Research Article

Enhancing Sample Generation of Diffusion Models using Noise Level Correction

Abulikemu Abuduweili[1], Chenyang Yuan[2], Changliu Liu[1], Frank Permenter[2]

1. Carnegie Mellon University, United States; 2. Toyota Research Institute

The denoising process of diffusion models can be interpreted as a projection of noisy samples onto the data manifold. Moreover, the noise level in these samples approximates their distance to the underlying manifold. Building on this insight, we propose a novel method to enhance sample generation by aligning the estimated noise level with the true distance of noisy samples to the manifold. Specifically, we introduce a noise level correction network, leveraging a pre-trained denoising network, to refine noise level estimates during the denoising process. Additionally, we extend this approach to various image restoration tasks by integrating task-specific constraints, including inpainting, deblurring, super-resolution, colorization, and compressed sensing. Experimental results demonstrate that our method significantly improves sample quality in both unconstrained and constrained generation scenarios. Notably, the proposed noise level correction framework is compatible with existing denoising schedulers (e.g., DDIM), offering additional performance improvements.

1. Introduction

Generative models have significantly advanced our capability of creating high-fidelity data samples across various domains such as images, audio, and text[1][2][3]. Among these, diffusion models have emerged as one of the most powerful approaches due to their superior performance in generating high-quality samples from complex distributions[1][2][4]. Unlike previous generative models, such as generative adversarial networks (GANs)[5] and variational autoencoders (VAEs)[6], diffusion models adds multiple levels of noise to the data, and the original data is recovered through a learned denoising process[7][8]. This allows diffusion models to handle high-dimensional, complex data distributions, making them especially useful for tasks where sample quality and diversity are critical[9]. Their capability to generate complex, high-resolution data has led to widespread applications across numerous tasks, from image

 doi.org/10.32388/NSYEO2

generation in models like DALL·E[10] and Stable Diffusion[11] to use in robotic path-planningand control[12][13], as well as text generation[14].

Previous studies[15][16] have interpreted the denoising process in diffusion models as an approximate projection onto the data manifold, with the noise level σ approximating the distance between noisy samples and the data manifold. This perspective views the sampling process an optimization problem, where the goal is to minimize the distance between noisy samples and underlying data manifold using gradient descent. The gradient direction is approximated by the denoiser output with step size determined by the noise level schedule. However, the denoiser requires an estimate of the noise level as input. We claim that accurately estimating the noise level during the denoising process—essentially the distance to the data manifold—is crucial for convergence and accurate sampling.

The expressive capabilities of diffusion models have also made them a compelling choice for image restoration tasks, where generating high-quality, detailed images is essential[17]. Diffusion models can be used as an image-prior for capturing the underlying structure of image manifold and have shown significant promise for constrained generation such as image restoration[18][19][20]. Plug-and-play methods were proposed to utilize pre-trained models without the need for extensive retraining or end-to-end optimization for linear inverse problems such as super-resolution, inpainting and compressed sensing[19][21][22]. These methods can be interpreted as alternating taking gradient steps towards the constraint set (linear projection for linear inverse problems) and the image manifold (noise direction estimated by the learned denoiser) to find their intersection. However, they may suffer from inconsistency issues if, after each projection step, the noise level no longer approximates distance (2). Thus correcting the noise level after each step could increase the accuracy of image restoration tasks.

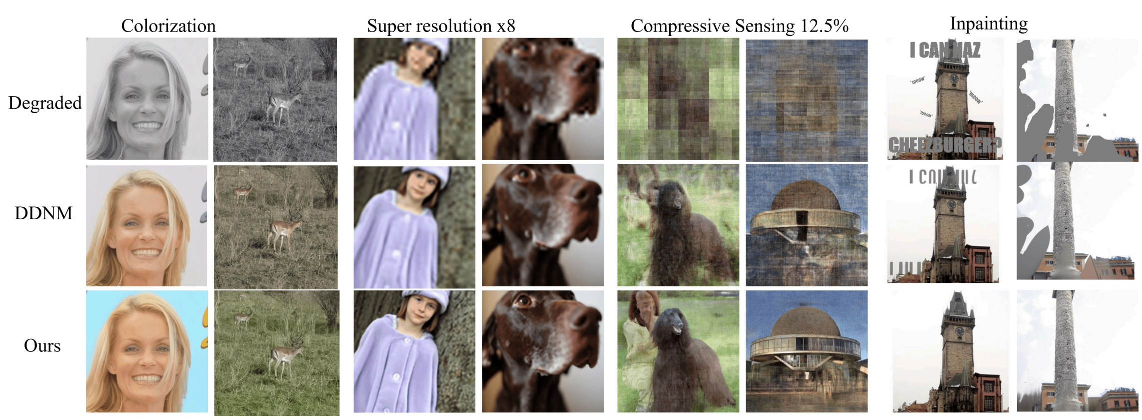


Figure 1. Qualitative results of constrained image generation.

doi.org/10.32388/NSYEO2

In this work, we propose a novel noise level correction method to refine the estimated noise level and enhance sample generation quality. Our approach introduces a noise level correction network that aligns the estimated noise level of noisy samples more closely with their true distance to the data manifold. By dynamically adjusting the sampling step size based on this corrected noise level estimation, our method improves the sample generation process, significantly enhancing the quality of generated data. Furthermore, our approach integrates seamlessly with existing denoising scheduling methods, such as DDPM (Denoising Diffusion Probabilistic Models)[7], DDIM (Denoising Diffusion Implicit Models)[23], and EDM[24]. Furthermore, we extend the application of noise level correction to various image restoration tasks, showing its ability to improve the performance of diffusion-based models such as DDNM (Diffusion Null-Space Model)[22]. Our method achieves improved results across tasks including inpainting, deblurring, super-resolution, colorization, and compressed sensing, as illustrated in Figure 1. Additionally, we introduce a parameter-free lookup table as an approximation of the noise level correction network, providing a computationally efficient alternative for improving the performance of unconstrained diffusion models. In summary, our contributions are:

- We propose a noise level correction network that improves sample generation quality by dynamically refining the estimated noise level during the denoising process.
- We extend the proposed method to constrained tasks, achieving significant performance improvements in various image restoration challenges.
- We develop a parameter-free approximation of the noise level correction network, offering a computationally efficient tool to improve diffusion models.
- Through extensive experiments, we demonstrate that the proposed noise level correction method consistently provides additional performance gains when applied on top of various denoising methods

2. Background

2.1. Diffusion Models

Diffusion models represent a powerful class of latent variable generative models that treat datasets as samples from a probability distribution, typically assumed to lie on a low-dimensional manifold $\mathcal{K} \subset \mathbb{R}^n$ [2][7]. Given a data point $z_0 \in \mathcal{K}$, diffusion models aim to learn a model distribution $p_\theta(z_0)$ that can approximate this manifold and enable the generation of high-quality samples. The process involves gradually corrupting the data with noise during a forward diffusion process and incrementally denoising it to reconstruct the original data through a reverse generative process.

 doi.org/10.32388/NSYEO2

Diffusion (forward) process. In the forward process, noise is added progressively to the data. Starting with a clean sample z_0, the noisy version at step t, denoted z_t, is a linear combination of z_0 and Gaussian noise ϵ:

$$z_t = \sqrt{\alpha_t} z_0 + \sqrt{1-\alpha_t}\epsilon, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, I), \tag{1}$$

where the noise schedule α_t controls the amount of noise injected at each step. Typically, $1 \geq \alpha_1 > \alpha_2 > \cdots > \alpha_T \geq 0$, ensuring that $p(z_T) \sim \mathcal{N}(0, I)$ for large enough T[7]. For mathematical convenience, a reparameterization is often employed, defining new variables $x_t = z_t / \sqrt{\alpha_t}$, which results in:

$$x_t = x_0 + \sigma_t \epsilon, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, I), \tag{2}$$

$$\sigma_t = \sqrt{\frac{1-\alpha_t}{\alpha_t}}, \quad x_t = \frac{z_t}{\sqrt{\alpha_t}}, x_0 = z_0. \tag{3}$$

Where σ_t denotes the noise level. Note that the diffusion process is originally presented in variable z_t, we use the formulation $x_t = \frac{z_t}{\sqrt{\alpha_t}}$ to simplify the forward diffusion process and lays the groundwork for the reverse denoising process. A similar formulation can be found in[25][24].

Denoiser. Diffusion models are trained to estimate the noise vector added to a sample during the forward process. The learned denoiser, denoted as ϵ_θ, is to predict the noise vector ϵ from the noisy sample x_t and the corresponding noise level σ_t. The denoiser is optimized using a loss function that minimizes the difference between the predicted and true noise vectors:

$$L(\theta) := \mathbb{E}\|\epsilon_\theta(x_t, \sigma_t) - \epsilon\|^2 = \mathbb{E}_{x_0,t,\epsilon}\|\epsilon_\theta(x_0 + \sigma_t\epsilon, \sigma_t) - \epsilon\|^2. \tag{4}$$

Here, x_0 is sampled from the data distribution, σ_t drawn from a discrete predefined noise level schedule, and ϵ is drawn from a standard Gaussian distribution, $\mathcal{N}(0, I)$. Training is typically performed using gradient descent, where randomly sampled triplets $(x_0, \epsilon, \sigma_t)$ are used to update the denoiser's parameters θ. Once the denoiser is trained, we can apply a one-step estimation to approximate the clean sample $\hat{x}_{0|t} \approx x_0$:

$$\hat{x}_{0|t} = x_t - \sigma_t \epsilon_\theta(x_t, \sigma_t). \tag{5}$$

Denoising (sampling) process. The one-step estimation, Eq. (5), may lack accuracy, in which case the trained denoiser is applied iteratively through the denoising process. This process aims to progressively denoise a noisy sample x_T and recover the original data x_0. Sampling algorithms construct a sequence of intermediate estimates $(x_T, x_{T-1}, \ldots, x_0)$, starting from an initial point x_T drawn from a Gaussian distribution, $x_T \sim \mathcal{N}(0, I)$. One of the widely used samplers, the deterministic DDIM[23], follows the recursion:

$$x_{t-1} = x_t + (\sigma_{t-1} - \sigma_t)\epsilon_\theta(x_t, \sigma_t) = \hat{x}_{0|t} + \sigma_{t-1}\epsilon_\theta(x_t, \sigma_t) \tag{6}$$

$$x_T = \frac{z_T}{\sqrt{\alpha_t}} = \sqrt{\sigma_T^2 + 1} \cdot z_T, \quad z_T \sim \mathcal{N}(0, I), \tag{7}$$

where ϵ_θ is the predicted noise at step t. This iterative process continues until x_0 is obtained, which represents a denoised sample. On the other hand, the randomized DDPM[7] follows the update rule:

$$x_{t-1} = x_t + (\sigma_{t'} - \sigma_t)\epsilon_\theta(x_t, \sigma_t) + \eta\omega_t = \hat{x}_{0|t} + \sigma_{t'}\epsilon_\theta(x_t, \sigma_t) + \eta\omega_t \tag{8}$$

$$\sigma_{t'} = \frac{\sigma_{t-1}^2}{\sigma_t}, \quad \eta = \sqrt{\sigma_{t-1}^2 - \sigma_{t'}^2}, \quad \omega_t \sim \mathcal{N}(0, I). \tag{9}$$

2.2. *Additional Related Works*

Diffusion models have gained significant attention for their ability to learn complex data distributions, excelling in diverse applications such as image generation[7][26], audio synthesis[27], and robotics[13]. Several methods have been proposed to improve the quality of generated samples while reducing the number of iterations. These include techniques like distillation for reducing sampling steps[28], progressive distillation[29], consistency models[8], and improved design space of diffusion models[24]. On the theoretical side, significant research has explored the non-asymptotic convergence rates of various diffusion samplers, including DDPM[30] and DDIM[31], contributing to a deeper understanding of the optimization processes underlying diffusion-based models.

Diffusion models have also demonstrated effectiveness in image restoration tasks, including super-resolution[19], inpainting[32], deblurring, and compressed sensing[22]. These tasks often require strict adherence to data consistency constraints, making diffusion models particularly suitable for addressing such challenges. Prior works such as DDRM[32] and DDNM[22] pioneered the use of diffusion models for solving linear inverse problems by projecting noisy samples onto the subspace of solutions that satisfy linear constraints. Other methods have employed alternative approaches, such as computing full projections by enforcing linear constraints or using the gradient of quadratically penalized constraints[33][34]. Recent work has extended the application of diffusion models to more complex non-convex constraint functions. In these cases, iterative methods such as gradient descent are utilized to guide the sampling process toward satisfying the constraints, as demonstrated in Universal Guidance[35]. Furthermore, a provably robust framework for score-based diffusion models applied to image reconstruction was introduced by[36], offering robust performance in handling nonlinear inverse problems while ensuring consistency with the observed data.

3. Methods

3.1. Noise Level as Distance from Noisy Samples to the Manifold

This work builds on the insight that denoising in diffusion models can be interpreted as an approximate projection onto the support of the training-set distribution. Previous studies[15][16] have established this connection. The distance function of a set $\mathcal{K} \subset \mathbb{R}^n$, denoted as $\mathrm{dist}_{\mathcal{K}}(x)$, is defined as the minimum distance from a point x to any point $x_0 \in \mathcal{K}$:

$$\mathrm{dist}_{\mathcal{K}}(x) := \inf\{\|x - x_0\| : x_0 \in \mathcal{K}\}. \tag{10}$$

The projection of x onto $\mathcal{K}$, denoted $\mathrm{proj}_{\mathcal{K}}(x)$, refers to the point (or points) on $\mathcal{K}$ that achieves this minimum distance. Assuming the projection is unique, we can express it as:

$$\mathrm{proj}_{\mathcal{K}}(x) := \{x_0 \in \mathcal{K} : \mathrm{dist}_{\mathcal{K}}(x) = \|x - x_0\|\}. \tag{11}$$

The **Manifold Hypothesis** suggests that many real-world datasets lie approximately on low-dimensional manifolds embedded in high-dimensional spaces[37][38]. In this context, we assume that $\mathcal{K}$ is a manifold of dimension d, where $d \ll n$. Leveraging this assumption, we outline the following informal theorem, which summarizes results from[16] (for formal proofs refer to[16]):

Theorem 1 (Informal) *Let $x_0 \in \mathcal{K}$, $\sigma_t > 0$, and $\epsilon \sim \mathcal{N}(0, I)$. Define $x_t = x_0 + \sigma_t \epsilon \in \mathbb{R}^n$. Assuming the manifold hypothesis holds:*

a. *Denoising as Approximate Projection:*

$$\|x_t - \sigma_t \epsilon_\theta(x_t, \sigma_t) - \mathrm{proj}_{\mathcal{K}}(x_t)\| \leq \eta \mathrm{dist}_{\mathcal{K}}(x_t) \tag{12}$$

when (x_t, σ_t) satisfies: $\frac{1}{\nu}\mathrm{dist}_{\mathcal{K}}(x_t) \leq \sqrt{n}\sigma_t \leq \nu \mathrm{dist}_{\mathcal{K}}(x_t)$ for constants $1 > \eta \geq 0$ and $\nu \geq 1$.

b. *Noise Level as Approximate Distance:*

$$\sqrt{n}\sigma_t \approx \mathrm{dist}_{\mathcal{K}}(x_t) \tag{13}$$

c. *Projection Interpretation via DDIM: Assume $\epsilon^*_\theta(x, \sigma)$ is the zero-error denoiser, and the initial distance satisfies $\mathrm{dist}_{\mathcal{K}}(x_T) = \sqrt{n}\sigma_T$. Then the DDIM sampler generates the sequence $(x_T, \ldots, x_0)$ by performing gradient descent on the objective function $f(x) := \frac{1}{2}\mathrm{dist}_{\mathcal{K}}(x)^2$ with a step-size of $\beta_t := 1 - \sigma_{t-1}/\sigma_t$:*

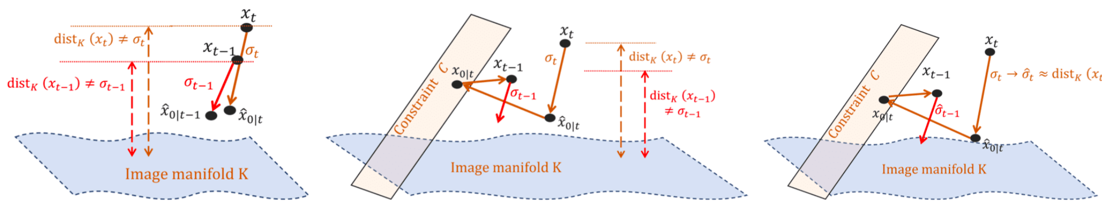
$$x_{t-1} = x_t - \beta_t \nabla f(x_t) = x_t - \beta_t \cdot \mathrm{dist}_{\mathcal{K}}(x_t) \cdot \nabla \mathrm{dist}_{\mathcal{K}}(x_t), \tag{14}$$

$$\mathrm{dist}_{\mathcal{K}}(x_t) = \sqrt{n}\sigma_t, \quad \nabla \mathrm{dist}_{\mathcal{K}}(x_t) = \epsilon^*_\theta(x_t, \sigma_t)/\sqrt{n} \tag{15}$$

Theorem 1(a) demonstrates that the estimated clean sample generated by the denoising process $\hat{x}_{0|t} = x_t \sigma_t \epsilon_\theta(x_t, \sigma_t)$, serves as an approximation of the projection of the noisy sample x_t onto the

doi.org/10.32388/NSYEO2

manifold $\mathcal{K}$. Similarly, Theorem 1(b) establishes that the distance of a noisy sample x_t onto manifold $\mathcal{K}$, can be approximated by the noise level $\sqrt{n}\sigma_t$. With a zero-error denoiser, Theorem 1(c) shows that the DDIM process generates a sequence that monotonically decreases the distance to the manifold by following the gradient descent of the objective function $f(x)$. Specifically, the distance of each noisy sample from the manifold is determined by the noise level, $\text{dist}_{\mathcal{K}}(x_t) = \sqrt{n}\sigma_t$, while the direction of the denoising process, or projection, is guided by the estimated noise vector $\nabla \text{dist}_{\mathcal{K}}(x_t) = \epsilon^*_\theta(x_t, \sigma_t)/\sqrt{n}$.



(a) Denoising Process (b) Constrained Denoising (c) Denoising with NLC

Figure 2. Denoising approximates projection. (a) Denoising process using DDIM. (b) Constrained denoising process viewed as an alternative projection. Note that in both (a) and (b), the estimated noise level σ_t does not always match the distance $\text{dist}_{\mathcal{K}}(x_t)$. (c) Constrained denoising process with noise level correction (NLC). By replacing the prior noise level σ_t with the more accurate noise level $\hat{\sigma}_t$, the projection more closely aligns with the manifold $\mathcal{K}$.

However Theorem 1(c), (15), relies on two key assumptions: (1) an zero-error denoiser, where $\epsilon_\theta(x_t, \sigma_t) = \epsilon_t$, and (2) an initial condition where the distance at $t = T$, $\text{dist}_{\mathcal{K}}(x_T) = \sqrt{n}\sigma_T$. For sample generation, cumulative errors introduced during the denoising process can lead to biases due to the imperfections of the denoiser[39]. These errors become particularly significant by the final steps, where deviations in the denoising process result in a large mismatch between the true distance $\text{dist}_{\mathcal{K}}(x_t)$ and the estimated noise level $\sqrt{n}\sigma_t$, as illustrated in Figure 2a. As observed, a mismatch between the true distance and the estimated noise level can lead to the clean image estimation using eq. (5) falling outside the manifold $\mathcal{K}$. In constrained sample generation tasks, such as image restoration and guided sample generation, this issue is further exacerbated. As shown in Figure 2b, deviations introduced by guidance terms or projection steps onto the constraint $\mathcal{C}$ amplify the discrepancy between the estimated noise level $\sqrt{n}\sigma_t$ and the actual distance $\text{dist}_{\mathcal{K}}(x_t)$. These deviations are particularly impactful in the later denoising steps and can prevent the reconstructed sample from accurately lying on the data manifold $\mathcal{K}$. Additional details are provided in Appendix A.

 doi.org/10.32388/NSYEO2

3.2. Noise Level Correction

To address the issue of inaccurate distance estimation caused by relying on a predefined noise level σ_t, as discussed in section 3.1, we propose a method called **noise level correction** to better align the corrected noise level with the true distance. This approach replaces the predefined noise level scheduler σ_t with a corrected noise level $\hat{\sigma}_t$ during the denoising process, enabling more accurate distance estimation and improved sample quality. As shown in Figure 2c, using the corrected noise level brings the estimated clean sample $\hat{x}_{0|t}$ closer to the data manifold $\mathcal{K}$ compared to the naive denoising process that relies solely on σ_t. The corrected noise level is defined as $\hat{\sigma}_t := \sigma_t[1+\hat{r}_t] \approx \mathrm{dist}_{\mathcal{K}}(x_t)/\sqrt{n}$, where $\hat{r}_t$ represents the residual and can be modeled using either a neural network or a non-parametric function. This residual alignment approach is effective because the residual $\hat{r}_t$ is stable across noise levels, making it easier to model, while the noise level itself may become unbounded during large diffusion time steps.

For $\mathrm{dist}_{\mathcal{K}}(x_t)$, calculating the ground-truth distance to the manifold is generally not feasible. Instead, we approximate it using the distance between the noisy sample and its clean counterpart in the forward diffusion process. Specifically, given the noisy samples x_t generated by the diffusion process from $x_0 \in \mathcal{K}$ in eq. (2), we estimate the distance as $\mathrm{dist}_{\mathcal{K}}(x_t) \approx |x_t - x_0|$. This approximation is reasonable because, according to the manifold hypothesis, the random noise ϵ is orthogonal to the manifold $\mathcal{K}$. Therefore, the projection of x_t onto $\mathcal{K}$ satisfies $\mathrm{proj}_{\mathcal{K}}(x_t) = \mathrm{proj}_{\mathcal{K}}(x_0 + \sigma_t \epsilon) \approx x_0$, as also illustrated in[16].

We introduce a neural network to learn $\hat{r}_t = r_\theta(x_t, \sigma_t)$ for noise level correction. To minimize additional computational costs, we design the noise level correction network $r_\theta(\cdot)$ to be small and efficient. It leverages the encoder module of the denoiser's UNet architecture, followed by compact layers that fully utilize the pre-trained denoiser's capabilities. As shown in Figure 3, the denoiser network uses a UNet structure to estimate the noise vector (denoising direction) based on the noisy image x_t and σ_t. Meanwhile, the noise level correction network utilizes the shared encoder, followed by additional neural network blocks, to predict the residual noise level. We train the noise level correction network $r_\theta(\cdot)$ alongside a fixed, pre-trained denoiser $\epsilon_\theta(\cdot)$, ensuring coordinated improvement in denoising accuracy. In training, to further enhance the noise level correction network, we introduce a scaling factor λ to expand the input-output space of $r_\theta(x_t, \sigma_t)$ within the diffusion process. The objective function for noise level correction is defined as:

$$L_{r_\theta} := \mathbb{E}\left[\|\sqrt{n}\sigma_t[1 + r_\theta(\hat{x}_t, \sigma_t)] - \sigma_t \lambda \|\epsilon_t\|\|\right] \tag{16}$$

$$\hat{x}_t = x_0 + \sigma_t \lambda \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I), \quad \lambda \sim \mathcal{U}(1-\delta, 1+\delta) \tag{17}$$

The scaling factor λ is sampled from a uniform distribution $\mathcal{U}(1-\delta, 1+\delta)$, with $\delta = 0.5$ in our experiment, to control the level of variation introduced into the noise level correction.

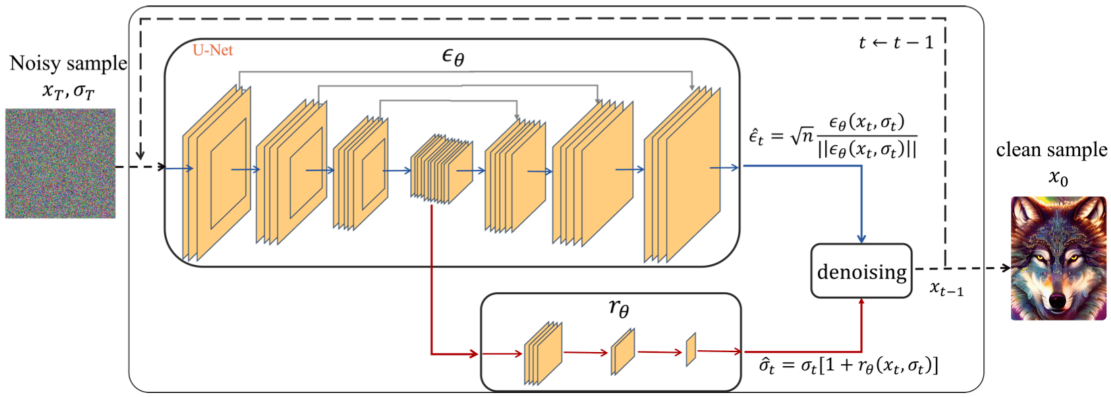


Figure 3. Neural Network Architecture

3.3. Enhancing Sample Generation with Noise Level Correction

The trained noise level correction network r_θ can be integrated into various existing sampling algorithms to improve sample quality. For algorithms that include an initial sample estimate $x_{0|t}$, eq. (5), we can reformulate the one-step estimation as follows:

$$\hat{x}_{0|t} = x_t - \hat{\sigma}_t \hat{\epsilon}_t \tag{18}$$

$$\hat{\sigma}_t = \sigma_t[1 + r_\theta(\hat{x}_t, \sigma_t)], \quad \hat{\epsilon}_t = \sqrt{n}\frac{\epsilon_\theta(x_t, \hat{\sigma}_t)}{\|\epsilon_\theta(x_t, \hat{\sigma}_t)\|} \tag{19}$$

We normalize the noise vector $\epsilon_\theta(\cdot)$ during the sampling as in eq. (19), process to decouple noise level (magnitude) correction $\hat{\sigma}_t$ from direction estimation $\hat{\epsilon}_t$. Empirical experiments show that normalizing $\epsilon_\theta(\cdot)$ and using $\hat{\sigma}_t$ to account for magnitude yields better results compared to not normalizing $\epsilon_\theta(\cdot)$. In the training loss function eq. (16), normalization of the noise vector ϵ is unnecessary because the randomly sampled noise ϵ naturally concentrates around the norm $\sqrt{n}$. However, the neural network-estimated noise vector $\epsilon_\theta(\cdot)$ does not maintain a constant norm.

Using eq. (19), we integrate noise level correction into the DDIM and DDPM sampling algorithms, as illustrated in Algorithm 1, with the modifications from the original DDIM/DDPM algorithms highlighted in blue. In this algorithm, lines 3 and 4 represent the current and next-step noise level corrections, respectively, while line 5 provides the normalized noise vector. Lines 6 through 8 follow the steps of the original DDIM and DDPM algorithms. Similarly, noise level correction can be integrated into the EDM sampling algorithm, as shown in Algorithm 3. Note that in EDM with noise level correction, we do not normalize the noise vector $\epsilon_\theta(x_t, \hat{\sigma}_t)$, since EDM employs a second-order Heun solver to improve noise vector estimation. By incorporating noise level correction, these algorithms produce higher-quality

doi.org/10.32388/NSYEO2

samples with improved accuracy by considering both the direction and distance to the data manifold during the denoising process.

Algorithm 1 DDIM/DDPM with Noise Level Correction (DDIM/DDPM-NLC)

Input: Denoiser ϵ_θ and noise level corrector r_θ
Input: Noise scheduler σ_t, randomness scale η ($\eta = 0$ for deterministic DDIM and $\eta = 1$ for DDPM)
Output: samples $x_0 \in \mathcal{K}$

1: $x_T = \sqrt{\sigma_T^2 + 1} \cdot z_T,\ \ z_T \sim \mathcal{N}(0, I),$
2: **for** $t = T, T-1, \cdots, 1$ **do**
3: $\quad \hat{\sigma}_t = \sigma_t[1 + r_\theta(x_t, \sigma_{t)})]$
4: $\quad \hat{\sigma}_{t-1} = \hat{\sigma}_t \frac{\sigma_{t-1}}{\sigma_t}$
5: $\quad \hat{\epsilon}_t = \sqrt{n}\epsilon_\theta(x_t, \hat{\sigma}_t)/\|\epsilon_\theta(x_t, \hat{\sigma}_t)\|$
6: $\quad \sigma_{noise} = \eta \frac{\hat{\sigma}_{t-1}}{\hat{\sigma}_t}\sqrt{\hat{\sigma}_t^2 - \hat{\sigma}_{t-1}^2}$
7: $\quad \sigma_{signal} = \sqrt{\hat{\sigma}_{t-1}^2 - \sigma_{noise}^2}$
8: $\quad x_{t-1} = x_t + (\sigma_{signal} - \hat{\sigma}_t)\hat{\epsilon}_t + \sigma_{noise}\omega_t,$ where $\omega_t \sim \mathcal{N}(0, I)$
9: **end for**

3.4. Constrained Sample Generation

The noise level correction method can also improve performance in constrained sample generation tasks, such as image restoration. Let $\mathcal{K}$ denote the data manifold, and let $\mathcal{C}$ represent specific constraints, such as masked pixel matching in inpainting tasks. Constrained sample generation aims to generate samples x that satisfy both the manifold and constraint requirements, meaning $x \in \mathcal{K} \cap \mathcal{C}$. Similar to DDIM-NLC sampling approach in Algorithm 1, noise level correction can be incorporated into existing constrained sample generation methods, such as DDNM[22], a DDIM-based image restoration algorithm. An example of this is shown in Algorithm 4.

To further enhance constrained sample generation, we propose a flexible iterative projection algorithm inspired by the alternating projection technique[40][41]. This approach iteratively projects samples onto each constraint set to approximate a solution that lies in the intersection of $\mathcal{K}$ and $\mathcal{C}$. The iterative projection process can be expressed as follows:

$$\hat{x}_{0|k} = \mathrm{proj}_{\mathcal{K}}(x_{(k)}), \quad x_{0|k} = \mathrm{proj}_{\mathcal{C}}(\hat{x}_{0|k}), \quad x_{(k+1)} = x_{0|k} + \bar{\epsilon}, \quad k = 0, 1, \cdots \tag{20}$$

Where $\hat{x}_{0|k}$ and $x_{0|k}$ represent the k-th estimates of points satisfying $x \in \mathcal{K}$ and $x \in \mathcal{C}$, respectively. The iterative rule ensures that $x_{0|k}$ approximates a point in $\mathcal{K} \cap \mathcal{C}$. x_{k+1} introduces a small noise term $\bar{\epsilon}$, which helps avoid convergence to local minima in non-flat regions. This noise term is analogous to the one used in DDPM (eq. 8) and facilitates iterative refinement toward the final clean samples. The noise term $\bar{\epsilon}$ can be gradually reduced over iterations or set to zero once a satisfactory iteration $k = K_{\max}$ is achieved. At this point, the algorithm returns a final estimate such that $x_{K_{\max}} \in \mathcal{K} \cap \mathcal{C}$.

 doi.org/10.32388/NSYEO2

Considering diffusion models, Theorem 1 demonstrates that the projection operator $\mathrm{proj}_{\mathcal{K}}(\cdot)$ can be approximately computed using the denoising operator. Starting from an initial random point $x_0 = \sigma_{\max}\epsilon$, the projection onto the manifold $\mathcal{K}$ can be iteratively refined using eq. (18) and eq. (19), as follows:

$$\mathrm{proj}_{\mathcal{K}}(x_{(k)}) = \hat{x}_{0|k} = x_{(k)} - \hat{\sigma}_{(k)}\,\hat{\epsilon}_{(k)} \tag{21}$$

For the additional constraint projection, $\mathrm{proj}_{\mathcal{C}}(x)$, the specific calculation depends on the nature of the constraint. The constraints for many image restoration tasks are linear, including inpainting, colorization, super-resolution, deblurring, and compressed sensing. For tasks with linear constraints, the projection can be computed directly or optimized using gradient descent. Consider an image restoration task formulated as $y = \mathrm{A}x_0$, where x_0 represents the ground-truth image, y is the degraded observation, A is the linear degradation operator. Given the degraded image y and the current estimate $\hat{x}_{0|k}$, the projection onto the constraint can be computed as:

$$x_{0|k} = \mathrm{proj}_{\mathcal{C}}(\hat{x}_{0|k}) = \mathrm{A}^{\dagger}y + (\mathrm{I} - \mathrm{A}^{\dagger}\mathrm{A})\hat{x}_{0|k} \tag{22}$$

Where $\mathrm{A}^{\dagger}$ is the pseudo-inverse of A.In this work, we adopt the values of $\mathrm{A}^{\dagger}$ for image restoration tasks as provided in [22]. Here, $x_{0|k}$ is the $k-$th th estimate satisfying $x_0 \in \mathcal{K} \cap \mathcal{C}$. This iterative estimation requires a predefined noise scheduler $\sigma_1, \ldots, \sigma_{(k)}, \ldots$ to generate $x_{0|k}$ and $x_{(k+1)}$. To allow flexible and potentially unlimited refinement steps, we define the noise schedule $\sigma_{(k)}$ with a maximum noise level $\sigma_{(0)} = \sigma_{max}$ and a minimum level σ_{min}, decaying by a predefined factor $\eta < 1$. If the noise level reaches σ_{min}, the process can either stop with returning $x_{0|k}$ or restart from $\sigma_{restart}$. This strategy permits an arbitrary number of refinement steps, stopping either at a desired loss threshold or continuing indefinitely. Since $\sigma_{(k)}$ represents the distance of noisy samples from the manifold, this decaying schedule incrementally reduces $x_{(k)}$'s distance from the manifold.

doi.org/10.32388/NSYEO2

Algorithm 2 Constrained Sample Generation with Noise Level Correction (IterProj-NLC)

Input: Denoiser ϵ_θ and noise level corrector r_θ
Input: Constraint $\mathcal{C}$, distance decay α, noise scale η
Input: σ_{max}, σ_{min}, and $\sigma_{restart}$, and maximum iterations K_{max}
Output: samples $x_{0|K} \in \mathcal{K} \cap \mathcal{C}$

1: $x_{(0)} = \sigma_{(0)}\epsilon,\ \epsilon \sim \mathcal{N}(0, I),\ \sigma_{(0)} = \sigma_{max}$
2: **for** $k = 0, 1, 2, \cdots,$ **do**
3: $\quad \hat{\sigma}_{(k)} = \sigma_{(k)}[1 + r_\theta(x_{(k)}, \sigma_{(k)})]$
4: $\quad \hat{\epsilon}_{(k)} = \sqrt{n}\epsilon_\theta(x_{(k)}, \hat{\sigma}_{(k)})/\|\epsilon_\theta(x_{(k)}, \hat{\sigma}_{(k)})\|$
5: $\quad \hat{x}_{0|k} = x_{(k)} - \hat{\sigma}_{(k)}\hat{\epsilon}_{(k)}$
6: $\quad x_{0|k} = \mathrm{Proj}_{\mathcal{C}}(\hat{x}_{0|k})$ (For image restoration tasks, refer eq. (22))
7: $\quad \sigma_{(k+1)} = \alpha\sigma_{(k)}$
8: $\quad$ **if** $\sigma_{(k+1)} < \sigma_{min}$ **then**
9: $\quad\quad \sigma_{(k+1)} = \sigma_{restart}$
10: $\quad$ **end**
11: $\quad \tilde{\epsilon}_{(k)} = \sqrt{1 - \eta^2}\hat{\epsilon}_{(k)} + \eta\epsilon,$ where $\epsilon \sim \mathcal{N}(0, I)$
12: $\quad x_{(k+1)} = x_{0|k} + \sigma_{(k+1)}\tilde{\epsilon}_{(k)}$
13: $\quad$ **if** $k \geq K_{max}$ **or** $\|x_{0|k} - x_{0|k-1}\|$ is small enough **then**
14: $\quad\quad$ **return** $x_{0|k}$
15: $\quad$ **end**
16: **end for**

Algorithm 2 presents the proposed constrained generation approach, termed IterProj-NLC (Iterative Projection with Noise Level Correction). Lines 3 to 5 project onto the data manifold $\mathcal{K}$, while Line 6 projects onto the constraint set $\mathcal{C}$. Lines 7 to 10 update the noise level, and Lines 11 and 12 compute the next noisy sample $x_{(k+1)}$ from the current clean estimate $x_{k|0}$. This step also acts as a convex combination of the current clean estimate $x_{k|0}$ and the previous noisy sample $x_{(k)}$.

4. Experiments

4.1. Toy Experiments

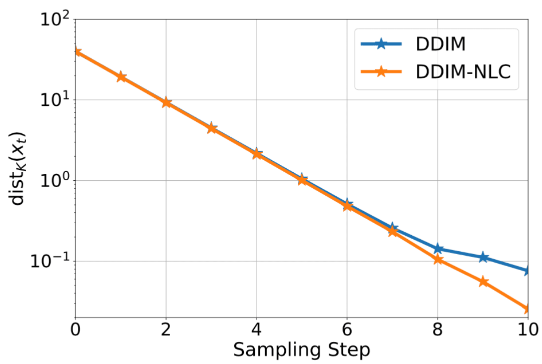
We conducted a toy experiment to demonstrate the effectiveness of noise level correction in sample generation for diffusion models. The objective was to generate samples on d-sphere manifold. The toy training dataset was sampled from d-dimensional sphere manifold embedded within an n-dimensional data space, where $d < n$. Detailed experimental design information is available in Appendix C.1.

After training, we applied the proposed 10-step DDIM with Noise Level Correction (DDIM-NLC), as detailed in Algorithm 1 to generate samples. This method was compared to the 10-step DDIM baseline. Our evaluation metric measured the distance between the generated samples and the ground-truth d-sphere manifold $\mathcal{K}$ (where lower distance indicates better results). Figure 4a presents the sample quality (measured as the distance to the manifold) for different methods across each denoising step. As shown, DDIM-NLC outperforms the baseline DDIM by generating samples that are consistently closer to the target manifold.

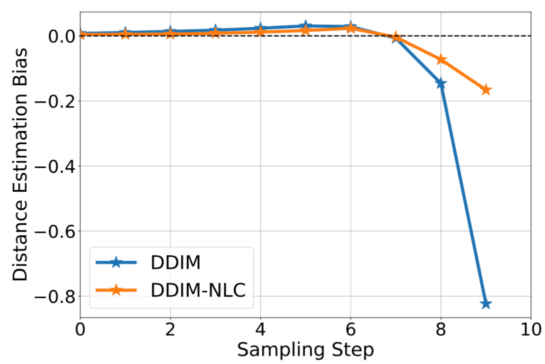
As we discussed in Theorom 1, the distance to the manifold can be approximated by the noise level, as shown in eq. (13). We further evaluated the accuracy of the noise level correction by examining its proximity to the actual distance $\text{dist}_{\mathcal{K}}(\hat{x}_t)$. Figure 4b displays the relative distance estimation bias during sampling, calculated as

$$\text{Distance Estimation Bias} = \frac{\text{dist}_{\mathcal{K}}(\hat{x}_t) - \sqrt{n}\hat{\sigma}_t}{\sqrt{n}\sigma_t}.$$

Where $\hat{\sigma}_t = \sigma_t$ for DDIM, and $\hat{\sigma}_t = \sigma_t[1 + r_\theta(\hat{x}_t, \sigma_t)]$ for DDIM-NLC. The results show that DDIM-NLC achieves a significantly smaller distance estimation bias than DDIM, particularly in the later steps as samples approach the manifold. The results of constrained sample generation are shown in Appendix C.2.



(a) Sample Quality



(b) Distance Estimation Bias

Figure 4. Results from experiments on toy models. (a) Generated sample quality was evaluated the distance to the manifold. (b) Distance Estimation Error.

4.2. Unconstrained Image Generation

We conducted experiments to evaluate the effectiveness of noise level correction in unconstrained image generation tasks. The noise level correction network was trained on top of a pre-trained denoiser network. Notably, the noise level correction network is approximately ten times smaller than the denoiser network. Additional details on the experimental setup are provided in Appendix D.1. We used the FID (Fréchet Inception Distance) score as the evaluation metric to assess the quality of generated samples, where lower scores indicate better quality, following standard practice in image generation tasks[42]. The experimental results for the DDIM/DDPM framework on the CIFAR-10 dataset are shown in Table 1. As observed, DDPM-NLC and DDIM-NLC, which incorporate noise level correction, outperform the original DDPM and DDIM models across all sampling steps. Specifically, our proposed noise level correction approach improves

doi.org/10.32388/NSYEO2

DDIM performance by 32%, 31%, 22%, 33%, and 28% for 10, 20, 50, 100, and 300 sampling steps, respectively.

Method\Step	**1000**	**300**	**100**	**50**	**20**	**10**
DDPM	2.99	2.95	3.37	4.43	10.41	23.19
DDPM-NLC	**2.35**	**2.21**	**2.39**	**2.74**	6.44	19.27
DDIM	4.29	4.32	4.66	5.17	8.25	14.21
DDIM-NLC	3.11	3.11	3.12	4.04	**5.66**	**9.61**

Table 1. FID on DDIM/DDPM sampling on CIFAR-10 with and without noise level correction.

We evaluate the effectiveness of noise level correction with EDM[24], as it achieves state-of-the-art sampling quality with few sampling steps. Specifically, we assess the impact of NLC using both a first-order Euler ODE solver and a second-order Heun ODE solver within the EDM framework. As shown in Table 2, the proposed NLC also enhances the performance of EDM-based sampling methods. Notably, as a robust sampling technique, noise level correction improves the performance of the Heun sampler by 10% with just 13 sampling steps.

Method\Step	**35**	**21**	**13**
Euler	3.81	6.29	12.28
Euler-NLC	2.79	4.21	8.17
Heun	1.98	2.33	7.22
Heun-NLC	**1.95**	**2.22**	**6.56**

Table 2. FID on EDM sampling on CIFAR-10 with and w/o NLC.

doi.org/10.32388/NSYEO2

4.3. Image Restoration

In this section, we evaluate the effectiveness of noise level correction on five common image restoration tasks, $4\times$ super-resolution (SR) using bicubic downsampling, deblurring with a Gaussian blur kernel, colorization using an average grayscale operator, compressed sensing (CS) with a Walsh-Hadamard sampling matrix at a 0.25 compression ratio, and inpainting with text masks. These experiments are conducted on the ImageNet[43] and CelebA-HQ[44] datasets. We compare our method with recent state-of-the-art diffusion-based image restoration methods, including ILVR[45], RePaint[46], DDRM[32], and DDNM[22]. For a fair comparison, all diffusion-based methods utilize the same pretrained denoising networks with the same 100-step denoising process (100 number of inference steps), following the experimental setup in[22]. To evaluate sample quality, we use FID, PSNR (Peak Signal-to-Noise Ratio), and SSIM (Structural Similarity Index Measure). For colorization, where PSNR and SSIM are less effective metrics[22], we additionally use a Consistency metric, denoted as "Cons" and calculated as $\|Ax_0 - y\|_1$. As a baseline, we also include the inverse solution for each image restoration task, given by $\hat{x} = \mathrm{A}^\dagger y$, which achieves zero constraint violation but lacks the data manifold information.

The results on the ImageNet dataset are summarized in Table 3, while those for CelebA-HQ are shown in Table 4. Tasks not supported by certain methods are marked as "N/A." As the results indicate, integrating noise level correction (as in Algorithm 4) enhances sample generation performance for DDNM. Furthermore, the proposed IterProj-NLC method (Algorithm 2), achieves the best performance across all benchmarks. For instance, IterProj-NLC outperforms the baseline DDNM in FID score by 6%, 59%, 14%, and 50% on $4\times$ SR, Deblurring, CS 25%, and Inpainting tasks, respectively. It also improves Consistency in colorization by 9%. Qualitative comparisons are shown in Figure 1, with additional results in Appendix E.1.

doi.org/10.32388/NSYEO2

ImageNet	4 x SR	Deblurring	Colorization	CS 25%	Inpainting
Method	PSNR↑/SSIM↑/FID↓	PSNR↑/SSIM↑/FID↓	Cons↓/FID↓	PSNR↑/SSIM↑/FID↓	PSNR↑/SSIM↑/FID↓
$\mathbf{A}^{\dagger}y$	24.26 / 0.684 / 134.4	18.46 / 0.6616 / 55.42	0.0 / 43.37	15.65 / 0.510 / 277.4	14.52/ 0.799 / 72.71
ILVR	27.40 / 0.870 / 43.66	N/A	N/A	N/A	N/A
RePaint	N/A	N/A	N/A	N/A	31.87 / 0.968 / 13.43
DDRM	27.38 / 0.869 / 43.15	43.01 / 0.992 / 1.48	260.4 / 36.56	19.95 / 0.704 / 97.99	31.73 / 0.966 / 10.82
DDNM	27.45 / 0.870/ 39.56	44.93 / 0.993 / 1.17	42.32 / 36.32	21.62 / 0.748 / 64.68	31.60 / 0.946 / 9.79
DDNM-NLC	27.50 / 0.872 / 37.82	46.20 / 0.995 / 0.79	41.60 / 35.89	21.27 / 0.769 / 58.96	32.51 / 0.957 / 7.20
IterProj-NLC	**27.56 / 0.873 / 37.48**	**48.24 / 0.997 / 0.48**	**38.30/ 35.66**	**22.27 / 0.771 / 55.69**	**33.58 / 0.966 / 4.90**

Table 3. Comparative results of five image restoration tasks on ImageNet.

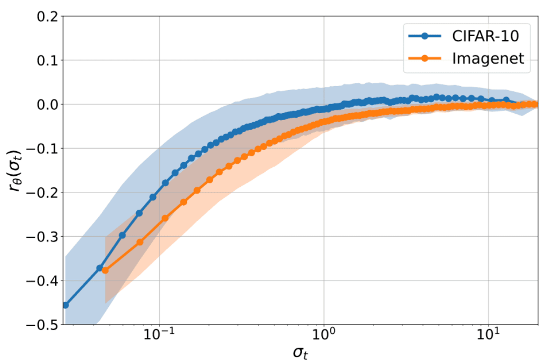
Celeba-HQ	4 x SR	Deblurring	Colorization	CS 25%	Inpainting
Method	PSNR↑/SSIM↑/FID↓	PSNR↑/SSIM↑/FID↓	Cons↓/FID↓	PSNR↑/SSIM↑/FID↓	PSNR↑/SSIM↑/FID↓
$\mathbf{A}^{\dagger}y$	27.27 / 0.782 / 103.3	18.85 / 0.741 / 54.31	0.0 / 68.81	15.09 / 0.583 / 377.7	15.57 / 0.809 / 181.56
ILVR	31.59 / 0.945 / 29.82	N/A	N/A	N/A	N/A
RePaint	N/A	N/A	N/A	N/A	35.20 / 0.981 /18.21
DDRM	31.63 / 0.945 / 31.04	43.07 / 0.993 / 6.24	455.9 / 31.26	24.86 / 0.876 / 46.77	34.79 / 0.978 /16.35
DDNM	31.63 / 0.945 / 22.50	46.72 / 0.996 / 1.42	26.25 / 26.78	27.52 / 0.909 / 28.80	35.64 / 0.979 / 12.21
DDNM-NLC	31.78 / 0.947 / 22.10	46.78 / 0.997 / 1.36	24.92 / 25.81	27.63 / 0.914 / 24.72	36.48 / 0.980 / 11.60
IterProj-NLC	**31.93 / 0.949 / 21.96**	**46.97 / 0.997 / 1.29**	**24.65 / 25.30**	**27.78 / 0.916 / 23.45**	**36.57 / 0.981 / 11.07**

Table 4. Comparative results of five image restoration tasks on Celeba-HQ.

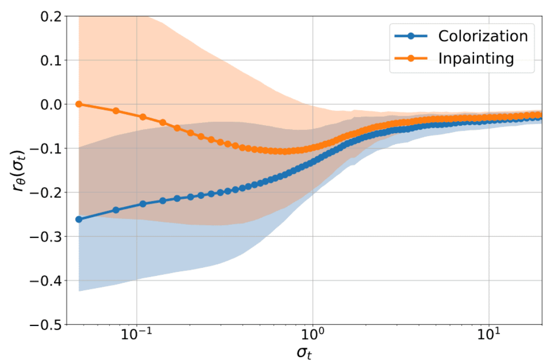
doi.org/10.32388/NSYEO2

4.4. Lookup Table for Noise Level Correction

In this section, we explore the statistical properties of the noise level correction network and demonstrate how these properties can be leveraged to create a lookup table for correcting noise levels without neural network inference. The lookup table for noise level correction is defined as $\hat{\sigma}_t = \sigma_t[1 + \hat{r}_t]$ where $\hat{r}_t$ is a non-parametric function that approximates the actual distance to the data manifold. As illustrated in the toy experiment shown in Figure 4b, distance estimation error using noise levels is lower in the initial sampling steps and increases in later stages when the true distance to the manifold decreases. This trend is expected: in the early stages, noisy samples are farther from the manifold, making approximate projections easier and reducing relative distance estimation error. More specifically, at the initial steps, the true distance $\mathrm{dist}_{\mathcal{K}}(x_t)$ is slightly larger than the estimation from the noise level $\sqrt{n}\sigma_t$ as supported by eq. (24). In later steps, however, $\mathrm{dist}_{\mathcal{K}}(x_t)$ decreases more rapidly than $\sqrt{n}\sigma_t$.



(a) Unconstrained sample generation



(b) Constrained sample generation

Figure 5. Plot of $r_\theta(\sigma_t)$ versus σ_t in the unconstrained DDIM-NLC denoising process and constrained DDNM-NLC denoising process. The curve represents the average over samples, with shaded regions indicating the standard deviation. The larger variance (right) illustrates that the corrections applied by $r_\theta(\sigma_t)$ are too complex for a simple look-up table in the context of constrained generation.

We conducted an experiment to analyze the statistical behavior of the neural network-based noise level corrector $r_\theta(\cdot)$ for unconstrained sample generation on the CIFAR-10 and ImageNet datasets. Figure 5a presents the relationship between $r_\theta(\sigma_t)$ and σ_t, averaged over the samples x_t during the DDIM-NLC denoising process. As seen, $r_\theta(\sigma_t)$ values are negative for smaller σ_t corresponding to the final denoising steps (higher time steps t), and increase as σ_t increases. This trend aligns with the observation in the toy experiment Figure 4b, indicating that distance decreases in the final steps and thus requires reducing $\hat{\sigma}_t$ for accurate distance representation. Moreover, a similar trend is observed across different datasets, such as CIFAR-10 and ImageNet. We further analyzed the statistical behavior of $r_\theta(\cdot)$ in constrained sample generation tasks on the ImageNet dataset. These tasks introduce additional variability due to constraint projections [22], resulting in higher variance in $r_\theta(\cdot)$ across samples. Notably, even within the same dataset, constraints such as colorization and inpainting exhibit distinct trends during the final denoising steps (i.e., at small noise levels). Moreover, the variance at small noise levels is substantially higher in constrained tasks compared to unconstrained scenarios.

Using the values of $r_\theta(\sigma_t)$ recorded in the average value curve of Figure 5, we created a lookup table-based noise level correction (LT-NLC) search $\hat{r}_t$ to estimate $r_\theta(\sigma_t)$. We evaluated the effectiveness of LT-NLC in unconstrained sample generation tasks. The experimental results for LT-NLC applied to the DDIM framework on the CIFAR-10 dataset are shown in Table 5. As expected, the trained noise level correction (NLC) achieves the best performance. However, LT-NLC also significantly improves the original DDIM, enhancing performance by 14%, 20%, and 15% for 10, 20, and 50 sampling steps, respectively. The results for LT-NLC applied to the EDM framework on the CIFAR-10 dataset are presented in Table 6. Similar to the DDIM results, LT-NLC improves the performance of EDM-based sampling methods, demonstrating its effectiveness as a network inference-free enhancement. The results for constrained generation can be found in Appendix D.3. As illustrated in Figure 5, the variance of $r_\theta(\sigma_t)$ in constrained generation tasks, such as image restoration, is significantly higher. Consequently, the performance improvements achieved by LT-NLC are smaller compared to those of the neural network-based NLC, as LT-NLC applies the same correction across all samples. Therefore, in constrained image generation tasks, the neural network-based NLC remains essential for achieving optimal performance.

Method\Step	**1000**	**300**	**100**	**50**	**20**	**10**
DDIM	4.29	4.32	4.66	5.17	8.25	14.21
DDIM-LT-NLC	4.01	3.97	3.83	4.37	6.54	11.21
DDIM-NLC	3.11	3.11	3.12	4.04	5.66	9.61

Table 5. FID on DDIM sampling on CIFAR-10 with lookup table noise level correction.

doi.org/10.32388/NSYEO2

Method\Step	35	21	13
Heun	1.98	2.33	7.22
Heun-LT-NLC	1.97	2.27	6.84
Heun-NLC	1.95	2.22	6.56

Table 6. FID on EDM sampling on CIFAR-10 with and with LT-NLC.

The noise level correction (NLC) network is significantly smaller than the denoiser, resulting in minimal additional computational overhead. Detailed comparisons of the training and inference times for the proposed NLC method are provided in Appendix D.2.

5. Conclusions

In this work, we explore the relationship between noise levels in diffusion models and the distance of noisy samples from the underlying data manifold. Building on this insight, we propose a novel noise level correction method, utilizing a neural network to align the corrected noise level with the true distance of noisy samples to the data manifold. This alignment significantly improves sample generation quality. We further extend this approach to constrained sample generation tasks, such as image restoration, within an alternating projection framework. Extensive experiments on both unconstrained and constrained image generation tasks validate the effectiveness of the proposed noise level correction network. Additionally, we introduce a lookup table-based approximation for noise level correction. This parameter-free method effectively enhances performance in various unconstrained sample generation tasks, offering a computationally efficient alternative to the neural network-based approach.

Appendix A. Discrepancy between $\sqrt{n}\sigma_t$ and the Distance $\mathrm{dist}_{\mathcal{K}}(x_t)$

Previous works[7][24] have primarily focused on improving the estimation of the noise vector $\epsilon_\theta(x_t, \sigma_t) \approx \epsilon_t$ eq. (4), aligning with the first assumption of a zero-error denoiser. However, these approaches typically rely on a predefined noise level scheduler σ_t for distance estimation, often without explicit validation. This reliance can lead to inaccuracies, even at the initial step, where $\mathrm{dist}_{\mathcal{K}}(x_T) \neq \sqrt{n}\sigma_T$.

Consider the DDIM as an example. The initial noisy point x_T is sampled as follows:

$$x_T = \frac{z_T}{\sqrt{\alpha_t}} = \sqrt{\sigma_T^2 + 1} \cdot z_T, \quad z_T \sim \mathcal{N}(0, I) \tag{23}$$

Let $x_0^* = \mathrm{Proj}_{\mathcal{K}}(x_t) \in \mathcal{K}$ denotes the projection of x_t onto the manifold $\mathcal{K}$. In the context of an image manifold, we have $\|x_0^*\| > 0$. If $\langle z_T, x_0^* \rangle \leq 0$, the expectation of squared distance is given by:

$$\begin{aligned} \mathbb{E}\left[\|x_T - x_0^*\|^2\right] &= \mathbb{E}\left[\|\sqrt{\sigma_T^2+1} \cdot z_T - x_0^*\|^2\right] = \mathbb{E}\left[(\sigma_T^2+1)\|z_T\|^2 + \|x_0^*\|^2 - 2\sqrt{\sigma_T^2+1}\langle z_T, x_0^* \rangle\right] \\ &\geq (\sigma_T^2+1)\mathbb{E}[\|z_T\|^2] + \mathbb{E}[\|x_0^*\|^2] \\ &= (\sigma_T^2+1)n + \mathbb{E}[\|x_0^*\|^2] > \sigma_T^2 n \end{aligned} \tag{24}$$

where the final equality uses the fact that $\mathbb{E}[|z_T|^2] = n$ for for $z_T \sim \mathcal{N}(0, I^{n \times n})$. Equation (24) implies that, with high probability, $\mathrm{dist}_{\mathcal{K}}(x_T) > \sqrt{n}\sigma_T$. Consequently, at any step $t = T, \ldots, 0$, these deviations may result in $\mathrm{dist}_{\mathcal{K}}(x_t) \neq \sqrt{n}\sigma_t$, potentially causing the final sample to deviate from the manifold $\mathcal{K}$. It is worth noting that the EDM sampling method initializes with a random step as $x_T = \sigma_T z_T$. However, it leads to the same conclusion: for $\langle z_T, x_0^* \rangle \leq 0$, we have $\mathrm{dist}_{\mathcal{K}}(x_T) \neq \sqrt{n}\sigma_T$.

Appendix B. Sampling with Noise Level Correction

Algorithm 3 presents the algorithm for incorporating noise level correction within EDM. Unlike the approach used in DDIM with noise level correction Algorithm 1, the EDM version does not normalize the noise vector $\epsilon_\theta(x_t, \hat{\sigma}_t)$.

Algorithm 3 EDM with Noise Level Correction (EDM-NLC)

Input: Denoiser ϵ_θ and noise level corrector r_θ
Input: Noise scheduler σ_t
Output: samples $x_0 \in \mathcal{K}$

```
x_T = σ_T ε,  ε ~ N(0, I),
for  t = T, T − 1, ··· , 1 do
  σ̂_t = σ_t[1 + r_θ(x_t, σ_t))]
  σ̂_{t−1} = σ̂_t σ_{t−1}/σ_t
  ε̂_t = ε_θ(x_t, σ̂_t)
  x_{t−1} = x_t + (σ̂_{t−1} − σ̂_t)ε̂_t
  if t > 1 then
    ε̂_{t−1} = ε_θ(x_{t−1}, σ̂_{t−1})
    ε̄_t = 0.5ε̂_t + 0.5ε̂_{t−1}
    x_{t−1} = x_t + (σ̂_{t−1} − σ̂_t)ε̄_t
  end
end for
```

Algorithm 4 presents the algorithm for incorporating noise level correction within DDNM for linear image restoration tasks $y = \mathrm{A}x$.

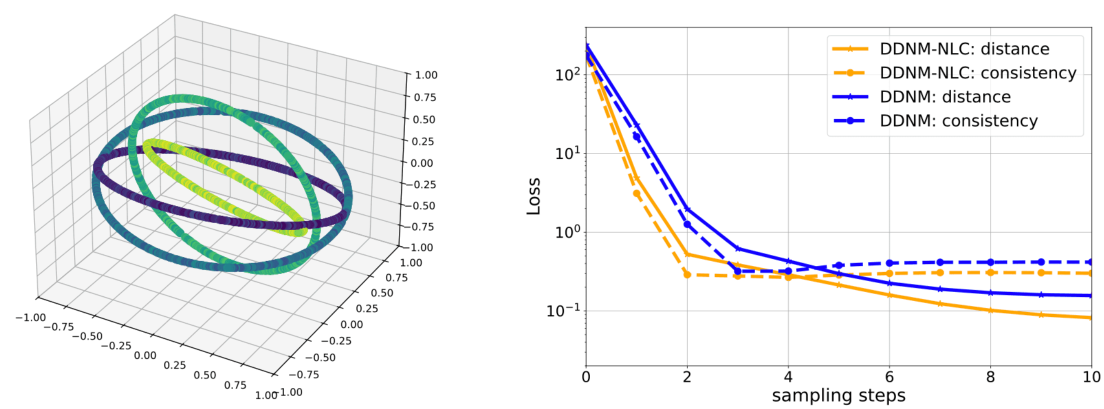
doi.org/10.32388/NSYEO2

Algorithm 4 DDNM with Noise Level Correction (DDNM-NLC)

Input: Denoiser ϵ_θ and noise level corrector r_θ
Input: Noise scheduler σ_t, randomness scale η,
Input: Linear degradation operator $\mathbf{A}$, and pseudo-inverse operator $\mathbf{A}^\dagger$ for image restoration constraint $\mathcal{C}$,
Input: Degraded image y,
Output: samples $x_0 \in \mathcal{K} \cap \mathcal{C}$

$x_T = \sqrt{\sigma_T^2 + 1} \cdot z_T,\ \ z_T \sim \mathcal{N}(0, I),$
for $t = T, T-1, \cdots, 1$ **do**
  $\hat{\sigma}_t = \sigma_t[1 + r_\theta(x_t, \sigma_t))]$
  $\hat{\sigma}_{t-1} = \hat{\sigma}_t \frac{\sigma_{t-1}}{\sigma_t}$
  $\hat{\epsilon}_t = \sqrt{n}\epsilon_\theta(x_t, \hat{\sigma}_t) / \|\epsilon_\theta(x_t, \hat{\sigma}_t)\|$
  $\hat{x}_{0|t} = x_t - \hat{\sigma}_t \hat{\epsilon}_t$
  $x_{0|t} = \mathbf{A}^\dagger y + (\mathbf{I} - \mathbf{A}^\dagger \mathbf{A})\hat{x}_{0|t}$
  $\sigma_{noise} = \eta \frac{\hat{\sigma}_{t-1}}{\hat{\sigma}_t} \sqrt{\hat{\sigma}_t^2 - \hat{\sigma}_{t-1}^2}$
  $\sigma_{signal} = \sqrt{\hat{\sigma}_{t-1}^2 - \sigma_{noise}^2}$
  $x_{t-1} = x_{0|t} + \sigma_{signal}\hat{\epsilon}_t + \sigma_{noise}\omega_t,$ where $\omega_t \sim \mathcal{N}(0, I)$
end for

Appendix C. Toy Experiments



(a) Sphere datasets in toy experiments

(b) Sample generation with linear constraint

Figure 6. (a) Sphere datasets in toy experiments; (b) shows the results of sample generation with linear constraint $Ax = b$.

C.1. Experimental design for sphere manifold.

The dataset consists of samples from a d-dimensional sphere manifold, denoted as $s \sim \mathcal{S}^d$ embedded within an n-dimensional data space. To create training samples x we apply m different linear projections (rotations) to the original sphere s and add a small amount of Gaussian noise x_{noise} to the d-sphere signal x_{signal}. Let $\mathcal{K}$ represent the d-sphere manifold, such that $x_{\text{signal}} \in \mathcal{K}$. Each training sample $x \in \mathbb{R}^n$ is generated according to the following equations:

doi.org/10.32388/NSYEO2

$$x = x_{\text{signal}} + x_{\text{noise}}, \quad x_{\text{noise}} \sim \mathcal{N}(0, 10^{-6} I) \qquad (25)$$

$$x_{\text{signal}} = R_k s \in \mathbb{R}^n, \quad \text{where } k \sim \mathcal{U}(\{1, 2, \cdots, m\}),\ R_k R_k^T = I^{(n \times n)} \qquad (26)$$

$$s \sim \mathcal{S}^d, \quad \text{where } \sum_{i=1}^{d+1} s_i^2 = 1,\ s_{d+2} = s_{d+3} = \cdots = s_n = 0 \qquad (27)$$

Where $\mathcal{N}(0, \Sigma)$ denotes the normal distribution, $\mathcal{U}(\{\cdot\})$ denotes the uniform distribution. $\mathcal{S}^d$ refers to a d-dimensional sphere. The matrices $R_1, R_2, \cdots, R_m$ are fixed random orthogonal matrices that are utilized to "hide" zeros in certain coordinates. For our experiments, we focus primarily on a dataset with parameters $n = 100$, $d = 1$, and $m = 4$, resulting in a 100-dimensional dataset composed of 4 circles. For illustration, we also generate a 3-dimensional dataset with parameters ($n = 3$, $d = 1$, $m = 4$), as shown in Figure 6a. A total of 10,000 samples were used to train the model. The denoiser $\epsilon_\theta(\cdot)$ is implemented with a fully connected network containing 5 layers, each with a hidden dimension of 128. For noise level correction, $r_\theta(\cdot)$ is implemented with a 2-layer fully connected network, also using a hidden dimension of 128.

C.2. Experimental result of constrained sample generation

In Section 4.1, we demonstrate the effectiveness of noise level correction in unconstrained sample generation. Here, we extend the evaluation to constrained generation, using a linear constraint $Ax = b$ with $A \in \mathbb{R}^{1 \times n}$ as a random variable and $b = 0$. We applied the proposed 10-step DDNM with Noise Level Correction (DDNM-NLC), as detailed in Algorithm 4 to generate samples and compared this method to the 10-step DDNM baseline. We report the distance (to measure sample quality) and Consistency error $||Ax - b||$ (to measure constraint satisfaction) as shown in Figure 6b. The proposed method shows superior performance in both metrics, generating high-quality samples that satisfy the constraint more effectively than the baseline.

Appendix D. Image Generation Experiments

D.1. Experimental design

Implementation. In this experiment, we used a pretrained denoiser $\epsilon_\theta(\cdot)$, and trained the noise level correction network $r_\theta(\cdot)$ to enhance the denoising process. For unconstrained image generation experiments, we employed the pretrained 32×32 denoising network from [23] in DDIM-based experiments on the CIFAR-10 dataset, and the pretrained 32×32 denoising network from [24] in EDM-based experiments on CIFAR-10. For image restoration experiments, we utilized the pretrained

256×255 denoising network from [47] for ImageNet experiments and the pretrained 256×255 denoising network from [46] for CelebA-HQ experiments.

Network architecture. The noise correction network $r_\theta(\cdot)$ is designed to be significantly smaller than the denoiser $\epsilon_\theta(\cdot)$, while still incorporating residual and attention blocks similar to those used in the original DDPM denoiser [7]. Table 7 outlines the architecture of the noise correction network for CIFAR-10, ImageNet, and CelebA-HQ datasets. For all datasets, we employ 2 residual blocks, 1 attention block, and 4 attention heads. The primary architectural difference lies in the varying feature sizes (input channels and input dimensions) generated by the denoiser's encoder. For comparison, the ImageNet denoiser contains 9 attention blocks in the encoder and 13 in the decoder. Consequently, the noise correction network is approximately ten times smaller than the denoiser network. The last two rows of Table 7 provide a parameter count comparison between the noise correction network r_θ and the denoiser σ_θ.

Network	**Heperparameter**	**CIFAR-10**	**ImageNet**	**Celeba-HQ**
Network Architecture for $r_\theta(\cdot)$	Input Channel (feature channel)	256	1024	512
	Input Size (feature size)	4	8	8
	Residual Blocks	2	2	2
	Attention Blocks	1	1	1
	Attention Heads	4	4	4
# Parameters	$r_\theta(\cdot)$	14 M	234 M	59 M
	$\epsilon_\theta(\cdot)$(Frozen, not trained)	218 M	2109 M	434 M

Table 7. Architecture of the noise level correction network $r_\theta(\cdot)$

Training. The noise correction network is trained following the original DDPM training procedure. Key hyperparameters for training are listed in Table 8. Notably, the noise correction network is trained until 19.2 million samples have been drawn from the training set. In comparison, training the denoiser requires 200 million samples for CIFAR-10 and over 2000 million samples for ImageNet.

Settings		CIFAR-10	ImageNet	Celeba-HQ
Hyperparameters	Batch size	128	64	64
	Learning rate	0.0003	0.0003	0.0003
	# Iterations	150 K	300 K	300 K
# Samples	Training $r_\theta(\cdot)$	19.2 M	19.2 M	19.2 M
	Training $\epsilon_\theta(\cdot)$	≈200 M	≈2500 M	≈1000 M

Table 8. Hyperparameters used for the training

D.2. Time and Memory Cost

Training time. The training time is shown in the last two rows of Table 8. This results in significantly faster training for the noise level correction network. For example, training the ImageNet denoiser on 8 Tesla V100 GPUs takes approximately two weeks, while training the noise correction network requires only about one day. This efficiency is due to the smaller size of the noise level correction network and its use of the pretrained denoiser.

Inference time. The inference times for 10-step sample generation with a batch size of 1 are presented in Table 9. As shown, incorporating noise level correction adds only a modest increase in inference time, approximately $\approx 10\%$.

Inference time	CIFAR-10	ImageNet
DDIM	0.32	0.93
DDIM-NLC	0.38	1.19

Table 9. Inference time of DDIM and DDIM-NLC.

D.3. Lookup table method for image restoration

We evaluate the performance of the lookup table noise level correction (LT-NLC) in image restoration tasks. The results on the ImageNet dataset are summarized in Table 10. As shown, the DDNM image

doi.org/10.32388/NSYEO2

restoration method achieves additional performance gains with LT-NLC. However, these improvements are notably smaller compared to those achieved with the neural network-based NLC method.

ImageNet	**4 x SR**	**Deblurring**	**Colorization**	**CS 25%**	**Inpainting**
Method	PSNR↑/SSIM↑/FID↓	PSNR↑/SSIM↑/FID ↓	Cons↓/FID↓	PSNR↑/SSIM↑/FID↓	PSNR↑/SSIM↑/FID ↓
DDNM	27.45 / 0.870/ 39.56	44.93 / 0.993 / 1.17	42.32 / 36.32	21.62 / 0.748 / 64.68	31.60 / 0.946 / 9.79
DDNM-LT-NLC	27.47 / 0.870/ 39.03	45.14 / 0.993 / 1.02	42.12 / 36.07	21.48 / 0.751 / 62.64	31.84 / 0.951 / 9.19
DDNM-NLC	27.50 / 0.872 / 37.82	46.20 / 0.995 / 0.79	41.60 / 35.89	21.27 / 0.769 / 58.96	32.51 / 0.957 / 7.20

Table 10. Comparative results of image restoration tasks on ImageNet for lookup-table noise level correction.

Appendix E. Qualitative study

E.1. Image Restoration

We present qualitative comparisons between the proposed method, IterProj-NLC, and the baseline, DDNM, across various image restoration tasks. These tasks include compressive sensing, shown in fig. 7, colorization, shown in fig. 8, inpainting, shown in fig. 9, and super-resolution, shown in fig. 10. The comparisons demonstrate the effectiveness of IterProj-NLC in producing visually superior results over the baseline.

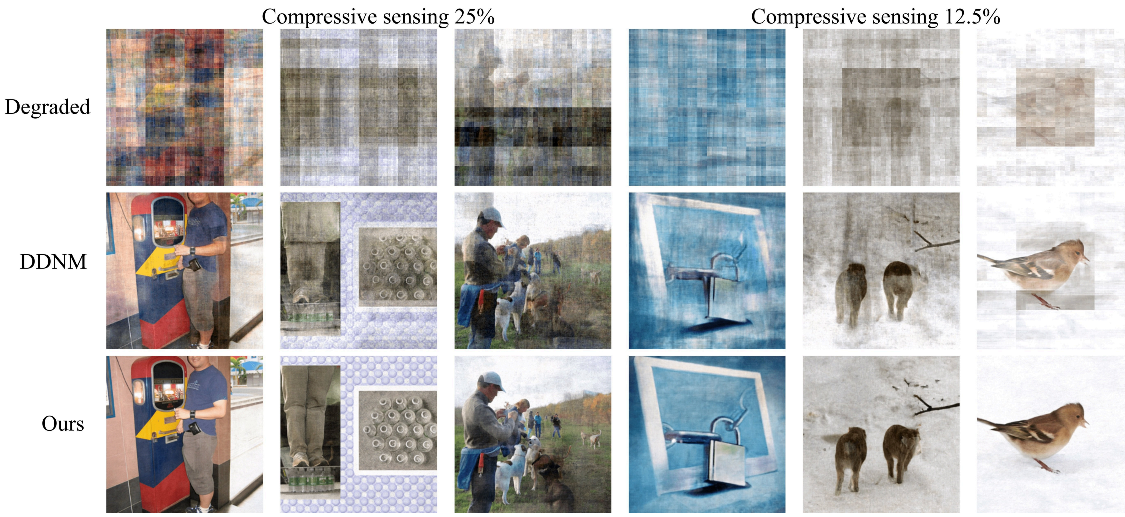


Figure 7. Qualitative results of compressive sensing.

Figure 8. Qualitative results of colorization.

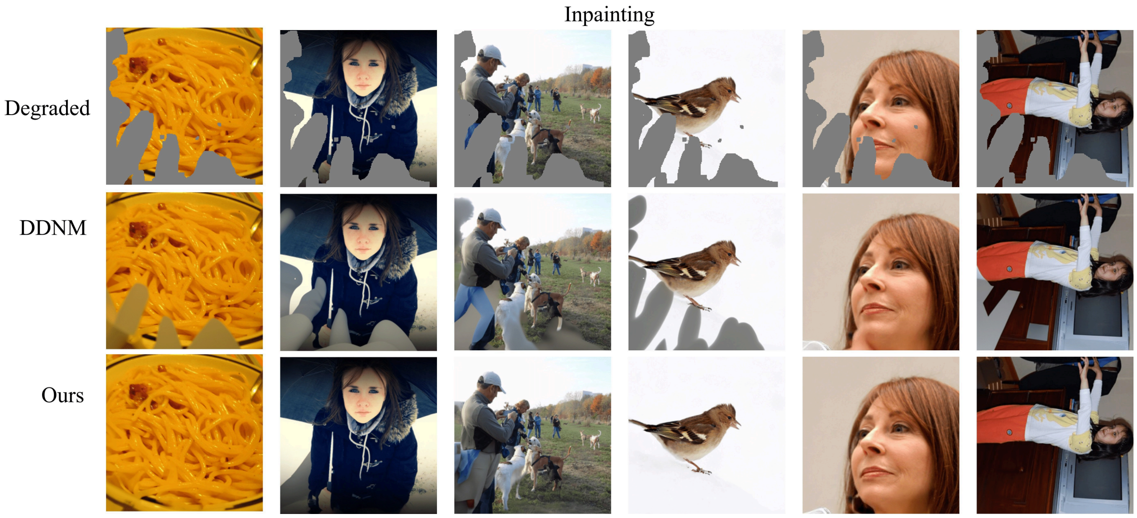


Figure 9. Qualitative results of inpainting.

Figure 10. Qualitative results of super resolution.

E.2. Unconstrained Image Generation

The example results of CIFAR-10 generated using 100-step sampling of DDIM-NLC are presented in fig. 11.

doi.org/10.32388/NSYEO2

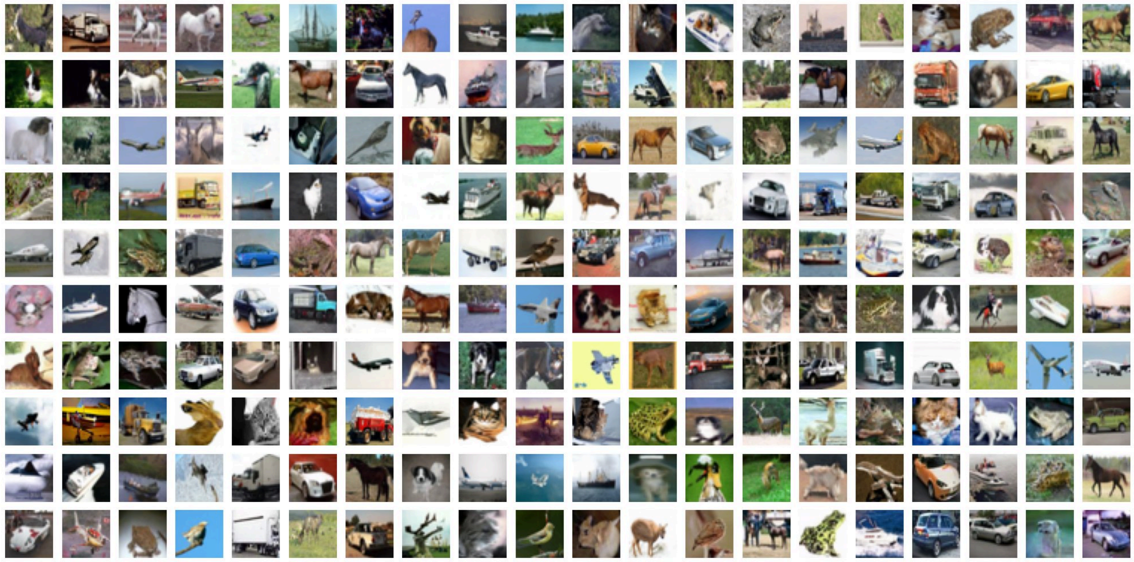


Figure 11. Example results of CIFAR-10 generated using DDIM-NLC.

References

1. ^a, b^ *Song Y, Ermon S (2019). "Generative modeling by estimating gradients of the data distribution". Advances in neural information processing systems. 32.*
2. ^a, b, c^ *Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S. Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning. PMLR; 2015. p. 2256-2265.*
3. ^ *Oussidi A, Elhassouny A. Deep generative models: Survey. In: 2018 International conference on intelligent systems and computer vision (ISCV). IEEE; 2018. p. 1-8.*
4. ^ *Li D, Li J, Hoi S. "Blip-diffusion: Pre-trained subject representation for controllable text-to-image generation and editing". Advances in Neural Information Processing Systems. 36; 2024.*
5. ^ *Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014). "Generative adversarial nets". Advances in neural information processing systems. 27.*
6. ^ *Diederik P. Kingma, Max Welling (2014). "Auto-encoding variational bayes". 2nd International Conference on Learning Representations (ICLR).*
7. ^a, b, c, d, e, f, g, h^ *Ho J, Jain A, Abbeel P (2020). "Denoising diffusion probabilistic models". Advances in Neural Information Processing Systems. 33: 6840–6851.*
8. ^a, b^ *Song Y, Dhariwal P, Chen M, Sutskever I. "Consistency Models." In: International Conference on Machine Learning. PMLR; 2023. p. 32211-32252.*

 doi.org/10.32388/NSYEO2

9. ^Cao H, Tan C, Gao Z, Xu Y, Chen G, Heng PA, Li SZ (2024). "A survey on generative diffusion models". IEEE Transactions on Knowledge and Data Engineering. 2024.
10. ^Ramesh A, Dhariwal P, Nichol A, Chu C, Chen M (2022). "Hierarchical text-conditional image generation with clip latents". arXiv preprint arXiv:2204.06125. 1 (2): 3.
11. ^Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2022). "High-resolution image synthesis with latent diffusion models". Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695.
12. ^Janner M, Du Y, Tenenbaum J, Levine S. "Planning with diffusion for flexible behavior synthesis." In: International Conference on Machine Learning. PMLR; 2022. p. 9902-9915.
13. a, bChi C, Xu Z, Feng S, Cousineau E, Du Y, Burchfiel B, Tedrake R, Song S (2023). "Diffusion policy: Visuomotor policy learning via action diffusion". The International Journal of Robotics Research. pp. 02783649241273668.
14. ^Li X, Thickstun J, Gulrajani I, Liang PS, Hashimoto TB (2022). "Diffusion-lm improves controllable text generation". Advances in Neural Information Processing Systems. 35: 4328–4343.
15. a, bRick Chang JH, Li C-L, Poczos B, Vijaya Kumar BVK, Sankaranarayanan AC. One network to solve them all--solving linear inverse problems using deep projection models. In: Proceedings of the IEEE International Conference on Computer Vision. 2017. p. 5888–5897.
16. a, b, c, d, ePermenter F, Yuan C. "Interpreting and Improving Diffusion Models from an Optimization Perspective". In: Forty-first International Conference on Machine Learning; 2024.
17. ^Wang J, Yue Z, Zhou S, Chan KC, Loy CC (2024). "Exploiting diffusion prior for real-world image super-resolution". International Journal of Computer Vision. pp. 1–21.
18. ^Song Y, Shen L, Xing L, Ermon S. Solving inverse problems in medical imaging with score-based generative models. In: International Conference on Learning Representations; 2022.
19. a, b, cChung H, Kim J, Mccann MT, Klasky ML, Ye JC. "Diffusion Posterior Sampling for General Noisy Inverse Problems." In: The Eleventh International Conference on Learning Representations; 2023.
20. ^Wu L, Trippe B, Naesseth C, Blei D, Cunningham JP (2024). "Practical and asymptotically exact conditional sampling in diffusion models". Advances in Neural Information Processing Systems. 36.
21. ^Dou Z, Song Y. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In: The Twelfth International Conference on Learning Representations; 2024.
22. a, b, c, d, e, f, g, h, iWang Y, Yu J, Zhang J. "Zero-shot image restoration using denoising diffusion null-space model." In: The Eleventh International Conference on Learning Representations; 2023.

 doi.org/10.32388/NSYEO2

23. ^a, b, c^ *Song J, Meng C, Ermon S. "Denoising Diffusion Implicit Models". International Conference on Learning Representations. 2021.*

24. ^a, b, c, d, e, f^ *Karras T, Aittala M, Aila T, Laine S (2022). "Elucidating the design space of diffusion-based generative models". Advances in neural information processing systems. **35**: 26565–26577.*

25. ^ *Song Y, Sohl-Dickstein J, Kingma DP, Kumar A, Ermon S, Poole B. Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations; 2021.*

26. ^ *Arechiga N, Permenter F, Song B, Yuan C (2023). "Drag-guided diffusion models for vehicle image generation". arXiv preprint arXiv:2306.09935. Available from: https://arxiv.org/abs/2306.09935.*

27. ^ *Kong Z, Ping W, Huang J, Zhao K, Catanzaro B. "DiffWave: A Versatile Diffusion Model for Audio Synthesis." In: International Conference on Learning Representations; 2021.*

28. ^ *Nichol AQ, Dhariwal P (2021). "Improved denoising diffusion probabilistic models". In: International conference on machine learning. PMLR. pp. 8162–8171.*

29. ^ *Salimans T, Ho J (2022). "Progressive Distillation for Fast Sampling of Diffusion Models". International Conference on Learning Representations.*

30. ^ *Benton J, De Bortoli V, Doucet A, Deligiannidis G. Nearly d-linear convergence bounds for diffusion models via stochastic localization. In: The Twelfth International Conference on Learning Representations; 2024.*

31. ^ *Li G, Wei Y, Chen Y, Chi Y (2024). "Towards Non-Asymptotic Convergence for Diffusion-Based Generative Models". The Twelfth International Conference on Learning Representations.*

32. ^a, b, c^ *Kawar B, Elad M, Ermon S, Song J (2022). "Denoising diffusion restoration models". Advances in Neural Information Processing Systems. **35**: 23593–23606.*

33. ^ *Chung H, Sim B, Ryu D, Ye JC (2022). "Improving diffusion models for inverse problems using manifold constraints". Advances in Neural Information Processing Systems. **35**: 25683–25696.*

34. ^ *Chung H, Lee S, Ye JC (2023). "Fast Diffusion Sampler for Inverse Problems by Geometric Decomposition". arXiv preprint arXiv:2303.05754. Available from: https://arxiv.org/abs/2303.05754.*

35. ^ *Bansal A, Chu HM, Schwarzschild A, Sengupta S, Goldblum M, Geiping J, Goldstein T (2023). "Universal guidance for diffusion models". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 843–852.*

36. ^ *Xu X, Chi Y (2024). "Provably robust score-based diffusion posterior sampling for plug-and-play image reconstruction". arXiv preprint arXiv:2403.17042.*

37. ^ *Bengio Y, Courville A, Vincent P (2013). "Representation learning: A review and new perspectives". IEEE Transactions on Pattern Analysis and Machine Intelligence. **35** (8): 1798–1828.*

38. ^Fefferman C, Mitter S, Narayanan H (2016). "Testing the manifold hypothesis". Journal of the American Mathematical Society. 29 (4): 983–1049.

39. ^Ning M, Sangineto E, Porrello A, Calderara S, Cucchiara R. "Input perturbation reduces exposure bias in diffusion models." In: Proceedings of the 40th International Conference on Machine Learning. 2023. pp. 26245–26265.

40. ^Bauschke HH, Borwein JM (1996). "On projection algorithms for solving convex feasibility problems". SIAM review. 38 (3): 367–426.

41. ^Lewis AS, Malick J (2008). "Alternating projections on manifolds". Mathematics of Operations Research. 33 (1): 216–234.

42. ^Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S (2017). "Gans trained by a two time-scale update rule converge to a local nash equilibrium". Advances in neural information processing systems. 30.

43. ^Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. Ieee; 2009. p. 248–255.

44. ^Karras T, Aila T, Laine S, Lehtinen J (2018). "Progressive Growing of GANs for Improved Quality, Stability, and Variation". International Conference on Learning Representations.

45. ^Choi J, Kim S, Jeong Y, Gwon Y, Yoon S. ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). IEEE; 2021. p. 14347-14356.

46. a, bLugmayr A, Danelljan M, Romero A, Yu F, Timofte R, Van Gool L. Repaint: Inpainting using denoising diffusion probabilistic models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022. p. 11461-11471.

47. ^Dhariwal P, Nichol A (2021). "Diffusion models beat gans on image synthesis". Advances in neural information processing systems. 34: 8780–8794.

Declarations

Funding: No specific funding was received for this work.

Potential competing interests: No potential competing interests to declare.