KOOPMAN OPERATOR BASED DIFFUSION MODELS

UDAY KIRAN REDDY TADIPATRI SHAILESH SRIDHAR

Abstract. Diffusion models represent a sophisticated class of deep-generative models crucial for transforming target data distributions into known distributions, such as Gaussian. Governed by underlying stochastic differential equations (SDEs), these processes pose significant computational challenges. In this work, we introduce a novel stochastic control perspective to address the nonlinear dynamics inherent in diffusion models. Leveraging the Koopman Operator Theory framework, we effectively translate nonlinear dynamics into linear dynamics within the Koopman eigenspace, offering computational efficiency gains during inference. Building upon this theoretical foundation, we extend diffusion modeling to the domain of video generation, pioneering a two-time scale diffusion approach for modeling video dynamics. Finally, we validate the efficacy of our Koopman-based diffusion model through experimentation on a toy circular distribution, demonstrating its versatility and applicability.

1. Introduction

Diffusion models are state-of-the-art deep-generative models that transform a given Gaussian distribution into a desired Data Distribution. This is done in two stages. In the first step, namely, Forward Diffusion, the original data is corrupted with Gaussian noise recursively. With the right set of scales one could make any data distribution converge to Gaussian. In the second step, namely, Reverse Diffusion, the data is "de-corrupted" via prediction of the mean and variance using a Deep Neural Network (DNN). Steering the wheel towards Control Theory, the process of Reverse Diffusion can be viewed as a non-linear dynamical system where the observables are just the states themselves. The Koopman Operator is a theoretical paradigm originating from Control theory which transforms **non-linear systems** (typically finite-dimensional) to **linear infinite-dimensional systems**. In the Koopman Theory literature [4], [15], [8] there are work-arounds to find appropriate finite-dimensional representations .

To reconstruct data from Gaussian noise, in Diffusion Models we require T queries from the DNNs, where T being the number of time steps. Due to a large number of parameters in the DNN, this task is quite compute intensive.

Traditionally, one can write diffusion models as Stochastic Differential Equations (SDE)s. Let, $x_0 \in \mu_x$, a distribution defined on $\mathbb{R}^{n_x}$, be drawn from a data-distribution. One can move these particles to any arbitrary distribution by the following SDE.

$$\text{Forward Diffusion: } d\mathbf{x}_t = f(t, \mathbf{x}_t)dt + \sigma(t, \mathbf{x}_t)d\mathbf{w}_t \tag{1}$$

We can reverse this process. Here we sample, $x_T \in \mathcal{N}(0, I)$, and move the particles systematically to the original data distribution, μ_X.

$$\text{Reverse Diffusion: } d\overleftarrow{\mathbf{x}}_t = \tilde{f}(T-t, \overleftarrow{\mathbf{x}}_t)dt + \tilde{\sigma}(T-t, \overleftarrow{\mathbf{x}}_t)d\overleftarrow{\mathbf{w}}_t \tag{2}$$

here, $\overleftarrow{\mathbf{x}}_t$ represents that time is moving from $T \to 0$. From the seminal work by Anderson [3], it is shown that,

$$\tilde{f}(t, \mathbf{x}_t) = \frac{\nabla_{\mathbf{x}} \left[p(t, \mathbf{x}_t)\sigma(t, \mathbf{x}_t)\sigma^T(t, \mathbf{x}_t)\right]}{p(t, \mathbf{x}_t)} - f(t, \mathbf{x}_t); \tilde{\sigma}(t, \mathbf{x}_t) = \sigma(t, \mathbf{x}_t) \tag{3}$$

The above choices will yield a time reversal of SDEs. In practice the SDEs are implemented post discretizations using various SDE samplers like Euler-Maruyama, Predictive Correction [13].

In the literature of diffusion models [5, 13], most works are centered around generating images. However, existing diffusion models for videos [6, 9, 14], treat videos as volumes, using Variational Autoencoders [7] to reduce the dimensions and then perform diffusion goverened by equations 1, and 2. *As far as authors are aware, there are no works that exploit the dynamics in videos for generation using a diffusion process.* This

paper, has attempted to develop mathematical tools for modeling video generation using diffusion process, via two-time scales.

In the reverse diffusion process, 3, one needs to have access to $p(t, \mathbf{x}_t)$ and having a closed form expression is very hard. Practitioners, parameterize this with Neural Networks using various approaches. DDPM [5] tries to predict the mean and variance of the noise. Score based models [13] predict the score, $-\nabla_{\mathbf{x}} p(t, \mathbf{x}_t)$. To reverse the process of diffusion we need to have T-queries to a DNN. In practice, this is a huge bottleneck, due to non-linearity the process cannot be accelerated via parallelism. We exploit Koopman theory to linearize the non-linear dynamics in the koopman eigenfucntions space. However, Koopman operators lies in infinite-dimensional space which is intractable to realize in practice. To achieve this we need to parameterize a NN for computing the representation, both encoding and decoding, and a state transition matrix for the linear system. Theoretically, this would save many FLOating Point Operations (FLOPs) in comparison to vanilla reverse diffusion.

1.1. **Contributions.** We highlight the key main contributions of this work:

- We provide stochastic Koopman perspective of diffusion SDEs.
- We propose diffusion process theory for Videos which is first of its kind.
- We propose a novel approache to speed-up the reverse diffusion process.

2. Stochastic Koopman Perspective of Diffusion SDEs

Let $\mathbf{x} \in \mathbb{R}^{n_x}$, consider the SDE below for the diffusion process for T time-steps.

$$\text{Forward Diffusion: } d\mathbf{x}_t = f(t, \mathbf{x}_t)dt + \sigma(t, \mathbf{x}_t)d\mathbf{w}_t \tag{4}$$

$$\text{Reverse Diffusion: } d\overleftarrow{\mathbf{x}}_t = \tilde{f}(T-t, \overleftarrow{\mathbf{x}}_t)dt + \tilde{\sigma}(T-t, \overleftarrow{\mathbf{x}}_t)d\overleftarrow{\mathbf{w}}_t \tag{5}$$

Here, w_t is the Wiener process or simply, $dw_t \sim \sqrt{dt}\mathcal{N}(0, I)$. From the ground-breaking result from Anderson [3], we have

$$\tilde{f}(t, \mathbf{x}_t) = \frac{\nabla_{\mathbf{x}} \left[p(t, \mathbf{x}_t)\sigma(t, \mathbf{x}_t)\sigma^T(t, \mathbf{x}_t) \right]}{p(t, \mathbf{x}_t)} - f(t, \mathbf{x}_t)$$
$$\tilde{\sigma}(t, \mathbf{x}_t) = \sigma(t, \mathbf{x}_t)$$

The above result can be be obtained by reversing the flow of probability measures governed by Fokker-Plank Equations.

2.0.1. *Koopman Operators.* Consider the below dynamical system, in this work we ignore the control inputs,

$$\dot{\mathbf{x}}_t = f(\mathbf{x}_t) \tag{6}$$
$$\mathbf{y}_t = g(\mathbf{x}_t) \tag{7}$$

here, $f(X \to X) \in F \subseteq L^p(X)$, $g(: X \to Y) \in G \subseteq L^p(X)$.

The koopman operators are defined as the following, $K : G \to G$, where,

$$K := g \circ f \tag{8}$$

$$Kg(\mathbf{x}_t) = g(\mathbf{x}_{t+1})$$

from the above property we see that one can use the Koopman operator and perform linear dynamic systems on observables. We have a similar notion for time-varying random dynamical systems (RDS), [15].

$$d\mathbf{x}_t = f(t, \mathbf{x}_t)dt + \sigma(t, \mathbf{x}_t)d\mathbf{w}_t$$
$$\mathbf{y}_t = g(\mathbf{x}_t)$$

From [15, Equation 24], we have that, for example, $t \geq t_0$.

$$K^t g(\mathbf{x}_{t_0}) = \mathbb{E}[g(\mathbf{x}_t)] \tag{9}$$

We will state a proposition that will be useful for our work,

Proposition 1. [15, Proposition 6] *Let ϕ be eigenfunction of the stochastic Koopman generator, K^S*

$$K^S g(\mathbf{x}) = f(\mathbf{x}_t)\nabla g(\mathbf{x}) + \frac{1}{2} tr(\sigma(\mathbf{x})\nabla^2 g(\mathbf{x})\sigma(\mathbf{x})^T)$$

associated with RDS generated by SDE 7 with the corresponding eigenvalue, λ. Then

$$d\phi(\mathbf{x}_t) = \lambda\phi(\mathbf{x}_t)dt + \nabla\phi(\mathbf{x}_t)\sigma(\mathbf{x}_t)d\mathbf{w}_t \tag{10}$$

ϕ is the eigenfunction of the stochastic Koopman operator, K^t also,

$$K^t\phi(\mathbf{x}) = e^{\lambda t}\phi(\mathbf{x})$$

2.1. **Approximation of Koopman Operators.** By lifting the observables to the Koopman eigenspace we have obtained a Linear RDS 10. For any generic computing the eigenfunctions, ϕ is intractable problem, therefore in this section we propose learning these functions using a Deep Neural Network [8, 10, 12].

For this work, it is the case that observables are state themselves, i.e, $g = I \implies \mathbf{y}_t = \mathbf{x}_t$.

We define, $\mathbf{x} \in X \subseteq \mathbb{R}^{n_x}$, $\mathbf{z} \in \mathbf{Z} \subseteq \mathbb{R}^{n_z}$, where $n_z \leq n_x$. We define an encoder, $\psi_E : X \to \mathbf{Z}$, this mimics the eigenfunction, ϕ. A linear RDS with state transition matrix, $A \in \mathbb{R}^{n_z \times n_z}$ which governs the linear dynamics in Koopman eigenspace. We have an inverse operation, $\psi_D := \psi_E^{-1} : \mathbf{Z} \to X$, which takes use from Koopman eigenspace to observables. We have the following properties,

- $\mathbf{x}_t = \psi_D \circ A^{(t-t_0)} \circ \psi_E(\mathbf{x}_{t_0})$.
- $\mathbf{x} = \psi_D \circ \psi_E(\mathbf{x})$.

To connect the above SDEs to practically implemented diffusion models, we describe a few illustrations.

Example 1: Denoising Diffusion Probabilistic Models (DDPM) [5]
Choose $f(t, \mathbf{x}_t) = -\frac{1}{2}\beta_t\mathbf{x}_t$, and $g(t, \mathbf{x}_t) = \sqrt{\beta_t}$, where $\beta_t \in \mathbb{R}^+$ follow some noise scheduling sequence. Now we use tools from Itô calculus to obtain, $p(t, \mathbf{x}_t) = N\left(exp(-\frac{1}{2}\int_0^t \beta_s ds)\mathbf{x}_t, \left(1 - exp(-\frac{1}{2}\int_0^t \beta_s ds)\right) I\right)$. From, these choices we have that,

$$\begin{aligned}
\tilde{f}(t, \mathbf{x}_t) &= \beta_t \nabla_{\mathbf{x}_t} lnp(t, \mathbf{x}_t) + \frac{1}{2}\beta_t\mathbf{x}_t \\
&= -\beta_t \left(1 - exp(-\frac{1}{2}\int_0^t \beta(s)ds)\right)^2 \mathbf{x}_t + \frac{\beta_t}{2}\mathbf{x}_t
\end{aligned}$$

$$\tilde{\sigma}(t, \mathbf{x}_t) = \sqrt{\beta_t}$$

By performing Euler-Maruyama discretization and using a neural network to predict score, $-\nabla_{\mathbf{x}} lnp(t, \mathbf{x}_t)$, we get,

$$\mathbf{x}_{t-1} = c_{1,t}F_\theta(t, \mathbf{x}_t) + c_{2,t}\mathbf{x}_t + c_{3,t}\epsilon$$

where, $\epsilon \sim N(0, I)$, for some constants, $c_{1,t}, c_{2,t}, c_{3,t}$ dependent on β_t. In practice, the addition of the noise, ϵ is ignored due the propagation of noise from the DNNs.

$$\mathbf{x}_{t-1} \approx c_{1,t}F_\theta(t, \mathbf{x}_t) + c_{2,t}\mathbf{x}_t$$

Example 2: Score Based Diffusion Models [13]
In this type of diffusion models, there are many choices of $f(t, \mathbf{x}_t)$, $\sigma(t, \mathbf{x}_t)$. We tabulate (Appendix 6) the results the variants and provide the discrete version of the algorithms. Sampling the provided SDEs like demonstrated in the Example 1, we would be getting a recursive diffusion steps.

3. Diffusion process theory for Videos

Consider the probability space, $(X, B(X), \mu_X)$. Let, $x_{t,s} \in X$ be s'th frame at t'th diffusion time step. We model that video follow dynamics in the image manifold, $M(X)$.

$$\mathbf{x}_{0,s+1} = f_v(0, s, \mathbf{x}_{0,s})$$

where, $f_v : T \times S \times X \to X$. Consider, the below diffusion process in two time scales, i.e, (1) diffusion process that moves the probability distributions, (2) diffusion process that moves images to generate images. The stochastic differential equation will of the form,

$$d\mathbf{x}_{t,s} = f_d(t, s, \mathbf{x}_t)dt + f_v(t, s, \mathbf{x}_t)ds + g_d(t, s)d\mathbf{w}_t + g_v(t, s)d\mathbf{w}_s \tag{11}$$

where, $f_d : T \times S \times X \to X$, $g_d, g_v : T \times S \to \mathbb{R}$. $\mathbf{w}_t$, $\mathbf{w}_s$ are wiener process with correlation coefficient, $\rho(t,s)$. For simplicity we can consider that,

$$\begin{bmatrix} d\mathbf{w}_t \\ d\mathbf{w}_s \end{bmatrix} = \begin{bmatrix} I & 0 \\ \rho(t,s) & \sqrt{1-\rho^2(t,s)} \end{bmatrix} \begin{bmatrix} d\mathbf{v}_t \\ d\mathbf{v}_s \end{bmatrix}$$

where, $d\mathbf{v}_t \sim \mathcal{N}(0, Idt)$, and $d\mathbf{v}_s \sim N(0, Ids)$.

Theorem 1. *Give the forward SDE for two-time scale diffusion*

$$d\mathbf{x}_{t,s} = f_d(t,s,\mathbf{x}_t)dt + f_v(t,s,\mathbf{x}_t)ds + g_d(t,s)d\mathbf{w}_t + g_v(t,s)d\mathbf{w}_s$$

we have that the reverse SDE would be,

$$d\overleftarrow{\mathbf{x}}_{t,s} = \left[-f_d + \frac{1}{2}(g_d^2 + 2\rho(t,s)g_d g_v)\partial_{\mathbf{x}} lnp\right]dt + \left[-f_v + \frac{1}{2}g_v^2 p \partial_{\mathbf{x}} lnp\right]ds + g_d d\overleftarrow{\mathbf{w}}_t + g_v d\overleftarrow{\mathbf{w}}_s \tag{12}$$

Proof: Refer to Appendix 6.

4. Accelerating Reverse Diffusion

4.1. **Tracking non-linear trajectories of reverse diffusion model.** Observe that Equations 1, and 2 just describe stochastic non-linear systems, which is $[c_{1,t}F_\theta + c_{2,t}I]$. For the non-linearity, F_θ we perform Koopman representation, namely encoder $\psi_e : \mathbb{R}^{n_x} \to \mathbb{R}^{n_k}$, $\psi_d : \mathbb{R}^{n_k} \to \mathbb{R}^{n_x}$, and $A_t \in \mathbb{R}^{n_x \times n_x}$.

$$F_\theta(t, \mathbf{x}_t) = \psi_d \circ A_t \circ \psi_e(\mathbf{x}_t)$$

The idea is to encode the gaussian sample, $x_T \sim \mathcal{N}(0, I)$ and perform linear system on the encoded space to traverse T-timesteps and revert back to original space.

$$\mathbf{x}_t = \left[\sum_{k=t+1}^{T} c_{1,k}\psi_d \circ \left[\prod_{j=1}^{k} A_j\right] \circ \psi_e + c_{2,T}I\right](\mathbf{x}_T) \tag{13}$$

Suppose that, ψ_d was linear, then we can simplify that,

$$\mathbf{x}_t = \left[\psi_d \circ \sum_{k=t+1}^{T} c_{1,k}\left[\prod_{j=1}^{k} A_j\right] \circ \psi_e + c_{2,T}I\right](\mathbf{x}_T) \tag{14}$$

Let us parameterize the singular space of A_k's and keeping the Singular Spaces Time-Invariant, such as choosing, Fourier, or Cosine basis.

$$A_k = E\Sigma(k)E^H$$

Then we have that,

$$\mathbf{x}_t = \left[\psi_d \circ E \sum_{k=t+1}^{T} c_{1,k}\left[\prod_{j=1}^{k} \Sigma(k)\right] E^H \circ \psi_e + c_{2,T}I\right](\mathbf{x}_T) \tag{15}$$

The computation of the BLUE term is very fast as we have only, n_k non-zero elements. The main limitation is that decoder ψ_d is linear, however, we can make ψ_e non-linear.

On the contrary, if we were to use a small non-linear decoder, and a large encoder, then we would also save our computation.

$$\mathbf{x}_t = \sum_{k=t+1}^{T} c_{1,k}\left[\psi_d\left(\left[\prod_{j=1}^{k} A_j\right]\psi_e(\mathbf{x}_T)\right) + c_{2,T}I\right](\mathbf{x}_T) \tag{16}$$

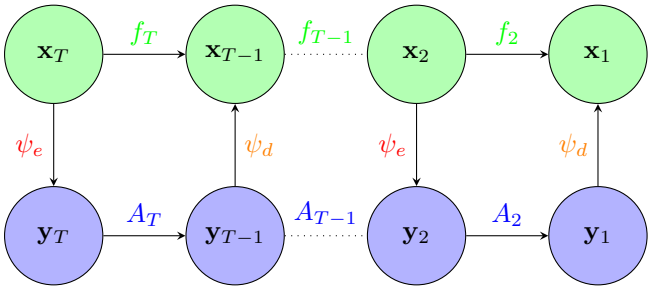

FIGURE 1. Non-Linear Systems vs Koopman Linear System. $\mathbf{x}_t$'s are the diffusion variables. f_t is a DNN. ψ_e, and ψ_d are the functions that let us move from original space to Koopman space. $\mathbf{y}_t$'s are the states in the Koopman Linear System.

5. Experiments and Results

5.1. **Setup.** Building off of [2], our setting consists of a sample of n points in $\mathbb{R}^2$, within limits. In a low noise situation, the points are arranged to form a circle centered at the origin. This corresponds to $t = 0$, with forward diffusion we want move from this circular distribution to an isotropic Gaussian distribution, as shown in Figure 6.3 and backwards with reverse diffusion, as shown in Figure 6.3. (Appendix)

We use a simple MLP to learn the reverse diffusion process, consisting of an encoder, linear system and decoder. The encoder and decoder consist of blocks of linear layers and GeLU activations. The linear system consists of a single linear layer without bias, with equal input and output units and no bias unit

We a use sinusoidal positional embeddings for both x and y ($\mathbb{R}^2$) components, as well as a sinusoidal embedding for time.

The loss term involves a mean square error loss between the expected noisy output, x_{t-1} using noise scheduling equations, and $\hat{x}_{t-1}$ predicted by the model.

We also experiment with adding additional terms to the loss function $A_j z_t = z_{t-1}$.

5.2. **Experiments and Results.**

5.2.1. *DDPM* [5]. We find that this simple model, with an encoder, linear system and decoder is able to learn the standard DDPM reverse process quite well. 6.3

5.2.2. *KOT Diffusion.* Standard DDPM learns a model to predict the noise ϵ. However, referring to 14, with c_2 set to 0 we attempt to predict x_{t-1} directly from x_t This works reasonably well, showing that we can indeed use Koopman operator theory for more efficient diffusion. We find that Neural networks require much more tuning and training on this task than on standard DDPM.

Let k indicate the number of time-steps propagated in the koopman eigenfunction space. Let, c be the function the computes the complexity time of the argument function. To perform T revere diffusion, we need $\mathcal{O}\left(\frac{T}{n}(c(\psi_e) + c(\psi_d)) + c(\psi_d)\right)$ amount of compute. We observe that using $k = 2$, and skipping an entire encoding decoding cycle, we get reasonable results. A plot can be seen in the appendix. 8

Beyond $k = 3$, we see that results are not as impressive. This indicates a tradeoff between quality and compute time.

6. Conclusion and Future Work

In conclusion, this work attempts to contribute significantly to the advancement of diffusion models, particularly in the context of deep-generative modeling and video generation. By introducing a novel stochastic control perspective and leveraging the Koopman Operator Theory framework, we address the computational challenges inherent in diffusion processes, offering efficiency gains during inference. Our novel extension of diffusion modeling to video dynamics via a two-time scale approach opens up new avenues for research and application in the field. Furthermore, our proposed method to accelerate the reverse diffusion process demonstrates promising potential for improving computational efficiency. Through these contributions, we provide valuable insights and tools for advancing generative modeling techniques, with implications for various domains such as image and video generation, imitation learning, and trajectory estimation.

References

[1] Reverse Time Stochastic Differential Equations [for generative modelling] — ludwigwinkler.github.io. `https://ludwigwinkler.github.io/blog/ReverseTimeAnderson/`. [Accessed 14-05-2024].

[2] tinydiffusion. https://github.com/tanelp/tiny-diffusion.

[3] P. W. Anderson. Absence of diffusion in certain random lattices. *Phys. Rev.*, 109:1492–1505, Mar 1958.

[4] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. 11(2):e0150171.

[5] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models.

[6] J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet. Video diffusion models.

[7] D. P. Kingma and M. Welling. Auto-encoding variational bayes.

[8] P. Laferrière, S. Laferrière, S. Dahdah, J. R. Forbes, and L. Paull. Deep koopman representation for control over images (DKRCI). In *2021 18th Conference on Robots and Vision (CRV)*, pages 158–164.

[9] Y. Liu, K. Zhang, Y. Li, Z. Yan, C. Gao, R. Chen, Z. Yuan, Y. Huang, H. Sun, J. Gao, L. He, and L. Sun. Sora: A review on background, technology, limitations, and opportunities of large vision models.

[10] B. Lusch, J. N. Kutz, and S. L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. 9(1):4950. Publisher: Nature Publishing Group.

[11] H. Risken. Fokker-planck equation. In *The Fokker-Planck Equation: Methods of Solution and Applications*, pages 63–95. Springer Berlin Heidelberg.

[12] H. Shi and M. Q.-H. Meng. Deep koopman operator with control for nonlinear systems.

[13] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations.

[14] S. Yu, W. Nie, D.-A. Huang, B. Li, J. Shin, and A. Anandkumar. Efficient video diffusion models via content-frame motion-latent decomposition.

[15] N. **v**Crnjarić **v**Zic, S. Maće**v**sić, and I. Mezić. Koopman operator spectrum for random dynamical systems.

Appendix

Type	$f(t,\mathbf{x}_t)$	$\tilde{f}(t,\mathbf{x}_t)$	$\sigma(t,\mathbf{x}_t)$	$\tilde{\sigma}(t,\mathbf{x}_t)$
Variance Exploding	0	$\sigma'^2(t)\nabla lnp(t,X_t)$	$\sqrt{\sigma'^2(t)}$	$\sqrt{\sigma'^2(t)}$
Variance Preserving	$-\frac{1}{2}\beta(t)\mathbf{x}_t$	$\beta(t)\nabla lnp(t,X_t)+\frac{1}{2}\beta(t)\mathbf{x}_t$	$\sqrt{\beta(t)}$	$\sqrt{\beta(t)}$
Sub-variance Preserving	$-\frac{1}{2}\beta(t)\mathbf{x}_t$	$\beta(t)(1-e^{-2\int_0^t\beta(s)ds})\nabla lnp(t,\mathbf{x}_t)+\frac{1}{2}\beta(t)\mathbf{x}_t$	$\sqrt{\beta(t)(1-e^{-2\int_0^t\beta(s)ds})}$	$\sqrt{\beta(t)(1-e^{-2\int_0^t\beta(s)ds})}$

Table 1. Score based Diffusion Models [13]

6.1. Score based diffusion.

6.2. Proof of Video Diffusion.

Theorem 2. *Given the forward SDE for two-time scale diffusion*

$$d\mathbf{x}_{t,s}=f_d(t,s,\mathbf{x}_t)dt+f_v(t,s,\mathbf{x}_t)ds+g_d(t,s)d\mathbf{w}_t+g_v(t,s)d\mathbf{w}_s$$

the reverse SDE becomes,

$$d\overleftarrow{\mathbf{x}}_{t,s}=\left[-f_d+\frac{1}{2}(g_d^2+2\rho(t,s)g_dg_v)\partial_{\mathbf{x}}lnp\right]dt+\left[-f_v+\frac{1}{2}g_v^2p\partial_{\mathbf{x}}lnp\right]ds+g_d d\overleftarrow{\mathbf{w}}_t+g_v d\overleftarrow{\mathbf{w}}_s \tag{17}$$

Proof. **Strategy**

- We take some function, f that is measurable w.r.t the Borel set, $B(X)$.
- We perform a 2-nd order approximation of the increment, df, and then compute the expectation, $\mathbb{E}[df]$.
- We compute $\frac{d}{dt}\mathbb{E}[f]$ from the first principle of expectation. This yields, the so called *Fokker-Plank Equations* [11], Since our equations consist of two-time scales, we will obtaining a slightly different set of equations than usual [1].
- Then we use the time reversal form of Fokker-Plank Equations to get the time reversal SDE.

To avoid clumsy notation, we drop the arguments to the functions and indices for $d\mathbf{x}_{t,s}$.
Take a function, $f\in M(B(X))$. Performing a second order taylor approximation,

$$df = \langle\partial_{\mathbf{x}}f,d\mathbf{x}\rangle+\frac{1}{2}d\mathbf{x}^T\partial_{\mathbf{x}}f\partial_{\mathbf{x}}f^Td\mathbf{x} \tag{18}$$

$$\tag{19}$$

From the forward SDE 11 we have,

$$\begin{aligned}(\partial_x f^T d\mathbf{x})^2 &= d\mathbf{x}^T\partial_x f\partial_x f^T d\mathbf{x}\\ &= \|\partial_x f^T f_d\|_2^2(dt)^2+\|\partial_x f^T f_v\|_2^2(ds)^2+g_d^2\|\partial_x f^T d\mathbf{w}_t\|_2^2+g_v^2\|\partial_x f^T d\mathbf{w}_s\|_2^2\\ &+ 2\langle\partial_x f^T f_d dt,\partial_x f^T f_v ds\rangle+2\langle\partial_x f^T f_d dt,\partial_x f^T g_d d\mathbf{w}_t\rangle+2\langle\partial_x f^T f_d dt,\partial_x f^T g_v d\mathbf{w}_s\rangle\\ &+ 2\langle\partial_x f^T f_v ds,\partial_x f^T g_d d\mathbf{w}_t\rangle+2\langle\partial_x f^T f_v ds,\partial_x f^T g_v d\mathbf{w}_s\rangle+2\langle\partial_x f^T g_d d\mathbf{w}_t,\partial_x f^T g_v d\mathbf{w}_s\rangle\end{aligned}$$

We can ignore the BLUE terms, as $(dt)^2,(ds)^2\to 0$, and $dtd\mathbf{w}_t=\mathcal{O}(dt^{1.5})\to 0$, $dsd\mathbf{w}_s=\mathcal{O}(ds^{1.5})\to 0$.

$$\begin{aligned}(\partial_x f^T d\mathbf{x})^2 &\approx g_d^2\|\partial_x f^T d\mathbf{w}_t\|_2^2+g_v^2\|\partial_x f^T df\mathbf{w}_s\|_2^2\\ &+ 2\langle\partial_x f^T f_d dt,\partial_x f^T f_v ds\rangle+2\langle\partial_x f^T f_d dt,\partial_x f^T g_v d\mathbf{w}_s\rangle\\ &+ 2\langle\partial_x f^T f_v ds,\partial_x f^T g_d d\mathbf{w}_t\rangle+2\langle\partial_x f^T g_d d\mathbf{w}_t,\partial_x f^T g_v d\mathbf{w}_s\rangle\end{aligned}$$

Now we apply expectation,

$$\mathbb{E}_{\mathbf{w}_t,\mathbf{w}_s}\left[(\partial_x f^T d\mathbf{x})^2\right] \approx (g_d^2+2\rho(t,s)g_dg_v)\|\partial_x f\|_2^2dt+g_v^2\|\partial_x f\|_2^2ds+2\partial_x f^T f_d\partial_x f^T f_v dtds$$

$$\begin{aligned}\mathbb{E}_{\mathbf{w}_t,\mathbf{w}_s}[\langle\partial_{\mathbf{x}}f,d\mathbf{x}\rangle] &= \mathbb{E}[\langle\partial_{\mathbf{x}}f,f_d dt+f_v ds+g_d d\mathbf{w}_t+g_v d\mathbf{w}_s\rangle]\\ &= \langle\partial_{\mathbf{x}}f,f_d\rangle dt+\langle\partial_{\mathbf{x}}f,f_v\rangle ds\end{aligned}$$

Finally stitching the above equations and ignoring the term $dtds$,

$$\mathbb{E}[df] = \mathbb{E}_{\mathbf{x},\mathbf{w}_t,\mathbf{w}_s}[df] = \mathbb{E}_{\mathbf{x}}\left[\left[\left(\frac{1}{2}g_d^2 + \rho(t,s)g_d g_v\right)\|\partial_x f\|_2^2 + \langle\partial_{\mathbf{x}} f, f_d\rangle\right]dt + \left[\frac{1}{2}g_v^2\|\partial_x f\|_2^2 + \langle\partial_{\mathbf{x}} f, f_v\rangle\right]ds\right] \tag{20}$$

From the first principles of expectation we have that,

$$\mathbb{E}[df] = \int_{\mathbf{x}\in X}\left[\left(\frac{1}{2}g_d^2 + \rho(t,s)g_d g_v\right)\|\partial_x f\|_2^2 + \langle\partial_{\mathbf{x}} f, f_d\rangle\right]pdtd\mathbf{x} + \int_{\mathbf{x}\in X}\left[\frac{1}{2}g_v^2\|\partial_x f\|_2^2 + \langle\partial_{\mathbf{x}} f, f_v\rangle\right]pdsd\mathbf{x}$$

Now we apply integration by parts we get that,

$$\mathbb{E}[df] = \int_{\mathbf{x}\in X} f\left[\langle -\partial_{\mathbf{x}} f_d p + \frac{1}{2}\partial_{\mathbf{x}}^2(g_d^2 p + 2\rho(t,s)g_d g_v p), d\mathbf{x}\rangle\right]dt + \int_{\mathbf{x}\in X} f\left[\langle -\partial_{\mathbf{x}} f_v p + \frac{1}{2}\partial_{\mathbf{x}}^2(g_v^2 p), d\mathbf{x}\rangle\right]ds$$

From the first principles we have that,

$$\mathbb{E}[df] = \int_{\mathbf{x}\in X} f\partial_t p d\mathbf{x}dt + \int_{\mathbf{x}\in X} f\partial_s p d\mathbf{x}ds \tag{21}$$

Comparing the above two we get,

$$\partial_t p = -\partial_{\mathbf{x}} f_d p + \frac{1}{2}\partial_{\mathbf{x}}^2(g_d^2 p + 2\rho(t,s)g_d g_v p) = \partial_{\mathbf{x}}\left(f_d + \frac{1}{2}(g_d^2 + 2\rho(t,s)g_d g_v)\partial_{\mathbf{x}} lnp\right) \tag{22}$$

$$\partial_s p = -\partial_{\mathbf{x}} f_v p + \frac{1}{2}\partial_{\mathbf{x}}^2(g_v^2 p) = \partial_{\mathbf{x}}\left(f_v + \frac{1}{2}g_v^2 p\partial_{\mathbf{x}} lnp\right) \tag{23}$$

If we define, $t : T - t$, then our equation will flip and look like,

$$\partial_t p = \partial_{\mathbf{x}} f_d p - \frac{1}{2}\partial_{\mathbf{x}}^2(g_d^2 p + 2\rho(t,s)g_d g_v p) = \partial_{\mathbf{x}}\left(f_d - \frac{1}{2}(g_d^2 + 2\rho(t,s)g_d g_v)\partial_{\mathbf{x}} lnp\right) \tag{24}$$

$$\partial_s p = \partial_{\mathbf{x}} f_v p - \frac{1}{2}\partial_{\mathbf{x}}^2(g_v^2 p) = \partial_{\mathbf{x}}\left(f_v - \frac{1}{2}g_v^2 p\partial_{\mathbf{x}} lnp\right) \tag{25}$$

Therefore, our reverse SDE would look like,

$$d\overleftarrow{\mathbf{x}}_{t,s} = \left[-f_d + \frac{1}{2}(g_d^2 + 2\rho(t,s)g_d g_v)\partial_{\mathbf{x}} lnp\right]dt + \left[-f_v + \frac{1}{2}g_v^2 p\partial_{\mathbf{x}} lnp\right]ds + g_d d\overleftarrow{\mathbf{w}}_t + g_v d\overleftarrow{\mathbf{w}}_s \tag{26}$$

□

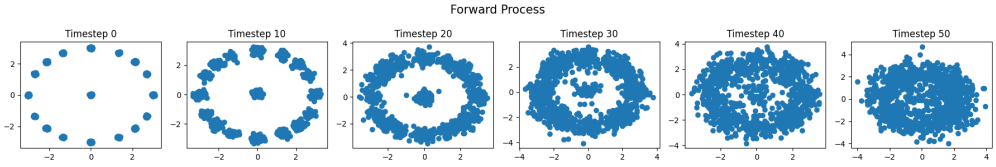

FIGURE 2. Forward Diffusion process, from a circle towards and isotropic gaussian distribution

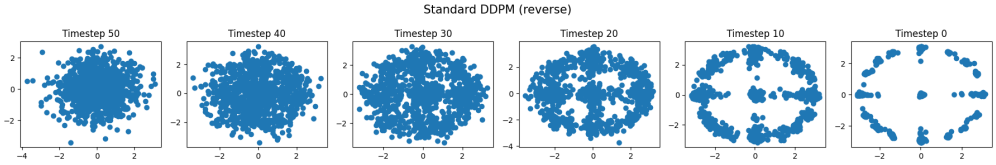

FIGURE 3. Standard Reverse Diffusion process in DDPM

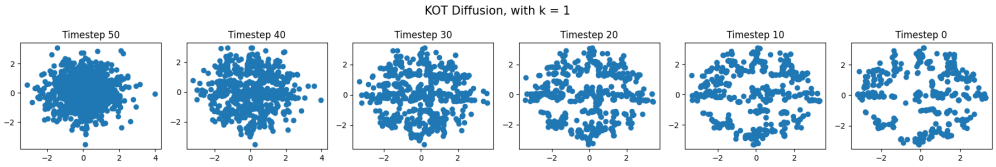

FIGURE 4. $k = 1$

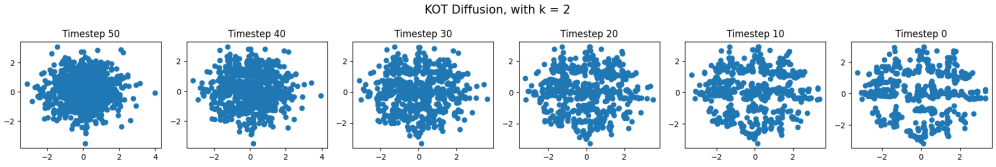

FIGURE 5. $k = 2$

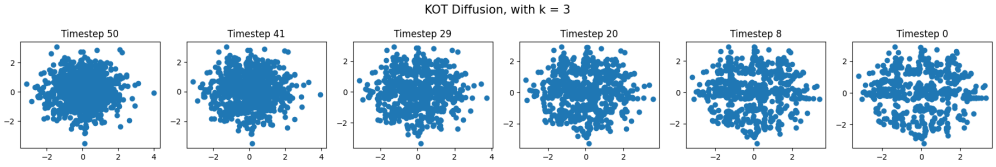

FIGURE 6. $k = 3$

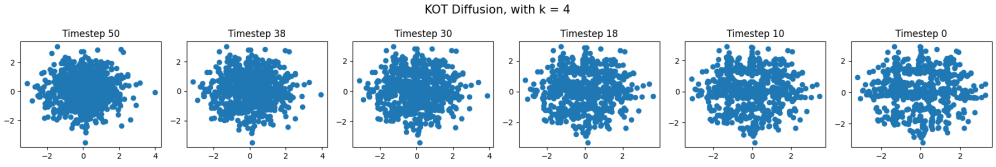

FIGURE 7. $k = 4$

FIGURE 8. KOT based Reverse Diffusion

6.3. **Experiment Results.**