets with increasing complexity. Finally, we evaluated its performance on genetic data, where Schrödinger bridges can be used to model the time evolution of single-cell RNA measurements.

Keywords: Schrödinger bridge problem; score estimation; reverse-time stochastic processes

Citation: Winkler, L.; Ojeda, C.; Opper, M. A Score-Based Approach for Training Schrödinger Bridges for Data Modelling. *Entropy* **2023**, *25*, 316. https://doi.org/10.3390/e25020316

Academic Editors: Andrea Prati, Luis Javier García Villalba and Vincent A. Cicirello

Received: 21 December 2022
Revised: 30 January 2023
Accepted: 3 February 2023
Published: 8 February 2023

1. Introduction

Recently, there has been an increasing interest in the application of continuous-time stochastic processes as generative data models, e.g., the so-called *diffusion models* have recently achieved state-of-the-art performance in generating data with unmatched fidelity and granularity [1–4]. This approach to generative modelling proceeds by incrementally adding noise to the original data until they are indistinguishable from samples drawn from a prior distribution, which one selects such that it is easy to sample from, e.g., the Gaussian distribution. One then is required to learn the denoising procedure, whereby one is able to recover from the prior the original data distributions. The authors in [1] showed that one can model the noising procedure as a stochastic process in terms of a stochastic differential equation (SDE), whose drift function is defined such that the desired prior becomes the marginal distribution of the process at the end time. If one defines the noise-injecting process as the forward process, an equivalent SDE, which runs backward in time from the prior to the data, can be used as the denoiser to generate data samples. This approach can be made computationally tractable because the drift of the backward process at each time is given explicitly in terms of the score function, which equals the gradient of the logarithm of the marginal density of the forward process. Hence, if in the forward process, one uses simple Gaussian transition probabilities (related to Ornstein–Uhlenbeck processes), the required scores can be estimated from exact samples and be represented by neural networks as function approximators.

A more general version of generative models is based on the so-called Schrödinger bridges. Here, one tries to construct a stochastic process that has *two* given distributions as marginals at the initial and end times. In addition, the process has to stay close (in a probabilistic sense) to a reference process. While this scenario can obviously be applied to a Gaussian prior and a data distribution, it allows for more general applications, where, e.g., one tries to interpolate between two different arbitrary data distributions.

Schrödinger bridges were introduced by E. Schrödinger in 1931 [5,6] as the most-likely temporal evolution of the probability density for diffusing particles between given initial and end distributions. Recent reviews of the theoretical foundations, formulations as a stochastic control problem, and relations to optimal transport theory can be found, e.g., in [7,8]. Connections to particle-based filtering and smoothing problems occurring in the field of data assimilation were reviewed in [9].

In contrast to the previously mentioned diffusion models, the solution of Schrödinger bridge problems is computationally more demanding. Feasible methods are usually based on the so-called Iterative Proportional Fitting (IPF) algorithm [10] (also known as the Sinkhorn algorithm), which can be understood from the formulation of the bridges as entropically regularised optimal transport problems.

IPF solves the bridge problem by creating a convergent sequence of simpler forward and backward processes, known as half-bridges. For those sub-problems, only *one* of the two distributions at the boundaries of the time interval is kept fixed (alternating between initial and end-points). Recent algorithms differ in the way these half-bridge problems are solved. The authors of [11] presented a method that is based on sequential Monte Carlo techniques for efficiently sampling processes in both directions. References [12,13] are, to our knowledge, the first papers to discuss Schrödinger bridges as generative data models from a machine learning perspective. The construction of the half-bridges of the IPF algorithm is formulated in terms of an estimation for the drift functions of the corresponding SDE. A drift functions is learned from samples created by the half-bridge of the previous iteration using either a Gaussian process regression approach or by training a deep neural network.

The estimation of the required (backward) drift functions from samples of the forward processes (and vice versa) is less simple compared to the case of diffusion models. The exact representation of reverse drifts in terms of score functions could in principle be applied to the Schrödinger problem by a sample-based estimation of score functions using the so-called *score matching* method [14]. However, this approach was deemed as computationally too demanding [12] because it would represent the drift at a given iteration as a sum of scores obtained in the previous steps. Using neural networks as function approximators for score functions would require the storage of an increasing number of neural networks, two per iteration. References [12,13], in contrast, used methodologies that explicitly rely on the *Euler–Maruyama* (EM) discretisation of the SDE. Reference [12] derived an approximation for score functions that allows for an estimation of the reverse drift as a regression problem involving the states at neighbouring discrete times. In a similar way, the approach [13] uses likelihood approximation based on EM for drift estimation. In principle, both estimators depend on the temporal discretisation, which adds another approximation to the solution of the Schrödinger bridge problem.

In this paper, we developed a novel score-based approach to solving the half-bridge problems for Schrödinger bridges within the IPF algorithm. It relies on the exact representation of the drift for the backward process in terms of forward drift and a score function. We developed a variational formulation, which generalises the original score matching approach and which does not rely on the EM discretisation. It is based on a cost functional with a minimiser that agrees with the reverse drift. A sample-based empirical approximation of the cost functional can be minimised using a neural-network-based representation of the drift function. This requires the storage of only a single neural network during the algorithm. We evaluated the quality of our method on two synthetic datasets, as well as on a single-cell RNA sequencing benchmark problem and, for the latter case, showing an improvement upon previous methods by a significant margin.

2. Materials and Methods

In the following subsections, we review the definition of the Schrödinger bridge problem. We discuss how it can be solved by the *Iterative Proportional Fitting* (IPF) method, which relies on estimating drift functions for time-reversed stochastic processes. We

show how this estimation can be performed using a modified score matching estimator. Finally, we discuss the practical implementation of the estimator using a feed-forward neural network.

2.1. Stochastic Differential Equations

We considered the dynamics of a D-dimensional state variable $\{X_t : 0 \leq t \leq 1\}$ in continuous time t, which is defined by a stochastic differential equation (SDE) [15] of the form:

$$dX_t = \overrightarrow{\mu}(X_t, t)dt + \sigma dW_t \tag{1}$$

Here, dX_t is the change of X_t during an infinitesimal time interval dt. $\overrightarrow{\mu}(X_t, t)$ is the drift function, i.e., the deterministic part of the driving force. We introduce the harpoons to indicate the direction of integration, as we will later on introduce reverse-time stochastic processes. σdW_t denotes the diffusion part, which describes a stochastic, white noise force term defined by the infinitesimal change of a Wiener process W_t, where the diffusion strength is set by the constant scalar σ. The formal definition of the drift is given by the (conditional) expected infinitesimal change of X_t by

$$\overrightarrow{\mu}(x, t) = \lim_{h \to 0} \frac{1}{h} \mathbb{E}[X_{t+h} - X_t | X_t = x] \tag{2}$$

2.2. Schrödinger Bridge Problem

We here give a short, informal introduction to the Schrödinger bridge problem. A more detailed and rigorous discussion can, e.g., be found in [8]. The Schrödinger bridge problem consists of constructing an SDE (with fixed given diffusion σ) such that the probability densities of the corresponding state variables $X_{t=0}$ and $X_{t=1}$ at initial and final times (which, for simplicity, we take to be $t = 0$ and $t = 1$) coincide with given densities $\pi_0(x)$ and $\pi_1(x)$. In order to make the problem unique, one imposes the additional constraint that the probability measure $\mathbb{P}$ over the corresponding paths of the stochastic process should be close to a given reference measure $\mathbb{Q}_0$. The latter is itself defined by a drift function $\overrightarrow{\mu}^{\mathbb{Q}_0}(X_t, t)$ and (for simplicity) a given *initial* density $\pi_0(x)$.

If we define $\mathbb{D}(\pi_0, \pi_1)$ to be the set of probability measures over paths $\{X_t : 0 \leq t \leq 1\}$ with fixed marginal densities π_0, π_1, the measure $\mathbb{P}^*$ over paths of the SDE that solves the Schrödinger bridge is defined by the solution of the minimisation problem:

$$\mathbb{P}^* = \arg \inf_{\mathbb{P} \in \mathbb{D}(\pi_0, \pi_1)} \mathbb{KL}[\, \mathbb{P} \,||\, \mathbb{Q}_0 \,]. \tag{3}$$

The explicit expression of the KL-divergence between two different path measures $\mathbb{P}$ and $\mathbb{Q}$ (with the same σ) induced by two SDEs with drift functions $\overrightarrow{\mu}^P(x, t)$ and $\overrightarrow{\mu}^Q(x, t)$ and *initial* densities $\pi_0^P(x)$ and $\pi_0^Q(x)$ (for $X_{t=0}$) is given by

$$\mathbb{KL}[\, \mathbb{P} \,||\, \mathbb{Q} \,] = \mathbb{KL}\Big[\, \pi_0^P \,\Big|\Big|\, \pi_0^Q \,\Big] + \frac{1}{2\sigma^2} \int_0^1 \mathbb{E}_{\mathbb{P}}\Big[\Big(\overrightarrow{\mu}^P(X_t, t) - \overrightarrow{\mu}^Q(X_t, t) \Big)^2 \Big] dt \tag{4}$$

where

$$\mathbb{KL}\Big[\, \pi_0^P \,\Big|\Big|\, \pi_0^Q \,\Big] = \int \pi_0^P(x) \ln\left(\frac{\pi_0^P(x)}{\pi_0^Q(x)} \right) dx \tag{5}$$

denotes the usual KL-divergence between probability densities in $\mathbb{R}^D$. Hence, the Schrödinger bridge problem can be understood as a problem of optimal stochastic control, where one has to find a drift function $\overrightarrow{\mu}^{P^*}(x, t)$ as a state and time-dependent control variable that steers the stochastic dynamical system (1) in such a way that the marginal density of the state variable evolving from a given initial density reaches a predefined end

density. In addition, control variables are quadratically penalised by the KL-divergence (4) to stay close on average to the drift of the reference system $\overrightarrow{\mu}^{\mathbb{Q}}(x,t)$.

2.3. Iterative Proportional Fitting in Schrödinger Bridges

A popular methodology to solve the Schrödinger bridge problem is via Iterative Proportional Fitting (IPF) [16,17], which solves the problem iteratively, where in each iteration step i, two so-called *half-bridge* problems have to be solved. These half-bridges are defined by the recurrent optimisation problems:

$$\mathbb{P}_i^* = \arg \inf_{\mathbb{P} \in \mathbb{D}(\cdot, \pi_1)} \mathbb{KL}[\, \mathbb{P} \,||\, \mathbb{Q}_{i-1}^* \,] \tag{6}$$

$$\mathbb{Q}_i^* = \arg \inf_{\mathbb{Q} \in \mathbb{D}(\pi_0, \cdot)} \mathbb{KL}[\, \mathbb{Q} \,||\, \mathbb{P}_i^* \,] \tag{7}$$

for $i = 1, 2, \ldots$ with an initial measure defined by the reference process, i.e., $\mathbb{Q}_0^* = \mathbb{Q}_0$ for $i = 1$. In the first half-bridge, one minimises the KL-divergence with only the end condition π_1 fixed, whereas for the second half-bridge, only the initial condition π_0 is fixed. As $i \to \infty$, the sequences $\mathbb{P}_i^*$ and $\mathbb{Q}_i^*$ converge to the solution of the Schrödinger bridge problem. For a proof, see [18]. In Figure 1, we provide a visual intuition of IPF applied to the Schrödinger bridge problem.

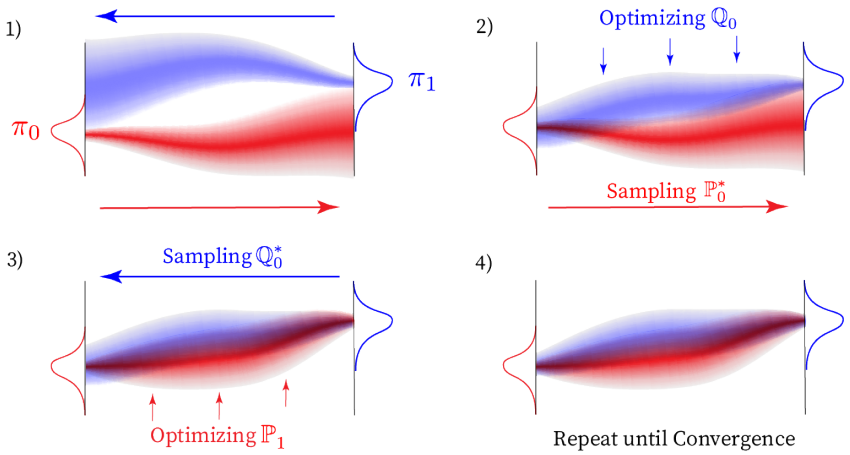

Figure 1. Visualization of the convergence of a one dimensional Schrödinger bridge problem via an Iterative Proportional Fitting style optimization. Subplot 1) shows the initial forward $\mathbb{P}_0^*$ and backward $\mathbb{Q}_0^*$ in red and blue. In the first half-bridge in subplot 2), $\mathbb{P}_0^*$ is held fixed and $\mathbb{Q}_0^*$ is obtained by optimizing equation (7). Consequently, corresponding to equation (6), $\mathbb{P}_1$ is fitted on a constant $\mathbb{Q}_0^*$ in subplot 3). This procedure is repeated until both $\mathbb{Q}_i^*$ and $\mathbb{P}_i^*$ converge according to some predetermined criterion as indicated by subplot 4).

To solve a half-bridge problem, one can use the fact that a given SDE with drift function $\overrightarrow{\mu}(x,t)$ can also be solved backwards in time, where the resulting backward process is also represented by an SDE. We define the reversed time as $\tau \doteq 1 - t$ and the backward SDE as

$$dZ_\tau = \overleftarrow{\mu}(Z_\tau, \tau)d\tau + \sigma dW_\tau \tag{8}$$

with a *backward drift* function that is given by the conditional expectation:

$$\overleftarrow{\mu}(z, 1-t) = \lim_{h \to 0} \frac{1}{h} \mathbb{E}[X_{t-h} - X_t | X_t = z] \tag{9}$$

in terms of the forward process X_t. It can be shown that the statistics of the ensemble of paths $\{Z_{1-t} : 0 \leq t \leq 1\}$ coincides with that of the forward process $\{X_t : 0 \leq t \leq 1\}$ when the initialisation $Z_{\tau=0}$ is drawn at random from the density state variable $X_{t=1}$. The

KL-divergence between path measures can also be expressed in terms of the backward processes and drifts as

$$\mathbb{KL}[\,\mathbb{P}\,||\,\mathbb{Q}\,] = \mathbb{KL}\Big[\,\pi_1^P\,\Big|\Big|\,\pi_1^Q\,\Big] + \frac{1}{2\sigma^2}\int_0^1 \mathbb{E}_{\mathbb{P}}\Big[\Big(\overleftarrow{\mu}^P(Z_\tau,\tau) - \overleftarrow{\mu}^Q(Z_\tau,\tau)\Big)^2\Big]d\tau \tag{10}$$

Equations (4) and (10) show that, for given initial or final densities, respectively, the KL-divergences are minimised by matching the drift functions of the processes (the KL-divergences between initial/end marginal densities equal zero). Hence, if we assume that the mapping:

$$\overrightarrow{\mu}(\cdot,t) \leftrightarrow \overleftarrow{\mu}(\cdot,\tau) \tag{11}$$

is known explicitly, the solution of the half-bridges becomes simple. The minimiser $\mathbb{P}_i^*$ of the KL-divergence in Equation (6) corresponds to an SDE that has the backward drift corresponding to the forward SDE given by the process $\mathbb{Q}_{i-1}^*$, but is started with the density $Z_{\tau=0} \sim \pi_1$ in backward time. The same construction holds for $\mathbb{Q}_i^*$ in Equation (7). This is given by a new SDE with a forward drift, which corresponds to the backward drift of $\mathbb{P}_i^*$ and is started from $\pi_0(x)$ in forward time. Hence, the IPF algorithm reduces the Schrödinger bridge problem to the computation of backward and forward drift functions from the corresponding forward and backward processes.

The *explicit relation* between forward and backward drifts was published first [19] and discussed in [20,21] and is given by

$$\overleftarrow{\mu}(x,\tau) = -\overrightarrow{\mu}(x,1-\tau) + \sigma^2\nabla_x \ln \overrightarrow{p}_{1-\tau}(x) \tag{12}$$

$$\overrightarrow{\mu}(x,t) = -\overleftarrow{\mu}(x,1-t) + \sigma^2\nabla_x \ln \overleftarrow{p}_{1-t}(x). \tag{13}$$

Keeping in mind the relationship between the forward time index t and reverse time index $\tau = 1-t$, $\overrightarrow{p}_{1-\tau}(x)$ is the marginal density of the state variable X_t. Likewise, the density $\overleftarrow{p}_{1-t}(x)$ corresponds to the marginal density of the backward state variable Z_τ evaluated for x. The superimposed harpoons indicate the flow of time with $\overrightarrow{\mu}$ being the drift of the forward process and $\overleftarrow{\mu}$ being the corresponding drift of the backward process. For the interested reader, we provide a derivation of the reverse-time drift resulting in the relationship above in Appendix A.

Figure 2 exemplifies visually how the reverse drift can be obtained from the respective forward drift and the score of the probability distribution over paths induced by the forward SDE. It is only possible in rare cases to compute this density analytically by solving the Fokker–Planck equation. Luckily, there is a numerical method *score matching* [14] that allows for a direct estimation of the gradient of log-densities in (12) from an ensemble of simulated data. This technique is well established in the field of machine learning.

This approach has been previously suggested in the literature, but deemed to be impractical [12] for a solution of the Schrödinger problem. The direct implementation of score matching to (13) in the IPF iterations would create considerable algorithmic problems. Every full iteration of the IPF algorithm would add a score term to the previous reverse drift, which in later iterations would itself be a sum of previous drifts and the score over the previous probability of paths that scales with each IPF iteration i.

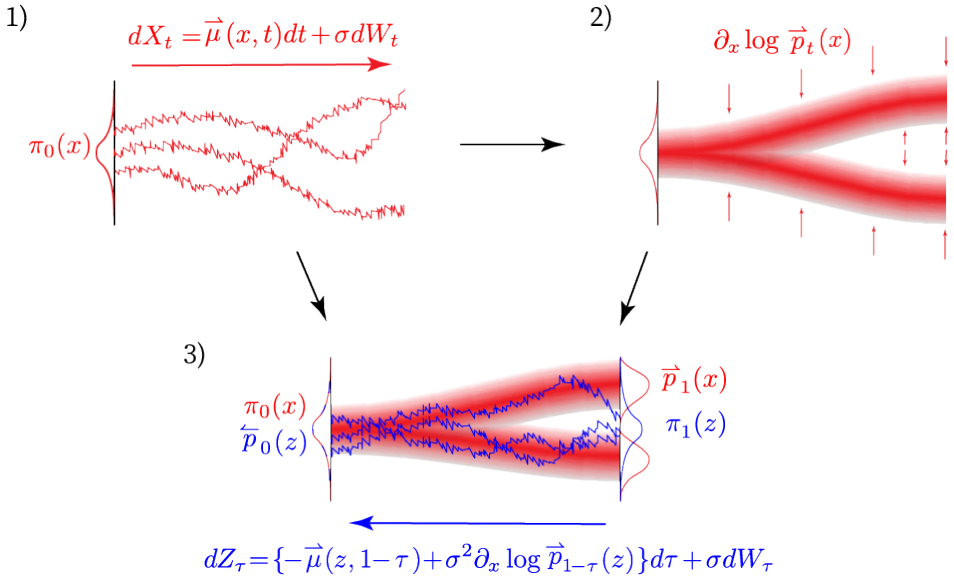

Figure 2. Visualization of the construction of the reverse process. The subplot 1) exemplifies three trajectories generated by solving the forward stochastic differential equation dx_t. The path measure $\mathbb{P}$ induces a probability distribution $\overrightarrow{p}_t(x)$ from which the score is estimated in subplot 2). Finally subplot 3) shows three possible trajectories of the reverse process starting from π_1 and finishing in π_0. A simpler visualization of the score can be found in subplot A), which serves as a figurative illustration of the behaviour of the score $-\partial_x \log p(x)$ on a simple one dimensional distribution.

If one represents both the score estimator and the current (e.g., the forward) drift by a nonlinear function approximation such as a neural network, the updated (backward) drift (12) becomes the sum of two neural networks, which is not easily represented as a single one. Hence, during the iterations, one would have to store the entire sequence of past drift functions in order to compute the present one. This would make the algorithm extremely complicated and slow, as we would have to keep in memory $2i$ score matching neural networks of the i'th IPF iteration. This would also mean that, for a single drift evaluation in the i'th IPF iteration, we would have to evaluate the $2i-1$ and $2i$ neural networks for a single drift evaluation at the i'th IPF iteration. For this reason, the score matching approach has not been applied to the Schrödinger bridge problem.

Alternative approaches to computing the drift functions are based on the *Euler–Maruyama* (EM) [15] temporal discretisation of forward and backward SDEs. From its conditional Gaussian transition densities, one can obtain a likelihood function for the drift function evaluated at the discrete time points. Using samples obtained from a forward process, one can estimate the corresponding backward drift using a maximum likelihood or Bayesian approach. This method was applied to the generation of half-bridges by [13], where Gaussian processes were used as a prior distribution over functions. Reference [12] developed a different method that used the conditional Gaussian transition densities of the EM discretisation to approximate the score function. This expression can then be converted into an approximation (which becomes exact in the limit when time interval used for discretisation goes to zero) for the drift function. Both approaches could be viewed as methods for approximating the backward drift (9) using a small time interval h and by computing the conditional expectations within a regression framework. This approach needs strong regularisation, as denoted in [12], which required running averages of the entire function approximators to guard against fatal training divergences as the drifts were trained on local estimates dependent on the interval h, as in Equation (2).

In summary, previous approaches in [12,13] approximated the drifts with the expected infinitesimal change defined in Equation (2) in order to yield a locally tractable reverse drift, thus omitting the influence of the score necessary for the analytical reverse process. In this work, we propose to include a surrogate form of the score term in the reverse drift

such that the respective reverse drifts are trained to approximate the complete reverse drift and not just its localised estimates. We will show in the following that the representation of the drifts (12) and (13) can be directly estimated in a straightforward way by a modification of the score matching approach.

2.4. Score Matching with a Reference Function

To simplify the notation, we denote by $\mu(x_t, t) : \mathbb{R}^D \times [0,1] \to \mathbb{R}^D$ one of the two drift functions and a corresponding marginal density $p_t(x)$ induced by the SDE with drift $\mu(x_t, t)$ and a scaled Wiener process with constant diffusion σ. Following [22], we define the following cost functional of the smooth vectorial function $\phi(x, t) : \mathbb{R}^D \times [0,1] \to \mathbb{R}^D$:

$$\mathcal{L}[\phi, \mu] = \int_0^1 dt \int dx p_t(x) \Big\{ \phi(x,t)^T \phi(x,t) + 2\mu(x,t)^T \phi(x_t,t) + 2\sigma^2 \mathrm{Tr}\Big[\mathrm{J}_\phi(x,t)\Big] \Big\} \quad (14)$$

where $\mathrm{Tr}[J_f(x)]$ is the trace of the Jacobian of a function $f(x) : \mathbb{R}^D \to \mathbb{R}^D$. With respect to the Schrödinger bridge problem, $\mu(x,t)$ would be the drift of the reference process and $\phi(x,t)$ would represent its reverse-time process. We purposefully withheld the harpoons denoting the flow of time earlier, as each half-bridge in Equations (6) and (7) alternates its reference and reverse drift. This score matching with a reference function does not require access to the true score, but instead, uses a surrogate function that is constructed from readily available numerical quantities. By straightforward integration by parts, we can show (see Appendix B) that

$$\phi^*(x,t) \doteq \arg\min_\phi \mathcal{L}[\phi, \mu](x,t) = -\mu(x,t) + \sigma^2 \nabla_x \log p_t(x). \quad (15)$$

Hence, a comparison with Equations (12) and (13) shows that the minimiser of the functional, for a given forward or backward drift, provides the corresponding reverse drift. For a practical computation of the cost function, the integrals over time and over the unknown density in (14) are approximated by numerically generating N_X independent trajectories of the process sampled at N_t regular time points t_j.

Hence, we approximated the cost function by its sample-based estimator:

$$\hat{\mathcal{L}}[\phi, \mu] = \sum_{j=1}^{N_t} \sum_{i=1}^{N_x} \Big\{ \phi(x_{t_j}^{(i)}, t_j)^T \phi(x_{t_j}^{(i)}, t_j) + 2\mu(x_{t_j}^{(i)}, t_j)^T \phi(x_{t_j}^{(i)}, t_j) + 2\sigma^2 \mathrm{Tr}\Big[\mathrm{J}_\phi(x_{t_j}^{(i)}, t_j)\Big] \Big\} \quad (16)$$

For a finite sample size, the empirical cost function must be regularised by controlling the complexity of the functions $\phi(\cdot, \cdot)$. In contrast to [22], we modelled $\phi(x,t) : \mathbb{R}^{D \times [0,1]} \to \mathbb{R}^D$ by a *single nonlinear parametric* function, which is given by a multilayer neural network (rather working with time slices using a distinctive function of x for each). In such a way, we implicitly incorporated the smoothness of the drift in both space x and time t. The construction of the reverse drift and the use of function approximators is exemplified in Figure 3.

2.5. Numerical Considerations and Implementation Details

The trace of the Jacobian requires the evaluation of the derivative of a single output with respect to the single input in the same dimension d, independent of all other outputs and inputs. Since we evaluated a single drift jointly for all dimensions D, this makes the computation of the analytical Jacobian expensive for higher dimensions, as we have to perform D independent backward passes to compute each entry in the Jacobian matrix. The number of gradient computations required for the Jacobian in a vector-valued function thus scales quadratically with the number of dimensions.

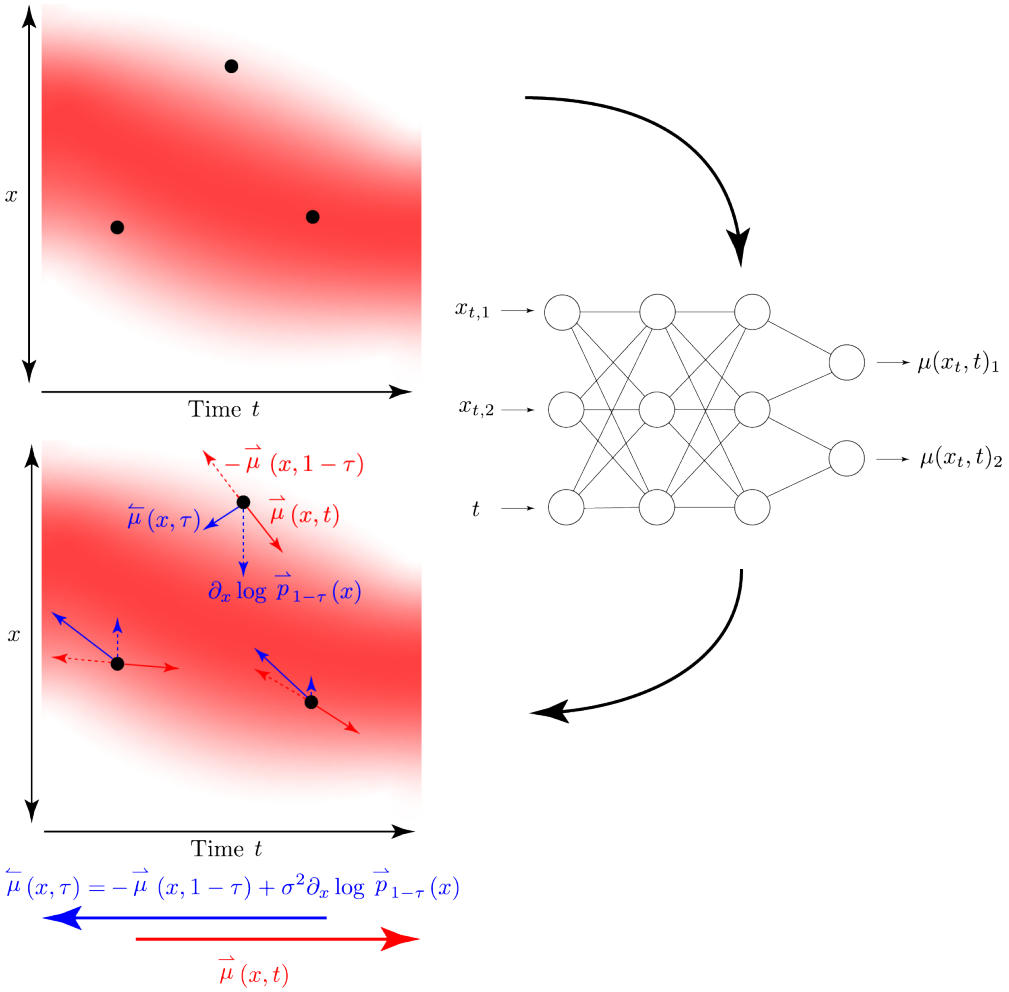

Figure 3. Visualisation of the construction of the backward stochastic process parameterised by the backward SDE with drift $\overleftarrow{\mu}(x, \tau)$. The red gradient represents the marginal distribution $\overrightarrow{p}_t(x)$ induced by the forward SDE with drift $\overrightarrow{\mu}(x, t)$ and marked in the colour red. We employed neural networks to learn both the forward and backward drift, as it allows for a single function approximator per process for the entire input domain as neural networks are inherently able to model vector-valued data.

For data with few dimensions, computing the diagonal terms of the Jacobian can be performed via batched the backpropagation of one-hot output gradients. This approach falters computationally and memorywise when we consider data in higher dimensions. For higher-dimensional data, we opted for the trace estimation trick of Hutchinson with samples from an i.i.d. Rademacher distribution in $\mathbb{R}^D$, namely

$$\mathrm{Tr}\left[J_\phi(x_t, t)\right] = \mathbb{E}_{z \sim p(z)}\left[z^T \nabla_x \left[\phi(x_t, t)^T z\right]\right]. \tag{17}$$

An elaboration on the trace estimation trick can be found in Appendix C.1. The main idea of the stochastic approximation of the trace is that we are only interested in the diagonal elements of the Jacobian. The Hutchinson trace estimation trick proceeds by computing the derivative with respect to the input of the inner product of the prediction $\phi(x_t, t$ and a random variable z. The same random variable is then applied a second time in an inner product to obtain an approximation of the scalar quantity of the trace of the Jacobian. The advantage of the trace estimation trick is that only a single additional derivative evaluation on the inner product $\phi(x_t, t)^T z$ is required, which scales linearly with the number of samples of z.

Taking the gradients of the loss with respect to the parameters requires a second derivative such that a function approximator trained on the reverse drift has to be twice-differentiable. Enforcing this property in neural networks requires us to use at least twice-differentiable evaluations of the prediction with respect to the spatial input x_t, which necessitates twice-differentiable activation functions such as the hyperbolic tangent or Gelu activation functions [23].

If the function approximator is only once differentiable as with the use of rectified linear unit activation functions [24], we can employ Stein's lemma to estimate the trace of the Jacobian. For this estimator, following [25,26], we define an isotropic Gaussian perturbation distribution $z \sim \mathcal{N}(x_t, \sigma_z^2 I)$ with $x_t, z \in \mathbb{R}^D$ around each data point $x_t \in \mathbb{R}^D$ and average the gradients in the z-neighbourhood of the data point:

$$\mathrm{Tr}[J_\phi(x_t, t)] = \lim_{\sigma_z \downarrow 0} \mathbb{E}_{p(z)} \left[\phi(x_t + z, t)^T \frac{z}{\sigma_z^2} \right] \tag{18}$$

The derivation of Stein's lemma can be followed up in Appendix C.2, and its application to the trace estimation therefrom is detailed in Appendix C.3. For a practical implementation, we approximated the Stein estimator using a sufficiently small σ_z by drawing only a single random vector $z_j^{(i)}$ for each trajectory i and each time point j. The perturbed ϕ function values can be computed along side the unperturbed values in a single forward pass through the neural network.

3. Results

We evaluated our proposed method on artificially generated datasets with varying dimensions and with and without dependencies between the dimensions at the two target marginals. The first set of experiments were performed on the construction of the Schrödinger bridge between Gaussian mixture models with which we could explore the behaviour with a changing set of dimensions. The second collection of experiments focused on manifold learning in which implicit distributions were learned to be generated from a standard normal distribution. Finally, we employed our proposed framework on the generation of intermediary distributions of embryoid single-cell RNA as a real-world application.

3.1. Experimental Setup

For all our experiments, we used a deep neural network, taking both the spatial input x_t and the time t as distinct inputs. We fixed the size of the fully connected layers in the hidden layers of the deep neural networks to an integer multiple of the spatial dimension D. As a rule of thumb, we used $10D$ neurons in the hidden layers and scaled the depth of the deep neural network with $\max(2, D/5)$.

We used LayerNorm [27] before the spatial features of the hidden layers so as to not destroy the time embeddings. LayerNorm normalises the representation of each sample to a standard normal distribution and has empirically been shown to numerically aid the gradient computation. Thus, we used blocks of the shape $x_{i+1} = x_i$ + Tanh(Linear(LayerNorm(x_i), Embedding(t))). We used the Adam optimiser [28] and cosine annealing [29], training each half-bridge for 1000 steps and annealing the learning rate from 10^{-3} to 10^{-5}. We drew $N_x = 128$ trajectories per bridge and stored them in a buffer of 512 sample trajectories, discarding old trajectories as needed to maintain the fixed buffer size. We constructed the Schrödinger bridge by running 10 IPF iterations.

The simulations of the SDE were performed using the *Euler–Maruyama* [15] approximation with a step size dt with N_t, and we found $N_t = 100$ and $dt = 0.01$ to be robust values working well for our experiments. Our choice of the diffusion σ was motivated by the idea that the samples of the half-bridge process at the first IPF iteration should sufficiently cover the marginal distribution, increasing the possibility that this distribution

is "hit" with at least some sampled trajectories. As the diffusion parameter is yet another hyperparameter, we chose the diffusion according to $\sigma = 1/N_t \cdot dt$.

The drifts of both processes, forward and backward, received as input the spatial information x and the time t. We normalised the time index $t \in [0, N_t \cdot dt]$ to $t \in \{0, 1\}$ as the time index remained fixed over the course of the entire training of the bridge.

The question remains how the reference process $\mathbb{Q}_0^*$ should be chosen for our applications. The authors in [12] used an Ornstein–Uhlenbeck (OU) process [30] for $\mathbb{Q}_0^*$. Its marginal distributions can be computed analytically and do not require solving an SDE numerically, which saves time and computational resources during the very first half-bridge. This choice of a Gaussian reference process could be also motivated from the fact that, for larger times t, the marginal of the process converges to a stationary Gaussian density that could approximately match Gaussian targets used for a denoising-style data generating application of Schrödinger bridges.

In our implementations, we did not want to make any specific assumptions on the end marginals. Hence, a choice of a zero initial drift $\mu(x, t) \equiv 0$ (reducing $\mathbb{Q}_0^*$ to a simple Wiener process) seemed more natural. However, practical considerations suggested a slightly different approach, in which $\mathbb{Q}_0^*$ corresponds to a a drift function represented by a neural network that has (untrained) small random weights, which serve as useful initial conditions for the subsequent training [31,32]. We observed experimentally that for the first half-bridge $\mathbb{Q}_0^*$, the Wiener process dominates the characteristics of the sampled trajectories.

For our applications, the marginal densities at the initial and end times $\overrightarrow{p}_0$ and $\overrightarrow{p}_1$, which are generated by the bridge models, as well as the desired targets π_0 and π_1 are represented by random samples, rather than by analytical expressions. In order to evaluate the quality of the converged bridge, we computed the Wasserstein-1 distances $W_1(\overrightarrow{p}_1, \pi_1)$ and $W_1(\pi_0, \overleftarrow{p}_0)$. The Wasserstein-1 distance [33] between two probability measures μ and ν is defined as

$$W_1(\mu, \nu) = \inf_{\gamma \in \mathbb{D}(\mu,\nu)} \mathbb{E}[||x - y||] \tag{19}$$

where $\mathbb{D}(\mu, \nu)$ is the set of all couplings of μ and ν. These can be straightforwardly evaluated on empirical distributions, which are given by samples. For the underlying optimal transport problem and its efficient solution via linear programming in its dual representation, see, e.g., [17]. The Wasserstein-1 distance is also known as the Earth Movers' Distance (EMD) in computer science.

3.2. Multimodal Parametric Distributions

We modelled the marginal distribution $\pi_0(x)$ as a Gaussian distribution with a diagonal covariance matrix. The opposite marginal distribution $\pi_1(x)$ was a Gaussian mixture model with two modes with a uniform prior over the GMM component centres. The mean values of all Gaussian distributions, uni-modal in $\pi_0(x)$, as well as bi-modal in $\pi_1(x)$, were sampled uniformly from $\mathcal{U}(-2.5, 2.5)$, and a standard deviation of 1.0 was used throughout. The visualisation of the inferred Schrödinger bridge highlights the learning of the time-dependent drift and the ability to model bifurcations in the case of bi-modal GMMs, as seen in Figure 4.

The dataset was created as the more tractable experiment in comparison to subsequent datasets. The use of GMM's allows for the analytical evaluation of the probability of the generated data at the marginal distributions. Furthermore, the modes of the GMM could be handcrafted, which turned out to be important to validate numerous design choices of the drift approximators. The authors deemed it equally important to visually verify the generated trajectories, ensuring that the drift approximators were able to model, for example, bifurcations and how they dealt with changing hyperparameters such as increased diffusion.

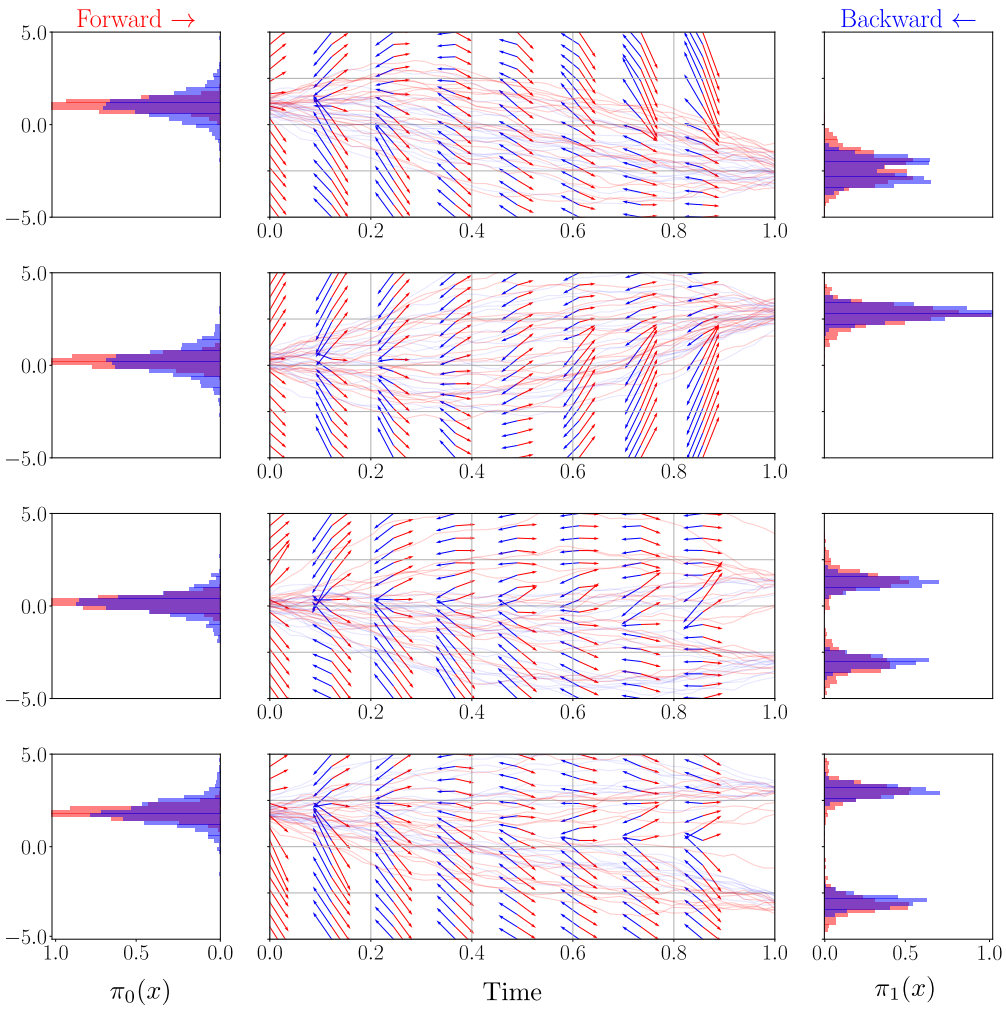

Figure 4. Visualisation of a solved Schrödinger bridge in $\mathbb{R}^4$ in which each dimension is plotted independently. The sampled trajectories and drift of the forward process and its initial condition π_0 are shown in red, whereas the backward process is shown in blue.

Estimating the score in regions with little probability mass is a difficult problem, as an insufficient amount of samples may be drawn from that region to accurately model the score. This was explored in detail in [1,34] and was one of the inspirations to the perturbation protocol in diffusion models. Diffusion models have the advantage of computing an analytical score at any point in space and time through their analytical perturbation kernels defined in the forward process.

The Schrödinger bridge problem offers none of these luxuries, as no reference process is available that yields analytical scores. Thus, we require sufficient data even in low-probability regions to enable us to estimate the necessary score. We found in our experiments that the hyperparameter with the single largest influence was the number of trajectories that were sampled from the path measures. We can thus see in Figure 5 that increasing the number of trajectories decreases the Wasserstein-1 distance with respect to the marginal distributions $\pi_0(x)$ and $\pi_1(x)$ as the score estimation becomes more precise, as even low-probability regions of the stochastic processes, on which the score is estimated, are sampled adequately.

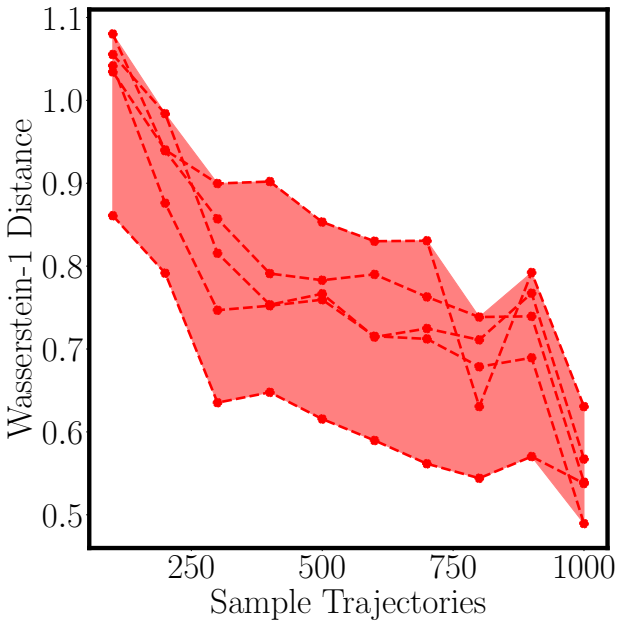

Figure 5. A comparison of the Wasserstein-1 distance on the number of sample trajectories sampled per IPF iteration for the multi-modal distribution problem in Section 3.2. The Wasserstein-1 distances $W_1(\overrightarrow{p}_1, \pi_1)$ and $W_1(\pi_0, \overleftarrow{p}_0)$ were averaged and show an overall decrease relative to the number of trajectories sampled from the stochastic processes.

We chose the multi-modal dataset as a test dataset to compare Hutchinson's trace estimator with the trace estimation via Stein's lemma. The result can be seen in Figure 6, in which we evaluated the Wasserstein-1 distance over a fixed set of dimensions. We observed that the Wasserstein-1 distance increases linearly with the number of dimensions for a fixed number of samples, and it was interesting to observe that the Schrödinger bridges with different trace estimators behaved very similar in terms of performance. We compared Stein's trace estimator as detailed above and in Appendix C.3 with varying perturbation scales with Hutchinson's trace estimation and the ground truth Wasserstein distance between the marginal distributions $\pi_0(x)$ and $\pi_1(x)$. In theory, Hutchinson's trace estimation can also be applied with a normal distribution sampling the random projection vectors, yet Rademacher's distribution has the lowest estimator variance. Stein's trace estimation can only be performed with the normal distribution, as it relies on integration by parts and the special derivative of the normal distribution. This led us to hypothesise that the inherently higher variance of the normal distribution in Stein's trace estimator leads to worse performance. However, we aim at examining this in future work.

3.3. Manifold Datasets

For the second dataset, we trained the forward and backward drifts on the generated manifold data via the Sklearn machine learning package [35]. The manifolds used were "make_swiss_roll", "make_s_curve", and "make_moons" from the sklearn.dataset code base, which were concatenated to create a higher-dimensional manifold. This increased the complexity of the manifold, setting it apart form earlier manifold modelling approaches, as in [12].

We trained the stochastic processes to predict multiple manifolds at once by modelling them jointly with a single fully connected neural network. For $\pi_0(x)$, we chose a standard normal multivariate distribution $\pi_0(x) = \mathcal{N}(0, I)$, while $\pi_1(x)$ was the implicit distribution generated by samples on the manifold. Whereas the previous Gaussian mixture models were statistically independent in each dimension, the manifolds explicitly modelled the statistical correlation between different dimensions. A visualisation of the Schrödinger bridge between the two distributions can be seen in Figure 7.

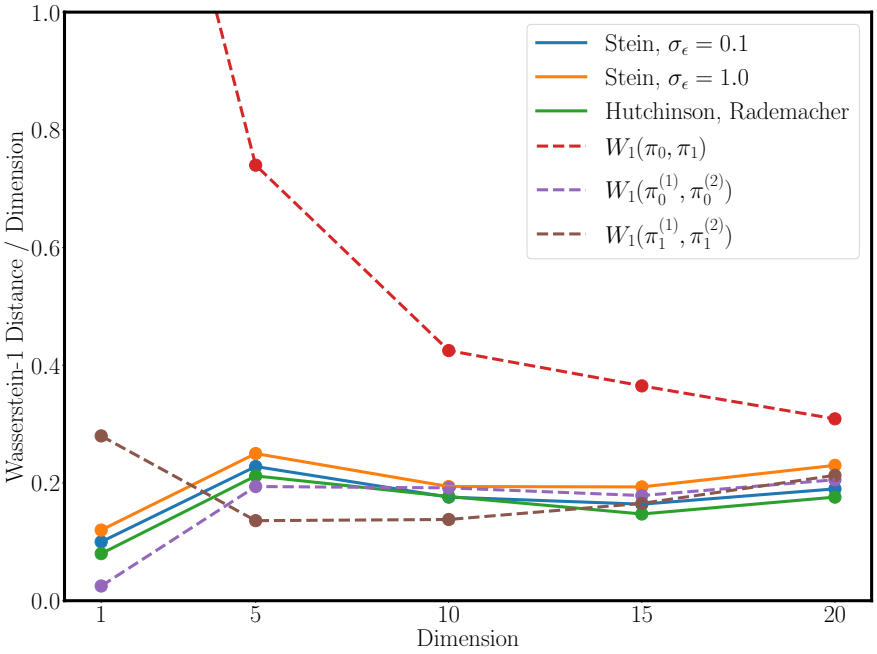

Figure 6. A comparison of the gradient estimators on an increasing number of dimensions with the average Wasserstein-1 distance per dimension between the true marginal and the predicted marginal distributions of the forward and backward process. As baselines, the Wasserstein-1 distances between the respective marginals is shown. One can see that the gradient estimators performed as well as the Wasserstein-1 distances between samples drawn from the marginals denoted as $W_1(\pi_0(x), \pi_0(x))$ and $W_1(\pi_1(x), \pi_1(x))$. The Hutchinson trace estimator performed best while requiring a second derivation. Interestingly, smaller sampling variances for the Gaussian distribution in the Stein gradient estimator yielded better overall performance on matching the marginal distributions.

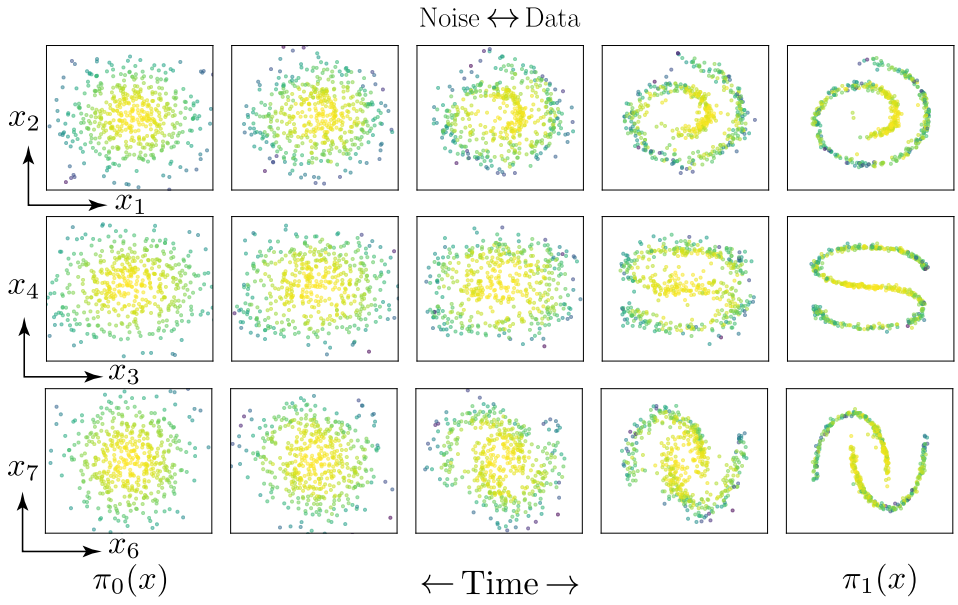

Figure 7. Visualisation of a constructed Schrödinger bridge for the manifold $\mathbb{R}^6$, which lies in $\mathbb{R}^{10}$. The left-most column represents samples from the marginal distribution $\pi_0(x)$, which are transformed into samples on the manifolds in the right-most column. The colour coding of each particle is according to its proximity to the mean of the prior distribution $\pi_0(x)$ such that we can distinguish which particles from $\pi_0(x)$ correspond to the particles on the manifold $p(x_1, 1)$.

The use of a tractable probability distribution as one marginal distribution was inspired by the purely generative task of diffusion models. The dataset is commonly used as a visual benchmark of new generative models and allows evaluating the drift approximators' ability to model nonlinear manifolds. A lack of this dataset is the absence of a tractable data likelihood under the marginal distribution, as the manifold generation function is modelled as an implicit distribution.

3.4. Embryoid Dataset

Single-cell RNA sequencing analyses the RNA of individual cells, destroying it unfortunately and making it inaccessible for further analysis. In a population of cells, we can remove individual cells and analyse their RNA. As each cell is eliminated from the population, we have to turn to a probabilistic method to simulate the development of the RNA at the *population* level.

Therefore, it is of interest to develop a methodology that can simulate the full trajectories of RNA sequences over time, which would allow for predicting the outcomes of such measurements on a single cell without actually having to perform them. As suggested in [36], an interesting solution to this problem would be the construction of a stochastic generative model for the possible measurements with the marginal distributions (for a population of cells) of the actual measurements at the initial and end time as boundary conditions. A visualisation of the application of the Schrödinger bridge to the RNA measurement task is provided in Figure 8. If we represent this generative model by a diffusion process, we naturally end up with the idea of applying Schrödinger bridges to this problem.

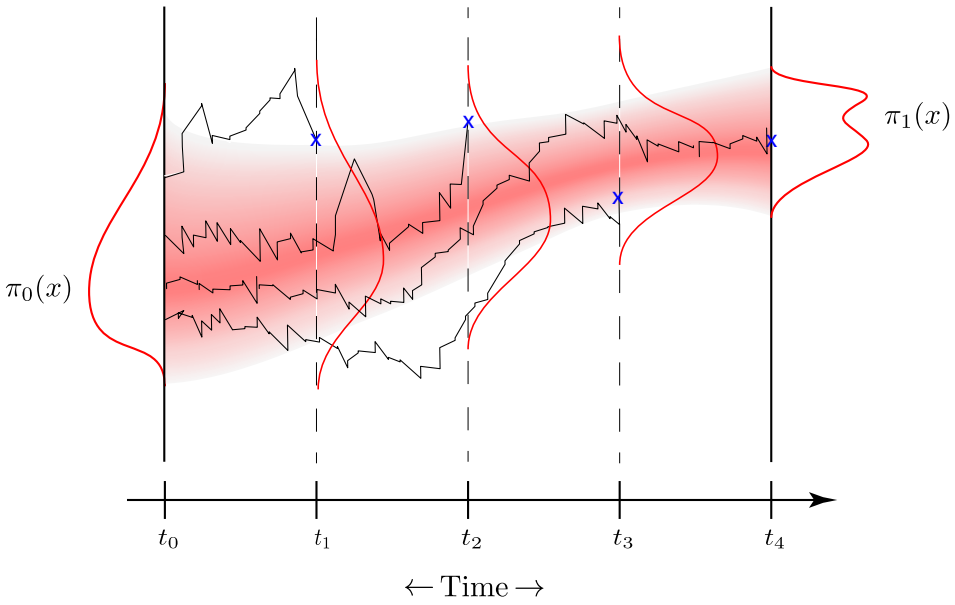

Figure 8. A visualisation of how the Schrödinger bridge is applied to the RNA measurements. The RNA of different cells is measured at the discrete time steps t_1, t_2, and t_3 and eliminated from the population marked by the blue crosses and the end of each of the four trajectories. Given these snapshots of the RNA distribution at discrete time steps, the task is now to construct a Schrödinger bridge between the discrete time steps as indicated by the stochastic process with a fading colour. The generative model can then be queried at any time step between t_0 and t_4.

We applied this approach to the embryoid dataset [37] for which single-cell RNA measurements were taken at five different times $t \in \{0, 1, 2, 3, 4\}$. The ensemble of measurements at each time index t consists of a varying number (2380, 4162, 3277, 3664, 3331) of RNA measurement samples. We considered the problem of constructing a Schrödinger bridge using only the first day and the very last day measurements. We then tried to infer the hold-out intermediate marginal distributions of the RNA measurements with the help of the constructed bridge. The samples drawn from the bridge were thus evaluated at the

intermediate marginal distributions at $t \in \{1, 2, 3\}$ using the Wasserstein-1 distance. In addition, we also evaluated the Schrödinger bridge at the two end-points by computing $W_1(\overleftarrow{p}_0, \pi_0)$ and $W_1(\overrightarrow{p}_1, \pi_1)$.

We compared our method with two other generative model approaches: *TrajectoryNet* [37] implements constrained normalising flows using neural networks. The paper was also the first to propose this benchmark. IPML [13] presents an alternative method to constructing a Schrödinger bridge in which the required drift functions are estimated from the trajectories using Gaussian process regression without computing score functions, as remarked upon in Section 2.3. Finally, we have also included a simpler method, which is based on Optimal Transport (OT). It computes a linear transport map between samples of the initial and end distributions. However, the comparison of this method with the others is slightly unfair. While the OT approach allows for making predictions of the marginals at intermediate time steps, it is not formulated as a generative model and, thus, could not be used to make predictions for measurements on single cells.

The RNA sequencing data were generated from FACS-sorted embryoid bodies via surface marker indication. The dataset was preprocessed with PHATE [36] to a dimensionality of $D = 5$, as performed in the reference methods. An important feature of the PHATE preprocessing is the reversibility of the dimensionality reduction. The preprocessing pipeline can be accessed and replicated with the public PHATE code base provided by the authors of [36]. Thus, it is possible to apply trajectory reconstructing algorithms in the low-dimensional representation of the genetic data and project the resulting trajectory back into the high-dimensional space. This is in contrast to commonly used dimensionality reduction algorithms, which do not offer these advantages. We refer the reader to [36] for an in-depth treatment of the algorithm. The experimental setup was kept identical for *IPFML* and TrajectoryNet.

Table 1 compares the performance of our Schrödinger bridge method with the other methods. Our method outperformed both Trajectorynet and the alternative Schrödinger bridge method IPML and was also better than the OT approach in two out of three instances. This corroborates the result from [13] that OT's linear transport map is not a well-fitting intermediate distribution for $t = 4$. Especially, the Wasserstein distance at the marginal distributions of the backward process at $t = 0$ and the forward process $t = 4$ were well modelled.

Table 1. Comparison of comparable methodologies on the embryoid dataset with the Wasserstein-1 distance The "Path" column denotes the average EMD of the intermediate time steps $T \in \{2, 3, 4\}$, whereas the "Full" column averages all time steps. The optimal transport linear transport map is only defined for the intermediate time steps, as it requires the two marginal distributions at $T \in \{1, 5\}$.

Method ↓	**t = 1**	**t = 2**	**t = 3**	**t = 4**	**t = 5**	**Path**	**Full**
TrajectoryNet	0.62	1.15	1.49	1.26	0.99	1.30	1.18
IPML EQ	0.38	1.19	1.44	1.04	0.48	1.22	1.02
IPML EXP	0.34	**1.13**	1.35	1.01	0.49	1.16	0.97
OT	N/A	1.13	1.10	1.11	N/A	1.16	N/A
Reverse-SDE	**0.23**	1.16	**1.00**	**0.64**	**0.28**	**0.93**	**0.66**

4. Discussion and Conclusions

We presented a method for solving the Schrödinger bridge problem based on the iterative proportional fitting algorithm. In contrast to the simpler diffusion models for data generation, Schrödinger bridges can provide interpolations between *two* arbitrary data distributions. Unfortunately, this advantage comes with the drawback that the score functions that can be used to compute drift functions for the time-reversed process are not analytically available. The novel aspect of our approach is the formulation of reverse-time drift functions as solutions of a minimisation problem (an extension of the score matching method) involving expectations over the forward process. This allows for an efficient

training of neural networks representing the drift functions on simulated trajectories of the corresponding stochastic differential equations.

To evaluate our approach, we conducted experiments on two synthetic datasets, which allowed us to analyse its performance within a controlled environment. Finally, we applied our method to a single-cell mRNA dataset in which a Schrödinger bridge was constructed between intermediate measurements, where the initial and end distributions were both non-Gaussian. On this dataset, we outperformed similar methodologies, also beating a linear transportation map in two out of three instances.

Our variational approach of estimating drift functions in the IPF algorithm for solving Schrödinger bridges could be extended in various ways:

- The variational formulation of our drift estimators is independent of the temporal discretisation used for creating sample trajectories. It only depends on the *marginal distributions of the state variables*. This fact opens up alternative possibilities for generating appropriate samples by forward and backward simulations, which could allow for larger stepsizes. One might, e.g., consider *weak* approximation schemes [38] for numerically simulating SDEs. A different alternative is the application of *deterministic*, particle-based simulations [22], where the reduced variance of estimators might allow keeping the number of trajectories small. Finally, *exact sampling* methods (see, e.g., [39]), which entirely avoid temporal discretisation, would be interesting candidates.
- Reliably estimating the score-based drift functions in regions with small marginal probability densities remains a challenging problem, especially for higher dimensions. This is evident in our synthetic experiments, as the number of trajectory samples remained the most-important hyperparameter to ensure the convergence of the Schrödinger bridge (see also [34] for similar observations). It would be interesting to see if prior knowledge expressed by exact analytical results for asymptotic scaling of densities could be implemented in the function approximators to improve on that problem.
- It is relatively straightforward to adapt the variational approach (see e.g., [22]), together with a corresponding change in the relations (12) and (13) to solve Schrödinger bridge problems for more general types of stochastic differential equations. Interesting cases could include SDEs with (fixed) state and time-dependent diffusion matrices, as well as processes based on *Langevin dynamics* (i.e., systems of second-order SDEs), well-known for modelling physical systems that are also used for Hamilton Monte Carlo simulations.
- It would be interesting to investigate possible simplifications of our method for the special case in which the drift of the reference process is the gradient of a potential function. The relations (12) and (13) show that all half-bridges in the IPF algorithm will inherit this property. One could then modify the variational formulation and learn directly the potential, rather than the D components of its gradient individually. This built-in symmetry might increase the accuracy of estimation, but the need for second derivatives in the cost function could lead to an increase of the numerical complexity of training a neural network.

Author Contributions: Conceptualisation, M.O. and C.O.; methodology, L.W.; software, L.W.; validation, L.W.; data curation, L.W.; writing—original draft preparation, L.W.; writing—review and editing, C.O.; visualisation, L.W.; supervision, C.O. All authors have read and agreed to the published version of the manuscript.

Funding: M.O. and C.O. were partially funded by Deutsche Forschungsgemeinschft (DFG) Project-ID 318763901-SFB1294. The research of L.W. was funded by the BIFOLD-Berlin Institute for the Foundations of Learning and Data (Reference 01IS18025A and Reference 01IS18037A).

Data Availability Statement: The RNA data can be found in this github repository.

Acknowledgments: Ludwig Winkler would like to thank Klaus-Robert Müller for fruitful discussions and proof reading and Jason Salomon Rinnert for technical support. M.O. would like to thank Sebastian Reich for many fruitful discussions. We acknowledge support by the German Research Foundation and the Open Access Publication Fund of TU Berlin.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Reverse-Time Evolution of Probability Distributions

Both the Kolmogorov forward (KFE) and the Kolmogorov backward (KBE) [40] equations are partial differential equations that describe the evolution of a probability distribution forward and backward in time. The Kolmogorov forward equation is identical to the Fokker–Planck equation and states

$$\partial_t p(x_t) = -\partial_{x_t}[\mu(x_t)p(x_t)] + \frac{1}{2}\partial^2_{x_t}\Big[\sigma^2(x_t)\; p(x_t)\Big]. \tag{A1}$$

The Kolmogorov backward equation for $s \geq t$ is defined as

$$-\partial_t p(x_s|x_t) = \mu(x_t)\; \partial_{x_t} p(x_s|x_t) + \frac{1}{2}\; \sigma^2(x_t)\; \partial^2_{x_t} p(x_s|x_t). \tag{A2}$$

While a stochastic differential equation of the form of $dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$ induces a path measure $p(x_t, t)$, Reference [20] showed that there is an equivalent stochastic differential equation inducing a path measure $p(x_\tau, \tau)$.

Applying the Bayes theorem allows us to factorise by conditioning: $p(x_s, x_t) = p(x_s|x_t)p(x_t)$ with the time ordering $t \leq s$. For deriving the reverse-time stochastic differential equation, we start out by applying the negative time differential to the joint distribution of $p(x_s, x_t)$:

$$\begin{aligned} -\partial_t p(x_s, x_t) &= -\partial_t[p(x_s|x_t)p(x_t)] && \text{(A3)}\\ &= \underbrace{-\partial_t p(x_s|x_t)}_{\text{KBE}}\; p(x_t) - p(x_s|x_t)\; \underbrace{\partial_t p(x_t)}_{\text{KFE}} && \text{(A4)} \end{aligned}$$

into which we can substitute the Kolmogorov forward (KFE) and Kolmogorov backward (KBE) equations:

$$\begin{aligned} &-\partial_t p(x_s, x_t) && \text{(A5)}\\ =& -\partial_t p(x_s|x_t)p(x_t) - p(x_s|x_t)\partial_t p(x_t) && \text{(A6)}\\ =& \left(\mu(x_t)\; \partial_{x_t} p(x_s|x_t) + \frac{1}{2}\; \sigma^2(x_t)\; \partial^2_{x_t} p(x_s|x_t)\right) p(x_t) && \text{(A7)}\\ &+ p(x_s|x_t)\left(\partial_{x_t}[\mu(x_t)p(x_t)] - \frac{1}{2}\partial^2_{x_t}\Big[\sigma^2(x_t)\; p(x_t)\Big]\right) && \text{(A8)} \end{aligned}$$

The derivative occurring in the backward Kolmogorov equation is

$$\begin{aligned} \partial_{x_t} p(x_s|x_t) &= \partial_{x_t}\left[\frac{p(x_s, x_t)}{p(x_t)}\right] && \text{(A9)}\\ &= \frac{\partial_{x_t} p(x_s, x_t)p(x_t) - p(x_s, x_t)\partial_{x_t} p(x_t)}{p^2(x_t)} && \text{(A10)}\\ &= \frac{\partial_{x_t} p(x_s, x_t)}{p(x_t)} - \frac{p(x_s, x_t)\partial_{x_t} p(x_t)}{p^2(x_t)} && \text{(A11)} \end{aligned}$$

Next, we evaluate the derivative of the products in the forward Kolmogorov equation:

$$\begin{aligned} \partial_{x_t}[\mu(x_t)p(x_t)] &= \partial_{x_t}\mu(x_t)\; p(x_t) + \mu(x_t)\; \partial_{x_t} p(x_t) && \text{(A12)}\\ \partial^2_{x_t}\Big[\sigma^2(x_t)\; p(x_t)\Big] &= \partial^2_{x_t}\sigma^2(x_t)\; p(x_t) + 2\; \partial_{x_t}\sigma^2(x_t)\; \partial_{x_t} p(x_t) + \sigma^2(x_t)\; \partial^2_{x_t} p(x_t). && \text{(A13)} \end{aligned}$$

Substituting the derivatives of the probability distributions accordingly, we obtain

$$
\begin{aligned}
-\partial_t p(x_s, x_t) &= -\partial_t [p(x_s|x_t)p(x_t)] && \text{(A14)}\\
&= -\partial_t p(x_s|x_t)p(x_t) - p(x_s|x_t)\partial_t p(x_t) && \text{(A15)}\\
&= \left(\mu(x_t)\ \partial_{x_t} p(x_s|x_t) + \frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\right) p(x_t) && \text{(A16)}\\
&\quad + p(x_s|x_t)\left(\partial_{x_t}[\mu(x_t)p(x_t)] - \frac{1}{2}\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right]\right) && \text{(A17)}\\
&= \mu(x_t)\ \partial_{x_t} p(x_s|x_t)\ p(x_t) + \frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\ p(x_t) && \text{(A18)}\\
&\quad + p(x_s|x_t)\partial_{x_t}\mu(x_t)\ p(x_t) + p(x_s|x_t)\mu(x_t)\ \partial_{x_t} p(x_t) && \text{(A19)}\\
&\quad - \frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right] && \text{(A20)}\\
&= \mu(x_t)\ \left(\frac{\partial_{x_t} p(x_s, x_t)}{\cancel{p(x_t)}} - \frac{p(x_s, x_t)\partial_{x_t} p(x_t)}{p^{\cancel{2}}(x_t)}\right) \cancel{p(x_t)} && \text{(A21)}\\
&\quad + p(x_s|x_t)\partial_{x_t}\mu(x_t)\ p(x_t) + p(x_s|x_t)\mu(x_t)\ \partial_{x_t} p(x_t) && \text{(A22)}\\
&\quad + \frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\ p(x_t) - \frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right] && \text{(A23)}\\
&= \mu(x_t)\ \left(\partial_{x_t} p(x_s, x_t) - \frac{p(x_s, x_t)\partial_{x_t} p(x_t)}{p(x_t)}\right) && \text{(A24)}\\
&\quad + p(x_s|x_t)\partial_{x_t}\mu(x_t)\ p(x_t) + p(x_s|x_t)\mu(x_t)\ \partial_{x_t} p(x_t) && \text{(A25)}\\
&\quad + \frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\ p(x_t) - \frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right] && \text{(A26)}\\
&= \mu(x_t)\ \left(\partial_{x_t} p(x_s, x_t) - \cancel{p(x_s|x_t)\partial_{x_t} p(x_t)}\right) && \text{(A27)}\\
&\quad + p(x_s, x_t)\partial_{x_t}\mu(x_t) + \cancel{p(x_s|x_t)\mu(x_t)\ \partial_{x_t} p(x_t)} && \text{(A28)}\\
&\quad + \frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\ p(x_t) - \frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right] && \text{(A29)}\\
&= \underbrace{\mu(x_t)\ \partial_{x_t} p(x_s, x_t) + p(x_s, x_t)\partial_{x_t}\mu(x_t)}_{\text{product rule}} && \text{(A30)}\\
&\quad + \frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\ p(x_t) - \frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right] && \text{(A31)}\\
&= \partial_{x_t}[\mu(x_t)\ p(x_s, x_t)] && \text{(A32)}\\
&\quad + \underbrace{\frac{1}{2}\ \sigma^2(x_t)\ \partial^2_{x_t} p(x_s|x_t)\ p(x_t)}_{(1)} - \underbrace{\frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right]}_{(2)} && \text{(A33)}
\end{aligned}
$$

In order to transform the partial differential equation above into a form from which we can deduce an equivalent stochastic differential equation, we match the terms of the second-order derivatives with the following identity:

$$
\begin{aligned}
&\frac{1}{2}\partial^2_{x_t}\left[p(x_s, x_t)\sigma^2(x_t)\right] && \text{(A34)}\\
=&\frac{1}{2}\partial^2_{x_t}\left[p(x_s|x_t)p(x_t)\sigma^2(x_t)\right] && \text{(A35)}\\
=&\frac{1}{2}\partial^2_{x_t} p(x_s|x_t)p(x_t)\sigma^2(x_t) + \partial_{x_t}\left[p(x_t)\sigma^2(x_t)\right]\partial_{x_t} p(x_s|x_t) + \frac{1}{2}\partial^2_{x_t}\left[p(x_t)\sigma^2(x_t)\right] p(x_s|x_t) && \text{(A36)}\\
=&\underbrace{\frac{1}{2}\sigma^2(x_t)\partial^2_{x_t} p(x_s|x_t)p(x_t)}_{(A37.1)} + \underbrace{\partial_{x_t}\left[p(x_t)\sigma^2(x_t)\right]\partial_{x_t} p(x_s|x_t)}_{(A37.2)} + \underbrace{\frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\left[p(x_t)\sigma^2(x_t)\right]}_{(A37.3)} && \text{(A37)}
\end{aligned}
$$

by observing that the terms (1) and (2) occurring in both equations. We can see from the expansion of the derivative above that we can combine the terms in our derivation if we expand the "center term". Furthermore, we can employ the identity $-\frac{1}{2}X = -X + \frac{1}{2}X$ to obtain

$$
\begin{aligned}
-\partial_t p(x_s, x_t) =& \partial_{x_t}[\mu(x_t)\ p(x_s, x_t)] && \text{(A38)}\\
&+ \frac{1}{2}\,\sigma^2(x_t)\,\partial^2_{x_t} p(x_s|x_t)\ p(x_t) - \frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big] && \text{(A39)}\\
=& \partial_{x_t}[\mu(x_t)\ p(x_s, x_t)] && \text{(A40)}\\
&+ \frac{1}{2}\,\sigma^2(x_t)\ p(x_t)\ \partial^2_{x_t} p(x_s|x_t) \underbrace{-\frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big]}_{-\frac{1}{2}X = -X + \frac{1}{2}X} && \text{(A41)}\\
&\underbrace{\pm \partial_{x_t} p(x_s|x_t)\partial_{x_t}\Big[p(x_t)\sigma^2(x_t)\Big]}_{\text{complete the square}} && \text{(A42)}\\
=& \partial_{x_t}[\mu(x_t)\ p(x_s, x_t)] + \overbrace{\frac{1}{2}\,\sigma^2(x_t)\,\partial^2_{x_t} p(x_s|x_t)\ p(x_t)}^{(A37.1)} && \text{(A43)}\\
&\underbrace{-p(x_s|x_t)\partial^2_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big] + \overbrace{\frac{1}{2} p(x_s|x_t)\partial^2_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big]}^{(A37.2)}}_{-\frac{1}{2}X = -X + \frac{1}{2}X} && \text{(A44)}\\
&+ \overbrace{\partial_{x_t} p(x_s|x_t)\partial_{x_t}\Big[p(x_t)\sigma^2(x_t)\Big]}^{(A37.3)} - \partial_{x_t} p(x_s|x_t)\partial_{x_t}\Big[p(x_t)\sigma^2(x_t)\Big] && \text{(A45)}\\
=& \partial_{x_t}[\mu(x_t)\ p(x_s, x_t)] + \overbrace{\frac{1}{2}\partial^2_{x_t}\Big[p(x_s|x_t)p(x_t)\sigma^2(x_t)\Big]}^{(A34)} && \text{(A46)}\\
&\underbrace{-p(x_s|x_t)\partial^2_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big] - \partial_{x_t} p(x_s|x_t)\partial_{x_t}\Big[p(x_t)\sigma^2(x_t)\Big]}_{-\partial_{x_t}\left[p(x_s|x_t)\partial_{x_t}[\sigma^2(x_t)\ p(x_t)]\right]\ \text{(product rule)}} && \text{(A47)}\\
=& \partial_{x_t}[\mu(x_t)\ p(x_s, x_t)] + \frac{1}{2}\partial^2_{x_t}\Big[p(x_s, x_t)\sigma^2(x_t)\Big] && \text{(A48)}\\
&- \partial_{x_t}\Big[p(x_s|x_t)\partial_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big]\Big]. && \text{(A49)}
\end{aligned}
$$

What remains to be done is to combine the joint probability and the conditional probability in the first-order derivative terms to combine them:

$$
\begin{aligned}
-\partial_t p(x_s, x_t) =& \partial_{x_t}\Big[\mu(x_t)\ p(x_s, x_t) - p(x_s|x_t)\partial_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big]\Big] && \text{(A50)}\\
&+ \frac{1}{2}\partial^2_{x_t}\Big[p(x_s, x_t)\sigma^2(x_t)\Big] && \text{(A51)}\\
=& \partial_{x_t}\Big[p(x_s, x_t)\Big(\mu(x_t) - \frac{1}{p(x_t)}\partial_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big]\Big)\Big] && \text{(A52)}\\
&+ \frac{1}{2}\partial^2_{x_t}\Big[p(x_s, x_t)\sigma^2(x_t)\Big] && \text{(A53)}\\
=& -\partial_{x_t}\Big[p(x_s, x_t)\Big(-\mu(x_t) + \frac{1}{p(x_t)}\partial_{x_t}\Big[\sigma^2(x_t)\ p(x_t)\Big]\Big)\Big] && \text{(A54)}\\
&+ \frac{1}{2}\partial^2_{x_t}\Big[p(x_s, x_t)\sigma^2(x_t)\Big] && \text{(A55)}
\end{aligned}
$$

the result of which is in the form of a Kolmogorov forward equation, although using the joint probability distribution $p(x_s, x_t)$. For the time ordering of $t \leq s$, we can observe that the term $-\partial_t p(x_s, x_t)$ describes the change of the probability distribution as we move backward in time. In accordance with Leibniz' rule, we can marginalise over x_s without interfering with the partial derivative ∂_t, to obtain

$$-\partial_t p(x_t) = -\partial_{x_t}\left[p(x_t)\left(-\mu(x_t) + \frac{1}{p(x_t)}\partial_{x_t}\left[\sigma^2(x_t)\ p(x_t)\right]\right)\right] \tag{A56}$$

$$+\frac{1}{2}\partial^2_{x_t}\left[p(x_t)\sigma^2(x_t)\right] \tag{A57}$$

and introduce the time reversal $\tau \doteq 1 - t$, which, with respect to the integration with respect to the flow of time, yields

$$-\partial_t p(x_t) = \partial_\tau p(x_{1-\tau}) \tag{A58}$$

$$= -\partial_{x_{1-\tau}}\left[p(x_{1-\tau})\left(-\mu(x_{1-\tau}) + \frac{1}{p(x_{1-\tau})}\partial_{x_{1-\tau}}\left[\sigma^2(x_{1-\tau})\ p(x_{1-\tau})\right]\right)\right] \tag{A59}$$

$$+\frac{1}{2}\partial^2_{x_{1-\tau}}\left[p(x_{1-\tau})\sigma^2(x_{1-\tau})\right] \tag{A60}$$

which finally gives us a stochastic differential equation analogous to the Fokker–Planck/forward Kolmogorov equation, which we can solve backward in time:

$$dX_\tau = \left(-\mu(x_{1-\tau}) + \frac{1}{p(x_{1-\tau})}\partial_{x_{1-\tau}}\left[\sigma^2(x_{1-\tau})\ p(x_{1-\tau})\right]\right)d\tau + \sigma(x_{1-\tau})dW_\tau \tag{A61}$$

where W_τ is a Wiener process that flows backward in time.

By keeping the $\sigma^2(x_{1-\tau})$ constant and independent of x_t and applying the log-derivative trick, the drift simplifies to

$$dX_\tau = \Big(-\mu(x_{1-\tau}) + \frac{1}{p(x_{1-\tau})}\partial_{x_{1-\tau}}[\overbrace{\sigma^2(x_{1-\tau})}^{=\sigma^2}\ p(x_{1-\tau})]\Big)d\tau + \sigma(x_{1-\tau})dW_\tau \tag{A62}$$

$$= \left(-\mu(x_{1-\tau}) + \frac{\sigma^2}{p(x_{1-\tau})}\partial_{x_{1-\tau}}\ p(x_{1-\tau})\right)d\tau + \sigma(x_{1-\tau})dW_\tau \tag{A63}$$

$$= \Big(-\mu(x_{1-\tau}) + \sigma^2\partial_{x_{1-\tau}}\ \log p(x_{1-\tau})\Big)dt + \sigma(x_{1-\tau})d\tilde{W}_\tau \tag{A64}$$

Appendix B. Log-Gradient-Density Estimation and Implicit Score Estimation

Here, we present the proof of the equivalency that the optimum of the criterion does indeed yield the correct optimum when $\phi(x) = -\mu(x) + \sigma^2 \nabla_x \log p(x)$. The proof goes as follows:

$$\mathcal{L}[\phi, \mu] = \mathbb{E}\Big[\phi^2(x) + 2\mu(x)\phi(x) + 2\sigma^2 \partial_x \phi(x)\Big] \tag{A65}$$

$$= \int p(x)\Big(\phi^2(x) + 2\mu(x)\phi(x) + 2\sigma^2 \partial_x \phi(x)\Big)dx \tag{A66}$$

$$= \int p(x)\Big(\phi^2(x) + 2\mu(x)\phi(x)\Big)dx + 2\sigma^2 \underbrace{\int p(x)\partial_x \phi(x)dx}_{\text{integration by parts}} \tag{A67}$$

$$= \int p(x)\Big(\phi^2(x) + 2\mu(x)\phi(x)\Big)dx - 2\sigma^2 \int \underbrace{\partial_x p(x)\phi(x)}_{\text{log-derivative trick}} dx \tag{A68}$$

$$= \int p(x)\Big(\phi^2(x) + 2\mu(x)\phi(x) - 2\sigma^2 \phi(x)\partial_x \log p(x))\Big)dx \tag{A69}$$

$$= \int p(x)\Big(\underbrace{\phi^2(x) + 2\phi(x)\{\mu(x) - \sigma^2 \partial_x \log p(x)\}}_{\text{complete the square}}\Big)dx \tag{A70}$$

$$= \int p(x)\Bigg(\Big(\phi(x) + \{\mu(x) - \sigma^2 \partial_x \log p(x)\}\Big)^2\Bigg) - \int p(x)\Big(\sigma^2 \partial_x \log p(x) - \mu(x)\Big)^2 dx \tag{A71}$$

$$= \mathbb{E}\Bigg[\Big(\phi(x) - \{-\mu(x) + \sigma^2 \partial_x \log p(x)\}\Big)^2\Bigg] - \mathbb{E}\Big[(-\mu(x) + \sigma^2 \partial_x \log p(x))^2\Big] \tag{A72}$$

The criterion above achieves its optimum with respect to $\phi(x)$ when the first term evaluates to zero, namely $\arg\min_\phi \mathcal{L}[\phi, \mu] = -\mu(x) + \sigma^2 \nabla_x \log p(x)$. Furthermore, we can see that the original criterion, which does not require the explicit score, trains the reverse drift $\phi(x)$ implicitly on the correct score up to an additive term.

The integration by parts can be understood as an operation opposite to the product rule of differentiation. Under the assumption of the vanishing probability of data at infinity, namely $\lim_{x \to \pm\infty} p(x) = 0$, we obtain

$$\int_{-\infty}^{\infty} p(x)\partial_x \phi(x)dx = \overbrace{[p(x)\phi(x)]_{-\infty}^{\infty}}^{=0} - \int \partial_x p(x)\phi(x)dx \tag{A73}$$

$$= -\int \partial_x p(x)\phi(x)dx \tag{A74}$$

Appendix C. Trace Estimation

Appendix C.1. Hutchinson's Stochastic Trace Estimation

For a square matrix $A \in \mathbb{R}^{d \times d}$, the trace is defined as

$$\mathrm{Tr}[A] = \sum_i^d A_{ii} \tag{A75}$$

which sums over the diagonal terms of the matrix A.

We can approximate the exact trace with a sampled approximation. We, therefore, haverandom samples $Z \in \mathbb{R}^D$ for which the mean is a zero vector and the covariance matrix is a identity matrix, i.e., $\Sigma[Z] = I$.

The Rademacher distribution, which samples from the set $\{-1, +1\}$ with equal probability, offers the lowest estimator variance and is commonly used in the trace estimation trick for this reason.

$$\begin{aligned} \mathrm{Tr}[A] &= \mathrm{Tr}[IA] && \text{(A76)} \\ &= \mathrm{Tr}\left[\mathbb{E}_{z\sim p(z)}\left[zz^T\right]A\right] && \text{(A77)} \\ &= \mathbb{E}_{z\sim p(z)}\left[z^TAz\right] && \text{(A78)} \end{aligned}$$

where the trace operator disappears as $z^TAz \in \mathbb{R}$ is a scalar value for which the trace is a superfluous operation.

For estimating the trace of the Jacobian, we can circumvent the quadratic nature of the Jacobian by reducing the network output with a random vector z to a scalar, which can then be readily derived with a single backward pass.

$$\begin{aligned} \mathrm{Tr}\left[J_f(x)\right] &= \mathbb{E}_{z\sim p(z)}\left[z^TJ_f(x)z\right] && \text{(A79)} \\ &= \mathbb{E}_{z\sim p(z)}\left[z^T\nabla_x[f(x)^T]z\right] && \text{(A80)} \\ &= \mathbb{E}_{z\sim p(z)}\left[z^T\nabla_x[f(x)^Tz]\right] && \text{(A81)} \end{aligned}$$

Appendix C.2. Stein's Lemma

Let $X \in \mathbb{R}^N$ be a normally distributed random variable $p(x) = \mathcal{N}(x; \mu, \sigma^2)$ with mean μ and variance σ^2. Let the derivative of the normal distribution with respect to x be

$$\begin{aligned} \partial_x p(x) &= \partial_x\left[\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}\right] && \text{(A82)} \\ &= -\frac{(x-\mu)}{\sigma^2}\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}} && \text{(A83)} \\ &= -\frac{(x-\mu)}{\sigma^2}p(x). && \text{(A84)} \end{aligned}$$

Integration by Parts (IbP) yields the often-used identity:

$$\int_{x=-\infty}^{\infty} u(x)\partial_x v(x)dx = [u(x)v(x)]_{x=-\infty}^{\infty} - \int_{x=-\infty}^{\infty} \partial_x u(x)v(x)dx. \quad \text{(A85)}$$

In practice, the property that either $u(x)$ or $v(x)$ or both evaluate to zero at $x = \pm\infty$ as is the case with common probability distributions is leveraged as an algebraic trick to "switch the derivative to the other function".

Given a function $g(x)$, we can obtain a gradient estimator with the following steps via integration by parts:

$$\mathbb{E}_{p(x))}[g(x)(x-\mu)] = \int g(x)(x-\mu)\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}dx \tag{A86}$$

$$= -\sigma^2 \int g(x) \underbrace{\frac{(x-\mu)}{-\sigma^2}\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}}_{\partial_x p(x)} dx \tag{A87}$$

$$= -\sigma^2 \underbrace{\int g(x)\partial_x p(x)dx}_{\text{IbP}} \tag{A88}$$

$$= -\sigma^2\{\underbrace{[g(x)p(x)]_{x=-\infty}^{\infty}}_{p(\pm\infty)=0} - \int \partial_x g(x)p(x)dx\} \tag{A89}$$

$$= \sigma^2 \mathbb{E}_{p(x)}[\partial_x g(x)] \tag{A90}$$

Appendix C.3. Trace Estimation with Stein's Lemma

By choosing a perturbation $\epsilon \sim p(0, \sigma_\epsilon^2)$ with zero mean and a small variance σ_ϵ^2, we can define a perturbed data point $x' \sim p(x, \sigma_\epsilon^2)$ via $x' = x + \epsilon$. This transforms Stein's lemma into

$$\mathbb{E}_{p(\nu))}\left[g(x')(x'-x)\right] = \mathbb{E}_{p(\epsilon))}[g(x+\epsilon)\epsilon] = \sigma_\epsilon^2 \mathbb{E}_{p(\epsilon)}\left[\partial_{x'} g(x')\right]. \tag{A91}$$

In practice, we rescaled with $1/\sigma_\epsilon^2$ and evaluated the left side of the following identity:

$$\mathbb{E}_{p(\epsilon)}\left[g(x+\epsilon)\frac{\epsilon}{\sigma_\epsilon^2}\right] = \mathbb{E}_{p(\epsilon)}[\partial_{x+\epsilon} g(x+\epsilon)]. \tag{A92}$$

which gives us an estimator of the gradient $\partial_x g(x)$ by averaging the gradients in the ϵ-neighbourhood of x. For a function $g : \mathbb{R}^M \to \mathbb{R}^N$, the gradient estimation with Stein's lemma estimates the trace of the Jacobian $J_g(x+\epsilon)$:

$$\mathbb{E}_{p(\epsilon)}\left[g(x+\epsilon)\frac{\epsilon}{\sigma_\epsilon^2}\right] = \mathbb{E}_{p(\epsilon)}\left[\mathrm{Tr}\left[J_g(x+\epsilon)\right]\right]. \tag{A93}$$

In the limit of $\sigma_\epsilon \to 0$, we obtain the trace estimator:

$$\mathrm{Tr}\left[J_g(x)\right] = \lim_{\sigma_\epsilon \downarrow 0} \mathbb{E}_{p(\epsilon)}\left[\mathrm{Tr}\left[J_g(x+\epsilon)\right]\right] = \lim_{\sigma_\epsilon \downarrow 0} \mathbb{E}_{p(\epsilon)}\left[g(x+\epsilon)\frac{\epsilon}{\sigma_\epsilon^2}\right] \tag{A94}$$

in which we computed the right-most term to obtain the left-most term.

The scaling of the perturbation scale σ_ϵ offers at least in theory intriguing similarities to the forward diffusive process of diffusion models. These models estimate the scores of the data distribution $x_t' \sim p(x, \sigma_t^2)$ in which x is a sample from the true data distribution, which is modelled, and the perturbation scale σ_t is time dependent, which decreases as the generative process is integrated in time. Thus, to stabilise the score estimation in higher dimensions, we aimed to to make the perturbation scale in the Stein trace estimator time dependent.

Appendix D. Sampling with Predictor–Corrector Scheme

Recent papers [1,34] have suggested a method to correct original samples x_t in order to create better representatives of the marginal densities $p_t(x)$ (required by our estimators),

which might deviate otherwise too far from the high-probability regions of p_t. The method is based on Langevin sampling [41–43], which performs the iteration:

$$x_t^{(m+1)} = x_t^{(m)} + \epsilon \nabla_x \log p_t(x_t^{(m)}) + \sqrt{2\epsilon} z_m \tag{A95}$$

while integrating the forward and backward SDE. The superscript (m) denotes the m'th step of the Langevin sampler at which a standard normal random variable $z_m \sim \mathcal{N}(0,1)$ is drawn. For small ϵ and as $m \to \infty$, the samples are distributed as $x_t^{(m)} \sim p_t(\cdot)$. Note that, during the Langevin sampling, the time t remains fixed.

Unfortunately, in our approach, as well as in previous work, the direct estimation of the score $\nabla_x \log p_t(x)$ is avoided due to its intractability. However, by using (12) and (13), we can still recompute the score by adding forward and backward drifts:

$$\nabla_x \log \overrightarrow{p}_t^{(k-1)}(x_t) = \frac{1}{\sigma^2}\left(\overleftarrow{\mu}^{(k)}(x_t,t) + \overrightarrow{\mu}^{(k-1)}(x_t,t)\right) \tag{A96}$$

$$\nabla_x \log \overleftarrow{p}_t^{(k-1)}(x_t) = \frac{1}{\sigma^2}\left(\overleftarrow{\mu}^{(k)}(x_t,t) + \overrightarrow{\mu}^{(k-1)}(x_t,t)\right) \tag{A97}$$

which can be easily performed by simply evaluating both the forward and backward drift.

We evaluated the predictor–corrector sampling scheme on the manifold dataset. As the data lie on a multi-dimensional manifold, we chose this dataset to examine the validity of the predictor–corrector sampling scheme. Therefore, we integrated the forward process with the Euler–Maryuama integration scheme for 200 time steps with an integration step size of $dt = 0.001$. Subsequently, after each integration time step, we ran a varying number of Langevin sampling steps M with stepsize $\epsilon = 0.0001$ to obtain more representative samples from the conditional distribution $p(x_t^{(M)}|t)$. Yet, we observed, similarly to other publications on standard diffusion models, that a single corrector step yielded slightly better results, while more than a single step yielded significantly worse Wasserstein-1 distances between the data distributions and the terminal distributions of the processes. This behaviour can be seen in Figure A1 in more detail.

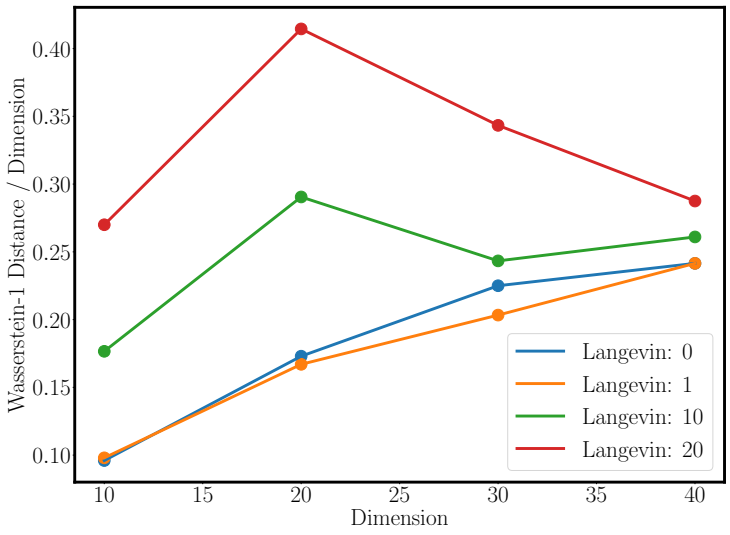

Figure A1. The performance of the predictor–corrector sampling scheme dependent on the number of Langevin steps per integration step. After each Euler–Maryuama step, a fixed number of Langevin sampling steps are performed. Similar to other publications, we found that a single Langevin corrector step improves the performance slightly, while more steps of Langevin sampling lead to significantly worse performance. The Wasserstein-1 distance was evaluated on the true marginal and predicted marginal distributions.

References

1. Song, Y.; Sohl-Dickstein, J.; Kingma, D.P.; Kumar, A.; Ermon, S.; Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv* **2020**, arXiv:2011.13456.
2. Saharia, C.; Chan, W.; Saxena, S.; Li, L.; Whang, J.; Denton, E.; Ghasemipour, S.K.S.; Ayan, B.K.; Mahdavi, S.S.; Lopes, R.G.; et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *arXiv* **2022**, arXiv:2205.11487.
3. Dhariwal, P.; Nichol, A. Diffusion models beat gans on image synthesis. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 8780–8794.
4. Ho, J.; Jain, A.; Abbeel, P. Denoising diffusion probabilistic models. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 6840–6851.
5. Schrödinger, E. *Über die Umkehrung der Naturgesetze*; Verlag der Akademie der Wissenschaften in Kommission bei Walter De Gruyter: Berlin, Germany, 1931.
6. Schrödinger, E. Sur la théorie relativiste de l'électron et l'interprétation de la mécanique quantique. *Ann. L'Inst. Henri Poincaré* **1932**, *2*, 269–310.
7. Léonard, C. A survey of the Schrödinger problem and some of its connections with optimal transport. *arXiv* **2013**, arXiv:1308.0215.
8. Chen, Y.; Georgiou, T.T.; Pavon, M. On the relation between optimal transport and Schrödinger bridges: A stochastic control viewpoint. *J. Optim. Theory Appl.* **2016**, *169*, 671–691.
9. Reich, S. Data assimilation: The Schrödinger perspective. *Acta Numer.* **2019**, *28*, 635–711.
10. Chen, Y.; Georgiou, T.T.; Pavon, M. Optimal transport in systems and control. *Annu. Rev. Control. Robot. Auton. Syst.* **2021**, *4*, 89–113.
11. Bernton, E.; Heng, J.; Doucet, A.; Jacob, P.E. Schrödinger Bridge Samplers. *arXiv* **2019**, arXiv:1912.13170.
12. De Bortoli, V.; Thornton, J.; Heng, J.; Doucet, A. Diffusion Schrödinger bridge with applications to score-based generative modeling. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 17695–17709.
13. Vargas, F.; Thodoroff, P.; Lamacraft, A.; Lawrence, N. Solving schrödinger bridges via maximum likelihood. *Entropy* **2021**, *23*, 1134.
14. Hyvärinen, A.; Dayan, P. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.* **2005**, *6*, 695–709.
15. Oksendal, B. *Stochastic Differential Equations: An Introduction with Applications*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
16. Sinkhorn, R. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Stat.* **1964**, *35*, 876–879.
17. Peyré, G.; Cuturi, M. Computational optimal transport: With applications to data science. *Found. Trends Mach. Learn.* **2019**, *11*, 355–607.
18. Ruschendorf, L. Convergence of the iterative proportional fitting procedure. *Ann. Stat.* **1995**, *23*, 1160–1174.
19. Nelson, E. Derivation of the Schrödinger equation from Newtonian mechanics. *Phys. Rev.* **1966**, *150*, 1079.
20. Anderson, B.D. Reverse-time diffusion equation models. *Stoch. Process. Appl.* **1982**, *12*, 313–326.
21. Nelson, E. Stochastic mechanics and random fields. In *École d'Été de Probabilités de Saint-Flour XV–XVII, 1985–1987*; Springer: Berlin/Heidelberg, Germany, 1988; pp. 427–459.
22. Maoutsa, D.; Reich, S.; Opper, M. Interacting particle solutions of Fokker–Planck equations through gradient–log–density estimation. *Entropy* **2020**, *22*, 802.
23. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
24. Agarap, A.F. Deep learning using rectified linear units (relu). *arXiv* **2018**, arXiv:1803.08375.
25. Vincent, P. A connection between score matching and denoising autoencoders. *Neural Comput.* **2011**, *23*, 1661–1674.
26. Boffi, N.M.; Vanden-Eijnden, E. Probability flow solution of the Fokker–Planck equation. *arXiv* **2022**, arXiv:2206.04642.
27. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
28. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
29. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
30. Uhlenbeck, G.E.; Ornstein, L.S. On the theory of the Brownian motion. *Phys. Rev.* **1930**, *36*, 823.
31. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
32. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile 7–13 December 2015; pp. 1026–1034.
33. Villani, C. *Optimal Transport: Old and New*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 338.
34. Song, Y.; Ermon, S. Generative modeling by estimating gradients of the data distribution. *Adv. Neural Inf. Process. Syst.* **2019**, 11918–11930.
35. Buitinck, L.; Louppe, G.; Blondel, M.; Pedregosa, F.; Mueller, A.; Grisel, O.; Niculae, V.; Prettenhofer, P.; Gramfort, A.; Grobler, J.; et al. API design for machine learning software: Experiences from the scikit-learn project. *arXiv* **2013**, arXiv:1309.0238.
36. Moon, K.R.; van Dijk, D.; Wang, Z.; Gigante, S.; Burkhardt, D.B.; Chen, W.S.; Yim, K.; Elzen, A.V.D.; Hirn, M.J.; Coifman, R.R.; et al. Visualizing structure and transitions in high-dimensional biological data. *Nat. Biotechnol.* **2019**, *37*, 1482–1492.
37. Tong, A.; Huang, J.; Wolf, G.; Van Dijk, D.; Krishnaswamy, S. Trajectorynet: A dynamic optimal transport network for modeling cellular dynamics. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020; pp. 9526–9536.

38. Kloeden, P.E.; Platen, E.; Schurz, H. *Numerical Solution of SDE through Computer Experiments*; Springer Science & Business Media: Abingdon, UK, 2002.
39. Beskos, A.; Papaspiliopoulos, O.; Roberts, G.O.; Fearnhead, P. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *J. R. Stat. Soc. Ser. B (Stat. Methodol.)* **2006**, *68*, 333–382.
40. Kolmogorov, A.N. On analytic methods in probability theory. *Uspekhi Mat. Nauk* **1938**, *5*, 5–41.
41. Parisi, G. Correlation functions and computer simulations. *Nucl. Phys. B* **1981**, *180*, 378–384.
42. Grenander, U.; Miller, M.I. Representations of knowledge in complex systems. *J. R. Stat. Soc. Ser. B (Methodol.)* **1994**, *56*, 549–581.
43. Welling, M.; Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Washington, DC, USA, 28 June–2 July 2011; pp. 681–688.