Diffusion Models for Image Generation and Inverse Problems

Zin Yamin Tun

Abstract—Diffusion models have recently emerged as powerful generative models capable of high-quality image synthesis and solving challenging inverse problems. In this project, we explored the use of a pretrained diffusion model for image denoising, unconditional image generation, and inverse problem reconstruction. First, we demonstrate single-step image denoising under different noise levels to evaluate the model's ability to recover clean images from corrupted observations. We then implement the Denoising Diffusion Probabilistic Model (DDPM) sampling procedure to generate images from pure Gaussian noise through iterative reverse diffusion. Building on this framework, we apply diffusion models as priors for solving inverse problems, including image inpainting and deconvolution, using SDEdit, ScoreALD, and Diffusion Posterior Sampling (DPS). We compare these methods qualitatively and quantitatively using PSNR and LPIPS metrics. Our results highlight the effectiveness of diffusion-based posterior sampling methods, with DPS providing improved reconstruction stability and visual fidelity compared to alternative approaches.

Index Terms—Diffusion Models, Computational Imaging

1 Introduction

In real-world applications, images are often corrupted by noise, blur, or missing regions, making the reconstruction of clean images an inherently ill-posed problem. Traditional approaches typically rely on explicit priors or handcrafted regularization techniques to address these challenges. In contrast, diffusion models learn to reverse the noise corruption process through a probabilistic framework, enabling them to serve as powerful generative priors for solving inverse imaging problems.

In this project, we explore two primary applications of diffusion models. First, we perform unconditional image generation by starting from Gaussian noise and iteratively denoising it to synthesize realistic human face images. Second, we investigate diffusion-based approaches for solving inverse problems, specifically image inpainting (recovering missing regions) and deconvolution (restoring blurry images). We compare three methods: SDEdit, which reconstructs images by denoising partially corrupted measurements; ScoreALD, which incorporates likelihood gradients to guide the reconstruction; and Diffusion Posterior Sampling (DPS), which uses normalized likelihood gradients to stabilize posterior sampling. All methods follow the framework of Denoising Diffusion Probabilistic Models (DDPM) [1], which generate high-quality images by sampling from noise and iteratively refining them through a learned denoising process.

To evaluate the performance of these methods, we conduct both qualitative and quantitative analyses. Quantitatively, we measure reconstruction quality using Peak Signal-to-Noise Ratio (PSNR) and Learned Perceptual Image Patch Similarity (LPIPS). Our results demonstrate that diffusion models can generate high-quality images and effectively solve inverse problems, with DPS outperforming the other methods in reconstruction quality and stability.

- *Z. Tun is with the Department of Electrical Engineering, Stanford University, Stanford, CA, 94301.*

2 Related Work

Diffusion models have recently emerged as a powerful class of generative models for high-quality image synthesis. Denoising Diffusion Probabilistic Models (DDPM), introduced by Ho et al. (2020) [1], provide an alternative to traditional generative models such as GANs and VAEs, which often suffer from training instability or limited sample diversity. DDPM learns a probabilistic mapping between data and noise through two processes: a forward diffusion process that gradually corrupts images by adding Gaussian noise, and a learned reverse diffusion process that iteratively removes this noise.

Variance-Preserving Forward Diffusion:

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1} \tag{1}$$

One-Step Diffusion (Closed Form):

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{2}$$

$$x_t \sim \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\, x_0,\ (1-\bar{\alpha}_t) I\right) \tag{3}$$

Definitions:

$$\alpha_t = 1 - \beta_t, \qquad \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s, \qquad \epsilon \sim \mathcal{N}(0, I) \tag{4}$$

Reverse Diffusion:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\, \epsilon_\theta(x_t, t) \right) + \sqrt{\beta_t}\, \epsilon_t \tag{5}$$

In the above equations, x_0 denotes the clean image at time step $t = 0$, while x_t represents the noisy image at diffusion step t. The variable ϵ denotes Gaussian noise added during the forward diffusion process.

During training, a neural network is optimized to predict the noise (or equivalently the score function) at each diffusion timestep. By learning this mapping, the network approximates the reverse diffusion process. During sampling, the model progressively transforms random Gaussian noise into realistic images by repeatedly applying the learned denoising steps, effectively generating samples from the underlying data distribution.

Algorithm 1 Training

repeat
  $x_0 \sim q(x_0)$
  $t \sim \text{Uniform}(\{1, \dots, T\})$
  $\epsilon \sim \mathcal{N}(0, I)$
  Take a gradient descent step on

$$\nabla_\theta \left\| \epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon,\ t\right) \right\|^2$$

until convergence

Algorithm 2 Sampling

$x_T \sim \mathcal{N}(0, I)$
for $t = T, \dots, 1$ **do**
  $z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$$

end for
return x_0

3 Proposed Method

Our approach utilizes the **Denoising Diffusion Probabilistic Model (DDPM)** to perform image denoising and generation. The overall method consists of three main steps:

1) **Forward Diffusion:** Gaussian noise is gradually added to a clean image according to the variance-preserving diffusion process described in Equation (2), producing progressively noisier images.
2) **Network Training:** A neural network is trained using gradient descent (Algorithm 1) to estimate the noise added at each timestep of the diffusion process.
3) **Reverse Denoising Process:** During sampling, the trained network iteratively removes noise from a noisy sample using the reverse diffusion process, ultimately reconstructing the original image.

3.1 Unconditional Image Generation

To generate images without any reference input, the trained diffusion model starts from pure Gaussian noise. Through the iterative reverse diffusion process, the network progressively removes noise and generates a realistic image sample. Since there is no conditioning input, the generated images represent random samples drawn from the learned data distribution.

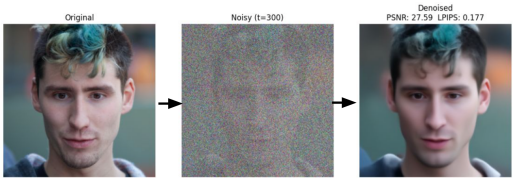

Fig. 1. Illustration of the DDPM denoising network. The model learns to progressively remove noise from a corrupted image through iterative reverse diffusion steps.

3.2 Conditional Image Generation

In conditional image generation, the diffusion process is guided by additional input information, such as a corrupted or partially observed image. For example, tasks such as image inpainting or deconvolution use the available measurements to guide the reconstruction process. In this case, the diffusion model refines the input image through the reverse denoising process while preserving consistency with the provided measurements. In this work, we explore three diffusion-based approaches for conditional image reconstruction.

3.2.1 SDEdit

SDEdit provides a simple and intuitive approach for conditional image generation. Given a corrupted image, Gaussian noise is first added through the forward diffusion process. The image is then denoised using the reverse diffusion process guided by the pretrained network. The amount of noise added determines the balance between preserving the input structure and allowing the model to generate new details. Adding excessive noise may cause the reconstructed image to deviate from the input, while adding too little noise may limit the model's ability to improve the image. Although SDEdit does not explicitly solve the inverse problem, it provides a practical method for image editing and enhancement.

3.2.2 ScoreALD

Score-based Annealed Langevin Dynamics (ScoreALD) is a diffusion-based posterior sampling method designed to solve inverse problems by incorporating measurement information during the denoising process. Unlike SDEdit, which primarily relies on denoising corrupted observations, ScoreALD explicitly incorporates measurement likelihood information to guide reconstruction toward solutions that are consistent with the observed data.

At each diffusion timestep, the algorithm combines two components: the score function predicted by the diffusion model and a gradient term derived from the measurement likelihood. The score function encourages samples to follow the learned data distribution, while the likelihood gradient enforces consistency with the measurements. By progressively reducing the noise level through annealed Langevin dynamics, ScoreALD generates samples that are both realistic and measurement-consistent.

ScoreALD relies on the approximation

$$\nabla_x \log p(y \mid x_0) \approx \nabla_x \log p(y \mid x_t), \tag{6}$$

Algorithm 3 ScoreALD Sampling

$x_T \sim \mathcal{N}(0, I)$
for $t = T, \dots, 1$ **do**
  $z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$
  Estimate clean image:

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\big(x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t)\big)$$

  Reverse diffusion:

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 + \sqrt{1-\alpha_t}z$$

  Likelihood correction:

$$x_{t-1} = x_{t-1} - \frac{1}{2(\sigma^2+\gamma_t^2)}\nabla_{x_t}\|\mathcal{A}(x_t) - y\|^2$$

end for
return x_0

Algorithm 4 Diffusion Posterior Sampling (DPS)

$x_T \sim \mathcal{N}(0, I)$
for $t = T, \dots, 1$ **do**
  $z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$
  Estimate clean image:

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t)\right)$$

  DDPM reverse update:

$$x'_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 + \sqrt{1-\alpha_t}\,z$$

  Posterior correction:

$$x_{t-1} = x'_{t-1} - \zeta_t\nabla_{x_t}\|\mathcal{A}(\hat{x}_0) - y\|^2$$

end for
return x_0

which assumes that the likelihood gradient evaluated at the noisy sample x_t approximates that of the clean image x_0.

Under a Gaussian measurement model, this gradient can be approximated as

$$\nabla_x \log p(y \mid x_t) \approx -\frac{1}{\sigma^2+\gamma_t^2}A^\top(Ax_t - y), \tag{7}$$

where σ^2 denotes the measurement noise variance, γ_t is a guidance-strength hyperparameter, and A represents the forward measurement operator.

During sampling, ScoreALD modifies the standard DDPM reverse diffusion step by introducing an additional gradient correction term. The updated sampling procedure is shown in Algorithm 3.

3.2.3 Diffusion Posterior Sampling (DPS)

Diffusion Posterior Sampling (DPS) further improves diffusion-based inverse problem solving by explicitly incorporating posterior sampling into the reverse diffusion process. DPS modifies the standard diffusion sampling procedure by introducing a normalized likelihood-gradient correction derived from the measurement model.

At each reverse diffusion step, the sample update combines the learned diffusion prior with a gradient term that enforces consistency with the observed measurements. The normalization of this gradient stabilizes the reconstruction process and prevents excessively large updates that may degrade image quality.

DPS approximates the likelihood gradient as

$$\nabla_x \log p(y \mid x_0) \approx \nabla_x \log p_t(y \mid x_0), \tag{8}$$

where the clean image x_0 is replaced by its conditional expectation given the noisy sample x_t.

Using Tweedie's formula for variance-preserving diffusion, the conditional expectation can be written as

$$\hat{x}_0 = \mathbb{E}[x_0 \mid x_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t + (1-\bar{\alpha}_t)\nabla_{x_t}\log p_t(x_t)\right). \tag{9}$$

In practice, the score function $\nabla_{x_t}\log p_t(x_t)$ is approximated using the trained diffusion network $s_\theta(x_t, t)$.

Similar to ScoreALD, DPS introduces a likelihood-guided correction during sampling. However, DPS evaluates the measurement consistency using the reconstructed clean estimate $\hat{x}_0$ rather than the noisy state x_t. The update becomes

$$x_{t-1} = x'_{t-1} - \zeta_t\nabla_{x_t}\|y - \mathcal{A}(\hat{x}_0)\|_2^2, \tag{10}$$

where $\mathcal{A}$ denotes the forward measurement operator and the adaptive step size is defined as

$$\zeta_t = \frac{\zeta}{\|\nabla_{x_t}\|\mathcal{A}(\hat{x}_0) - y\|_2^2\|}. \tag{11}$$

Compared to ScoreALD, DPS provides more stable sampling behavior and often produces higher-quality reconstructions. Due to this stability, DPS has become a widely used method for solving linear inverse problems such as image inpainting and deconvolution using pretrained diffusion models.

4 Experimental Results

4.1 Image Denoising using a Pretrained Diffusion Model

To evaluate the denoising capability of a pretrained diffusion model, we corrupt images by applying the forward diffusion process at different timesteps $t \in \{30, 100, 300\}$, where larger t corresponds to higher noise levels. At each timestep, Gaussian noise is progressively added to the image according to the diffusion schedule, and the pretrained model is then used to reconstruct the image through the reverse denoising process.

Figure 2 shows the reconstruction results under different noise levels. The diffusion model performs well when the corruption level is relatively low. In particular, the reconstruction at $t = 30$ closely resembles the original image, demonstrating strong recovery performance. As the noise level increases, reconstruction quality gradually degrades. While the model successfully preserves global semantic structure and high-level features, it struggles to recover fine high-frequency details. For example, at $t = 300$, facial details such as the beard appear faded, and the reconstructed image deviates slightly from the original appearance.

These results indicate that pretrained diffusion models are effective denoisers for moderate noise levels but may lose local texture information under severe corruption.

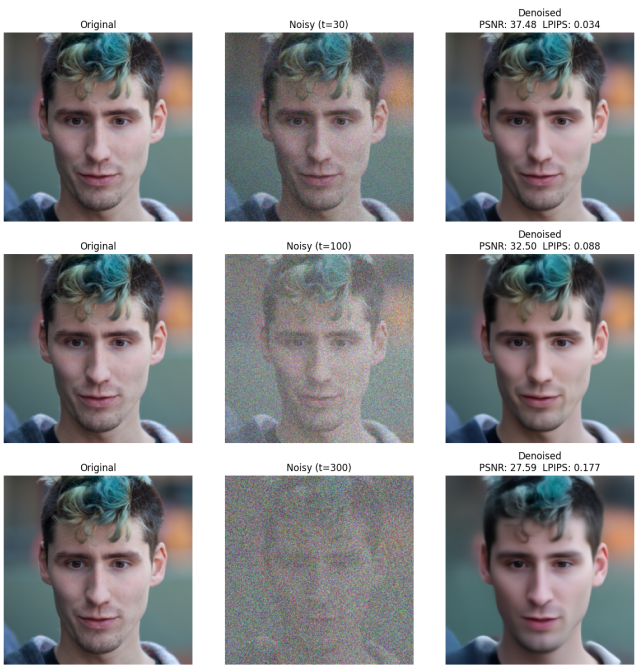

Fig. 2. Image denoising using a pretrained diffusion model at different diffusion timesteps. Lower noise levels lead to reconstructions that more closely resemble the original image.

Fig. 3. Unconditional image generation using a pretrained diffusion model. Starting from Gaussian noise, the model iteratively refines samples to produce realistic human faces.

4.2 Unconditional Image Generation

For unconditional image generation, the pretrained diffusion model is initialized with pure Gaussian noise sampled from $\mathcal{N}(0, I)$. The model then performs iterative reverse diffusion for 1000 sampling steps, progressively transforming noise into a realistic image sample.

Figure 3 presents examples of generated human faces. Since the sampling process begins from random Gaussian noise, each generation run produces a different image while remaining consistent with the learned data distribution. The results demonstrate the ability of diffusion models to synthesize diverse and visually realistic images without requiring any conditioning input.

4.3 Solving Inverse Problems

In this section, we evaluate the performance of three diffusion-based approaches for solving inverse problems: **SDEdit**, **ScoreALD**, and **Diffusion Posterior Sampling (DPS)**. The experiments focus on two common inverse imaging tasks: *image inpainting* and *image deconvolution*.

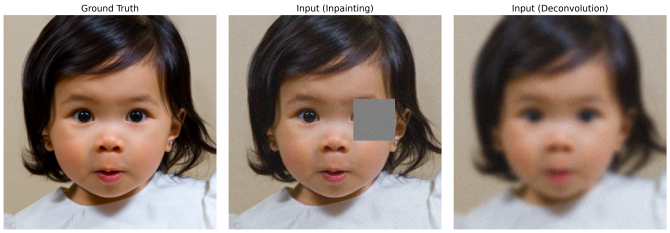

Fig. 4. Input images for inverse problem experiments. From left to right: ground truth image, inpainting input with missing regions, and blurred image used for deconvolution.

Figure 4 shows the input data used for evaluation. The left image represents the ground truth, the middle image corresponds to the inpainting input with missing regions, and the right image represents the blurred input used for the deconvolution task.

4.3.1 SDEdit

SDEdit solves inverse problems by first adding Gaussian noise to the corrupted input image and then applying the reverse diffusion process to denoise the image using a pretrained diffusion model. The key parameter controlling this process is the diffusion timestep t, which determines the amount of noise added before denoising. In our experiments, we evaluate three noise levels: $t \in \{300, 500, 700\}$.

Figure 5 illustrates the reconstruction results for both inpainting and deconvolution tasks. When the noise level is relatively small (e.g., $t = 300$), the reconstructed image remains close to the original corrupted input. For example, the gray missing region in the inpainting input remains partially visible, and the deconvolution result still exhibits noticeable blur. As the noise level increases, the diffusion model gains more flexibility to generate new image content. However, when the noise level becomes too large (e.g., $t = 700$), the reconstruction begins to deviate significantly from the ground truth. In this regime, the noisy input becomes close to random Gaussian noise, causing the model to behave similarly to unconditional image generation.

To quantitatively evaluate the reconstruction quality, we compute two common image similarity metrics: **Peak Signal-to-Noise Ratio (PSNR)** and **Learned Perceptual Image Patch Similarity (LPIPS)**. PSNR measures the signal fidelity between the reconstructed image and the ground truth, where higher values indicate better reconstruction quality. LPIPS measures perceptual similarity between image patches using deep feature representations, where lower values indicate higher perceptual similarity.

For $t = 300$, the inpainting result achieves a PSNR of 23.20 with LPIPS $= 0.139$, while the deconvolution result achieves a PSNR of 23.23 with LPIPS $= 0.165$. When the noise level increases to $t = 700$, reconstruction quality degrades significantly, with PSNR dropping to 16.38 and LPIPS increasing to 0.305 for the inpainting task, and PSNR $= 16.45$ with LPIPS $= 0.313$ for the deconvolution task. These results indicate that lower diffusion timesteps produce better reconstruction quality for SDEdit in both tasks.

TABLE 1
SDEdit reconstruction results. Entries report PSNR / LPIPS (higher PSNR and lower LPIPS indicate better reconstruction quality).

Timestep t	Inpainting	Deconvolution
300	23.20 / 0.139	23.23 / 0.165
500	20.66 / 0.191	20.24 / 0.207
700	16.38 / 0.305	16.45 / 0.313

Fig. 5. Inverse problem reconstruction using SDEdit under different diffusion timesteps. Lower noise levels preserve more structure from the input, while higher noise levels lead to results that deviate from the ground truth.

4.3.2 *ScoreALD*

For Score-based Annealed Langevin Dynamics (ScoreALD), we evaluate the effect of different annealing schedules, specifically $\{10\text{–}15,\ 15\text{–}20,\ 20\text{–}25\}$. The annealing range controls the noise levels used during sampling and determines the balance between measurement consistency and diffusion prior guidance.

For the inpainting task, the highest PSNR value is achieved with the annealing range 10–15, obtaining a PSNR of 25.52. However, similar to the SDEdit case with small diffusion timestep ($t = 300$), a faint gray patch remains visible in the reconstructed region. The relatively high PSNR may result from limited noise injection, which preserves much of the input structure while only partially correcting missing regions. Visually, the reconstruction produced using annealing 15–20 appears more natural despite having slightly lower quantitative metrics.

For the deconvolution task, the best performance is obtained using annealing 10–15, achieving PSNR $= 22.37$ and LPIPS $= 0.167$. Increasing the annealing range leads to progressively worse reconstruction quality, indicating that excessive stochasticity weakens measurement consistency.

Overall, ScoreALD produces improved reconstructions compared to SDEdit, with modest gains in both PSNR and LPIPS. However, the method occasionally introduces overexposure artifacts, likely caused by the likelihood-gradient approximation used during sampling. As a result, improvements in quantitative metrics remain limited despite noticeable visual enhancements.

TABLE 2
ScoreALD reconstruction results. Each entry reports PSNR / LPIPS (higher PSNR and lower LPIPS indicate better performance).

Annealing Range	Inpainting	Deconvolution
10–15	25.52 / 0.108	22.37 / 0.167
15–20	23.18 / 0.127	18.30 / 0.251
20–25	19.83 / 0.191	16.02 / 0.323

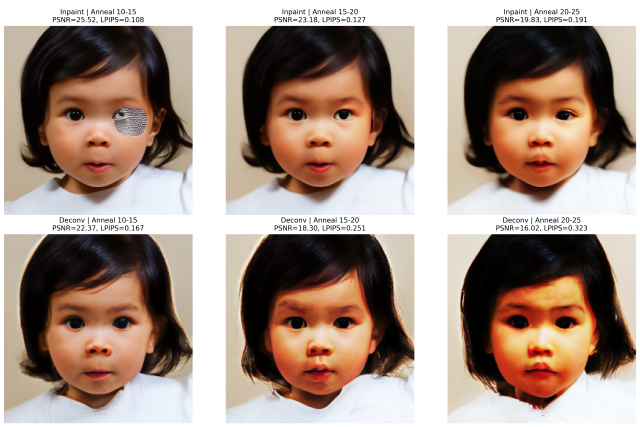

Fig. 6. Inverse problem reconstruction using ScoreALD under different annealing schedules for inpainting and deconvolution tasks.

4.3.3 *Diffusion Posterior Sampling (DPS)*

Diffusion Posterior Sampling (DPS) achieves the strongest reconstruction performance among the evaluated methods. We evaluate DPS using different guidance scale parameters $\{0.3,\ 0.5,\ 1.0\}$ for both inpainting and deconvolution tasks.

Qualitatively, DPS produces the most visually faithful reconstructions, recovering both global structure and fine image details. The best performance for both tasks is obtained at scale $= 0.5$. For inpainting, DPS achieves PSNR $= 35.77$ with LPIPS $= 0.011$, while for deconvolution it achieves PSNR $= 28.47$ with LPIPS $= 0.066$. These values are significantly better than those obtained using SDEdit and ScoreALD, demonstrating the effectiveness of DPS in incorporating measurement information through normalized likelihood gradients.

As the scale parameter increases beyond the optimal value, reconstruction quality begins to degrade. In particular, at scale $= 1.0$, the deconvolution results exhibit noticeable patch artifacts, suggesting that overly strong likelihood guidance can destabilize sampling. Although DPS provides superior reconstruction quality, it also incurs higher computational cost due to additional gradient computations during each diffusion step.

5 Conclusion

In this work, we explored diffusion models for both image generation and inverse problem reconstruction. Using a pretrained DDPM, we demonstrated unconditional image generation by progressively denoising Gaussian noise to synthesize realistic human faces. We also evaluated diffusion-based approaches for solving inverse problems, including image inpainting and deconvolution.

TABLE 3
DPS reconstruction results. Entries report PSNR / LPIPS (higher PSNR and lower LPIPS indicate better performance).

Scale	Inpainting	Deconvolution
0.3	33.84 / 0.024	28.41 / 0.063
0.5	**35.77 / 0.011**	**28.47 / 0.066**
1.0	34.76 / 0.022	27.70 / 0.107

Fig. 7. Inverse problem reconstruction using Diffusion Posterior Sampling (DPS) with different guidance scales for inpainting and deconvolution tasks.

Three methods were compared: SDEdit, ScoreALD, and Diffusion Posterior Sampling (DPS). Experimental results show that while SDEdit provides reasonable reconstructions under low noise levels, incorporating measurement guidance significantly improves reconstruction quality. Among the evaluated methods, DPS achieved the best overall performance, producing the highest PSNR and lowest LPIPS scores for both tasks.

These results highlight the effectiveness of diffusion models as powerful generative priors for solving ill-posed inverse imaging problems.

Acknowledgments

The authors would like to thank the course staffs of EE 367.

References

[1] H. Chung, J. Kim, M. T. McCann, M. L. Klasky, and J. C. Ye, "Diffusion posterior sampling for general noisy inverse problems," in *International Conference on Learning Representations (ICLR)*, 2023.

[2] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[3] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. Tamir, "Robust compressed sensing MRI with deep generative priors," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[4] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon, "SDEdit: Guided image synthesis and editing with stochastic differential equations," in *International Conference on Learning Representations (ICLR)*, 2022.

6 Extra Information

6.1 Task 1.1

The forward diffusion process progressively adds Gaussian noise to the image. At each timestep the process can be written as

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, z_{t-1}, \quad z_{t-1} \sim \mathcal{N}(0, I),\ t = 1, 2, \ldots, T \tag{12}$$

where we define

$$\alpha_t = 1 - \beta_t. \tag{13}$$

Thus the diffusion step becomes

$$x_t = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, z_{t-1}. \tag{14}$$

Similarly,

$$x_{t-1} = \sqrt{\alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_{t-1}}\, z_{t-2}. \tag{15}$$

Substituting this expression into the previous equation gives

$$x_t = \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\, x_{t-2} + \sqrt{1-\alpha_{t-1}}\, z_{t-2}\right) + \sqrt{1-\alpha_t}\, z_{t-1} \tag{16}$$

$$\tag{17}$$

$$= \sqrt{\alpha_t\alpha_{t-1}}\, x_{t-2} + \sqrt{\alpha_t}\sqrt{1-\alpha_{t-1}}\, z_{t-2} + \sqrt{1-\alpha_t}\, z_{t-1}. \tag{18}$$

Continuing this expansion recursively until x_0 yields

$$x_t = \sqrt{\alpha_t\alpha_{t-1}\cdots\alpha_1}\, x_0 + \sum_{i=1}^{t}\left(\sqrt{1-\alpha_i}\prod_{j=i+1}^{t}\sqrt{\alpha_j}\right) z_{i-1}. \tag{19}$$

Define

$$\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i. \tag{20}$$

Then the expression simplifies to

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sum_{i=1}^{t} c_i z_{i-1}. \tag{21}$$

Since $z_i \sim \mathcal{N}(0, I)$ and a linear combination of independent Gaussian variables is also Gaussian,

$$\sum_{i=1}^{t} c_i z_{i-1} \sim \mathcal{N}\left(0, \sum_{i=1}^{t} c_i^2 I\right). \tag{22}$$

The variance becomes

$$\mathrm{Var}(x_t|x_0) = (1-\alpha_t) + \alpha_t(1-\alpha_{t-1}) + \alpha_t\alpha_{t-1}(1-\alpha_{t-2}) + \cdots \tag{23}$$

$$\tag{24}$$

$$= 1 - \prod_{i=1}^{t}\alpha_i = 1 - \bar{\alpha}_t. \tag{25}$$

Substituting back yields the closed-form forward diffusion equation

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, z, \quad z \sim \mathcal{N}(0, I). \tag{26}$$

6.2 Task 1.2

The clean image estimate can be written as

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t + (1-\bar{\alpha}_t) s_\theta(x_t, t) \right) \tag{27}$$

The reverse diffusion step is defined as

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t} \hat{x}_0 \tag{28}$$

Substituting Equation (27) into Equation (28) gives

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t} \left(\frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t + (1-\bar{\alpha}_t) s_\theta(x_t, t) \right) \right) \tag{29}$$

Using the identity

$$\bar{\alpha}_t = \alpha_t \bar{\alpha}_{t-1} \tag{30}$$

we obtain

$$\sqrt{\bar{\alpha}_{t-1}} = \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{\alpha_t}} \tag{31}$$

Substituting this relation simplifies the expression to

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t + \frac{1-\alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} \left(x_t + (1-\bar{\alpha}_t) s_\theta(x_t, t) \right) \tag{32}$$

Expanding the terms yields

$$x_{t-1} = \left[\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} + \frac{1-\alpha_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} \right] x_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}} s_\theta(x_t, t) \tag{33}$$

Using the identity

$$\alpha_t(1-\bar{\alpha}_{t-1}) + (1-\alpha_t) = 1 - \bar{\alpha}_t \tag{34}$$

the expression simplifies to the final reverse diffusion update

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t + (1-\alpha_t) s_\theta(x_t, t) \right) \tag{35}$$

6.3 Task 1.3

From the reverse diffusion update derived earlier, we have

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t + (1-\alpha_t) s_\theta(x_t, t) \right) \tag{36}$$

Using Tweedie's formula for variance-preserving diffusion, the conditional expectation of the clean image can be written as

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t + (1-\bar{\alpha}_t) \nabla_{x_t} \log p_t(x_t) \right) \tag{37}$$

Let the score function predicted by the neural network be

$$s_\theta(x_t, t) = \nabla_{x_t} \log p_t(x_t) \tag{38}$$

Then the estimator becomes

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t + (1-\bar{\alpha}_t) s_\theta(x_t, t) \right) \tag{39}$$

From the variance-preserving forward diffusion process

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t} \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{40}$$

we can solve for x_0:

$$x_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}} \tag{41}$$

Thus,

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon \right) \tag{42}$$

Equating Equation (39) and Equation (42) gives

$$x_t + (1-\bar{\alpha}_t) s_\theta(x_t, t) = x_t - \sqrt{1-\bar{\alpha}_t}\epsilon \tag{43}$$

which simplifies to

$$s_\theta(x_t, t) = -\frac{\epsilon}{\sqrt{1-\bar{\alpha}_t}} \tag{44}$$

Substituting this relation into Equation (36) yields

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) \tag{45}$$