

Beyond Blur: A Fluid Perspective on Generative Diffusion Models

Grzegorz Gruszczyński^{1,2}

Jakub Meixner^{1,3}

Michał Włodarczyk^{1,4}

Przemysław Musalski^{1,5,6}

¹IDEAS NCBR ²Samsung AI Center Warsaw ³Polish Academy of Sciences

⁴Warsaw University of Technology ⁵IDEAS Research Institute ⁶New Jersey Institute of Technology

g.gruszczyns@samsung.com, {kubameixner, mwłodarz}{gmail.com, przem@njit.edu

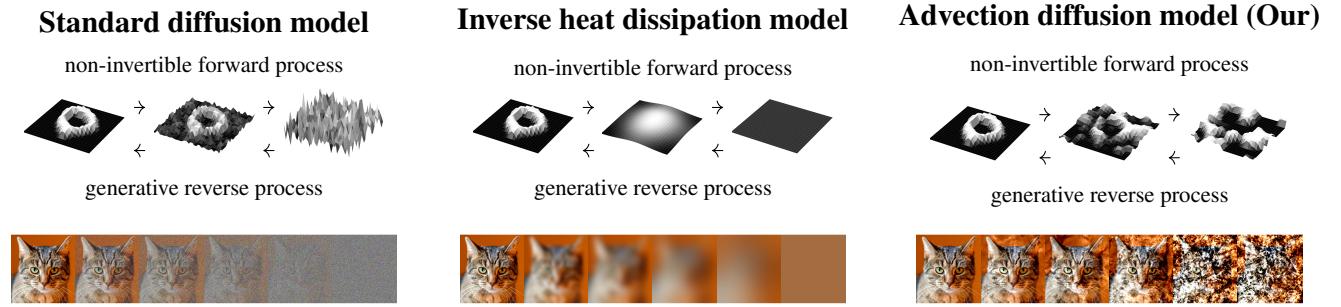


Figure 1. The standard diffusion model (DDPM, left) induces Gaussian noise for image corruption, inverse heat dissipation blurs the image using the heat equation (middle), and our proposed advection-diffusion method adds both blur and translation of pixels (right).

Abstract

We propose a novel, Partial Differential Equation (PDE)-driven, corruption process for generative image synthesis which generalizes existing PDE-based approaches. Our forward pass formulates image corruption via a physically motivated PDE that couples directional advection with isotropic diffusion and Gaussian noise, controlled by dimensionless numbers. We solve this PDE numerically through a GPU-accelerated Lattice Boltzmann solver for fast evaluation. To induce realistic “turbulence,” we generate stochastic velocity fields that introduce coherent motion and capture multi-scale mixing. In the generative process, a neural network learns to reverse the advection-diffusion operator thus constituting a novel generative model. We discuss how previous methods emerge as specific cases of our operator, demonstrating that our framework generalizes prior PDE-based corruption techniques. We illustrate how advection improves the diversity and quality of the generated images while keeping the overall color palette unaffected. This work bridges fluid dynamics, dimensionless PDE theory, and deep generative modeling, offering a fresh perspective on physically based inverse problems.

1. Introduction

Denoising probabilistic diffusion-based generative models have made striking strides in recent years, demonstrating high-quality image synthesis through iterative noise addition and subsequent denoising [7, 11, 33, 36].

A distinct branch of works focuses on introducing different image corruption processes, such as Cold Diffusion [2], Soft Diffusion [6], or more physically inspired processes, such as Inverse Heat Dissipation [27] or Blurring Diffusion [12]. The idea in these works is to replace or augment the pure Gaussian noise with other mechanisms aimed at better preserving color budgets, multi-scale detail, or interpretability. A notable subfamily are the PDE-based methods, which model the image frequencies explicitly and thus deliver a multi-scale perspective with clear frequency-domain interpretation. Nevertheless, previous PDE approaches remain purely *isotropic*, ignoring potentially compelling directional flows.

We introduce the advection–diffusion corruption processes, which allows to unlock *anisotropic* patterns of texture shifts and swirling motions in the forward corruption process. This forward operator is physically well-grounded and not covered by earlier works.

Our main contribution is hence an *Advection-Diffusion-Reaction Probabilistic Model* which goes *beyond blur*. It integrates not only a random reaction terms (Gaussian noise) and blurring (averaging) but also shift (advection) terms, generalizing the previous works into a common framework. It is inspired by fluid dynamics and it allows to unlock structured flows. The major advantage over previous methods lies in its physical grounding and its property to enable coherent texture shifts in the forward corruption trajectory that isotropic blur cannot represent.

To implement the aforementioned advection-diffusion operator on typical datasets efficiently, we propose a scalable GPU-based Lattice Boltzmann Method (LBM) solver, an established fluid simulation technique [19]. We also introduce a dimensionless formulation of the training process (using similarity numbers) and show that the intensity of the physical process can be easily transferred between images of different resolutions.

2. Related Work

The last few years have witnessed tremendous progress in diffusion-based generative methods for image generation and beyond. Several comprehensive surveys [4, 39] consolidate the growing literature. Building on early variants of *Denoising Diffusion Probabilistic Models* (DDPM) [11, 33], the diffusion processes have been generalized to continuous-time stochastic differential equations [36], revealing unifying insights across iterative noising–denoising paradigms. Notably, the notion of learning gradients of the data distribution can be traced back to Song and Ermon [35], which laid the foundation for modern score-based generative modeling. Subsequent refinements have focused on *improved sampling* [14, 25, 31, 34], *better architectures* [15], *improved training dynamics* [16], and *conditional guidance* [10], often yielding state-of-the-art image quality [7]. These advances have extended beyond unconditional image synthesis to tasks such as *super-resolution* [30] and *time-series imputation* [37], showcasing the versatility of diffusion approaches. A variational perspective on diffusion complements score-based methods [17], and *Latent Diffusion Models* [28] further scale to high-resolution generation by introducing a compressed latent space, improving computational efficiency and memory usage.

Recently, methods which introduce alternative processes for the corruption of the input image arised. Particularly, *Inverse Heat Dissipation* (IHD) introduced by Rissanen et al. [27] replaces the conventional Gaussian-based forward corruption with a heat-equation blur with minor additive noise, thereby offering a physically grounded coarse-to-fine scheme. Follow-up work has been done by Hoogeboom and Salimans [12], who combined blurring with a growing amount of Gaussian noise controlled by an appropriate scheduler [5]. On the other hand, Huang et al. [14] intro-

duced a blue noise characterized by a prescribed energy spectrum for the corruption process. Our approach generalizes this PDE-driven philosophy by infusing a velocity-driven *advection* term into the forward process.

In a so-called *cold diffusion*, Bansal et al. [2] proposed a sampling method that is able to invert a fully deterministic (without noise) degradation process. They introduced arbitrary deterministic operators (e.g. blur, masking) without noise, and learned to invert them without imposing a strict physical PDE. Daras et al. [6] presented a *soft score-matching* loss function and a momentum sampler with application to blurring with a limited amount of noise. Their model was trained to predict a clean image that would resemble the one after corruption.

In terms of computational fluid dynamics, the *Lattice Boltzmann Method* (LBM) have garnered growing attention [1, 3, 20, 21] due to its highly local computations suitable for GPU-based acceleration. Because LBM naturally solves advection-diffusion type equations, it has been chosen for our goal of embedding a velocity-dependent PDE in the forward corruption pass. Finally, the data-driven PDE solvers have exploited deep learning to accelerate numerical simulations [18, 22, 26].

3. Background on Diffusion Models

In diffusion-based generative models (e.g., DDPM), the pixel intensities are progressively corrupted by noise, which can be viewed in physical terms as a *random reaction* process. In contrast, recent works highlight that image intensities may also be *transformed* via partial differential equations (PDEs) before adding only marginal noise [6, 27]. Below, we first revisit the classic noise-driven diffusion formulation and then generalize to the PDE-based diffusion process.

3.1. Probabilistic Diffusion Models

Classical diffusion models [11, 33] treat the forward noising as a *Markov chain* $q(u_k | u_{k-1})$ that progressively corrupts an initial data sample u_0 . Interpreted in continuous time, the forward process can be seen as a *stochastic differential equation* (SDE),

$$du(t) = f(u(t), t) dt + g(t) dw(t), \quad (1)$$

where $w(t)$ is Brownian motion and $g(t)$ regulates the noise injection. The drift term is denoted with $f(u(t), t)$. The distribution of $u(t)$ then follows the *Fokker–Planck equation*:

$$\frac{\partial p(u, t)}{\partial t} = -\nabla_u \cdot (f(u, t) p(u, t)) + \frac{1}{2} \nabla_u^2 (g^2(t) p(u, t)). \quad (2)$$

In more intuitive terms, as t increases, $u(t)$ becomes increasingly randomized (e.g., converging to an isotropic Gaussian), and a learned *reverse process* iteratively “denoises”



Figure 2. Example of the corruption and generative process of our method, illustrated over 11 sequential frames in a chain. $\sigma = 20$, $Pe = 0.6$. Please refer to Section 4 for details.

from that noise-prior back to the data manifold. Such purely noise-driven approaches can be viewed physically as a *local “reaction”* term affecting each pixel independently. However, this local scalar reaction may overlook spatial coherence.

3.2. PDE-Based Diffusion Models

According to terminology coined in [36], the evolution of moments of the corrupted prior distribution can be described as either being Variance Preserving, $\text{Var}[p(u_t)] = I$, or Variance Exploding, $\text{Var}[p(u_t)] \rightarrow \infty$. At the same time, the mean of the prior distribution will be either shrinking, $\mathbb{E}[p(u_t)] \rightarrow 0$, or constant $\mathbb{E}[p(u_t)] = \mathbb{E}[p(u_0)]$. These properties arise from the interplay between the drift term and Gaussian noise injection in Eq. (1). When using a conservative PDE, like *blurring*, the forward corruption process by *design* preserves color “intensity”. Therefore, it can be classified as having a constant mean and exploding variance. For example, starting from any smoothed “blue canvas,” one can generate an image of a sky or an ocean but not an autumn forest.

Apart of that, the energy spectrum of an image behaves differently depending on the type of corruption process, as shown in Fig. 3.

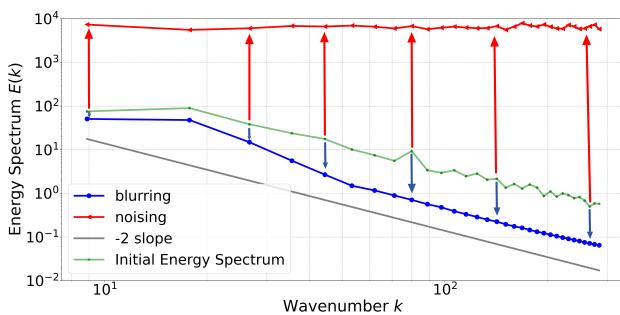


Figure 3. Comparison of the Energy Spectrum (ES) of an image subjected to different corruption processes. In the blurring process, the amplitude of energy components decays in a log-linear manner with a constant slope. The ES of a blurred image is represented by a blue line with circular markers. It is shifted downwards compared to the ES of the clean image denoted with green dotted line. On the other hand, the ES of the image in classical DDPM can be viewed as being gradually flooded with noise (red) starting from the higher frequencies..

Examples of PDE corruption operators. Recently, models based on that approach have been proposed in literature:

- **Heat dissipation (Isotropic Diffusion).** The simplest operator sets $\frac{\partial u}{\partial t} = \alpha \nabla^2 u$. The Inverse Heat Dissipation (IHD) [27] utilizes this PDE, generating a “coarse-to-fine” representation that naturally preserves color bias. To avoid the accumulation of numerical errors, a minor Gaussian perturbation is added during both training and sampling procedures.
- **Blurring Diffusion.** Here, the heat equation is recasted as a full Markov chain in the frequency domain, by injecting dimension-wise noise [12]. Although the multi-scale blurring is emphasized, the directional flow is not present.
- **Advection (Ours) or More Complex Dynamics.** One can further enrich the physical process with a velocity field to transport pixel intensities spatially.

In contrast to local Gaussian noising, the PDE-based corruptions *rearrange* intensities in a spatially coherent way. With a purely conservative, PDE-based corruption operator, one may still add small perturbations outside the PDE solution at each step to keep the chain stochastic. This framework ensures that color conservation arises from the PDE itself, which can be beneficial for preserving global color palettes.

4. Advection–Diffusion–Reaction Process

Our goal here is to perform the corruption process using a physically grounded mass transport equation, such as the *advection–diffusion–reaction* equation. Standard Gaussian-based noising can be seen as a special case of this equation in which both the advective transport and the averaging (blurring) terms are omitted. Let us discuss the *advection–diffusion–reaction* equation in the general form,

$$\frac{\partial u}{\partial t} + \underbrace{\nabla \cdot (\mathbf{v} u)}_{\text{advection}} = \underbrace{\nabla \cdot (\alpha \nabla u)}_{\text{diffusion}} + \underbrace{\dot{Q}(t)}_{\text{reaction}}. \quad (3)$$

The velocity field, $\mathbf{v} = \mathbf{v}(x, y)$, displaces the image intensities, $u = u(x, y, t)$. We omit the vector notation here, but the same approach is applied for each color channel. Over time, the diffusion term continues to reduce high-frequency features with a time-varying coefficient $\alpha = \alpha(t)$. Simultaneously, the reaction term, $\dot{Q}(t)$, can modify the amount of the quantity u over time. In the context of DDPM, the reaction term would correspond to the injection of the Gaussian

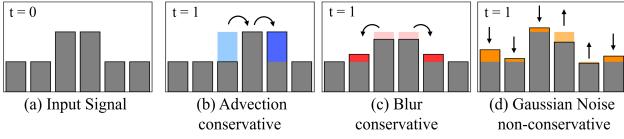


Figure 4. Corruption process: (a) input image, (b) advection and (c) blur “redistribute” the intensities but preserve the total “mass”, i.e., pixel-intensity sum (conservative). (d) Gaussian noise adds or subtracts “mass” (non-conservative).

noise, $\dot{Q} \sim \mathcal{N}(0, \sigma I)$, while the advection and diffusion terms may be interpreted as the drift term, $f(u(t), t)$, in Eq. 1. This kind of PDEs naturally arises in physical processes that describe the transport of quantities such as mass or energy and are known as conservation laws.

Fig. 4 shows that the redistribution of pixels’ intensities can be achieved solely by the reaction term (addition-/subtraction) or via the advection-diffusion operator which would shift the intensity (being a vector flux rather than a scalar reaction term). Interestingly, from the point of view of a standalone observer who sees only the effective change in intensity, the underlying processes may be indistinguishable. Our research aims to explore the advection effect while keeping the stochastic *scalar* noise marginal. Consequently, in our method we do *not* inject noise directly into the PDE itself in order to keep the process conservative. Let us define the $\mathcal{A}(t_k)[u]$ as shorthand for the advection-diffusion terms given in Eq. (3). This equation is solved numerically for k steps using a numerical LBM solver as described in Sec. 4.2. We add a small Gaussian perturbation just before passing the data to the neural network,

$$u_k(x, y) = \underbrace{\mathcal{A}(t_k)[u_0]}_{\text{advection-diffusion forward chain}} + \underbrace{\epsilon_t \sim \mathcal{N}(0, \sigma_T I)}_{\text{Gaussian training noise}}. \quad (4)$$

Here, $\sigma_T = 0.01$ is a small constant during all steps. In contrast to classical DDPM, the noise is not accumulated along the corruption process. As illustrated in Fig. 5, this separation keeps the corruption process physically consistent. The primary purpose of these small Gaussian perturbations is to alleviate the accumulation of numerical errors (c.f. Daras et al. [6]). Readers interested in work with high noise-to-blur ratios are referred to [12].

4.1. Turbulent Velocity Field Generator

The presented advection–diffusion process relies on a velocity field $\mathbf{v} = \mathbf{v}(\mathbf{x}, t)$ to transport image intensities during the forward process. In real-world phenomena, the time varying, turbulent velocity fields exhibit both large and small-scale coherent flows [38]. To capture these multi-scale effects we propose a *Spectral Turbulence Generator* that synthesizes a \mathbf{v} with controllable spectral energy.

The turbulent velocity field can swirl, convect, or otherwise *mix* image content over time, along the forward corruption trajectory. Our turbulence generator achieves this

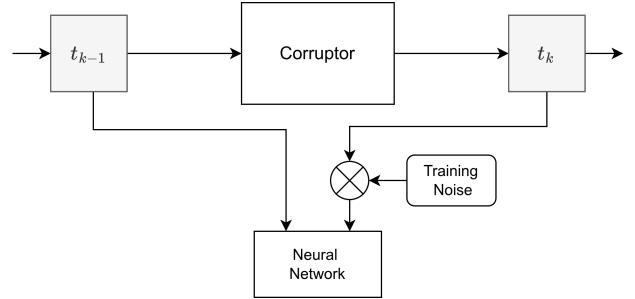


Figure 5. An overview of the NN training pipeline. The image corruptor applies advection–diffusion operator $\mathcal{A}(t_k)$ during each of the discrete time steps. The NN is trained on pairs of images destroyed up to the prescribed time, as dictated by the scheduler.

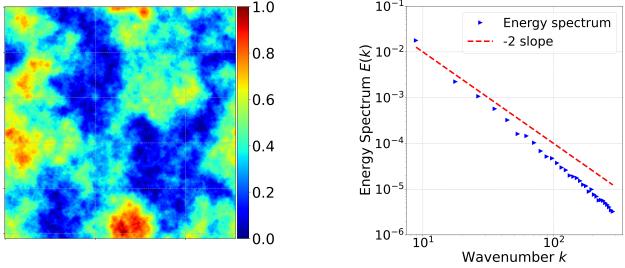
by parameterizing \mathbf{v} in the frequency domain and drawing random phases to yield diverse realizations. Moreover, the spectral parametrization allows to control energy at specific wavenumbers, thereby tailoring how features are displaced over different spatial scales.

Spectral Parameterization. The synthetic turbulent velocity field is inspired by the Random Fourier Modes (RFM) approach [8]. The core idea is to construct a velocity field in the Fourier space by assigning random phases to Fourier modes whose amplitudes are determined by a prescribed energy spectrum. Consider the two-dimensional domain $\Omega \subset \mathbb{R}^2$ with coordinates $\mathbf{x} = (x, y)$. We form discrete wavevectors $\boldsymbol{\kappa} = (\kappa_x, \kappa_y)$ on a grid determined by the spatial resolution $N \times N$. Let us denote the 2D velocity field $\mathbf{v} = [v_x, v_y]$ and its Fourier transform as $\hat{\mathbf{v}}(\mathbf{k}) = \mathcal{F}(\mathbf{v})$. In the RFM approach, the time-varying velocity components are generated in the spectral space as,

$$\begin{aligned} \hat{v}_x(\boldsymbol{\kappa}) &= A(\|\boldsymbol{\kappa}\|) e^{i(\phi(\boldsymbol{\kappa}) + \omega t)} \\ \hat{v}_y(\boldsymbol{\kappa}) &= A(\|\boldsymbol{\kappa}\|) e^{i(\phi(\boldsymbol{\kappa}) + \omega t)}, \end{aligned}$$

where $\|\boldsymbol{\kappa}\| = \sqrt{k_x^2 + k_y^2}$ is the wavenumber magnitude. The angular frequency ω is calculated as $\omega = \|\boldsymbol{\kappa}\| dt$ where we set $dt = 10^{-4}$ as a small increment so that phase evolution remains slow and stable in the spectral domain. The random phases are denoted as $\phi(\boldsymbol{\kappa})$ and are uniformly distributed in $[0, 2\pi]$. The amplitude $A(\|\boldsymbol{\kappa}\|)$ is determined by the prescribed energy spectrum $E(\|\boldsymbol{\kappa}\|) \propto \|\boldsymbol{\kappa}\|^{-2}$.

We restrict $\boldsymbol{\kappa}$ to a chosen band to shape the flows dominant frequencies; in our experiments we choose $[\kappa_{\min}, \kappa_{\max}] = [2\pi/N, 2\pi/(1024N)]$. By inverse Fourier transform we obtain the spatial velocity components $\mathbf{v}(\mathbf{x}, t)$ updated across time steps $t = 1, \dots, t_N$. This approach naturally handles swirling, drifting, and other turbulent-like motions when κ_{\max} is large enough to include fine-scale eddies. Fig. 6 depicts an example of the generated velocity field and its corresponding spectrum.



(a) Generated velocity field. Colored by the velocity field magnitude (normalized).

(b) Spectrum of the turbulent velocity field obeys the imposed decay slope.

Figure 6. Generated turbulent velocity field and its corresponding spectrum. Please refer to Section 4.1 for details.

Velocity Magnitude Limiting. Our generator can also cap the velocity magnitude to control numerical stability by maintaining realistic motion speeds. Specifically, after transforming to the spatial domain we apply a soft-limiter $\tanh(\cdot)$ -based function to ensure that the generated field does not exceed a maximum threshold, $\|\mathbf{v}(\mathbf{x}, t)\| < 10^{-3}$. One can vary this threshold to tune the swirl intensity or keep flows numerically stable.

Integration into the PDE. At forward step k the generator outputs $\mathbf{v}_k(\mathbf{x})$. Substituting into (3), we solve for the advected and diffused image field. Over multiple timesteps, this results in a gradual flow-driven smoothing that captures both large-scale drift and fine-grained turbulence.

4.2. Lattice Boltzmann Solver

While one can solve the heat equation in the frequency domain, the presence of advection makes an efficient real-space discretization necessary. We adopt the *Lattice Boltzmann Method* (LBM), which models the macroscopic field $u(x, y, t)$ via particle-like *distribution functions* at each grid node: $\{u_i(x, y, t)\}_{i=0,\dots,8}$ in a D2Q9 arrangement, where D is the number of dimensions and Q is the number of discrete distributions. Conceptually, $u_i(x, y, t)$ denotes distributions moving in one of nine discrete velocity directions, so that $u(x, y, t) = \sum_i u_i(x, y, t)$ recovers the physical density (or intensity) at (x, y) . It can be shown [19], that the LBM algorithm naturally handles advection and diffusion, and easily scales to large images on GPUs.

The LBM Routine. At each time step, the LBM applies the following fundamental stages:

1. *Compute equilibrium distribution:* The advection is driven by the externally generated velocity field \mathbf{v} which influences the equilibrium distribution:

$$u_i^{eq}(\mathbf{x}, t, \mathbf{v}) = w_i u \left[1 + \frac{\mathbf{c}_i \mathbf{v}}{c_s^2} + \frac{(\mathbf{c}_i \mathbf{v})^2}{2c_s^4} - \frac{\mathbf{v}^2}{2c_s^2} \right],$$

where $c_s^2 = \frac{1}{3}$, is the lattice speed of sound and w_i is a directional weight. The relaxation time τ is directly correlated with the diffusion coefficient $\alpha = c_s^2 (\tau - \frac{1}{2})$.

2. *Collision:* The local distributions u_i at each grid node relax toward an equilibrium $u_i^{eq}(u, \mathbf{v})$ at a rate $\frac{1}{\tau}$,

$$\tilde{u}_i(x, y, t) = u_i(x, y, t) - \frac{1}{\tau} [u_i(x, y, t) - u_i^{eq}].$$

3. *Streaming:* The post-collision distributions \tilde{u}_i shift (or “stream”) to their neighboring nodes according to the discrete lattice directions (c_{ix} , c_{iy}). On the D2Q9 lattice, each node has eight neighbors and the rest particle

$$u_i(x + c_{ix}, y + c_{iy}, t + \Delta t) = \tilde{u}_i(x, y, t). \quad (5)$$

After streaming, one sums the updated u_i to recover the macroscopic field $u = \sum_i u_i$. Physical boundary conditions are implemented using the bounce-back rule which simply reverses the distributions back into the domain and achieves the no-flux condition.

Computational Advantages. The LBM collision and streaming are purely local memory operations, ideal for GPU acceleration. This enables $\mathcal{O}(HW)$ updates on an $H \times W$ grid with limited communication overhead. The result is a stable, easily parallelizable solver that captures both directional flow and diffusion in one coherent framework. We implement the LBM scheme in CUDA using the Taichi framework [13]. The distribution functions are stored in a structure-of-arrays layout for memory coalescing. The explicit LBM stencil in the pseudo-code is available in the Appendix C of our work.

4.3. Dimensionless PDE Formulation

A key step to a fair comparison of our advection–diffusion method to others (we compare to the IHD method [27]) is to express the forward PDE in a dimensionless form. We do this by introducing two dimensionless parameters: the *Fourier number* (Fo) and the *Peclet number* (Pe) which together control how much diffusion and advection is applied. This strips away arbitrary scale factors (like image resolution or absolute blur sizes), letting us match the essential “diffusion budget” (Fourier number) and “flow to diffusion ratio” (Peclet number) between methods so that any performance differences are not merely due to mismatched scales.

Characteristic Scales. Let α be the diffusion coefficient from Eq. (3), L a characteristic length scale (e.g., the image width), and V the maximum or typical speed for the velocity field \mathbf{v} . We define the dimensionless spatial/time coordinates as

$$\mathbf{x}^* = \frac{\mathbf{x}}{L}, \quad t^* = \frac{\alpha t}{L^2},$$

with $\nabla = \frac{1}{L} \nabla^*$ and $\frac{\partial}{\partial t} = \frac{L^2}{\alpha} \frac{\partial}{\partial t^*}$. We then scale the velocity with $\mathbf{v}^*(x^*, t^*) = \frac{\mathbf{v}(x, t)}{V}$. Reinterpreting the image $u(x, t)$ as $u^*(x^*, t^*)$ completes the non-dimensional setup.

Fourier Number (Fo). When discretizing the PDE into time increments of length Δt , we define

$$\text{Fo} = \frac{\alpha \Delta t}{L^2} = \Delta t^*, \quad (6)$$

which dictates how much diffusion occurs per time step in dimensionless time. Equating Fo across different forward processes ensure that the same corruption schedule and the same physical time apply to data at different resolutions.

Peclet Number (Pe). To quantify the ratio of advective transport to diffusion rate, we use

$$\text{Pe} = \frac{V L}{\alpha}. \quad (7)$$

Increasing Pe intensifies the directional flow.

Dimensionless PDE. Starting from $\frac{\partial u}{\partial t} + \alpha(\mathbf{v} \cdot \nabla u) = \alpha \nabla^2 u$, and applying the above scalings yields

$$\frac{\partial u^*}{\partial t^*} + \text{Fo Pe} (\mathbf{v}^* \cdot \nabla^*) u^* = \text{Fo} \nabla^{*2} u^*, \quad (8)$$

where the diffusion term is scaled by Fo and the advection term by Fo Pe. When Pe = 0, Eq. (8) reduces to forward blur only (no advection). In practice, we fix the Fo schedule (thereby defining the diffusivity α per solver step) and then we choose Pe to modulate how much advective flow is added. This dimensionless framing ensures a clean comparison: both processes consume the same ‘‘diffusion budget’’, but differ in how strongly they advect.

4.4. Scheduler

In the work of Rissanen et al. [27], the amount of blur has been solely defined by a parameter σ . From the physical standpoint, it can be connected with Fo as $\sigma = \sqrt{2t}\alpha$ therefore $\text{Fo} = \frac{\sigma}{L^2}$. To compare our results to Rissanen et al. [27], we found the exponential schedule most suitable. The blur schedule, expressed in dimensionless time Fo_t for $t = 0, 1, \dots, T - 1$, is generated using an exponential spacing. The formula reads,

$$\text{Fo}_t = \left(\frac{\text{Fo}_{\max}}{\text{Fo}_{\min}} \right)^{\frac{t}{T-1}} \text{Fo}_{\min},$$

where Fo_{\min} and Fo_{\max} is the initial and final dimensionless time. The value T is the total number of denoising steps in the schedule and t is the index of the current step, ranging from 0 to $T - 1$. Notice that the number of denoising steps T does not have to coincide with the number of steps performed by the numerical solver. Therefore the relaxation coefficient in conductivity in Eq. 3 has to be calculated for each pair of Fo_t and Fo_{t+1} .

Algorithm 1 Training

```

 $\mathbf{u}_k, \mathbf{u}_{k-1}, k \leftarrow$  drawn a training batch
 $\epsilon_T \sim \mathcal{N}(\mathbf{0}, \sigma_T^2)$  ▷ Training noise
 $\hat{\mathbf{u}}_k \leftarrow \mathbf{u}_k + \epsilon_T$ 
 $\Delta^{\mathbf{u}} = \Psi_{\theta}(\hat{\mathbf{u}}_k, k)$ 
 $\mathcal{L}_{\text{MSE}} = \|\Delta^{\mathbf{u}} + (\mathbf{u}_{k-1} - \hat{\mathbf{u}}_k)\|_2^2$ 

```

Algorithm 2 Sampling

```

 $k \leftarrow K$  ▷ Start from terminal state
 $\mathbf{u} \sim p(\mathbf{u}_K)$  ▷ Sample from the blurry prior
while  $k > 0$  do
     $\epsilon_S \sim \mathcal{N}(\mathbf{0}, \sigma_S^2)$  ▷ Sampling noise
     $\hat{\mathbf{u}}_k \leftarrow \mathbf{u}_k + \epsilon_S$ 
     $\Delta^{\mathbf{u}} \leftarrow \Psi_{\theta}(\hat{\mathbf{u}}_k, k)$ 
     $\mathbf{u}_{k-1} \leftarrow \hat{\mathbf{u}}_k + \Delta^{\mathbf{u}}$  ▷ Reverse step
     $k \leftarrow k - 1$ 
end while

```

4.5. Learning the Reverse Process

Given an image, let us denote the field of its pixels intensities as u_k . It evolves in discrete time steps $k = 1, \dots, K$. We define a *forward* chain $q(u_{1:K} \mid u_0)$ by applying our PDE-based advection-diffusion operator plus noise. Rather than building a variational bound, the neural network Ψ_{θ} learns differences between chain elements $p_{\theta}(u_{k-1}) - p_{\theta}(u_k)$ in a purely in a regression-style manner. Specifically, we start from a corrupted prior $p(u_K)$ (e.g. kernel density estimates of blurred and advected training images). The training and sampling noise, denoted as ϵ_T and ϵ_S respectively, is injected between pairs (u_k, u_{k-1}) for all k . Their ratio is fixed as $\sigma_T/\sigma_S = 1.25$ and acts solely as a mild regularization.

Observe, that the MSE loss in Algorithm 1 corresponds to a finite difference of $\frac{\partial u}{\partial t} \approx \frac{u_{k-1} - u_k}{dt}$ where $dt = 1/K$. As the number of steps grows to infinity, $K \rightarrow \infty$, the loss converges to the denoising score matching objective (which is the gradient of the log-probability density), re-using the derivation presented in IHD App. A.4 [27].

To speed up the Algorithm 1, the \mathbf{u}_k and \mathbf{u}_{k-1} are actually stored in a precomputed training data tensor from which the batch is drawn. The precomputation procedure follows Eq. (4), $\mathbf{u}_k = \mathcal{A}(t_k)[u_0]$. The neural network is an U-Net [29] with attention layers, please refer to the supplemental material for further details.

5. Experiments

We evaluate our approach on datasets commonly used in generative modeling: FFHQ-128 (128 × 128 resolution, 70,000 training samples), MNIST (28 × 28 pixels, 60,000 training images, 10,000 testing). We present additional

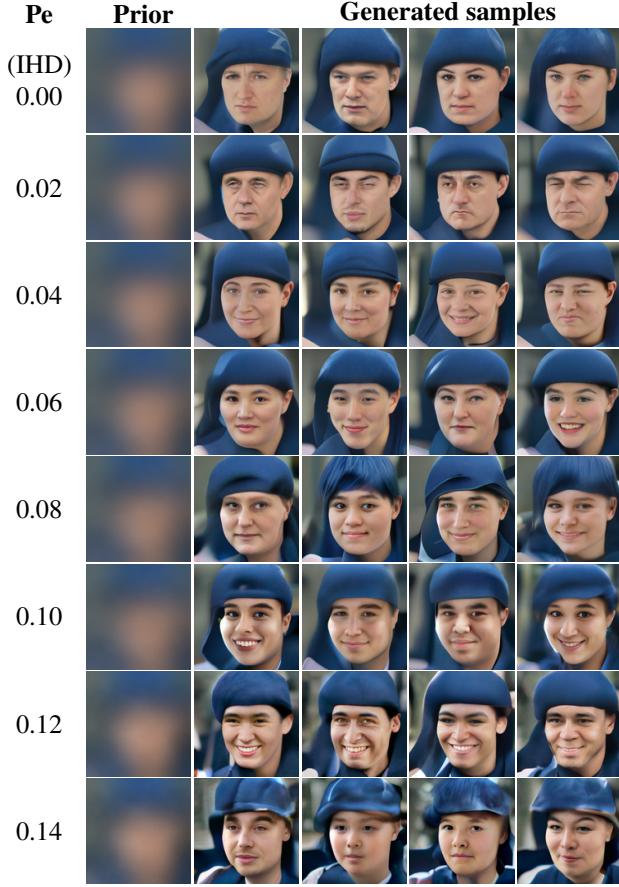


Figure 7. Samples for $\sigma = 16$, comparing different Peclet numbers.

qualitative results on LSUN Church dataset in the Appendix. To obtain the initial state (blurry prior), we corrupt the clean images with according to the PDE, as described in Sec. 4.

Impact of Peclet Number on Generated Samples. We first provide qualitative demonstrations of our models ability to generate high-fidelity images. In all datasets we observe that a directional flow (via $Pe \neq 0$) yields visually richer details. Figure 7 shows side-by-side generated images on the FFHQ-128 dataset, illustrating that even mild Peclet numbers can enhance the local structure.

To quantify the impact of our advection-diffusion model on image quality and generative performance, we report evaluation metrics in Tab. 1 for various Peclet numbers (Pe), which control the strength of the advective term. The experiments are conducted on the FFHQ dataset at 128×128 resolution using 200 sampling steps, and repeated under two final blur conditions, parameterized by spatial diffusion scales $\sigma=16$ and $\sigma=20$. We report Fréchet Inception Distance (FID)[9] to measure distributional alignment and the LPIPS metric[40]to capture perceptual diversity with respect to ground truth images. In addition, we evaluate

Table 1. Evaluation metrics for Peclet numbers on FFHQ 128×128 . Top: $\sigma=16$, bottom: $\sigma=20$. Lower FID is better. Higher μ_{LPIPS} and PRDC values indicate greater diversity and coverage. When $Pe = 0$, the model achieves a baseline values corresponding to the purely blurring approach.

Pe	FID \downarrow	$\mu_{\text{LPIPS}} \uparrow$	$\sigma_{\text{LPIPS}} \uparrow$	P \uparrow	R \uparrow	D \uparrow	C \uparrow
$\sigma = 16$							
0 (IHD)	52.10	0.238	0.011	0.789	0.118	0.763	0.507
0.02	52.33	0.255	0.013	0.788	0.112	0.770	0.506
0.04	53.54	0.260	0.014	0.797	0.119	0.803	0.499
0.06	46.93	0.265	0.012	0.812	0.139	0.858	0.539
0.08	46.24	0.248	0.017	0.806	0.153	0.870	0.565
0.10	47.57	0.297	0.017	0.800	0.171	0.892	0.550
0.12	45.36	0.296	0.018	0.808	0.173	0.956	0.587
0.14	38.10	0.302	0.023	0.798	0.223	0.958	0.627
$\sigma = 20$							
0 (IHD)	55.87	0.265	0.016	0.798	0.109	0.762	0.482
0.02	56.57	0.293	0.013	0.797	0.102	0.806	0.491
0.04	51.44	0.286	0.022	0.815	0.115	0.921	0.539
0.06	36.64	0.315	0.019	0.826	0.243	1.040	0.665
0.08	37.41	0.305	0.019	0.817	0.247	1.043	0.662
0.10	42.88	0.311	0.018	0.764	0.187	0.854	0.556
0.12	48.62	0.344	0.022	0.688	0.183	0.683	0.510
0.14	54.56	0.348	0.019	0.632	0.150	0.565	0.429

generation fidelity and sample diversity using the Precision, Recall, Density, and Coverage (PRDC) metrics [24]. As expected, increasing the Peclet number enhances sample diversity (higher σ_{LPIPS} and PRDC scores) but generally degrades fidelity (higher FID), consistent across both σ regimes. The arrow annotations in the column headers indicate the preferred direction for each metric.

Hoogeboom et al. [12] noticed that better FID metric can be achieved for higher final blur σ at the price of a more difficult (longer) training. Eventually, the optimal Pe depends on the final amount of blur.

Image Interpolation. We also evaluate our method on interpolated initial states and compare it with a baseline approach (IHD). Two samples from FFHQ dataset undergo the forward process, followed by linear interpolation in the latent space. The interpolated states are then denoised with generative noise and interpolated with SLERP Shoemake [32]. The analysis provides an insight into the smoothness and consistency of the learned latent space. By denoising the linearly interpolated noisy inputs we assess the model’s ability to generate coherent transitions. Figure 8 depicts the qualitative results.

Discussion and Limitations. An immediate drawback of introducing a nonlinear advection term into the diffusion process is the necessity of step-by-step computation up to



Figure 8. Visual comparison of interpolations between two FFHQ samples. Each undergoes the forward process up to $\sigma = 16$, followed by linear interpolation and denoising with SLERP-interpolated generative noise added at each step.

the given time. In practice, we solve the PDE which governs the corruption process for the dataset and store the result to avoid repeated runs. This procedure consumes $\sim 10\%$ of total training time. In our examples, the corruption process lead to the blurry prior from which new samples can be generated. As a consequence, we have not found a distribution (like a standard Gaussian) from which samples could be easily drawn. However, a sufficiently long process with properly tuned balance between advection and diffusion terms shall converge to a well defined stationary turbulent field analogous to the noising Markov process.

6. Future Outlook and Conclusions.

We proposed a novel PDE-based diffusion model based on the advection-diffusion equation. The model introduces a turbulent mixing term to the forward diffusion process,

which, to our knowledge, has not been attempted before. In our experiments, we showed that adding the advective term besides the diffusion one improve the quality (FID) of generated images compared to the baseline approach (IHD) [27]. The external velocity field, which can transfer pixels' intensities in a spatially coherent way, offers a new way of introducing corruption into the forward process. In the future, the influence of different turbulence generators and the slope of turbulent kinetic energy can be further investigated. The interplay with a Gaussian noise corruption as proposed in Hoogeboom and Salimans [12] also seems a natural direction of research. Finally, it would be interesting to evaluate the model using different training objective, for instance the *flow matching* approach [23]. Then, the effect of the “overall” destruction of the image (final state) could be investigated separately from the forward corruption trajectory dictated by a particular of PDE.

Acknowledgments

We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2025/017969. We would also like to thank Kamil Deja and Przemyslaw Spurek for aiding us with in-house reviews.

References

- [1] Mohammadmehdi Ataei and Hesam Salehipour. Xlb: A differentiable massively parallel lattice boltzmann library in python. *Comput. Phys. Commun.*, 300:109187, 2023. [2](#)
- [2] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. In *Advances in Neural Information Processing Systems*, pages 41259–41282. Curran Associates, Inc., 2023. [1, 2](#)
- [3] Mario Christopher Bedrunka, Dominik Wilde, Martin L. Kliemann, Dirk Reith, Holger Foysi, and Andreas Krämer. Lettuce: Pytorch-based lattice boltzmann framework. In *ISC Workshops*, 2021. [2](#)
- [4] Hanqun Cao, Cheng Tan, Zhangyang Gao, Yilun Xu, Guangyong Chen, Pheng-Ann Heng, and Stan Z. Li. A survey on generative diffusion models. *IEEE Transactions on Knowledge and Data Engineering*, 36:2814–2830, 2022. [2](#)
- [5] Ting Chen. On the importance of noise scheduling for diffusion models. *arXiv preprint arXiv:2301.10972*, 2023. [2](#)
- [6] Giannis Daras, Mauricio Delbracio, Hossein Talebi, Alexandros G. Dimakis, and Peyman Milanfar. Soft diffusion: Score matching for general corruptions, 2022. [1, 2, 4](#)
- [7] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *ArXiv*, abs/2105.05233, 2021. [1, 2](#)
- [8] J. C. H. Fung, J. C. R. Hunt, N. A. Malik, and R. J. Perkins. Kinematic simulation of homogeneous turbulence by unsteady random fourier modes. *Journal of Fluid Mechanics*, 236:281318, 1992. [4](#)
- [9] 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*, 2017. [7](#)
- [10] Jonathan Ho. Classifier-free diffusion guidance. *ArXiv*, abs/2207.12598, 2022. [2](#)
- [11] Jonathan Ho, Ajay Jain, and P. Abbeel. Denoising diffusion probabilistic models. *ArXiv*, abs/2006.11239, 2020. [1, 2](#)
- [12] Emiel Hoogeboom and Tim Salimans. Blurring diffusion models. *arXiv preprint arXiv:2209.05557*, 2022. [1, 2, 3, 4, 7, 8](#)
- [13] Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédéric Durand. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6):201, 2019. [5](#)
- [14] Xingchang Huang, Corentin Salaun, Cristina Vasconcelos, Christian Theobalt, Cengiz Oztireli, and Gurprit Singh. Blue noise for diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, New York, NY, USA, 2024. Association for Computing Machinery. [2](#)
- [15] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *ArXiv*, abs/2206.00364, 2022. [2](#)
- [16] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. *arXiv preprint arXiv:2312.02696*, 2023. [2](#)
- [17] Diederik P. Kingma, Tim Salimans, Ben Poole, Prafulla Dhariwal, Xi Chen, and Tim Chen. Variational diffusion models. In *Advances in Neural Information Processing Systems*, 2021. [2](#)
- [18] Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning-accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences of the United States of America*, 118, 2021. [2](#)
- [19] Timm Krüger, Halim Kusumaatmaja, Alexandr Kuzmin, Ornest Shardt, Goncalo Silva, and Erlend Magnus Viggen. *The Lattice Boltzmann Method*. Springer, Cham, first edition, 2017. [2, 5](#)
- [20] Łaniewski-Wołk and Rokicki. Adjoint lattice boltzmann for topology optimization on multi-gpu architecture. *Computers and Mathematics with Applications*, 71(3):833–848, 2016. [2](#)
- [21] Moritz Lehmann, Mathias J Krause, Giorgio Amati, Marcello Segà, Jens Harting, and Stephan Gekle. Accuracy and performance of the lattice boltzmann method with 64-bit, 32-bit, and customized 16-bit number formats. *Physical Review E*, 106(1):015308, 2022. [2](#)
- [22] Zong-Yi Li, Nikola B. Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *ArXiv*, abs/2010.08895, 2020. [2](#)
- [23] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023. [8](#)
- [24] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *International conference on machine learning*, pages 7176–7185. PMLR, 2020. [7](#)
- [25] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. [2](#)
- [26] Bogdan Raoni, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of pdes, 2023. [2](#)
- [27] Severi Rissanen, Markus Heinonen, and A. Solin. Generative modelling with inverse heat dissipation. *ArXiv*, abs/2206.13397, 2022. [1, 2, 3, 5, 6, 8](#)
- [28] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image

- synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2
- [29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. 6
- [30] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:4713–4726, 2021. 2
- [31] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. 2
- [32] Ken Shoemake. Animating rotation with quaternion curves. *SIGGRAPH Comput. Graph.*, 19(3):245254, 1985. 7
- [33] Jascha Narain Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *ArXiv*, abs/1503.03585, 2015. 1, 2
- [34] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ArXiv*, abs/2010.02502, 2020. 2
- [35] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019. 2
- [36] Yang Song, Jascha Narain Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ArXiv*, abs/2011.13456, 2020. 1, 2, 3
- [37] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In *Neural Information Processing Systems*, 2021. 2
- [38] D.C. Wilcox. *Turbulence Modeling for CFD*. Number v. 1 in Turbulence Modeling for CFD. DCW Industries, 2006. 4
- [39] Ling Yang, Zhilong Zhang, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Ming-Hsuan Yang, and Bin Cui. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56:1 – 39, 2022. 2
- [40] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. 7