LEARNING STOCHASTIC BEHAVIOUR FROM AGGREGATE DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning nonlinear dynamics from aggregate data is a challenging problem since the full trajectory of each individual is not available, namely, the individual observed at one time point may not be observed at next time point, or the identity of individual is unavailable due to technical limitations, experimental cost and/or privacy issues. This is in sharp contrast to learning dynamics with trajectory data, on which the majority of existing methods are based. In this paper, we present a complementary strategy designed for aggregate data. Our method uses weak form of Fokker Planck Equation(FPE) to describe density evolution of data in a sampling form, which is combined with Wasserstein generative adversarial network(WGAN) in training process. In such a sample-based framework we are able to study nonlinear dynamics from data without solving the partial differential equation(PDE). The model can also handle high dimensional cases with the help of neural networks. We demonstrate our approach in the context of a series of synthetic and real-world data sets.

1 INTRODUCTION

Aggregate data refers to the data sets that full trajectory of each individual is not available, meaning that there is no known individual level correspondence. Typical examples include data sets collected for DNA evolution, social gathering, density transport in control problems, bird migration during which it is impossible to identify individual bird, and many more. In those applications, some observed individuals at one time point may be unobserved at next time spot, or when the individual identities are block or unavailable due to various technical and ethical reasons. Rather than inferring the exact information for each individual, the main objective of learning dynamics in aggregate data is to recover and predict the evolution of distribution of all individuals together. **Trajectory data**, in contrast, is a kind of data that we are able to acquire the information of each individual all the time, although some studies considered the case that some individual trajectories are partially missing. However, the identities of those individuals, whenever they are observable, is always assumed available. For example, stock price, weather, customer behaviors and most training data sets for computer vision and natural language processing.

There are many existing models to learn dynamics of full-trajectory data. Popular ones include Hidden Markov Model (HMM)(Alshamaa et al., 2019; Eddy, 1996), Kalman Filter (KF)(Farahi & Yazdi, 2020; Harvey, 1990; Kalman, 1960) and Particle Filter (PF) (Santos et al., 2019; Djuric et al., 2003). These models and their variants (Deriche et al., 2020; Fang et al., 2019; Hefny et al., 2015; Langford et al., 2009) require full trajectories of each individual, which may not be directly applicable to the aggregate data as we mentioned earlier. On the other side, only a few methods are focused on aggregated data in the recent learning literature. In the work of Hashimoto et al. (2016), a stochastic differential equation(SDE) was adopted to capture the dynamics of particles directly on observations, in particular the drift coefficient is parameterized by a recurrent neural network. Furthermore, Wang et al. (2018) improved traditional HMM model by using an SDE to describe the evolving process of hidden states. To be the best of our knowledge, there is no method directly learning the evolution of the density of objects from aggregate data yet.

We propose to learn the dynamics of density through the weak form of Fokker Planck Equation (FPE), which is a parabolic partial differential equation (PDE) governing many dynamical systems subject to random noise perturbations, including the typical SDE models in existing studies. Our learning is accomplished by minimizing the Wasserstein distance between predicted distribution given by FPE and the empirical distribution from data samples. Meanwhile we utilize neural networks to handle

higher dimensional cases. More importantly, by leveraging the framework of Wasserstein Generative Adversarial Network (WGAN) (Arjovsky et al., 2017), our model is capable of approximating the distribution of samples at different time points without solving the SDE or FPE. More specifically, we treat the transport coefficient, the goal of learning, in the FPE as a generator, and the test functions in the weak form of FPE as adversarial. In other words, our method can also be regarded as a data-driven method to estimate transport coefficient in FPE, which corresponds to the drift coefficients in SDEs. We would like to mention that several methods of solving SDE and FPE (Weinan et al., 2017; Beck et al., 2018; Li et al., 2019) adopt opposite ways to our method, they utilize neural networks to estimate the distribution $P(x, t)$ with given drift and diffusion terms. Additionally, though we treat diffusion term as a constant in our model, it is straightforward to generalize it to be a neural network as well, which can be an extension of this work.

In conclusion, our contributions are:

- We design a formulation that uses the weak form of FPE to recover nonlinear dynamics from aggregate observations.
- We combine our weak form of FPE with WGAN so that we do not need to solve the FPE.
- Finally, we empirically demonstrate the effectiveness of our framework via some synthetic and real-world data sets.

2 Proposed Method

2.1 Fokker Planck Equation for the density evolution

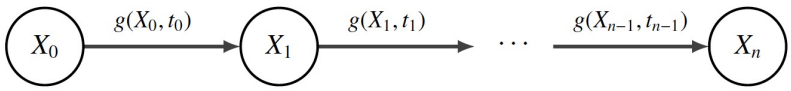


Figure 1: State model of the stochastic process X_t

We assume the individuals evolve in a pattern as shown in Figure 1. An example satisfies such process is the following stochastic differential equation(SDE) also known as Itô process:

$$d\boldsymbol{X} = g(\boldsymbol{X}, t)dt + \sigma d\boldsymbol{W} \tag{1}$$

where the distribution of $\boldsymbol{X}$ represents the observations along time, $d\boldsymbol{X}$ represents a tiny change of $\boldsymbol{X}$ along with time interval dt, $g(\boldsymbol{X}, t)$ is drift coefficient that drives the dynamics of the SDE, σ is the diffusion constant, $d\boldsymbol{W}$ is a stochastic process and $\mathcal{N}(0, 1)$ is a standard Gaussian noise.

Given the above SDE model, the density of the solution $\boldsymbol{X}$ is governed by the Fokker Planck Equation(FPE).

Lemma 1. *Suppose $\phi(\boldsymbol{x})$: $\mathbb{R}^d \mapsto \mathbb{R}^n$ has a continuous partial time derivative and continuous second partial space derivatives, then the density of ϕ that satisfies Equation 1 leads to the Fokker Planck Equation:*

$$\frac{\partial p}{\partial t} = \sum_{i=1}^{D} -\frac{\partial}{\partial x_i}\left[g(\boldsymbol{x})p(\boldsymbol{x}, t)\right] + \frac{1}{2}\sigma^2 \sum_{i,j=1}^{D} \frac{\partial^2}{\partial x_i \partial x_j} p(\boldsymbol{x}, t) \tag{2}$$

As a linear evolution PDE, FPE describes the evolution of density function of the stochastic process driven by a SDE. Due to this reason, FPE plays a crucial role in stochastic calculus, statistical physics and modeling (Nelson, 1985; Qi & Majda, 2016; Risken, 1989). Its importance is also getting more attention among statistic and machine learning community(Liu & Wang, 2016; Pavon et al., 2018; Rezende & Mohamed, 2015). In this paper, we utilize the weak form of FPE as a basis to study hidden dynamics of the whole group data without solving FPE.

Our task can be described as: assume that the individuals evolve with the process indicated by Figure 1, which can be simulated by Equation 1. Then given observations $\boldsymbol{x}_t$ along time axis, we aim to recover the transport coefficient $g(\boldsymbol{x}, t)$ in FPE. For simplicity we treat $g(\boldsymbol{x}, t)$ as a function uncorrelated to time t, namely, $g(\boldsymbol{x}, t) = g(\boldsymbol{x})$. Notice that though evolving process of individuals can be simulated by Equation 1, in reality since we lose identity information of individuals, the observed data become aggregate data, thus we need a new way other than traditional methods to study individuals' distribution.

2.2 Weak Form of Fokker Planck Equation

Given FPE stated in Lemma 1, if we multiply a test function $f \in H_0^1(\Omega; \mathbb{R})$ on both sides of the FPE, then the integration on both sides leads to the weak form of FPE:

$$\int \frac{\partial p}{\partial t} f(\boldsymbol{x}) d\boldsymbol{x} = \int \sum_{i=1}^{D} -\frac{\partial}{\partial x_i} \left[g(\boldsymbol{x}) p(\boldsymbol{x}, t) \right] f(\boldsymbol{x}) d\boldsymbol{x} + \frac{1}{2}\sigma^2 \int \sum_{i,j=1}^{D} \frac{\partial^2}{\partial x_i \partial x_j} p(\boldsymbol{x}, t) f(\boldsymbol{x}) d\boldsymbol{x}$$

where $H_0^1(\Omega; \mathbb{R})$ denote the Sobolev space. The first advantage of weak solution is that the strict solution of a PDE requires sufficient regularity of the problem thus may not exist in the classical sense, however the weak solution may exist and provide more advantages of solving PDEs(Zang et al., 2019). Specially, if $f(\boldsymbol{x})$ is a density function, the integral form then helps us investigate density in a samples level view, thus machine learning methods could be applied.

Suppose the observed samples at time points t_{m-1} and t_m are following true densities $\hat{p}(\boldsymbol{x}, t_{m-1})$ and $\hat{p}(\boldsymbol{x}, t_m)$, we utilize following method to approximate density $\tilde{p}(\boldsymbol{x}, t_m)$:

$$\tilde{p}(\boldsymbol{x}, t_m) = \hat{p}(\boldsymbol{x}, t_{m-1}) + \int_{t_{m-1}}^{t_m} \frac{\partial p(\boldsymbol{x}, \tau)}{\partial \tau} d\tau \tag{3}$$

where $\boldsymbol{x}$ is the solution satisfies Equation 1 and $p(\boldsymbol{x}, \tau)$ is the true density of $\boldsymbol{x}$ at time τ.

Then it is natural to compare the difference between two distributions $\tilde{p}(\boldsymbol{x}, t_m)$ and $\hat{p}(\boldsymbol{x}, t_m)$. Applying Kantorovich-Rubinstein duality form(Villani, 2008), the Wasserstein distance between $\tilde{p}$ and $\hat{p}$ is:

$$W(\hat{p}, \tilde{p}) = \max_{\|\nabla f\| \leq 1} \left(\mathbb{E}_{\boldsymbol{x}_r \sim \hat{p}(\boldsymbol{x}, t_m)}[f(\boldsymbol{x}_r)] - \mathbb{E}_{\boldsymbol{x}_g \sim \tilde{p}(\boldsymbol{x}, t_m)}[f(\boldsymbol{x}_g)] \right) \tag{4}$$

for some f whose gradient norm should be less or equal to 1. Notice that $\boldsymbol{x}_g$ is generated by Equation 1 from the initial data $\boldsymbol{x}_0$. If we describe this process with $\mathcal{G}(\boldsymbol{x}_0)$, representing the Euler increment form of Equation 1 starting with $\boldsymbol{x}_0$:

$$\mathcal{G}(\boldsymbol{x}_0) : \boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + g(\boldsymbol{x}_t)\Delta t + \sigma \sqrt{\Delta t} \mathcal{N}(0, 1) \tag{5}$$

then Equation 4 can be rewritten as a framework of Wasserstein generative adversarial network (WGAN)(Arjovsky et al., 2017; Goodfellow et al., 2014):

$$W(\hat{p}, \tilde{p}) = \max_{\|\nabla f\| \leq 1} \left(\mathbb{E}_{\boldsymbol{x}_r \sim \hat{p}(\boldsymbol{x}, t_m)}[f(\boldsymbol{x}_r)] - \mathbb{E}_{\boldsymbol{x}_0 \sim \tilde{p}(\boldsymbol{x}, t_0)}[f(\mathcal{G}(\boldsymbol{x}_0))] \right) \tag{6}$$

where we treat f as a discriminator and $\mathcal{G}$ as a generating process. Inspired by Equation 3, the way we deal with $\mathbb{E}_{\boldsymbol{x}_g \sim \tilde{p}(\boldsymbol{x}, t_m)}[f(\boldsymbol{x}_g)]$ is:

$$\mathbb{E}_{\boldsymbol{x}_g \sim \tilde{p}(\boldsymbol{x}, t_m)}[f(\boldsymbol{x}_g)] = \int f(\boldsymbol{x}) \hat{p}(\boldsymbol{x}, t_{m-1}) d\boldsymbol{x} + \int \int_{t_{m-1}}^{t_m} f(\boldsymbol{x}) \frac{\partial p(\boldsymbol{x}, \tau)}{\partial \tau} d\tau d\boldsymbol{x}$$

In above equation, the second term on the right is actually a weak form of FPE and $f(\boldsymbol{x})$ is a test function. We plug in Equation 3 and use integration by parts to solve this term, refer the proof of Theorem 1 to see more details.

Theorem 1. *Suppose f satisfies $\|\nabla f\| \leq 1$ and generated data evolves only one step from $\boldsymbol{x}_{t_{m-1}}$ by Equation 5, we define an operator:*

$$\mathcal{F}(\boldsymbol{x}_t) = \frac{1}{N} \sum_{k=1}^{N} \Big(\sum_{i=1}^{D} g(x_t^{(k)}) \frac{\partial}{\partial x_i} f(x_t^{(k)}) + \frac{1}{2}\sigma^2 \sum_{i,j=1}^{D} \frac{\partial^2}{\partial x_i \partial x_j} f(x_t^{(k)}) \Big)$$

then the Wasserstein distance between real data and generated data at time point t_m is:

$$\max_{\|\nabla f\|\leq 1}\Big(\frac{1}{N}\sum_{k=1}^{N} f(x_{t_m}^{(k)}) - \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m-1}}^{(k)}) - \frac{\Delta t}{2}\left[\mathcal{F}(\boldsymbol{x}_{t_{m-1}}) + \mathcal{F}(\boldsymbol{x}_{t_m})\right]\Big) \tag{7}$$

By Theorem 1 we are able to compute the Wasserstein distance between true and generated distributions at a single time point. Different with normal form of Wasserstein distance shown in Equation 4, we have two extra terms at the end.

2.3 Wasserstein Distance on Time Series

The next question is that in real cases, it is not realistic to observe the data on two consecutive time points, especially when Δt is small. To make our model more flexible, we extend our formulation to followings so that we are able to plug in data that is observed at arbitrary time points.

Theorem 2. *Suppose f satisfies $\|\nabla f\| \leq 1$ and generated data evolves multiple steps starting with arbitrary data $\boldsymbol{x}_{t_{m_0}}$ by Equation 5. We adopt the same operator $\mathcal{F}$ defined in Theorem 1, then the Wasserstein distance between real data and generated data at time point t_{m_n} is:*

$$\max_{\|\nabla f\|\leq 1}\Big\{\frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_n}}^{(k)}) - \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_0}}^{(k)}) - \frac{\Delta t}{2}(\mathcal{F}(\boldsymbol{x}_{t_{m_0}}) + \mathcal{F}(\boldsymbol{x}_{t_{m_n}}) + 2\sum_{s=1}^{n-1}\mathcal{F}(\boldsymbol{x}_{t_{m_s}}))\Big\}$$

Objective Function: Base on Theorem 2, finally we obtain objective function by minimizing the accumulated Wasserstein distance on time-series observations:

$$\min_{g}\sum_{n=1}^{J}\max_{f_n}\Big\{\frac{1}{N}\sum_{k=1}^{N} f_n(x_{t_{m_n}}^{(k)}) - \frac{1}{N}\sum_{k=1}^{N} f_n(x_{t_{m_0}}^{(k)}) - \frac{\Delta t}{2}(\mathcal{F}(\boldsymbol{x}_{t_{m_0}}) + \mathcal{F}(\boldsymbol{x}_{t_{m_n}}) + 2\sum_{s=1}^{n-1}\mathcal{F}(\boldsymbol{x}_{t_{m_s}}))\Big\} \tag{8}$$

We call our algorithm Fokker Planck Process(FPP) which is shown in Algorithm 1. We also provide an error analysis in Appendix.

Algorithm 1 Fokker Planck Process Algorithm

Require: Initialize f_{θ_n}, g_ω
for # training iterations **do**
  for k steps **do**
    for observed time t_{m_s}, t_{m_n} in $[t_{m_0} : T]$ **do**
      Generate $x_{t_{m_s}}^{(k)}$, $x_{t_{m_n}}^{(k)}$ by g_ω
      Sample real $\{x_{t_{m_n}}^{(k)}\}$ from $p(\boldsymbol{x}_{t_{m_n}})$
    end for
    For each f_n compute:
    $\mathcal{F}_n = \mathcal{F}(\boldsymbol{x}_{t_{m_0}}) + \mathcal{F}(\boldsymbol{x}_{t_{m_n}}) + 2\sum_{s=1}^{n-1}\mathcal{F}(\boldsymbol{x}_{t_{m_s}})$
    Update f_n by $\nabla_\theta(\frac{1}{N}\sum_{k=1}^{N} f_n(x_{t_{m_n}}^{(k)} - \frac{1}{N}\sum_{k=1}^{N} f_n(x_{t_{m_0}}^{(k)}) - \frac{\Delta t}{2}\mathcal{F}_n)$
  end for
  Update g_ω by $\nabla_\omega \sum_{f_n}(-\frac{\Delta t}{2}\mathcal{F}_n)$
end for

3 Experiments

In this section, we evaluate our model on various synthetic and realistic data sets.

Baselines: We compare our model with two recently proposed methods. One model(NN) adopts recurrent neural network(RNN) to learn dynamics directly from observations of aggregate data(Hashimoto et al., 2016). The other one model(LEGEND) learns dynamics in a HMM framework(Wang et al., 2018). The baselines in our experiments are two typical representatives that have state-of-the-art

performance on learning aggregate data. Furthermore, though we simulate the evolving process of the data as a SDE, which is on the same track with NN, as mentioned before, NN trains its RNN via optimizing Sinkhorn distance(Cuturi, 2013), our model starts with a view of weak form of PDE, focuses more on WGAN framework and easier computation.

3.1 Synthetic Data

We first evaluate our model on three synthetic data sets which are generated by three artificial dynamics: **Synthetic-1**, **Synthetic-2** and **Synthetic-3**.

Experiment Setup: In all synthetic data experiments, we set the drift coefficient g and the discriminator f as two simple fully-connected networks. The g network has one hidden layer and the f network has three hidden layers. Each layer has 32 nodes for both g and f. The only one activation function we choose is Tanh. Notice that since we need to calculate $\frac{\partial^2 f}{\partial x^2}$, the activation function of f must be twice differentiable to avoid loss of weight gradient. In terms of training process, we use the Adam optimizer (Kingma & Ba, 2014) with learning rate 10^{-4}. Furthermore, we use spectral normalization to realize $\|\nabla f\| \leq 1$(Miyato et al., 2018). We initialize the weights with Xavier initialization(Glorot & Bengio, 2010) and train our model by Algorithm 1.

Synthetic-1:

$$\boldsymbol{x}_0 \sim \mathcal{N}(0, \boldsymbol{\Sigma}_0), \qquad \boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t - (\boldsymbol{A}^T \boldsymbol{x}_t + \boldsymbol{b})\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0, 1)$$

Synthetic-2:

$$\boldsymbol{x}_0 \sim \mathcal{N}(0, \boldsymbol{\Sigma}_0), \qquad \boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t - \boldsymbol{G}\boldsymbol{x}_t\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0, 1)$$

Here we need to further state more information about $\boldsymbol{G}$:

$$\boldsymbol{G} = \begin{bmatrix} \frac{1}{\sigma_1}\frac{N_1}{N_1+N_2}(x_1-\mu_{11}) + \frac{1}{\sigma_2}\frac{N_2}{N_1+N_2}(x_1-\mu_{21}) & 0 \\ 0 & \frac{1}{\sigma_1}\frac{N_1}{N_1+N_2}(x_2-\mu_{12}) + \frac{1}{\sigma_2}\frac{N_2}{N_1+N_2}(x_1-\mu_{22}) \end{bmatrix}$$

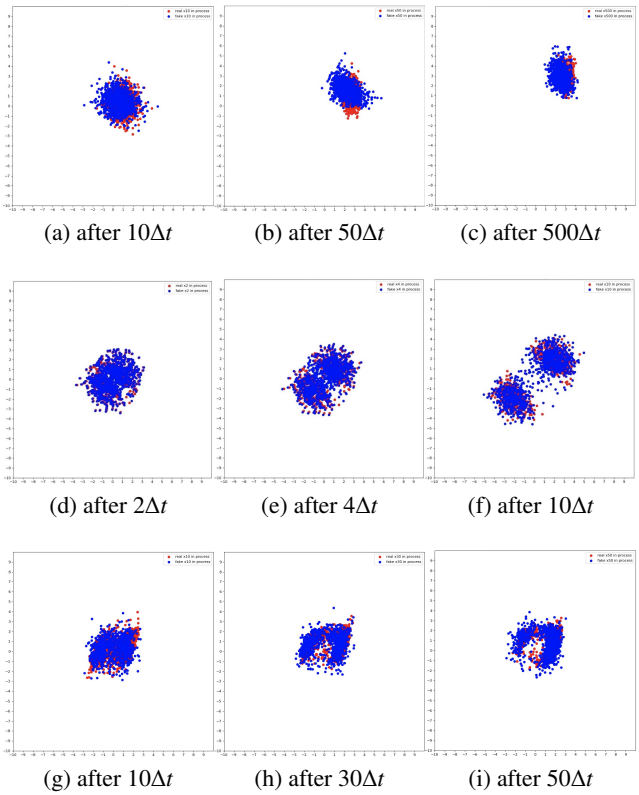
$$N_1 = \frac{1}{\sqrt{2\pi}\sigma_1} exp\Big(-\frac{(x_1-\mu_{11})^2}{2\sigma_1^2} - \frac{(x_1-\mu_{12})^2}{2\sigma_1^2}\Big), \quad N_2 = \frac{1}{\sqrt{2\pi}\sigma_2} exp\Big(-\frac{(x_2-\mu_{21})^2}{2\sigma_2^2} - \frac{(x_2-\mu_{22})^2}{2\sigma_2^2}\Big)$$

Synthetic-3 (Van der Pol oscillator(Li, 2018)):

$$\begin{aligned} &\boldsymbol{x}_0 \sim \mathcal{N}(0, \boldsymbol{\Sigma}_0) \\ &x_1^{t+\Delta t} = x_1^t + 10(x_2^t - \frac{1}{3}(x_1^t)^3 + x_1^t)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0, 1) \\ &x_2^{t+\Delta t} = x_2^t + 3(1 - x_1^t)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0, 1) \end{aligned}$$

The data size at each time point is $N = 2000$, we treat 1200 data points as the training set and the other 800 data points as the test set. In Synthetic-1, this is a two dimensional case following simple linear dynamics, where $\boldsymbol{A} = [(4, 0), (0, 1)]$, $\boldsymbol{b} = [-3, \ -3]^T$. We let $\Delta t = 0.01$, $\sigma = 1$, $\boldsymbol{\Sigma}_0 = [(1, 0), (0, 1)]$. We utilize true $\boldsymbol{x}_0$, $\boldsymbol{x}_{20}$ and $\boldsymbol{x}_{200}$ in training process and predict the distributions of $\boldsymbol{x}_{50}$ and $\boldsymbol{x}_{500}$. In Synthetic-2, this is a two dimensional case following complex nonlinear dynamics. We let $\Delta t = 0.01$, $\sigma = \sigma_1 = \sigma_2 = 1$, $\boldsymbol{\mu}_1 = [15, 15]^T$ and $\boldsymbol{\mu}_2 = [-15, -15]^T$. We utilize true $\boldsymbol{x}_0$, $\boldsymbol{x}_3$ and $\boldsymbol{x}_6$ in training process and predict $\boldsymbol{x}_2$, $\boldsymbol{x}_4$ and $\boldsymbol{x}_{10}$. In Synthetic-3, $\Delta t = 0.01$, $\sigma = 1$, we utilize true $\boldsymbol{x}_3$, $\boldsymbol{x}_7$ and $\boldsymbol{x}_{20}$ in training process then predict the distributions of $\boldsymbol{x}_{10}$, $\boldsymbol{x}_{30}$ and $\boldsymbol{x}_{50}$. We consider population evolution with three different dimensions: d = 2, d = 6 and d = 10. In each high dimensional case (d = 6, d = 10), to be convenience, we set every two dimensions follow the dynamics of low dimensional case(d = 2) in each data set.

Results: We first show the capability of our model for learning hidden dynamics of low-dimensional (d = 2) data. As visualized in Figure 2, the generated data(blue) covers all areas of ground truth(red), which demonstrates that our model is able to precisely learn the dynamics and correctly predict the future distribution of data. We then evaluate three models using Wasserstein distance as error metric for both low-dimensional (d = 2) and high-dimensional (d = 6, 10) data. As reported in Appendix Table 1, our model achieves lower Wasserstein error than the two baseline models in all cases.



(a) after $10\Delta t$ (b) after $50\Delta t$ (c) after $500\Delta t$

(d) after $2\Delta t$ (e) after $4\Delta t$ (f) after $10\Delta t$

(g) after $10\Delta t$ (h) after $30\Delta t$ (i) after $50\Delta t$

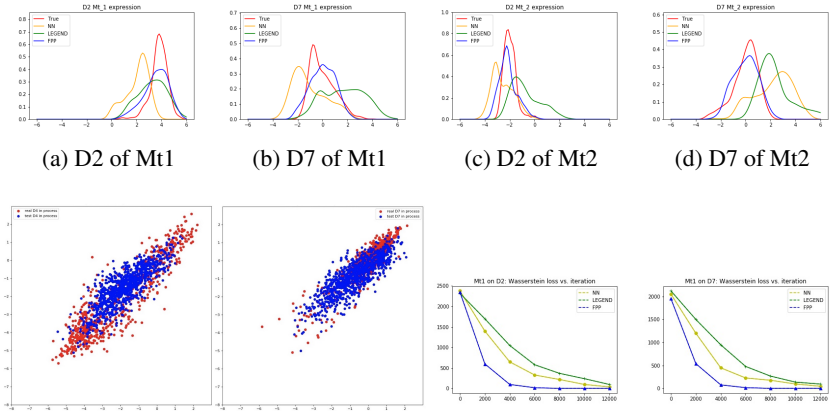
Figure 2: Comparison of generated data(blue) and ground truth(red) of Synthetic-1((a) to (c)), Synthetic-2((d) to (f)) and Synthetic-3((g) to (i)).

3.2 Realistic Data – RNA Sequence of Single Cell

In this section, we evaluate our model on a biology data set called Single-cell RNA-seq which is typically used for learning the diffusion process where embryonic stem cells differentiate into mature cells (Klein et al., 2015). The cell population begins to differentiate from embryonic stem cells after the removal of LIF (leukemia inhibitory factor) at day 0 (D0). Single-cell RNA-seq observations are sampled at day 0 (D0), day 2 (D2), day 4 (D4) and day 7 (D7). At each time point, the expression of 24,175 genes of several hundreds cells are measured (933, 303, 683 and 798 cells on D0, D2, D4 and D7 respectively), notice that there is only whole group's distribution but no trajectory information of each cell on different days. We focus on the data of cell differentiation for the two chosen makers, i.e., Mt1 and Mt2. In first task we treat gene expression level at D0, D4 and D7 as training data to learn the hidden dynamics and predict the distribution of gene expression level at D2. In second task we train the model by gene expression level at D0, D2 and D4, then predict the distribution of gene expression level at D7.

Experiment Setup: We set both f and g as fully connected three-hidden-layers neural networks, each layer has 64 nodes. The only activation function we choose is Tanh. The other setups of neural networks and training process are the same with the ones we use in Synthetic data. Notice that in realistic cases, Δt and $T/\Delta t$ become hyperparameters, here we choose $\Delta t = 0.05$, $T/\Delta t = 35$, which means the data evolves $10\Delta t$ from D0 to D2 , then $10\Delta t$ from D2 to D4 and finally $15\Delta t$ from D4 to D7. For preprocessing, we apply standard normalization procedures (Hicks et al., 2015) to correct batch effects and use non-negative matrix factorization to impute missing expression levels(Hashimoto et al., 2016; Wang et al., 2018).

Results: As shown in Table 1 in Appendix, when compared to other baselines, our model achieves lower Wasserstein error on both Mt1 and Mt2 data, which proves that our model is capable of learning the hidden dynamics of the two studied gene expressions. In Figure 3 (a) to (d), we visualized the



(a) D2 of Mt1 (b) D7 of Mt1 (c) D2 of Mt2 (d) D7 of Mt2

(e) Correlation on D2 (f) Correlation on D7 (g) W-loss of Mt1, D2(h) W-loss of Mt1, D7

Figure 3: (a) and (b): Wasserstein loss as iteration increases of Mt1. (c) to (f): the performance comparision among different models on D2 and D7 of Mt1 and Mt2. (g) and (h): true (red) and predicted (blue) correlations between Mt1(x-axis) and Mt2(y-axis) on D2 (left) and D7 (right).

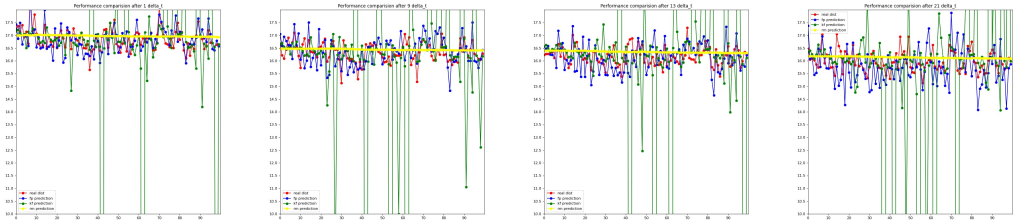
predicted distributions of the two genes. The distributions of Mt1 and Mt2 predicted by our model (curves in blue) are closer to the true distributions (curves in red) on both D2 and D7. Furthermore, our model precisely indicates the correlations between Mt1 and Mt2, as shown in Figure 3 (e) and (f), which also demonstrates the effectiveness of our model since closer to the true correlation represents better performance. In Figure 3 (g) and (h), we see that with simpler structure, the training process of our model is easier with least computation time.

3.3 Realistic Data – Daily Trading Volume

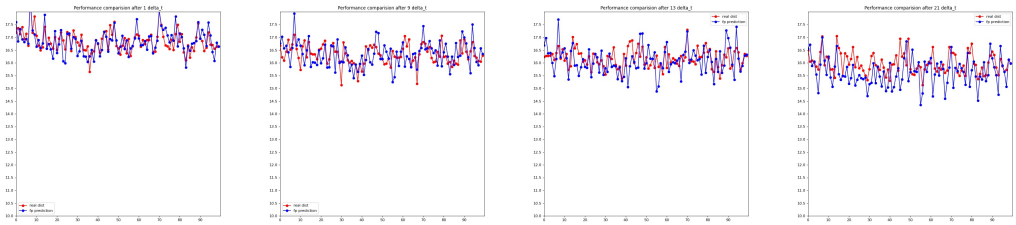
In this section we would like to demonstrate the performance of our model in financial area. Trading volume is the total quantity of shares or contracts traded for specified securities such as stocks, bonds, options contracts, future contracts and all types of commodities. It can be measured on any type of security traded during a trading day or a specified time period. In our case, daily volume of trade is measured on stocks. Predicting traded volume is an essential component in financial research since the traded volume, as a basic component or input of other financial algorithms, tells investors the market's activity and liquidity. The data set we use is the historical traded volume of the stock "JPM". The data covers period from January 2018 to January 2020 and is obtained from Bloomberg. Each day from 14:30 to 20:55, we have 1 observation every 5 minutes, totally 78 observations everyday. Our task is described as follows: given first two years data, we use the traded volume at 14:30, 14:40, 15:05, 15:20 and 16:20 as training data, namely, $\boldsymbol{x}_0, \boldsymbol{x}_2, \boldsymbol{x}_7, \boldsymbol{x}_{10}, \boldsymbol{x}_{22}$ to train our model, then for next 100 days we predict traded volume at 14:35, 15:15, 15:35 and 16:15, namely, $\boldsymbol{x}_1, \boldsymbol{x}_9, \boldsymbol{x}_{13}, \boldsymbol{x}_{21}$. One of baselines we choose is classical rolling means(RM) method, which predicts intraday volume of a particular time interval by the average volume traded in the same interval over the past days. The other one baseline is a kalman filter based model (Chen et al., 2016) that outperforms all available models in predicting intrady trading volume.

Experiment Setup: Following similar setup as we did for RNA data set, we utilize the same structures for neural networks here. For hyperparameters we set $\Delta t = 0.02$, $T/\Delta t = 22$, which means it takes one single Δt from $\boldsymbol{x}_t$ to $\boldsymbol{x}_{t+1}$. For preprocessing, we rescale data by taking natural logarithm of trading volume, which is a common way in trading volume research. We conduct experiments on two groups to show advantages of our method, for first group we train our model on complete data set, in this case the data has full trajectory; for second group we manually delete some trajectories of the data, for instance, we randomly kick out some samples of $\boldsymbol{x}_0, \boldsymbol{x}_2, \boldsymbol{x}_7, \boldsymbol{x}_{10}, \boldsymbol{x}_{22}$ then follow the same procedures of training and prediction.

Results: We present prediction results in Figure 4. As shown in first four figures, with full trajectory, prediction made by RM is almost a straight line, the prediction value bouncing up and down within a



(a) Comparison on 14:35 (b) Comparison on 15:15 (c) Comparison on 15:35 (d) Comparison on 16:15



(e) Comparison on 14:35 (f) Comparison on 15:15 (g) Comparison on 15:35 (h) Comparison on 16:15

Figure 4: (a) to (d): Group A: with full trajectory of training data, predictions of traded volume in next 100 days, RM(yellow) fails to capture the regularities of traded volume in time series, kalman filter based model(green) fails to capture noise information and make reasonable predictions, our model(blue) is able to seize the movements of traded volume and yield better predictions. (e) to (h): Group B: predictions of our model without full trajectory.

very small range, thus this model cannot capture the volume movements, namely, regularities existing in the time series; prediction made by the kalman filter based model captures the regularities better than RM model, but it fails to deal with noise component existing in the time series, thus some predictions are out of a reasonable range. Traded volume predicted by our model is closer to the real case, moreover, our model captures regularities meanwhile gives stable predictions. Furthermore, without full trajectory, kalman filter based model fails to be applied here and RM model still fails to capture the regularities, we randomly drop half of the training samples and display predictions made by our model in last four figures of Figure 4, we see our model still works well.

4 Conclusion

In this paper, we formulate a novel method to recover the hidden dynamics from aggregate data. In particular, our work shows one can simulate the evolving process of aggregate data as an Itô process, in order to investigate aggregate data, we derive a new model that employs the weak form of FPE as well as the framework of WGAN. Furthermore, in Appendix we prove the theoretical guarantees of the error bound of our model. Finally we demonstrate our model through experiments on three synthetic data sets and two real-world data sets.

References

D. Alshamaa, A. Chkeir, F. Mourad-Chehade, and P. Honeine. Hidden markov model for indoor trajectory tracking of elderly people. In *IEEE Sensors Applications Symposium (SAS)*, 2019.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. In *arXiv preprint arXiv:1701.07875*, 2017.

C. Beck, S. Becker, P. Grohs, N. Jaafari, and A. Jentzen. Solving stochastic differential equations and kolmogorov equations by means of deep learning. 2018.

R. Chen, Y. Feng, and D. Palomar. Forecasting intraday trading volume: a kalman filter approach. In *Available at SSRN 3101695*, 2016.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, 2013.

M. Deriche, A. A. Absa, A. Amin, and B. Liu. A novel approach for salt dome detection and tracking using a hybrid hidden markov model with an active contour model. *Journal of Electrical Systems*, 16(3):276–294, 2020.

P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez. Particle filtering. *IEEE Signal Processing Magazine*, 20(5):19–38, 2003.

S. R. Eddy. Hidden markov models. *Current Opinion in Structural Biology*, 6(3):361–365, 1996.

Y. Fang, C. Wang, W. Yao, X. Zhao, H. Zhao, and H. Zha. On-road vehicle tracking using part-based particle filter. *IEEE Transactions on Intelligent Transportation Systems*, 20(12):4538–4552, 2019.

F. Farahi and H. S. Yazdi. Probabilistic kalman filter for moving object tracking. *Signal Processing: Image Communication*, 82, 2020.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2010.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Neural Information Processing Systems*, 2014.

A. C. Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge University Press, 1990.

T. Hashimoto, D. Gifford, and T. Jaakkola. Learning population-level diffusions with generative rnns. In *International Conference on Machine Learning*, pp. 2417–2426, 2016.

A. Hefny, C. Downey, and G. J. Gordon. Supervised learning for dynamical system learning. In *Neural Information Processing Systems*, 2015.

S. C. Hicks, M. Teng, and R. A. Irizarry. On the widespread and critical impact of systematic bias and batch effects in single-cell rna-seq data. *bioRxiv*, 2015.

R. E. Kalman. A new approach to linear filtering and prediction problems. *arXiv preprint arXiv:1805.04099*, 1960.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

A.M. Klein, L. Mazutis, I. Akartuna, N. Tallapragada, A. Veres, V. Li, L. Peshkin, D.A. Weitz, and M.W. Kirschner. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, 2015.

J. Langford, R. Salakhutdinov, and T. Zhang. Learning nonlinear dynamic models. In *International Conference on Machine Learning*, pp. 593–600, 2009.

W.C. Li, S. Liu, H. Zha, and H. Zhou. Parametric fokker-planck equation. In *Geometry science of information*, 2019.

Y. Li. A data-driven method for the steady state of randomly perturbed dynamics. *arXiv preprint arXiv:1805.04099*, 2018.

Q. Liu and D. Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Neural Information Processing Systems*, pp. 2378–2386, 2016.

G. N. Milstein and M. V. Tretyakov (eds.). *Stochastic numerics for mathematical physics*. Springer Science & Business Media, 2013.

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

E. Nelson. *Quantum fluctuations*. Princeton University Press, 1985.

M. Pavon, E. G. Tabak, and G. Trigila. The data-driven schroedinger bridge. *arXiv preprint arXiv:1806.01364*, 2018.

D. Qi and A.J. Majda. Low-dimensional reduced-order models for statistical response and uncertainty quantification: Two-layer baroclinic turbulence. *Journal of the Atmospheric Sciences*, 73(12): 4609–4639, 2016.

D.J. Rezende and S. Mohamed. Variational inference with normalizing flows. In *arXiv preprint arXiv:1505.05770*, 2015.

H. Risken. The fokker-planck equation. *Springer Series in Synergetics*, 18:4609–4639, 1989.

N. P. Santos, V. Lobo, and A. Bernardino. Unmanned aerial vehicle tracking using a particle filter based approach. In *IEEE Underwater Technology (UT)*, 2019.

C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Y. Wang, B. Dai, L. Kong, S. M. Erfani, J. Bailey, and H. Zha. Learning deep hidden nonlinear dynamics from aggregate data. In *Uncertainty in Artificial Intelligence*, 2018.

E. Weinan, J. Han, and A. Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. In *Communications in Mathematics and Statistics*, pp. 349–380, 2017.

Y. Zang, G. Bao, X. Ye, and H. Zhou. Weak adversarial networks for high-dimensional partial differential equations. In *arXiv preprint arXiv:1907.08272*, 2019.

A Appendix

Table 1: The Wasserstein error of different models on Synthetic-1/2/3 and RNA-sequence data sets.

Data	Task	Dimension	NN	LEGEND	Ours
Syn–1	$\boldsymbol{x}_{50}$	2	1.37	0.44	**0.05**
		6	4.79	2.32	**0.06**
		10	9.13	2.89	**0.10**
	$\boldsymbol{x}_{500}$	2	0.84	0.18	**0.03**
		6	3.28	0.30	**0.03**
		10	8.05	1.79	**0.09**
Syn–2	$\boldsymbol{x}_{4}$	2	5.53	1.27	**0.05**
		6	9.97	4.20	**0.08**
		10	16.51	6.08	**0.12**
	$\boldsymbol{x}_{10}$	2	1.68	0.53	**0.05**
		6	4.84	1.08	**0.07**
		10	9.23	2.85	**0.11**
Syn–3	$\boldsymbol{x}_{30}$	2	4.13	1.29	**0.08**
		6	6.40	3.16	**0.17**
		10	11.76	8.53	**0.25**
	$\boldsymbol{x}_{50}$	2	3.05	0.87	**0.12**
		6	6.72	1.52	**0.16**
		10	9.81	3.55	**0.23**
RNA-Mt1	D2	10	33.86	10.28	**4.23**
	D7	10	12.69	7.21	**2.92**
RNA-Mt2	D2	10	31.45	13.32	**4.04**
	D7	10	11.58	7.89	**1.50**

B Error Analysis

In this section, we provide an error analysis of our model. Suppose the hidden dynamics is driven by $g_r(\boldsymbol{x})$, the dynamics that we learn from data is $g_f(\boldsymbol{x})$, then original Itô process, Euler process and Fokker Planck Process are as follows:

$$
\begin{aligned}
d\boldsymbol{X} &= g(\boldsymbol{X})dt + \sigma d\boldsymbol{W} \\
\boldsymbol{x}^r_{t+\Delta t} &= \boldsymbol{x}^r_t + g_r(\boldsymbol{x}^r_t)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \\
\boldsymbol{x}^f_{t+\Delta t} &= \boldsymbol{x}^f_t + g_f(\boldsymbol{x}^f_t)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1)
\end{aligned}
$$

Estimating the error between original Itô process and its Euler form can be very complex, hence we cite the conclusion from (Milstein & Tretyakov, 2013) and focus more on the error between original form and our model.

Lemma B.1. *With the same initial $\boldsymbol{X}_{t_0} = \boldsymbol{x}_{t_0} = \boldsymbol{x}_0$, if there is a global Lipschitz constant K which satisfies:*

$$|g(\boldsymbol{x},t) - g(\boldsymbol{y},t)| \leq K|\boldsymbol{x} - \boldsymbol{y}|$$

then after n steps, the expectation error between Itô process $\boldsymbol{x}_{t_n}$ and Euler forward process $\boldsymbol{x}^r_{t_n}$ is:

$$\mathbb{E}|\boldsymbol{x}_{t_n} - \boldsymbol{x}^r_{t_n}| \leq K\Big(1 + \mathbb{E}|X_0|^2\Big)^{1/2}\Delta t$$

Lemma B.1 illustrates that the expectation error between original Itô process and its Euler form is not related to total steps n but time step Δt.

Theorem B.1. *With the same initial $\boldsymbol{x}_0$, suppose the generalization error of neural network g is ε and existence of global Lipschitz constant K:*

$$|g(\boldsymbol{x}) - g(\boldsymbol{y})| \leq K|\boldsymbol{x} - \boldsymbol{y}|$$

then after n steps with step size $\Delta t = T/n$, the expectation error between Itô process $\boldsymbol{x}_{t_n}$ and approximated forward process $\boldsymbol{x}^f_{t_n}$ is bounded by:

$$\mathbb{E}|\boldsymbol{x}_{t_n} - \boldsymbol{x}^f_{t_n}| \leq \frac{\varepsilon}{K}(e^{KT} - 1) + K(1 + \mathbb{E}|\boldsymbol{x}_0|^2)^{1/2}\Delta t \tag{9}$$

Theorem B.1 implies that besides time step size Δt, our expectation error interacts with three factors, generalization error, Lipschitz constant of $g(x)$ and total time length. In our experiments, we find the best way to decrease the expectation error is reducing the value of K and n.

C Proofs

C.1 Proof of Lemma 1

Lemma C.1. *Suppose $\phi(\boldsymbol{x})$: $\mathbb{R}^d \mapsto \mathbb{R}^n$ have a continuous partial time derivative and continuous second partial space derivatives, then the density of ϕ that satisfies Equation 1 leads to the Fokker Planck Equation:*

$$\frac{\partial p}{\partial t} = \sum_{i=1}^{D} -\frac{\partial}{\partial x_i}\left[g(\boldsymbol{x})p(\boldsymbol{x},t)\right] + \frac{1}{2}\sigma^2 \sum_{i,j=1}^{D} \frac{\partial^2}{\partial x_i \partial x_j} p(\boldsymbol{x},t)$$

Proof. This result has been widely proved, here we just give a simple sketch of proof. Suppose $\phi(\boldsymbol{x}, t)$ is a solution of Equation 1, by Itô formula we have:

$$d\phi = \left(\frac{\partial \phi}{\partial t} + g\frac{\partial \phi}{\partial x} + \frac{1}{2}\sigma^2 \frac{\partial^2 \phi}{\partial x^2}\right)dt + \sigma \frac{\partial \phi}{\partial x} dW$$

Let $\frac{\partial \phi}{\partial t} = 0$, then we take expectation on both sides:

$$d\mathbb{E}\left[\phi\right] = \mathbb{E}\left[g\frac{\partial \phi}{\partial x} + \frac{1}{2}\sigma^2 \frac{\partial^2 \phi}{\partial x^2}\right]dt$$

$$\int \phi \frac{\partial p}{\partial t} d\boldsymbol{x} = \int g \frac{\partial \phi}{\partial x} p(\boldsymbol{x},t) d\boldsymbol{x} + \int \frac{1}{2}\sigma^2 \frac{\partial^2 \phi}{\partial x^2} p(\boldsymbol{x},t) d\boldsymbol{x}$$

Use integration by parts on right side and consider $p(\infty, t) = 0$, we have:

$$\int \phi \frac{\partial p}{\partial t} d\boldsymbol{x} = -\int \phi \frac{\partial}{\partial x}\left[g(\boldsymbol{x})p(\boldsymbol{x},t)\right] d\boldsymbol{x} + \int \frac{1}{2}\sigma^2 \phi \frac{\partial^2 p(\boldsymbol{x},t)}{\partial x^2} d\boldsymbol{x}$$

Finally we have 1-dimensional form of Fokker Planck Equation:

$$\frac{\partial p}{\partial t} = -\frac{\partial}{\partial x}\left[g(\boldsymbol{x})p(\boldsymbol{x},t)\right] + \frac{1}{2}\sigma^2 \frac{\partial^2 p(\boldsymbol{x},t)}{\partial x^2}$$

We extend 1-dimensional form to multi-dimensional form to get proof done. □

C.2 Proof of Theorem 1

Theorem C.1. *Suppose f satisfies $\|\nabla f\| \leq 1$ and generated data evolves only one step from $\boldsymbol{x}_{t_{m-1}}$ by (2), define an operator:*

$$\mathcal{F}(\boldsymbol{x}_t) = \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_t^{(k)})\frac{\partial}{\partial x_i} f(x_t^{(k)}) + \frac{1}{2}\sigma^2 \sum_{i,j=1}^{D} \frac{\partial^2}{\partial x_i \partial x_j} f(x_t^{(k)})\right) \tag{10}$$

then the weak form of Wasserstein distance between real data and generated data at time point t_m can be approximated by:

$$\max_{\|\nabla f\| \leq 1} \left(\frac{1}{N}\sum_{k=1}^{N} f(x_{t_m}^{(k)}) - \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m-1}}^{(k)}) - \frac{\Delta t}{2}\left[\mathcal{F}(\boldsymbol{x}_{t_{m-1}}) + \mathcal{F}(\boldsymbol{x}_{t_m})\right]\right) \tag{11}$$

Proof. Suppose $x_{t_m}^{(k)}$ and $x_{t_{m-1}}^{(k)}$ are our observed samples at t_m and t_{m-1} respectively, then expectations could be approximated by observed samples and generated data:

$$\mathbb{E}_{\boldsymbol{x}\sim\hat{p}(\boldsymbol{x},t_m)}[f(\boldsymbol{x})] = \int f(\boldsymbol{x})\hat{p}(\boldsymbol{x},t_m)d\boldsymbol{x} \approx \frac{1}{N}\sum_{k=1}^{N} f(x_{t_m}^{(k)}), \tag{12}$$

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{x}\sim\tilde{p}(\boldsymbol{x},t_m)}[f(\boldsymbol{x})] &= \int f(\boldsymbol{x})\tilde{p}(\boldsymbol{x},t_m)d\boldsymbol{x} = \int f(\boldsymbol{x})\left[\hat{p}(\boldsymbol{x},t_{m-1}) + \int_{t_{m-1}}^{t_m}\frac{\partial p(\boldsymbol{x},\tau)}{\partial t}d\tau\right]d\boldsymbol{x} \\
&= \int f(\boldsymbol{x})\hat{p}(\boldsymbol{x},t_{m-1})d\boldsymbol{x} + \int f(\boldsymbol{x})\int_{t_{m-1}}^{t_m}\frac{\partial p(\boldsymbol{x},\tau)}{\partial t}d\tau d\boldsymbol{x} \\
&\approx \frac{1}{N}\sum_{k=1}^{N} f(\boldsymbol{x}_{t_{m-1}}^{(k)}) + \underbrace{\int f(\boldsymbol{x})\int_{t_{m-1}}^{t_m}\left\{-\sum_{i=1}^{D}\frac{\partial}{\partial x_i}\left[g(\boldsymbol{x})p(\boldsymbol{x},\tau)\right] + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}p(\boldsymbol{x},\tau)\right\}d\tau d\boldsymbol{x}}_{I}
\end{aligned} \tag{13}$$

Then for the second term I above, it is difficult to calculate directly, but we can use integration by parts to rewrite I as:

$$\begin{aligned}
I &= \int_{t_{m-1}}^{t_m}\int\left[\sum_{i=1}^{D} -f(\boldsymbol{x})\frac{\partial}{\partial x_i}g(\boldsymbol{x})p(\boldsymbol{x},\tau) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D} f(\boldsymbol{x})\frac{\partial^2}{\partial x_i\partial x_j}p(\boldsymbol{x},\tau)\right]d\boldsymbol{x}d\tau \\
&= \int_{t_{m-1}}^{t_m}\int\left[\sum_{i=1}^{D} g(\boldsymbol{x})p(\boldsymbol{x},\tau)\frac{\partial}{\partial x_i}f(\boldsymbol{x}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D} p(\boldsymbol{x},\tau)\frac{\partial^2}{\partial x_i\partial x_j}f(\boldsymbol{x})\right]d\boldsymbol{x}d\tau \\
&= \int_{t_{m-1}}^{t_m}\left(\mathbb{E}_{\boldsymbol{x}\sim\hat{p}(\boldsymbol{x},t)}\left[\sum_{i=1}^{D} g(\boldsymbol{x})\frac{\partial}{\partial x_i}f(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{x}\sim\hat{p}(\boldsymbol{x},\tau)}\left[\frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(\boldsymbol{x})\right]\right)d\tau \\
&= \int_{t_{m-1}}^{t_m}\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x^{(k)})\frac{\partial}{\partial x_i}f(x^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x^{(k)})\right)d\tau
\end{aligned} \tag{14}$$

To approximate the integral from t_{m-1} to t_m, we adopt an average method between two endpoints, then we could rewrite the expectation in Equation (13) as:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{x}\sim\tilde{p}(\boldsymbol{x},t_m)}[f(\boldsymbol{x})] \approx{}& \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m-1}}^{(k)}) + \frac{\Delta t}{2}\left[\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m-1}}^{(k)})\frac{\partial}{\partial x_i}f(x_{t_{m-1}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m-1}}^{(k)})\right)\right. \\
&\left. + \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_m}^{(k)})\frac{\partial}{\partial x}f(x_{t_m}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_m}^{(k)})\right)\right] \\
={}& \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m-1}}^{(k)}) + \frac{\Delta t}{2}\left[\mathcal{F}(\boldsymbol{x}_{t_{m-1}}) + \mathcal{F}(\boldsymbol{x}_{t_m})\right]
\end{aligned} \tag{15}$$

Put (13), (14) and (15) together to finish the proof. □

C.3 Proof of Theorem 2

Theorem C.2. *Suppose f satisfies $\|\nabla f\| \leq 1$ and generated data evolves multiple steps from any initial data $x_{t_{m_0}}$ by (2), define the operator:*

$$\mathcal{F}(\boldsymbol{x}_t) = \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_t^{(k)})\frac{\partial}{\partial x_i}f(x_t^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_t^{(k)})\right)$$

then for arbitrary n, the weak form of Wasserstein distance between real data and generated data at time point t_{m_n} can be approximated by:

$$\max_{\|\nabla f\|\leq 1}\left\{\frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_n}}^{(k)}) - \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_0}}^{(k)}) - \frac{\Delta t}{2}\left(\mathcal{F}(\boldsymbol{x}_{t_{m_0}}) + \mathcal{F}(\boldsymbol{x}_{t_{m_n}}) + 2\sum_{s=1}^{n-1}\mathcal{F}(\boldsymbol{x}_{t_{m_s}})\right)\right\} \tag{16}$$

Proof. Given initial $\boldsymbol{x}_{t_{m_0}}$ and an updated g, we generate $\boldsymbol{x}_{t_{m_1}}$, $\boldsymbol{x}_{t_{m_2}}$, $\boldsymbol{x}_{t_{m_3}}$ ... $\boldsymbol{x}_{t_{m_n}}$ sequentially by Equation 5. Then the expectations can be rewritten as:

$$\mathbb{E}_{\boldsymbol{x}\sim\hat{p}(\boldsymbol{x},t_{m_n})}[f(\boldsymbol{x})] = \int f(\boldsymbol{x})\hat{p}(\boldsymbol{x},t_{m_n})d\boldsymbol{x} \approx \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_n}}^{(k)}), \tag{17}$$

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\sim\tilde{p}(\boldsymbol{x},t_{m_n})}[f(\boldsymbol{x})] \approx{}& \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_0}}^{(k)}) + \int_{t_{m_0}}^{t_{m_1}} \frac{1}{N}\sum_{k=1}^{N}\left[\sum_{i=1}^{D} g(x^{(k)})\frac{\partial}{\partial x_i}f(x^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x^{(k)})\right]d\tau \\ &+ \int_{t_{m_1}}^{t_{m_2}} \frac{1}{N}\sum_{k=1}^{N}\left[\sum_{i=1}^{D} g(x^{(k)})\frac{\partial}{\partial x_i}f(x^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x^{(k)})\right]d\tau + ... \\ &+ \int_{t_{m_{n-1}}}^{t_{m_n}} \frac{1}{N}\sum_{k=1}^{N}\left[\sum_{i=1}^{n} g(x^{(k)})\frac{\partial}{\partial x_i}f(x^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{n}\frac{\partial^2}{\partial x_i\partial x_j}f(x^{(k)})\right]d\tau\end{aligned} \tag{18}$$

which is:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\sim\tilde{p}(\boldsymbol{x},t_{m_n})}[f(\boldsymbol{x})] \approx{}& \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_0}}^{(k)}) + \frac{\Delta t}{2}\left[\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_0}}^{(k)})\frac{\partial}{\partial x_i}f(x_{t_{m_0}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_0}}^{(k)})\right)\right. \\ &\left.+ \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_1}}^{(k)})\frac{\partial}{\partial x}f(x_{t_{m_1}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_1}}^{(k)})\right)\right] \\ &+ \frac{\Delta t}{2}\left[\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_1}}^{(k)})\frac{\partial}{\partial x_i}f(x_{t_{m_1}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_1}}^{(k)})\right)\right. \\ &\left.+ \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_2}}^{(k)})\frac{\partial}{\partial x}f(x_{t_{m_2}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_2}}^{(k)})\right)\right] + ... \\ &+ \frac{\Delta t}{2}\left[\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_{n-1}}}^{(k)})\frac{\partial}{\partial x_i}f(x_{t_{m_{n-1}}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_{n-1}}}^{(k)})\right)\right. \\ &\left.+ \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_n}}^{(k)})\frac{\partial}{\partial x}f(x_{t_{m_n}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_n}}^{(k)})\right)\right] \\ ={}& \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_0}}^{(k)}) + \frac{\Delta t}{2}\left\{\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_0}}^{(k)})\frac{\partial}{\partial x_i}f(x_{t_{m_0}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_0}}^{(k)})\right)\right. \\ &+ \frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_n}}^{(k)})\frac{\partial}{\partial x}f(x_{t_{m_n}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_n}}^{(k)})\right) \\ &\left.+ 2\sum_{s=1}^{n-1}\left[\frac{1}{N}\sum_{k=1}^{N}\left(\sum_{i=1}^{D} g(x_{t_{m_s}}^{(k)})\frac{\partial}{\partial x_i}f(x_{t_{m_s}}^{(k)}) + \frac{1}{2}\sigma^2\sum_{i,j=1}^{D}\frac{\partial^2}{\partial x_i\partial x_j}f(x_{t_{m_s}}^{(k)})\right)\right]\right\}\end{aligned} \tag{19}$$

Finally it comes to:

$$\mathbb{E}_{\boldsymbol{x}\sim\tilde{p}(\boldsymbol{x},t_{m_n})}[f(\boldsymbol{x})] \approx \frac{1}{N}\sum_{k=1}^{N} f(x_{t_{m_0}}^{(k)}) + \frac{\Delta t}{2}\left(\mathcal{F}(\boldsymbol{x}_{t_{m_0}}) + \mathcal{F}(\boldsymbol{x}_{t_{m_n}}) + 2\sum_{s=1}^{n-1}\mathcal{F}(\boldsymbol{x}_{t_{m_s}})\right) \tag{20}$$

□

C.4 Proof of Error Analysis

Proof. The proof process of Lemma B.1 is quite long and out of the scope of this paper, for more details please see first two chapters in reference book (Milstein & Tretyakov, 2013). While for Theorem B.1, with initial X and first one-step iteration:

$$\begin{cases} \boldsymbol{x}_{t_0}^r = \boldsymbol{x}_{t_0} \\ \boldsymbol{x}_{t_0}^f = \boldsymbol{x}_{t_0} \end{cases} \tag{21}$$

$$\begin{cases} \boldsymbol{x}_{t_1}^r = \boldsymbol{x}_{t_0}^r + g_r(\boldsymbol{x}_{t_0}^r)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \\ \boldsymbol{x}_{t_1}^f = \boldsymbol{x}_{t_0}^f + g_f(\boldsymbol{x}_{t_0}^f)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \end{cases} \tag{22}$$

Then we have:

$$\mathbb{E}|\boldsymbol{x}_{t_0}^r - \boldsymbol{x}_{t_0}^f| = \mathbb{E}|\boldsymbol{x}_{t_0} - \boldsymbol{x}_{t_0}| = 0 \tag{23}$$

$$\begin{aligned} \mathbb{E}|\boldsymbol{x}_{t_1}^r - \boldsymbol{x}_{t_1}^f| &= \mathbb{E}|\boldsymbol{x}_{t_0}^r - \boldsymbol{x}_{t_0}^f + g_r(\boldsymbol{x}_{t_0}^r)\Delta t - g_f(\boldsymbol{x}_{t_0}^f)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) - \sigma\sqrt{\Delta t}\mathcal{N}(0,1)| \\ &\leq \mathbb{E}|\boldsymbol{x}_{t_0}^r - \boldsymbol{x}_{t_0}^f| + \mathbb{E}|g_r(\boldsymbol{x}_{t_0}^r) - g_f(\boldsymbol{x}_{t_0}^f)|\Delta t \\ &= \mathbb{E}|g_r(\boldsymbol{x}_{t_0}^r) - g_f(\boldsymbol{x}_{t_0}^r) + g_f(\boldsymbol{x}_{t_0}^r) - g_f(\boldsymbol{x}_{t_0}^f)|\Delta t \\ &\leq \mathbb{E}|g_r(\boldsymbol{x}_{t_0}^r) - g_f(\boldsymbol{x}_{t_0}^r)|\Delta t + \mathbb{E}|g_f(\boldsymbol{x}_{t_0}^r) - g_f(\boldsymbol{x}_{t_0}^f)|\Delta t \\ &\leq \varepsilon\Delta t + \mathbb{E}|g_f(\boldsymbol{x}_{t_0}^r) - g_f(\boldsymbol{x}_{t_0}^f)|\Delta t \\ &= \varepsilon\Delta t + \mathbb{E}|g_f'(\boldsymbol{x}_{t_0}^\xi)(\boldsymbol{x}_{t_0}^r - \boldsymbol{x}_{t_0}^f)|\Delta t \qquad (\boldsymbol{x}_{t_0}^\xi \in [\boldsymbol{x}_{t_0}^r, \boldsymbol{x}_{t_0}^f]) \\ &\leq \varepsilon\Delta t + K\mathbb{E}|\boldsymbol{x}_{t_0}^r - \boldsymbol{x}_{t_0}^f|\Delta t \\ &= \varepsilon\Delta t \end{aligned} \tag{24}$$

Follow the pattern we have:

$$\begin{cases} \boldsymbol{x}_{t_2}^r = \boldsymbol{x}_{t_1}^r + g_r(\boldsymbol{x}_{t_1}^r)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \\ \boldsymbol{x}_{t_2}^f = \boldsymbol{x}_{t_1}^f + g_f(\boldsymbol{x}_{t_1}^f)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \end{cases} \tag{25}$$

...

$$\begin{cases} \boldsymbol{x}_{t_n}^r = \boldsymbol{x}_{t_{n-1}}^r + g_r(\boldsymbol{x}_{t_{n-1}}^r)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \\ \boldsymbol{x}_{t_n}^f = \boldsymbol{x}_{t_{n-1}}^f + g_f(\boldsymbol{x}_{t_{n-1}}^f)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) \end{cases} \tag{26}$$

Which leads to:

$$\begin{aligned} \mathbb{E}|\boldsymbol{x}_{t_2}^r - \boldsymbol{x}_{t_2}^f| &= \mathbb{E}|\boldsymbol{x}_{t_1}^r - \boldsymbol{x}_{t_1}^f + g_r(\boldsymbol{x}_{t_1}^r)\Delta t - g_f(\boldsymbol{x}_{t_1}^f)\Delta t + \sigma\sqrt{\Delta t}\mathcal{N}(0,1) - \sigma\sqrt{\Delta t}\mathcal{N}(0,1)| \\ &\leq \mathbb{E}|\boldsymbol{x}_{t_1}^r - \boldsymbol{x}_{t_1}^f| + \mathbb{E}|g_r(\boldsymbol{x}_{t_1}^r) - g_f(\boldsymbol{x}_{t_1}^f)|\Delta t \\ &\leq \mathbb{E}|\boldsymbol{x}_{t_1}^r - \boldsymbol{x}_{t_1}^f| + \varepsilon\Delta t + K\mathbb{E}|\boldsymbol{x}_{t_1}^r - \boldsymbol{x}_{t_1}^f|\Delta t \\ &\leq (1 + K\Delta t)\varepsilon\Delta t + \varepsilon\Delta t \end{aligned} \tag{27}$$

...

$$\mathbb{E}|\boldsymbol{x}_{t_n}^r - \boldsymbol{x}_{t_n}^f| \leq \varepsilon\Delta t \sum_{i=0}^{n-1}(1 + K\Delta t)^i \tag{28}$$

Now let $S = \sum_{i=0}^{n-1}(1 + K\Delta t)^i$, then consider followings:

$$\begin{aligned} S(K\Delta t) &= S(1 + K\Delta t) - S \\ &= \sum_{i=1}^{n}(1 + K\Delta t)^i - \sum_{i=0}^{n-1}(1 + K\Delta t)^i \\ &= (1 + K\Delta t)^n - 1 \\ &= (1 + K\frac{T}{n})^n - 1 \\ &\leq e^{KT} - 1 \end{aligned} \tag{29}$$

Finally we have:

$$\mathbb{E}|\boldsymbol{x}_{t_n}^r - \boldsymbol{x}_{t_n}^f| \leq \frac{\varepsilon}{K}(e^{KT} - 1) \tag{30}$$

$$\mathbb{E}|\boldsymbol{x}_{t_n} - \boldsymbol{x}_{t_n}^f| \leq \frac{\varepsilon}{K}(e^{KT} - 1) + K(1 + E|\boldsymbol{x}_0|^2)^{1/2}\Delta t \tag{31}$$

□