RENORMALIZATION GROUP FLOW, OPTIMAL TRANSPORT AND DIFFUSION-BASED GENERATIVE MODEL

Artan Sheshmani^{1,2,3}, Yi-Zhuang You⁴, Baturalp Buyukates⁵, Amir Ziashahabi⁵ and Salman Avestimehr⁵

ABSTRACT. Diffusion-based generative models represent a forefront direction in generative AI research today. Recent studies in physics have suggested that the renormalization group (RG) can be conceptualized as a diffusion process. This insight motivates us to develop a novel diffusion-based generative model by reversing the momentum-space RG flow. We establish a framework that interprets RG flow as optimal transport gradient flow, which minimizes a functional analogous to the Kullback-Leibler divergence, thereby bridging statistical physics and information theory. Our model applies forward and reverse diffusion processes in Fourier space, exploiting the sparse representation of natural images in this domain to efficiently separate signal from noise and manage image features across scales. By introducing a scale-dependent noise schedule informed by a dispersion relation, the model optimizes denoising performance and image generation in Fourier space, taking advantage of the distinct separation of macro and microscale features. Experimental validations on standard datasets demonstrate the model's capability to generate high-quality images while significantly reducing training time compared to existing image-domain diffusion models. This approach not only enhances our understanding of the generative processes in images but also opens new pathways for research in generative AI, leveraging the convergence of theoretical physics, optimal transport, and machine learning principles.

MSC codes: 03B70, 03-04, 03D10, 11Y16

Keywords: Category theory, Renormalization Flow theory, Diffusion-based models, Categorical representation learning, Natural language processing (NLP).

CONTENTS

| Introduction | 2 |
|---|--------|
| 1. Background: Optimal transport theory, Monge and Kantorovich formulations | 4 |
| 1.1. Generalization to Fuzzy Transport | 4 |
| 1.2. Kantorovich Formulation | 5 |
| 1.3. Kantorovich Duality | 5 |
| 1.4. Wasserstein Distance | 5 |
| 2. Benamou-Brenier formulation of optimal transports and discretized diffusion equa | tion 6 |
| 3. RG flow, and its connection to Wasserstein Gradient flow and diffusion dynamics | 7 |
| 4. Diffusion-based model | 8 |
| 4.1. Forward Diffusion (Renormalization) | 8 |
| 4.2. Reverse Diffusion Process (Generation) | 9 |
| 5. Applications: Forward and reverse Diffusion in Fourier Space | 9 |
| 5.1. Overview and Motivation | 9 |
| 5.2. Fourier-Space Diffusion Model | 9 |
| 5.3. Description of FDDM | 11 |
| 5.4. Experiment Results | 13 |
| Appendix A. Discretization of Wasserstein distance function | 15 |
| References | 17 |

INTRODUCTION

Deep generative models have been applied to various domains, including image generation, text generation, and audio synthesis. They have been used for tasks such as image generation, data augmentation, anomaly detection, and even as tools for understanding the underlying structure of the data. The diffusion generative models are a special type of generative model aiming to learn the underlying distribution of the data by iteratively applying a diffusion process to a simple noise distribution. They model the data as the result of a sequence of transformations that gradually corrupt the noise until it resembles the desired data distribution. The core idea behind the diffusion process is to introduce noise into the generated samples and then gradually reduce the noise level over multifold steps. To summarize it: each step of the diffusion process consists of two operations: an update step and a sampling step. During the update step, the model takes a noisy sample and updates it to reduce the noise level. This is typically done using a neural network that takes the current noisy sample and produces an estimate of the noise that needs to be subtracted. During the sampling step, the model samples from the updated noisy sample to generate a new sample. This step introduces stochasticity into the process and allows for the exploration of the data distribution. The diffusion process is performed for multiple steps, progressively reducing the noise level until the generated sample resembles the target data distribution. The model is trained by optimizing the likelihood of the data given the diffusion process.

One advantage of diffusion generative models is that they allow one to generate high-quality samples and capture complex data distributions. They have shown impressive results in various domains, including image generation, text generation, and audio synthesis. The training process of diffusion models can be computationally intensive, however recent advancements have introduced techniques to improve training efficiency, such as using reversible networks and applying diffusion models in a hierarchical manner. The current article is an attempt to create a diffusion generative model, based on ideas borrowed from mathematics (algebraic geometry, category theory, and theory of optimal transports) and physics (theory of renormalization group flows). A part of this endeavor has already been established by the work of the first two named authors, that is, the construction of a flow-based autoencoding and auto-decoding algorithm based on category theory and renormalization group flows in quantum field theory.

Several recent works [4–6] have pointed out that the renormalization group (RG) can be conceptualized as an optimal transport of complicated probability distributions towards trivial distributions, described as a diffusion process. Following the early idea that inverse RG can be viewed as a generative model [2, 15, 21], there has been proposals of using generative model to learn the optimal RG flow [3, 9, 14]. In this work, we will explore the opposite direction of developing diffusion-based generative models by reversing the RG flow.

In [25] the authors constructed an algorithmic architecture capable of rapid classification of large-size data with few labels, as well as the discovery of new data that could potentially have similar characteristics to labeled data. The authors further showed as an example, that the construction can have applications in the field of "Sequence-to-Function-Mapping" which is currently a highly attractive focus of development in the Bio-Tech industry. The construction of the "RG-flow based categorifier" borrows ideas from the theory of renormalization group flows (RG) in quantum field theory, holographic duality, and hyperbolic geometry, and mixing them with neural ODE's (ordinary differential equations). The authors constructed a new algorithmic natural language processing (NLP) architecture, called the RG-flow categorifier or for short the RG categorifier, which is capable of data classification and generation in all layers. The authors in [25] showed that the RG categorifier is capable of

- Combining representation and generative learning of data sets
- Unsupervised classification of data sets
- Discovering fine and coarse features within the dataset

- Precise analysis of data sets with their given functional characteristics, in particular, it enjoys trackable likelihood estimation capability
- Being computationally efficient
- Having explainability features, that is, capable of exploring explicitly how local structures in the data sets, induce their global features

Despite having the advantages mentioned above, the RG categorifier shows low efficiency in speed which led the authors of the current article to envision the construction of a similar, yet less computationally intensive auto-encoder and decoder which works based on Diffusion dynamics in optimal transport theory.

Optimal transport theory, also known as transportation theory or Monge-Kantorovich theory, is a mathematical framework for studying the transportation of mass from one location to another while minimizing the cost of transportation. This theory has applications in a variety of fields, including economics, physics, and image processing.

At its core, optimal transport theory seeks to find the optimal way to move a certain amount of mass from one location to another, subject to certain constraints such as the cost of transportation or the total amount of mass that can be transported. This is done by formulating the problem as an optimization problem and solving it using various mathematical techniques such as linear programming, convex analysis, and partial differential equations.

One of the key concepts in optimal transport theory is the Wasserstein distance, which measures the distance between two probability distributions by computing the minimum amount of work required to transform one distribution into the other. This concept has important applications in image processing, where it can be used to compare two images and measure the amount of deformation required to transform one image into the other.

When it comes to artificial intelligence (AI) and machine learning (ML), one of the main areas of application of optimal transport theory is in generative models, where optimal transport can be used to measure the distance between probability distributions and help guide the training of models.

In particular, optimal transport theory has been used to develop a new class of generative models called Wasserstein generative adversarial networks (WGANs), which improve upon traditional GANs by using the Wasserstein distance to measure the distance between the generated and real data distributions. This results in more stable training and better sample quality.

Optimal transport theory has also been used in image synthesis, where it can be used to learn the mapping between two images or image domains. For example, optimal transport theory has been used to develop domain adaptation methods, where a model is trained on one set of images and then transferred to a different set of images by learning the optimal transport plan between the two image domains. Another area of application is in natural language processing (NLP), where optimal transport theory can be used to measure the similarity between two text documents or the distance between two-word embeddings. This has led to the development of new techniques for text classification and clustering based on optimal transport.

We apply the theory of optimal transport to a Fourier space diffusion problem and propose the Frequency Domain Diffusion Model (FDDM) as an alternative to image domain diffusion models. FDDM leverages the natural separation of components in the frequency domain and utilizes a scale-dependent noise schedule to intelligently add/remove noise during the diffusion process for efficient image generation. Combined with a JPEG-inspired design, FDDM achieves a computational speedup of $2.7-8.5\times$, with a modest impact on image quality.

This article is organized as follows: In Section 1 we review the Monge-Kantorovich formulation in optimal transport theory. Then in Section 3 we show that there is a close and deep relationship between RG flow theory and Monge-Kantorovich formulation, that is replacing the probabilistic metric used to optimize the Wasserstein distance function, with a certain derivative of the correlation matrix in RG dynamics, turns the RG flow equation into a Benamou-Brenier discretized diffusion equation on space of probability distributions on a given fixed space. After this mathematical construction, we

apply the theory of optimal transport to Fourier space associated with images in Section 5, and show the performance of an auto encoding and decoding algorithm which implements Benamou-Brenier discretized diffusion to encode and decode data.

1. BACKGROUND: OPTIMAL TRANSPORT THEORY, MONGE AND KANTOROVICH FORMULATIONS

In this section, we will first review the formulation of renormalization group (RG) flow as optimal transport developed by Cotler and Rezchikov [8]. In this formulation, the RG flow can be viewed as a trivializing flow of a probability distribution, described by a diffusion equation. The inverse diffusion can then be used to design a diffusion-based generative model.

Let X and Y be separable metric spaces with positive measure μ_X and μ_Y respectively. The Monge formulation of optimal transportation problem is to find a "transport" map $T:X\to Y$ such that for any measure subset $S\subset X$ we have that

$$\int_{S \subset X} \mu_X dx = \int_{T(S)} \mu_Y dy,$$

that is, using the pullback pushforward formula, we have that

$$T_*\mu_X = \mu_Y$$
.

The map T will be called an optimal transport if attains the infimum value of the expected cost function associate to the transport, that is

$$\inf\{\int_{X\times Y} c(x,y)d\eta_{X\times Y} \mid \eta_{X\times Y} \in \Gamma(\mu_X,\mu_Y)\},\$$

where $c(x,y): X\times Y\to [0,\infty]$ is a Borel measurable function and $\Gamma(\mu_X,\mu_Y)$ denotes the set of probability measures on $X\times Y$ with marginal probability measures μ_X on X and μ_Y on Y. As we elaborated the pushforward of Borel measure μ_X under transport map T is the Borel measure μ_Y . When the map T is given by a C^1 -smooth map, this identity is realized in appropriate local coordinate charts as

(1)
$$\mu_X(x) = \mu_Y(T(x)) \mid \det(\partial_x T(x)) \mid$$

This nonlinear condition can be relaxed to a KL divergence regularization.

(2)
$$\mathcal{L}(T,\lambda) = \int_{X} c(x,T(x)) + \lambda \log \frac{\mu_X(x)}{\mu_Y(T(x)) \mid \det(\partial_x T(x)) \mid}$$

1.1. **Generalization to Fuzzy Transport.** The expected cost function above relies on the assumption that the transport map $T:X\to Y$ is given deterministically. A fruitful generalization of this construction is to assume that the morphism T is given as a conditional probability condition $\mu_Y(y\mid x)$. The conditional distribution satisfies the identity

$$\int_{Y} \mu_X(x) \mu_Y(y \mid x) dx.$$

On the other hand the conditional distribution $\mu_Y(y)$ is related to the joint distribution $\pi(x,y)$ as $\mu_Y(y\mid x)=\frac{\pi(x,y)}{\mu_X(x)}$. Therefore we obtain

$$\mu_Y(y) = \int_X \pi(x, y) \ dx$$

and similarly

$$\mu_X(x) = \int_Y \pi(x, y) \ dy$$

These imply that

$$\mu_Y(y \mid x) = \frac{\pi(x, y)}{\mu_X(x)} = \frac{\pi(x, y)}{\int_{Y} \pi(x, y) \ dy}$$
$$\mu_X(x \mid y) = \frac{\pi(x, y)}{\mu_Y(y)} = \frac{\pi(x, y)}{\int_{X} \pi(x, y) \ dx}$$

- 1.2. **Kantorovich Formulation.** Given two spaces X and Y with positive measures μ_X on X and μ_Y on Y, the Kantorovich formulation of the optimal transport asks to find a positive measure $\pi_{X,Y}$ on $X \times Y$, such that the following conditions are satisfied:
- 1. The pushforward of $\pi_{X,Y}$ to X is μ_X and the pushforward of $\pi_{X,Y}$ to Y is μ_Y . As the pushforward of the constructible measurable function $\pi_{X,Y}$ is the integral along the fibers of the projective morpshism $\pi_1: X \times Y \to X$ and $\pi_2: X \times Y \to Y$ respectively, one can say that the compatibility of measures induces the marginal probability conditions on X and Y respectively

$$\mu_Y(y) = \int_X \pi(x, y) \ dx$$

and similarly

$$\mu_X(x) = \int_Y \pi(x, y) \ dy.$$

The measure π_X minimizes the expected cost

(3)
$$\mathcal{L}(\pi_{X,Y}) = \int_{X \times Y} c(x,y) \pi_{X,Y}(x,y) \ dx \, dy$$

for some cost function $c: X \times Y \to [0, \infty]$. This is a convex optimization problem, known as the primal Kantorovich problem, which is easier to solve compared to the Monge formulation.

1.3. **Kantorovich Duality.** The primal Kantorovich problem (3) admits a dual formulation, known as the dual Kantorovich problem,

$$\max \mathcal{L}(\pi_{X,Y})[\phi,\psi] = \int_X \mu_X(x)\phi(x) + \int_Y \mu_Y(y)\psi(y)dy$$

$$\phi(x) + \psi(y) \le c(x,y)$$
(4)

1.4. Wasserstein Distance.

Definition 1.1. Given a metric space (X, g) the optimum value of the Kantorovich problem defines the Wasserstein distance

(5)
$$\mathcal{W}(\mu_X) = \inf_{\pi \in \Gamma(\mu(x), \mu(y))} \left(\int_{X \times X} \pi(x, y) d_g(x, y) dx \ dy \right)^{1/2}$$

where $d_g(x,y)$ is the g-distance between any two points $x,y \in X$, and $\Gamma(\mu(x),\mu(y))$ is the space of probability distributions π on $X \times X$ such that

$$\mu(x) = \int_{X} \pi(x, y) \ dy$$

and similarly

$$\mu(y) = \int_X \pi(x, y) \ dx.$$

Definition 1.2. Given a scalar density functional $F: dens(X) \to \mathbb{R}$, its Wasserstein gradient $\delta F[\mu]/\delta \mu$ is defined by

(6)
$$F[\mu + \delta\mu] = F[\mu] + \left\langle \frac{\delta F[\mu]}{\delta\mu}, \delta\mu \right\rangle_{\mathcal{W}} + O(\delta\mu^2)$$

where $\left\langle \frac{\delta F[\mu]}{\delta \mu}, \delta \mu \right\rangle_{\mathcal{W}}$ is the Wasserstein distance evaluated on the two infinitesimal vectors $\frac{\delta F[\mu]}{\delta \mu}, \delta \mu$ defined as sections of the tangent sheaf over dens(X), the space of probability distributions on X.

Given this definition one can formulate Wasserstein gradient of energy functionals over the space of probability distributions on X. Our aim here is to show that Wasserstein variation of such integrals induces the diffusion dynamics on space of probability distributions on X. That is to say that the stochastic diffusion equation can be realized as the gradient flow on Wasserstein space. First we define the energy functional, then the entropy and interactions functionals. The diffusion equations are obtained by computing the Wasserstein gradient of the sum of energy, entropy and interaction functionals.

Definition 1.3. *Define the energy functional*

(7)
$$E[\mu] = \int_X \mu(x)\epsilon(x)dx$$

where $\epsilon(x)$ denotes the energy of x. Its Wasserstein gradient is given by

(8)
$$\frac{\delta E[\mu]}{\delta \mu(x)} = -\partial_x \cdot g^{-1} \cdot (\mu(x)\partial_x \epsilon(x))$$

Now we define the entropy functional

Definition 1.4. *Define the entropy functional*

(9)
$$S[\mu] = -\int_{Y} \mu(x) \log(\mu(x)) dx$$

Definition 1.5. Consider Definition 1.3 and Definition 1.4. The free energy functional is defined as

(10)
$$F[\mu] = E[\mu] - TS[\mu] = \int_X (\mu(x)\epsilon(x) + T\log(\mu(x))) dx$$

Now we are ready to derive a diffusion equation as the gradient flow of the Free energy functional defined above.

Definition 1.6. Consider Definition 1.5. The Fokker-Plank diffusion relaxation equation is defined as the Wasserstein gradient flow applied to the free energy functional. That is

(11)
$$\frac{\delta F[\mu]}{\delta \mu(x,t)} = -T\partial_x \cdot g^{-1} \cdot \partial_x \mu(x,t) + \partial_x \cdot g^{-1} \cdot (\mu(x,t)\partial_x \epsilon(x))$$

where the first term describes the diffusion, and the second term describes the relaxation.

2. BENAMOU-BRENIER FORMULATION OF OPTIMAL TRANSPORTS AND DISCRETIZED DIFFUSION EQUATION

On a manifold X, define the Wasserstein distance between probability measures μ, ν as

(12)
$$W_2^2 = \inf_{\mu_t, \nu_t} \{ \int_0^1 || \nu_t ||_{L^2(\mu_t)} dt \}$$

where the infimum runsa over all 2-absolutely continuous curves $\mu_t, t \in [0, 1]$ in the L^2 -Wasserstein space, which satisfy the continuity equation

(13)
$$\frac{d\mu_t}{dt} + \nabla \cdot (\nu_t \mu_t) = 0.$$

Here μ_t satisfies the boundary condition $\mu_0 = \mu$ and $\mu_1 = \nu$. It must be noted that this formulation is arising in fluid dynamics, where in the original formulation ν is a vector field in X and $\int ||\nu_t||_{L^2(\mu_t)}$ is the total kinetic energy. Here on the space of Borel probability measures on X. Now the natural task to carry out for our applications is to discretize the Wasserstein space and obtain a discretized version of the Benamou-Brenier flow. To do this one can assume a triangulation of the space of probability distributions on X, that is dens(X), is given. In Appendix. C. We have provided a detailed construction of the graph $G(x_i, a_{ij}), i, j = 0, \cdots, m$ associated to dens(X) on which we discretize the Wasserstein distance function. Assume the graph $G(x_i, a_{ij}), i, j = 0, \cdots, m$ is given. We can associate a discretized "tangent sheaf" to the graph, by assigning vector fields μ_i, ν_i to the vertices, and $\mu_i, \nu_{i,j}, i, j = 1, n$ characterizing the amount transportation from node i to j over the edges. Using this construction the Wasserstein distance between two probability measures μ, ν distributed over the graph is given as

$$\langle \mu, \nu \rangle = \sum_{ij} W_{ij}(\mu_i, \nu_j) = \sum_{i,j} c_{ij}(\rho) \mu_{i,j} \nu_{ij},$$

where $c_{ij}(\rho)$ is the "conductance coefficient" associated to the density distribution ρ on edge e_{ij} with length a_{ij} . Similarly, one defines the discretized inner product one can define the discretized divergence of a probability vector field distribution over the graph as

(14)
$$\operatorname{div}_{\rho}(\nu) := \sum_{i \neq j} c_{ij}(\rho) \nu_{ji}$$

Now the dynamic equation induced by Benamou-Brenier formalism is obtained as

(15)
$$\frac{d\mu_t}{dt} = \operatorname{div}_{\rho^t}(\mu_t).$$

3. RG FLOW, AND ITS CONNECTION TO WASSERSTEIN GRADIENT FLOW AND DIFFUSION DYNAMICS

As described in Appendix. A. the exact renormalization group is the non-perturbative renormalization scheme that builds on optimal transport towards an uncorrelated Gaussian target distribution. The partition functional here is the sum of free energy action and the interaction action which can be written as

(16)
$$Z = \int \exp\left(-\frac{1}{2}x \cdot \Sigma^{-1}.x - S_{int}(x)\right)$$

where Σ is the covariance matrix between random variables x. The action $S(x) = S_{free}(x) + S_{int}(x)$ specifies a probability distribution of random variable x, that is

$$\mu(x) = \frac{1}{Z}e^{-S(x)},$$

with $Z = \int e^{-S(x)} dx$ the partition function defined above. Now under RG flow we obtain the differential equation [8, 19, 20]

(17)
$$\partial_t \mu(x) = -\frac{\partial_x \cdot \partial_t \Sigma \cdot \partial_x}{2} \mu(x) - \partial_x \cdot \partial_t \Sigma \cdot \left((\Sigma^{-1} \cdot x) \mu(x) \right)$$

It can be seen that Equation (17) leads to the diffusion dynamics equation induced by the Wasserstein gradient of the free energy functional we defined earlier. The trick is to define the metric used in the Fokker-Plank equation as

$$(18) g^{-1} = \frac{\partial_t \Sigma}{2}$$

which is a positive definite matrix. Using this definition the exact RG flow equation (17) can be re-written in terms of metric q as

(19)
$$\partial_t \mu(x) = \partial_x \cdot g^{-1} \cdot \partial_x \mu(x) + 2\partial_x \cdot g^{-1} \cdot (\mu(x)\partial_x S_{free}(x))$$

The RG scheme must be such a design so that

$$\partial_t \Sigma \leq 0$$

otherwise the metric g will not be positive semi-definite, and the RG flow might be unstable. This means that the covariance of random variable x must generally decrease under RG flow, which is compatible with how we defined it in earlier sections of the current article. Comparing Equation (19) to Equation (11), we see that the corresponding functional for RG flow, must be given as

(20)
$$F[\mu] = \int \mu(x)(2S_{free}(x) + \log \mu(x))dx$$

which up to a constant corresponds to the KL-divergence of the actual probability $\mu(x)$ with respect to the hypothetical probability distribution q(x) given by

$$q(x) = -\frac{e^{-2S_{free}(x)}}{2}.$$

The Equation (20) then clearly shows that the RG flow can be interpreted as the optimal transport gradient flow that minimizes the RG functional $F[\mu]$ along the Wasserstein geodesic of the probability distributions supported on the metric space equipped with the metric $g = -2(\partial_t \Sigma)^{-1}$.

4. DIFFUSION-BASED MODEL

4.1. **Forward Diffusion (Renormalization).** We use construction in Section 2. That is we realize the diffusion equation as the gradient flow associated with the Wasserstein distance function induced by Benamou-Brenier equations. The induced diffusion-based model is the generative model inspired by non-equilibrium thermal dynamics. A diffusion model gradually introduces random noise into data using a sequence of Markov chain steps and is then trained to reverse the process for image generation.

Let x_0 be the data and $x_{1:T}$ be a sequence of noisy samples. The forward diffusion is defined [13] by a Markov process

(21)
$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t} x_{t-1}, \beta_t \mathbb{I}),$$

where $\beta_{1:T}$ denote the noise variance schedule. Under reparameterization, with $\epsilon_{t|t-1} \sim \mathcal{N}(0,\mathbb{I})$ we obtain

$$(22) x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon_{t|t-1}.$$

The diffusion chain is generated auto-regressively such that

(23)
$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1}).$$

In fact, marginalization to any time step is tractable

$$q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) \mathbb{I}),$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$.

4.2. **Reverse Diffusion Process (Generation).** When conditioned on x_0 , the reverse diffusion is defined by the Markov process

(24)
$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \bar{\mu}(x_t, x_0), \bar{\beta}_t \mathbb{I}),$$

where we have

(25)
$$\bar{\mu}(x_t, x_0) = \frac{\sqrt{\alpha_t} (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1 - \bar{\alpha}_t} x_0, \quad \bar{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t.$$

The landmark study [13] demonstrated the capability of diffusion models to generate high-quality images, sparking significant interest in this area. Studies pertinent to our work can be grouped into two main categories: (1) studies aiming to provide alternative diffusion (corruption) processes to the standard Gaussian process [1, 7, 10, 22] and (2) studies that focus on enhancing the performance of diffusion models [23, 26, 27, 31, 32]. In the next section, we propose the Frequency Domain Diffusion Model (FDDM), which lies at the intersection of the two, as we achieve performance improvements under FDDM.

5. APPLICATIONS: FORWARD AND REVERSE DIFFUSION IN FOURIER SPACE

5.1. **Overview and Motivation.** In this section, we propose the Frequency Domain Diffusion Model (FDDM). FDDM is a novel diffusion-based approach for image generation that maps data between image and frequency domains using the discrete cosine transform (DCT) [24]. Unlike the existing works that operate in the image domain, FDDM performs diffusion-based generative modeling in the frequency domain. This model is motivated by the fact that many natural images have a sparse representation in the frequency domain, which makes it easier to separate signals from noise.

During the forward process, we add noise to the image in the frequency domain, with the diffusion coefficient being determined by a novel scale-dependent noise schedule. We define this schedule using a dispersion relation, which describes the energy associated with each frequency component. The dispersion relation determines how quickly information diffuses through the image. The sparsity in the frequency domain means that most of the energy in an image is concentrated in a small number of frequency components, while the remaining components have very low energy (see Figure 1). With this sparsity, we separate signal from noise more effectively than in image space. This is because the noise is spread out across all frequency components, while the signal is concentrated in a small number of components (unlike in the image domain).

Another advantage of the FDDM is that it can handle both large-scale (low frequency) and small-scale (high frequency) features (components) in an image. By separating these in the frequency domain, we apply different diffusion coefficients to each scale. We then leverage ideas from JPEG encoding [11, 28, 30] and apply FDDM on patches of images (see Figure 2), significantly increasing the training/inference speed (compared to image domain diffusion), making FDDM suitable for time-critical applications such as medical imaging where there is a need for rapid image generation [16].

In short, in this section (1) We introduce a novel diffusion model, FDDM, that operates in the frequency domain using a scale-dependent noise schedule. (2) We combine our process with ideas from JPEG encoding and frequency domain learning to refine the training protocol, thereby achieving improved speed in both training and inference, without a significant drop in generated image quality.

5.2. Fourier-Space Diffusion Model. We denote the data in the image space as x and in the frequency space as \tilde{x} such that $\tilde{x} = \mathcal{F}(x) \Leftrightarrow x = \mathcal{F}^{-1}(\tilde{x})$. The inverse frequency transformation \mathcal{F}^{-1} maps the data back to its original domain. In this work, we use the discrete cosine transformation (DCT) [24] to ensure that real data maps to real features. DCT is similar to the Fourier transformation, using real-valued cosine functions instead of complex exponentials. It is more suitable for processing real-valued data like images.

Large-scale (small-scale) features correspond to low-frequency (high-frequency) components in the frequency domain. By leveraging this separation, we apply different diffusion coefficients to each scale to improve the denoising performance. In particular, we use momentum coordinates in the

FIGURE 1. Scale-dependent noise schedule in the frequency domain (top) and the corresponding image domain representation, without patching, i.e., the whole image is a single patch. The frequency representation of the original (leftmost) image exhibits a sparsity, which we leverage in the proposed FDDM.

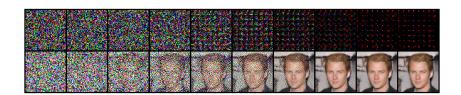


FIGURE 2. Image denoising and corresponding frequency domain representation for a 8 × 8 patched image. FDDM generates new samples by gradually denoising in the frequency domain.

frequency domain and label the components of the transformed data by their momentum k, defined as $k = (k_1, k_2)$, where k_1 and k_2 are wave numbers that coordinate the frequency domain. Then we apply different diffusion coefficients based on the location in the frequency domain.

5.2.1. Forward Diffusion in the Frequency Domain. Unlike in the image domain, as in (22), the forward diffusion process in FDDM gradually introduces noise from ultraviolet (UV), i.e., high frequency, to infrared (IR), i.e., low frequency, in the frequency domain (see Figure 1), using a stochastic process $\tilde{\boldsymbol{x}}_0 o \tilde{\boldsymbol{x}}_1 o \cdots o \tilde{\boldsymbol{x}}_t o \cdots$ defined by

$$\tilde{\boldsymbol{x}}_{t+1} = \sqrt{1 - \boldsymbol{\beta}_t} \odot \tilde{\boldsymbol{x}}_t + \sqrt{\boldsymbol{\beta}_t} \odot \boldsymbol{z}_t,$$

where we have $z_t \sim \mathcal{N}(0, I)$ and β_t is a scale-dependent diffusion coefficient that controls how much noise we introduce at each step. The symbol ⊙ denotes element-wise multiplication. Alternatively, the same process can be defined using an $\bar{\alpha}_t$ parameter that controls the SNR at each step:

(26)
$$\tilde{\boldsymbol{x}}_t = \sqrt{\bar{\boldsymbol{\alpha}}_t} \odot \tilde{\boldsymbol{x}}_0 + \sqrt{1 - \bar{\boldsymbol{\alpha}}_t} \odot \boldsymbol{z}_t,$$

where we have $z \sim \mathcal{N}(0, I)$. We specify the design of $\bar{\alpha}_t$ in Section 5.2.4, where we discuss the proposed scale-dependent noise schedule. The key idea of this noise schedule is to gradually add correlated noise to the image from small-scale to large-scale in the forward process. By controlling the diffusion coefficient, i.e., SNR, based on locations in the frequency space, we essentially apply different amounts of smoothing to different scales and locations in the frequency domain.

5.2.2. Backward Diffusion in the Frequency Domain. The backward diffusion process in FDDM learns to generate data from IR to UV in the frequency domain, using a noise prediction model ϕ_{θ} that takes the noisy frequency features \tilde{x}_t and predicts the clean image in the frequency space \tilde{x}_0 such that $\phi_{\theta}(\tilde{x}_t, t) \to \tilde{x}_0$. By utilizing the predicted clean image, we can perform the denoising operation. Following [26], we define the following relationship between \tilde{x}_{t-1} and \tilde{x}_t

(27)
$$\tilde{\boldsymbol{x}}_{t-1} = \underbrace{\sqrt{\bar{\boldsymbol{\alpha}}_{t-1}} \odot \boldsymbol{\phi}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_t, t) + \sqrt{1 - \bar{\boldsymbol{\alpha}}_{t-1} - \boldsymbol{\sigma}_t(\boldsymbol{\eta})} \odot \hat{\boldsymbol{z}}}_{\text{deterministic part}} + \underbrace{\boldsymbol{\sigma}_t(\boldsymbol{\eta}) \odot \boldsymbol{z}_t}_{\text{stochastic part}},$$

with
$$\sigma_t(\eta) = \eta \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}} \odot \sqrt{1-\frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}}$$
. The predicted noise is

(28)
$$\hat{z} = \frac{\tilde{x}_t - \phi_{\theta}(\tilde{x}_t, t)}{\sqrt{1 - \bar{\alpha}_t}}.$$

The backward diffusion process learns to generate cleaner data from noisy data in the frequency space by predicting the noise configuration and using it to recover the clean signal (see Figure 2). As in the forward process, in the backward process, we apply different amounts of smoothing to different scales and locations in the frequency space by controlling the diffusion coefficient (or SNR) based on momentum coordinates.

5.2.3. Objective Function and Training Approach. The objective function for training the noise prediction model ϕ_{θ} in FDDM is

(29)
$$\mathcal{L}_{\theta} = \underset{\boldsymbol{x}_0 \in \mathcal{D}}{\mathbb{E}} \underset{\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I)}{\mathbb{E}} \underset{t \sim \mathcal{U}(1, \dots, T)}{\mathbb{E}} \| (\tilde{\boldsymbol{x}}_0 - \boldsymbol{\phi}_{\theta}(\tilde{\boldsymbol{x}}_t, t)) \|^2,$$

where \mathcal{D} is the set of training images. x_0 is the image drawn from the training set, and \tilde{x}_t is the noisy frequency data obtained through the forward diffusion described by (26) given a noise configuration z. The objective function measures the mean squared error between the predicted clean image $\phi_{\theta}(\tilde{x}_t,t)$ and the true clean image \tilde{x}_0 over all training images. We note that the loss in (29) is computed in the frequency domain. Similarly, in FDDM, the forward and backward processes take place in the frequency domain.

We use Adam optimizer [17] to minimize the loss function in (29) by sampling a mini-batch of training images from \mathcal{D} and computing the gradients with respect to the parameters of the noise prediction model. Adam updates the parameters using a learning rate and adjusts the rate for each parameter based on the first and second moments of the gradients, improving the optimization's convergence properties. By updating the parameters iteratively over many epochs, we learn a noise prediction model that denoises images in the frequency domain.

5.2.4. Proposed Scale-Dependent Noise Schedule. The scale-dependent noise schedule in FDDM is a momentum-dependent function that controls the amount of noise introduced at each step of the forward process. We define it using a dispersion relation ϵ_k , which gives the energy associated with a frequency mode of momentum k. We use a tight-binding dispersion such that $\epsilon_k = -\cos \pi k_1$ $\cos \pi k_2$. In particular, the scale-dependent noise schedule is given by

(30)
$$\bar{\alpha}_{t,k} = \frac{1}{\exp\left(\frac{\epsilon_k - \mu_t}{T'}\right) + 1},$$

where μ_t is the Fermi level at time t, which controls the overall SNR. The noise schedule varies with momentum k, and can be packed into a vector $\bar{\alpha}_t = [\bar{\alpha}_{t,k}]_{k \in BZ}$, where BZ denotes the Brillouin zone. The Fermi level μ_t is expected to decrease with diffusion time t, such that $\bar{\alpha}_t$ decreases from 1 to 0 as more noise is introduced. This allows for a gradual introduction of noise from UV to IR in the frequency space. The temperature parameter T' controls how sharp or smooth this transition is, with lower values of T' leading to sharper transitions. With this scale-dependent noise schedule, we essentially apply different amounts of smoothing to different scales and locations in the frequency space. This allows for effective denoising while preserving important features in images.

5.3. **Description of FDDM.** FDDM consists of two algorithms: frequency-based forward process and sampling, which are given in Algorithms 1 and 2, respectively. Frequency-based forward process includes pre-processing, forward process, and training steps.

Frequency-Based Forward Process. We first sample an original image from a set \mathcal{D} of training images. This sample image is initially in the RGB format. Following the standard practice of JPEG encoding [30], we first convert it to YCbCr format and split it into patches to obtain enhanced training and inference speed. We note that this patch-based forward process is enabled by the FDDM, as it performs diffusion in the frequency domain. Next, we take the DCT of each patch separately. We

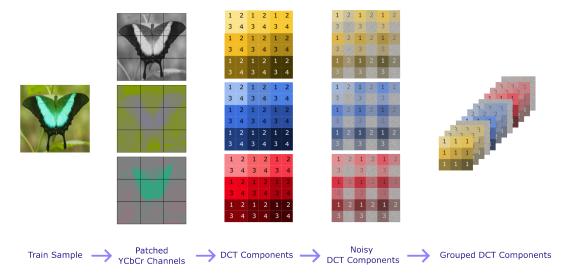


FIGURE 3. The proposed frequency-based forward process. We split YCbCr input image channels into patches and take the DCT of each patch. Then, we add the scaledependent noise to each patch in the frequency domain. Finally, we group the DCT components into output channels, with each channel containing components of the same frequency. We feed these output channels, i.e., grouped DCT components to the diffusion model during training.

then sample a timestep uniformly for the image and corrupt this image based on the forward process defined in (26). In particular, we apply the scale-dependent noise to each patch independently. Once the noise is applied, we obtain the noisy DCT components for each patch. The final step of the preprocessing is to group the components from the same frequency across patches into separate channels. In our design, this process is performed for a batch of images and we feed the associated uniformly sampled timestep of each image of the batch, i.e., noise level, to the neural network, together with the corrupted (and grouped) DCT components, i.e., channels. Finally, we take a gradient descent step using the L2 loss between the actual input image in the frequency domain and the predicted clean image in the frequency domain. This entire patchifying and forward process pipeline is demonstrated in Figure 3 and Algorithm 1 (which does not explicitly state the grouping of the noisy DCT components for ease of exposition).

We let n denote the size of the images, i.e., input images are of size $n \times n$. Assume the patching operation is performed with a patch size of $d \times d$. Then, the resulting patchified image has a total of $P = \left(\frac{n}{d}\right)^2$ patches. We note that this operation is repeated for each input channel (YCbCr images have 3 channels). In the example in Figure 3, the image size is n=6 and the patch size is d=2. Thus, we have a total of P = 9 patches in each image channel and the resulting tensor is of size $12 \times 3 \times 3$, where 12 is the number of output channels and 3×3 is the size of each output channel. We note that in the standard JPEG convention [28, 30], patches are of size 8×8 . Also, we note that the patch size directly determines the number of DCT components we have in each patch which is $d^2 = 4$, as also demonstrated in Figure 3.

Sampling (Inference). Sampling is essentially the backward process (denoising). We start by sampling pure noise in the frequency domain and gradually denoise it using the trained model, following (27). Algorithm 2 shows the denoising operation, which is also shown in Figure 2. We note that, as in the forward process, we work on patches in the backward process. That is, we perform the denoising operation described by (27) on the grouped DCT components, which are omitted in Algorithm 2 for ease of exposition. In particular, using the model prediction at timestep t_i , $\phi_{\theta}(\tilde{x}_{t_i}, t_i)$, we recover the less noisy DCT components from the previous timestep t_{i-1} , and these steps are repeated until timestep t_1 . Once the denoising is completed, we take the inverse DCT of the patches, combine the results (unpatch the image), and return the generated image. Here, t_i denotes the inference time steps

Algorithm 1 Frequency-Based Forward Process

```
Require: Input distribution D, # of training timesteps T, patch size d, image size n
   1: P \leftarrow (n/d)^2
                                                                                                                                                                            > Total number of patches
   2: repeat
                  oldsymbol{x}_0 \sim \mathcal{D}
   3:
                  t \sim \mathcal{U}(1,\ldots,T)
  4:
                  Split x_0 into d \times d patches \{x_0^p\}_{p \in \{1,\dots,P\}}
   5:
                  for p \in \{1, ..., P\} do
   6:
                           z \leftarrow \mathcal{N}(\mathbf{0}, I_d)
   7:
                          egin{aligned} & \tilde{oldsymbol{x}}_0^p \leftarrow \mathcal{F}(oldsymbol{x}_0^p) \ & \tilde{oldsymbol{x}}_t^p \leftarrow \sqrt{ar{oldsymbol{lpha}}_t} \odot 	ilde{oldsymbol{x}}_0^p + \sqrt{1-ar{oldsymbol{lpha}}_t} \odot oldsymbol{z} \end{aligned}
   8:
  9:
 10:
                  \tilde{oldsymbol{x}}_0 \leftarrow [\tilde{oldsymbol{x}}_0^0, \tilde{oldsymbol{x}}_0^1, \dots, \tilde{oldsymbol{x}}_0^P]
 11:
                                                                                                                                                                                    	ilde{oldsymbol{x}}_t \leftarrow [	ilde{oldsymbol{x}}_t^0, 	ilde{oldsymbol{x}}_t^1, \dots, 	ilde{oldsymbol{x}}_t^P]
12:
                  Take gradient descent step on \nabla_{\theta} \|\tilde{\boldsymbol{x}}_0 - \boldsymbol{\phi}_{\theta}(\tilde{\boldsymbol{x}}_t, t)\|^2
13:
14: until converged
```

Algorithm 2 Sampling (Inference)

```
Require: Inference time steps \{t_i\}_{i\in\{1,\dots,I\}}, total # of inference steps I, patch size d, image size n, \eta
    1: P \leftarrow (n/d)^2
   2: \{\tilde{\boldsymbol{x}}_0^p \leftarrow \mathcal{N}(\boldsymbol{0}, \boldsymbol{I_d})\}_{p \in \{1,\dots,P\}}
   3: \tilde{\boldsymbol{x}}_{t_I} \leftarrow [\tilde{\boldsymbol{x}}_{t_I}^0, \tilde{\boldsymbol{x}}_{t_I}^1, \dots, \tilde{\boldsymbol{x}}_{t_I}^P]
   4: for i = I downto 1 do
                        [\tilde{\boldsymbol{x}}_0^0, \tilde{\boldsymbol{x}}_0^1, \dots, \tilde{\boldsymbol{x}}_0^P] \leftarrow \boldsymbol{\phi}_{\theta}(\tilde{\boldsymbol{x}}_{t_i}, t_i)
    5:
                        if i > 1 then
    6:
                                    for p \in \{1, ..., P\} do
    7:
                                               oldsymbol{z} \leftarrow \mathcal{N}(oldsymbol{0}, oldsymbol{I_d})
    8:
                                              \boldsymbol{\sigma}_{t_i}(\eta) = \eta \sqrt{\frac{1 - \bar{\boldsymbol{\alpha}}_{t_{i-1}}}{1 - \bar{\boldsymbol{\alpha}}_{t_i}}} \odot \sqrt{1 - \frac{\bar{\boldsymbol{\alpha}}_{t_i}}{\bar{\boldsymbol{\alpha}}_{t_{i-1}}}}
    9:
                                              \hat{oldsymbol{z}} = rac{	ilde{oldsymbol{x}}_{t_i}^p - 	ilde{oldsymbol{x}}_0^p}{\sqrt{1 - ar{oldsymbol{lpha}}_t}} \ oldsymbol{x}_{t_{i-1}}^p = \sqrt{ar{oldsymbol{lpha}}_{t_{i-1}}} \odot 	ilde{oldsymbol{x}}_0^p + oldsymbol{\sigma}_{t_i}(\eta) \odot oldsymbol{z}
10:
11:
                                                                          +\sqrt{1-ar{oldsymbol{lpha}}_{t_{i-1}}-oldsymbol{\sigma}_{t_i}(\eta)}\odotoldsymbol{\hat{z}}
                                    end for
12:
13:
                        	ilde{oldsymbol{x}}_{t_{i-1}} \leftarrow [	ilde{oldsymbol{x}}_{t_{i-1}}^0, 	ilde{oldsymbol{x}}_{t_{i-1}}^1, \dots, 	ilde{oldsymbol{x}}_{t_{i-1}}^P]
15: end for
16: \{ \boldsymbol{x}_0^p \leftarrow \mathcal{F}^{-1}(\tilde{\boldsymbol{x}}_0^p) \}_{p \in \{1,\dots,P\}}
17: return combined (unpatched) [\boldsymbol{x}_0^0, \boldsymbol{x}_0^1, \dots, \boldsymbol{x}_0^P]
```

for $i \in \{1, ..., I\}$, with I being the total number of inference steps. Unlike the forward process, where the timesteps are consecutive, in the backward process, we can sparsely sample. That is, t_i and t_{i-1} are not necessarily consecutive for any $i \in \{1, ..., I\}$. Ideally, the resulting generated image should look as if it is from the original dataset.

5.4. **Experiment Results.** The Fermi level μ_t is linearly decreasing over time t from 4 to -4. The momentum coordinates in the frequency domain $\mathbf{k} = (k_1, k_2)$ take values between 0 and 1 with linear steps, where the step size is determined by the input image size. For example, for an image from the CelebA-64 [18] dataset of size 64×64 , the precision of momentum coordinate increments is $\frac{1}{64}$.



FIGURE 4. Uncurated generated samples on Fashion-MNIST.



FIGURE 5. Uncurated samples on CelebA-64 dataset.



FIGURE 6. Curated samples on CelebA-64 dataset.

By using these momentum coordinates, we effectively vary the applied noise intensity on the input image. We set the temperature parameter T' to 0.5.

In their seminal diffusion work in [13], authors utilize a U-Net structure based on ResNet blocks with downsampling followed by an upsampling process and as well as an attention module in between the convolutional blocks. Inspired by [30], we modify the U-Net structure in [13] to match the output of our frequency-based forward process. First, we change the number of input channels from 3 (RGB channels) to $3 \times d \times d$ (number of frequency channels). Second, we keep the resolution constant throughout the model. Since the neural network parameters are shared across time, we use sinusoidal position embedding to encode time so that the neural network knows at which noise level it operates during the denoising process. In addition, since in FDDM the noise is scale-dependent, by inputting the time, we implicitly feed the noise-scale to the neural network as well.

To report the performance of the proposed FDDM, we perform experiments on Fashion-MNIST [29] (resolution 28×28) and CelebA-64 (resolution 64×64) [18]. We configure the forward and reverse process using $\eta = 1$ and T = I = 1000. We use a batch size of 128 and train for a total of 800k steps. We use the Adam optimizer in training, with the learning rate set to 2×10^{-4} , without any sweeping. We carry out the experiments using NVIDIA A100 40GB.

We first present a set of uncurated samples generated by FDDM, using Fashion-MNIST with patch size d = 7 in Figure 4, demonstrating its image generation capabilities in the frequency domain. Next, we consider a more realistic CelebA-64 dataset and present the curated and uncurated generated

samples in Figs. 5 and 6 for d=4. Steps of the image denoising process and the corresponding frequency domain representation are as in Figure 2, showcasing denoising in UV and IR parts as a function of timesteps. Overall, these results for both datasets demonstrate that the proposed FDDM successfully generates compelling images.

Next, we compare the performance of FDDM with the seminal image domain diffusion approach of [13] named DDPM, as the FDDM architecture is based on DDPM and our goal is to offer a performance-utility trade-off for this design. For comparisons, we use MACs (Multiply-accumulate operations) [31], number of inference steps per second, and Fréchet Inception Distance (FID) [12] as our metrics. FID measures the similarity between two sets of images and is shown the correlate with human judgement. In general, a low FID score indicates a good generated sample quality.

First, we take a look at the number of inference steps per second and MACs of the two schemes. Results in Table 1 show that inference in our proposed FDDM is significantly faster than DDPM (around 2.7 to $8.5 \times$ faster). This is attributed to our method of patching and forming frequencybased channels (as explained in section 5.3), which effectively reduces the computations in the U-Net. The performance enhancement is evident from the MACs column (lower the better) in Table 1, demonstrating that our model requires significantly fewer MACs to generate 128 samples.

We compute the FID scores for the CelebA-64 dataset using 200000 samples and show the results in Table 1. As expected, our FID scores are higher than the baseline, considering our faster and more time-efficient design. These results indicate a possible trade-off between the sample quality and training runtime efficiency. One observation is that lower FID scores around 3 are achieved when the finer details of the images are present. This means that our FID score around 18 does not mean $6\times$ worse images, and in fact the images generated by the proposed FDDM may be suitable for certain downstream tasks, for which finer details are not as critical. For example, the images we generate in Figure 6 can be part of a synthetic dataset for training a classifier model that groups people according to their certain physical characteristics, e.g., hair color, glasses vs no glasses, and so on). For this task, finer details such as the background objects may not be as critical.

| Model | MACs (G) ↓ | # inference steps/s ↑ | FID score ↓ |
|------------------|------------|-----------------------|-------------|
| DDPM [13] | 3099 | 5.344 | 3.26 |
| Ours (4x4 patch) | 1781 | 14.655 | 18.17 |
| Ours (8x8 patch) | 449 | 45.76 | 20.65 |

TABLE 1. Performance comparison of the proposed FDDM with DDPM [13] for T =I = 1000 on CelebA-64.

APPENDIX A. DISCRETIZATION OF WASSERSTEIN DISTANCE FUNCTION

We would now elaborate on construction of a graph $G(x_i, a_{ij}), i, j = 0, \dots, m$ associated to dens(X) which induces the discretized Wasserstein distance function, this will lead to derivation of a discretized Benamou-Brenier dynamics equation on space of probability distributions on X.

We would first define the notion of Ricci curvature associated to a discrete graph induced by a lattice. Consider the discrete lattice \mathbb{Z}^n . Fix a point $x_0 \in \mathbb{Z}^n$ with coordinates $x := (t_1, \dots, t_n), t_i \in$ \mathbb{Z} . Now choose a number ϵ and consider n-dimensional balls $B_{x_0}(\epsilon)$ defined as set of points $x \in \mathbb{Z}^n$ such that $d(x,x_0) < 1 + \epsilon$ where $d(x,x_0)$ is the distance function with respect to the Eculidean metric in \mathbb{Z}^n . The latter inequality picks out a set of points in the lattice in ϵ -neighborhood of the point x_0 . We connect them to x_0 and label them as x_i , $i = 1, 2, \dots$, assuming that x_1 satisfies closest Euclidean distance to x_0 . Now we repeat the same procedure for points x_i , $i = 1, \dots$. Let us assume that, via iterating this procedure we obtain a graph $G(x_i, a_{ij}), i, j = 0, \dots, m$ with the central point x_0 as shown in Figure 7.

Now pick a vertex x_i in the graph and an edge with length a_{ij} connecting x_i and x_j for some $j \neq i$. We define the Wasserstein transport distance function induced by defining an optimal transport

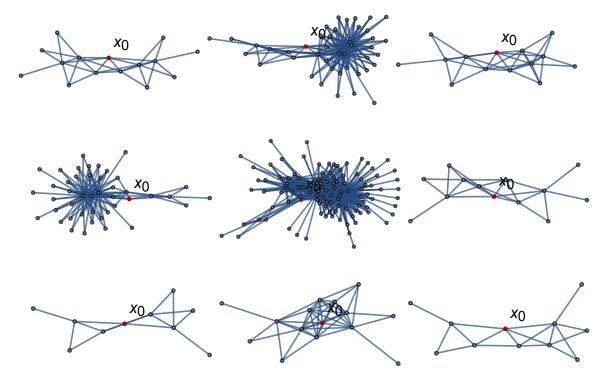


FIGURE 7. Samples of the local graph around a randomly picked point x_0 .

problem associated to this graph. We decorate the vertices of the graph with optimal transport weights as follows: We decorate x_i, x_j with $\epsilon, 1 - \epsilon$. In other words we associate to the vertex x_i a probability distribution $\Psi_{x_i}(x,\epsilon)$ with most of its weight at $x=x_i$ and a small amount of weight at neighboring vertices, so that the average distance from x_i to a vertex chosen from this distribution is much less than an edge length. Here the probability distribution function is defined by

$$\Psi_{x_i}(x,\epsilon) = \begin{cases} 1 - \frac{d_J(x_i)}{D_{x_i}} \epsilon & \text{if } x = x_i \\ \frac{J_{x_i x}}{D_{x_i}} \epsilon & \text{if } x \simeq x_i \\ 0 & \text{Otherwise,} \end{cases}$$

where $d_J(x_i) = \sum_{x \simeq x_i} J_{x_i x}$, D_{x_i} is the lapse function which determines how fast ϵ runs in different locations of the graph, and $J_{xy}=\frac{1}{a_{xy}^2}$. Now simplifying this for our situation, for $x_k, k\neq i,j$ we assign probability weight functions $\epsilon_{k|_i} := \Psi_{x_i}(x_k, \epsilon)$ which are given by

$$\epsilon_{k|_i} := \epsilon \cdot \frac{\left(\frac{1}{a_{ki}^2}\right)}{\sum_{m \neq i} \frac{1}{a_{im}^2}},$$

where a_{ki} are defined as the edge lengths obtained in the previous step. Now we define the Wasserstein transport distance function associated to the edge a_{ij}

(31)
$$W_{ij} = (1 - \epsilon - \epsilon_j) + \sum_{k \neq j,i} \epsilon_k \frac{(a_{ik} + a_{ij})}{a_{ij}}$$

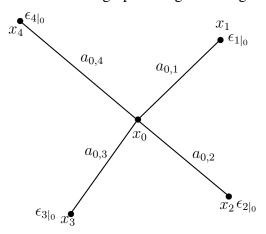
Now the Ricci curvature of the edge d_{ij} is defined by

$$(32) k_{ij} := \lim_{\epsilon \to 0} \frac{1 - W_{ij}}{\epsilon}.$$

Next one can calculate the Ricci curvature of the manifold passing through point x_i in ball of radius $1 + \epsilon$ centered at x_0 . The local curvature of the manifold is defined as the product of the maximum and minimum values attained by k_{ij} for all x_j adjacent to the vertex x_i .

Once the curvature is obtained the manifold can be constructed by gluing local patches with the given curvature.

Example: Let us assume that the obtained graph is as given in figure below.



Let us compute the curvature of $E_{0,1}$ which connects x_0 and x_1 . We associate to x_0, x_1 the probability weight function $\epsilon, 1 - \epsilon$ respectively. Now use the identity above the values of $\epsilon_2, \epsilon_3, \epsilon_4$ are given as

$$\epsilon_{0} = 1 - \epsilon$$

$$\epsilon_{1|_{0}} = \epsilon$$

$$\epsilon_{2|_{0}} = \epsilon \cdot \frac{\frac{1}{a_{0,2}^{2}}}{\frac{1}{a_{0,2}^{2}} + \frac{1}{a_{0,3}^{2}} + \frac{1}{a_{0,4}^{2}}}$$

$$\epsilon_{3|_{0}} = \epsilon \cdot \frac{\frac{1}{a_{0,2}^{2}}}{\frac{1}{a_{0,2}^{2}} + \frac{1}{a_{0,3}^{2}} + \frac{1}{a_{0,4}^{2}}}$$

$$\epsilon_{4|_{0}} = \epsilon \cdot \frac{\frac{1}{a_{0,4}^{2}}}{\frac{1}{a_{0,4}^{2}} + \frac{1}{a_{0,4}^{2}} + \frac{1}{a_{0,4}^{2}}}$$
(33)

Now using (31), we compute the Wasserstein transport distance function associated to the edge E_{01} is given by

$$(34) W_{01} = (1 - \epsilon - \epsilon_{1|0}) + \epsilon_{2|0} \frac{(a_{0,2} + a_{0,1})}{a_{0,1}} + \epsilon_{3|0} \frac{(a_{0,3} + a_{0,1})}{a_{0,1}} + \epsilon_{4|0} \frac{(a_{0,4} + a_{0,1})}{a_{0,1}}$$

Which can then induce the curvature of $E_{0,1}$ using (32). When applying our formalism to Benamou-Brenier formalism we obtain the following. Consider the metric graph $G(x_i, a_{ij}), i, j = 0, \cdots, m$ constructed over dens(X) with probability measures μ and ν .

REFERENCES

- [1] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. arXiv preprint arXiv:2208.09392, 2022.
- Deep learning and the renormalization group. [2] Cédric Bény. arXiv e-prints, page arXiv:1301.3124, January 2013.

- [3] Cédric Bény and Tobias J. Osborne. The renormalization group via statistical inference. New Journal of Physics, 17(8):083005, August 2015.
- [4] David S. Berman, Jonathan J. Heckman, and Marc Klinger. On the Dynamics of Inference and Learning. arXiv e-prints, page arXiv:2204.12939, April 2022.
- [5] David S. Berman and Marc S. Klinger. The Inverse of Exact Renormalization Group Flows as Statistical Inference. arXiv e-prints, page arXiv:2212.11379, December 2022.
- [6] David S. Berman, Marc S. Klinger, and Alexander G. Stapleton. Bayesian renormalization. *Machine Learning: Science and Technology*, 4(4):045011, December 2023.
- [7] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12413-12422, 2022.
- [8] Jordan Cotler and Semon Rezchikov. Renormalization group flow as optimal transport. *Physical* Review D, 108(2):025003, July 2023.
- [9] Jordan Cotler and Semon Rezchikov. Renormalizing Diffusion Models. arXiv e-prints, page arXiv:2308.12355, August 2023.
- [10] Giannis Daras, Mauricio Delbracio, Hossein Talebi, Alex Dimakis, and Peyman Milanfar. Soft diffusion: Score matching with general corruptions. Transactions on Machine Learning Research, 2023.
- [11] Lionel Gueguen, Alex Sergeev, Ben Kadlec, Rosanne Liu, and Jason Yosinski. Faster neural networks straight from jpeg. Advances in Neural Information Processing Systems, 31, 2018.
- [12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In Advances in Neural Information Processing Systems, volume 30, 2017.
- [13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems, 33:6840–6851, 2020.
- [14] Wanda Hou and Yi-Zhuang You. Machine Learning Renormalization Group for Statistical Physics. arXiv e-prints, page arXiv:2306.11054, June 2023.
- [15] Hong-Ye Hu, Shuo-Hui Li, Lei Wang, and Yi-Zhuang You. Machine learning holographic mapping by neural network renormalization group. Phys. Rev. Res., 2:023369, Jun 2020.
- [16] Amirhossein Kazerouni, Ehsan Khodapanah Aghdam, Moein Heidari, Reza Azad, Mohsen Fayyaz, Ilker Hacihaliloglu, and Dorit Merhof. Diffusion models in medical imaging: A comprehensive survey. Medical Image Analysis, page 102846, 2023.
- [17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
- [18] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), December 2015.
- [19] Han Ma and Sung-Sik Lee. Constraints on beta functions in field theories. arXiv e-prints, page arXiv:2009.11880, September 2020.
- [20] Masami Matsumoto, Gota Tanaka, and Asato Tsuchiya. Renormalization group and diffusion equation. arXiv e-prints, page arXiv:2011.14687, November 2020.
- [21] Pankaj Mehta and David J. Schwab. An exact mapping between the Variational Renormalization Group and Deep Learning. arXiv e-prints, page arXiv:1410.3831, October 2014.
- [22] Eliya Nachmani, Robin San Roman, and Lior Wolf. Non gaussian denoising diffusion models. arXiv preprint arXiv:2106.07582, 2021.
- [23] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In International Conference on Machine Learning, pages 8162–8171. PMLR, 2021.
- [24] K Ramamohan Rao and Ping Yip. Discrete cosine transform: algorithms, advantages, applications. Academic press, 2014.
- [25] Artan Sheshmani, Yi-Zhuang You, Wenbo Fu, and Ahmadreza Azizi. Categorical representation learning and rg flow operators for algorithmic classifiers. Machine Learning: Science and

- Technology, 4(1):015012, feb 2023.
- [26] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.
- [27] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. Advances in Neural Information Processing Systems, 33:12438–12448, 2020.
- [28] Gregory K. Wallace. The JPEG still picture compression standard. Communications of the ACM, 34(4):30–44, 1991.
- [29] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747, 2017.
- [30] Kai Xu, Minghai Qin, Fei Sun, Yuhao Wang, Yen-Kuang Chen, and Fengbo Ren. Learning in the frequency domain. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1740–1749, 2020.
- [31] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 22552-22562, 2023.
- [32] Oinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. arXiv preprint arXiv:2204.13902, 2022.
- artan@mit.edu, yzyou@physics.ucsd.edu, buyukate@usc.edu, ziashaha@usc.edu, avestime@usc.edu
- $^{
 m 1}$ MIT, Institute for Artificial Intelligence and Fundamental Interactions, 77 Mass Ave, Cam-**BRIDGE, MA 02138**
- ² Beijing Institute of Mathematical Sciences and Applications, No. 544, Hefangkou Village, Huaibei Town, Huairou District, Beijing 101408
 - 3 Harvard University, Jefferson Laboratory, 17 Oxford St, Cambridge, MA 02138 $^{\prime}$
- ⁴ UNIVERSITY OF CALIFORNIA SAN DIEGO, DEPARTMENT OF PHYSICS, CONDENSED MATTER GROUP, UC SAN DIEGO 9500 GILMAN DR. LA JOLLA, CA 92093
- 5 University of Southern California, Department of Electrical and Computer Engineering, Los ANGELES, CA 90089