Solving Inverse Problems with Diffusion Models

Daniel Song and Alex Liu

Abstract—Diffusion models have emerged as a strong class of generative models that synthesize high-fidelity images through iterative denoising. In this project, we study diffusion models for both unconditional image generation and inverse problems, focusing on image inpainting and deconvolution. We first implement the core Denoising Diffusion Probabilistic Model (DDPM) pipeline, including forward noising and reverse sampling, and then extend it to restoration settings using three conditioning strategies: SDEdit, Score-based Annealed Langevin Dynamics (Score-ALD), and Diffusion Posterior Sampling (DPS). We evaluate performance using perceptual quality (LPIPS) and reconstruction fidelity (PSNR), together with qualitative visual comparisons. Results show that diffusion priors provide strong restoration capability under corrupted measurements, with DPS giving the most stable and visually consistent reconstructions, while Score-ALD is more prone to artifacts and SDEdit tends to preserve less measurement-consistent detail. We also observe practical limitations: restoration quality depends strongly on the training distribution, and iterative sampling remains computationally expensive for real-time use.

Index Terms—Computational Imaging, Diffusion Model, DDPM, SDEdit, Score-ALD, DPS

1 Introduction

Diffusion models have become a leading paradigm for image generation by learning to reverse a gradual noising process. Starting from Gaussian noise, a trained denoiser iteratively recovers image structure, yielding samples with strong realism and diversity. Beyond generation, this probabilistic formulation makes diffusion models effective priors for inverse problems, where the goal is to recover clean images from incomplete or degraded measurements.

In this project, we study diffusion models in two settings. First, we implement the core Denoising Diffusion Probabilistic Model (DDPM) pipeline [1] for unconditional image synthesis, including forward diffusion, single-step denoising, and full reverse-time sampling. Second, we apply pretrained diffusion priors to image restoration tasks, specifically inpainting and deconvolution, where measurements are corrupted by masking, blur, and noise.

To solve inverse problems, we implement and compare three conditioning strategies: SDEdit [2], Score-based Annealed Langevin Dynamics (Score-ALD) [3], and Diffusion Posterior Sampling (DPS) [4]. These methods differ in how measurement consistency is enforced during reverse diffusion: SDEdit relies on partial noising and guided denoising, Score-ALD introduces likelihood-gradient corrections with annealing, and DPS uses normalized posterior-gradient updates for more stable conditioning.

We evaluate the three methods using quantitative metrics, including PSNR and LPIPS, together with qualitative visual inspection. Our experiments show that diffusion priors are effective for both generation and restoration, with DPS generally providing the most stable and highest-quality reconstructions. We also observe key practical limitations: performance is sensitive to the training distribution and image resolution assumptions, and iterative diffusion sampling remains computationally expensive for fast or real-time applications.

- *D. Song is with the Department of Computer Science, Stanford University, CA, 94305.*
- *A. Liu is with the Department of Computer Science, Stanford University, CA, 94305.*

2 Related Work

2.1 Generative Modeling Before Diffusion

Early deep generative modeling was dominated by variational and adversarial approaches. Variational Autoencoders (VAEs) introduced latent-variable generative learning with tractable optimization via variational inference [5]. Generative Adversarial Networks (GANs) improved sample realism through adversarial training [6], but often suffered from instability and mode collapse. In parallel, autoregressive models such as PixelRNN modeled exact likelihoods and produced strong image quality at high computational cost [7]. Normalizing flows (e.g., Real NVP) offered exact likelihood and invertible mappings, but required architectural constraints to preserve tractability [8].

2.2 From Score Matching to Diffusion Models

A key theoretical precursor to modern diffusion models is score matching, which estimates gradients of log-density without requiring normalized probabilities [9]. Diffusion-based generative modeling was then formalized as learning a reverse-time denoising process from a forward noising chain [10]. Later, noise-conditional score networks connected score estimation and annealed Langevin dynamics for high-quality sampling [11]. DDPMs provided a practical and stable framework for large-scale image generation [1], while follow-up work improved sampling efficiency and likelihood performance [12], [13]. The stochastic differential equation (SDE) view unified diffusion and score-based methods under a continuous-time framework and enabled flexible predictor-corrector samplers [14].

2.3 Inverse Problems and Learned Priors

Classical inverse problem methods relied on hand-crafted regularization, such as total variation (TV), to stabilize ill-posed reconstruction [15]. Plug-and-play priors replaced explicit regularizers with powerful denoisers inside optimization algorithms [16], bridging model-based and learning-

based reconstruction. Generative priors were further explored by constraining reconstructions to learned manifolds from deep generators [17]. Deep Image Prior demonstrated that network architecture itself can act as an implicit image prior even without external training data [18]. These directions collectively motivated diffusion-based posterior sampling methods for restoration, which combine strong generative priors with measurement consistency during iterative reconstruction.

3 Proposed Method

3.1 Baseline DDPM Sampling

We use a pretrained diffusion model for both generation and inverse problems. Diffusion models can be described using two main processes: forward diffusion and reverse diffusion. In the forward diffusion process, Gaussian noise is gradually added to a natural image x_0 over T steps, producing a sequence of increasingly noisy images until the final sample x_T resembles pure noise. As T becomes large, the distribution of x_T approaches an isotropic Gaussian distribution.

The reverse diffusion process aims to invert this process by progressively removing noise to recover a clean image. Diffusion models are trained to approximate this reverse process, which enables the generation of new images by starting from a sample of Gaussian noise and iteratively denoising it. This reverse diffusion process could be modeled as a stochastic differential equation (SDE):

$$d\boldsymbol{x} = [f(\boldsymbol{x}, t) - g^2(t)\nabla_x \log p_t(\boldsymbol{x})]dt + g(t)d\tilde{\boldsymbol{w}},$$

where $\boldsymbol{x}$ is the image, t is the timestep along the diffusion process, $\tilde{\boldsymbol{w}}$ is the reverse Wiener process, where $d\tilde{\boldsymbol{w}}$ represents a small change in white noise, $f(\boldsymbol{x}, t)$ is the drift coefficient representing the flow or movement of the mean, and $g(t)$ is the diffusion coefficient, determining the intensity of the noise added.

The $\nabla_x \log p_t(\boldsymbol{x})$ term represents the score function $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$, which acts as a vector field of directions that can be used to guide the current image at timestep t to the intended clean image distribution. The score function is what most diffusion models aim to learn as neural networks, either directly as a score estimation network or indirectly as a noise estimation network.

Algorithm 1 Baseline DDPM Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \dots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1 - \alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)) + \sqrt{1 - \alpha_t}\mathbf{z}$
end for
return $\mathbf{x}_0$

3.2 SDEdit

The baseline DDPM only performs unconditional image generation. To address inverse problems, a simple and naive approach is to take the observed measurement (e.g., a noisy, blurred, or incomplete image), apply a partial forward diffusion process to it, and then start the reverse diffusion process from the resulting noisy image to generate a reconstruction. By only applying a partial forward diffusion process, the hope is that it retains most of the coarse structure to help guide the reverse diffusion process. The partial forward diffusion can be formulated as a single-step process dependent on the starting image:

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1 - \bar{\alpha}_t}\mathbf{z}$$

3.3 ScoreALD

Despite its effectiveness, SDEdit does not formally solve inverse problems. One way to properly take the measurement into account is to use a Bayesian perspective to condition the sampling process based on the measurement. With this view, the diffusion model can be used as a prior $p(\boldsymbol{x})$ to sample from the posterior distribution $p(\boldsymbol{x}|\boldsymbol{b})$ using Bayes' rule $p(\boldsymbol{x}|\boldsymbol{b}) = p(\boldsymbol{b}|\boldsymbol{x})p(\boldsymbol{x})$. This changes the SDE formulation of the reverse diffusion process to:

$$d\boldsymbol{x} = [f(\boldsymbol{x}, t) - g^2(t)\nabla_x \log p_t(\boldsymbol{x}_t|\boldsymbol{b})]dt + g(t)d\tilde{\boldsymbol{w}},$$

where $\nabla_x \log p_t(\boldsymbol{x}_t|\boldsymbol{b}) = \nabla_x \log p_t(\boldsymbol{x}_t) + \nabla_x \log p_t(\boldsymbol{b} \mid \boldsymbol{x}_t)$. The score function $\nabla_x \log p_t(\boldsymbol{x}_t)$ could be represented by using a pretrained diffusion model, much like before. However, the likelihood $\nabla_x \log p_t(\boldsymbol{b}|\boldsymbol{x}_t)$ is analytically intractable to compute due to its dependence on the time t. ScoreALD sidesteps this by using the likelihood gradient of the clean image as an estimate,

$$\nabla_x \log p_t(\boldsymbol{b}|\boldsymbol{x}_t) \approx \nabla_x \log p_t(\boldsymbol{b}|\boldsymbol{x_0}),$$

In this approach, the formula derived for clean data $-\frac{1}{2\sigma^2}\nabla_x \|\mathcal{A}(\boldsymbol{x}) - \boldsymbol{y}\|^2$ is applied to the noisy image x_t. ScoreALD also introduces an annealing hyperparameter γ_t resulting in the estimate,

$$\nabla_x \log p_t(\boldsymbol{b}|\boldsymbol{x}_t) \approx -\frac{1}{2(\sigma^2 + \gamma_t^2)}\nabla_{x_t} \|\mathcal{A}(\boldsymbol{x}_t) - \boldsymbol{y}\|^2$$

Algorithmically, this only adds one additional step in the iterative sampling process compared to the original DDPM sampling procedure.

Algorithm 2 ScoreALD Sampling

$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \dots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1 - \alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)) + \sqrt{1 - \alpha_t}\mathbf{z}$
  $\boldsymbol{x}_{t-1} = \boldsymbol{x}_{t-1} - \frac{1}{2(\sigma^2 + \gamma_t^2)}\nabla_{x_t} \|\mathcal{A}(\boldsymbol{x}_t) - \boldsymbol{y}\|^2$
end for
return $\mathbf{x}_0$

3.4 DPS

DPS also attempts to estimate the gradient of the log likelihood term, similar to ScoreALD. However, instead of using the likelihood based on the distribution of clean images, it uses the distribution of the predicted clean image at timestep t, given by $\hat{\boldsymbol{x}}_0$.

$$\nabla_x \log p_t(\boldsymbol{b}|\boldsymbol{x}_t) \approx \nabla_x \log p_t(\boldsymbol{b}|\hat{\boldsymbol{x}}_0(x_t))$$

Now, this term is analytically tractable, since the measurement distribution is already provided. To find $\hat{\boldsymbol{x}}_0$, Tweedie's formula for variance preserving can be used:

$$\begin{aligned}\hat{\boldsymbol{x}}_{\mathbf{0}} &= \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t] \\ &= \frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_x \log p_t(\boldsymbol{x}_t))\end{aligned}$$

Using this approach leads to the derivation of the approximation for the likelihood term:

$$\nabla_x \log p_t(\boldsymbol{b}|\boldsymbol{x}_t) \approx -\zeta_t \nabla_{x_t} \|\mathcal{A}(\hat{\boldsymbol{x}}_0) - \boldsymbol{y}\|^2$$

In practice, we use the following as our approximation:

$$-\frac{\zeta}{\|\nabla_{x_t}\|\mathcal{A}(\hat{\boldsymbol{x}}_0)) - \boldsymbol{y}\|^2\|}\nabla_{x_t}\|\mathcal{A}(\hat{\boldsymbol{x}}_0) - \boldsymbol{y}\|^2$$

Algorithm 3 DPS Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \hat{\boldsymbol{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$
5: $\quad \boldsymbol{x}'_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{\boldsymbol{x}}_0 + \sqrt{1-\alpha_t}\mathbf{z}$
6: $\quad \boldsymbol{x}_{t-1} = \boldsymbol{x}'_{t-1} - \frac{\zeta}{\|\nabla_{x_t}\|\mathcal{A}(\hat{\boldsymbol{x}}_0)) - \boldsymbol{y}\|^2\|}\nabla_{x_t}\|\mathcal{A}(\hat{\boldsymbol{x}}_0) - \boldsymbol{y}\|^2$
7: **end for**
8: **return** $\mathbf{x}_0$

4 Experimental Results

To evaluate and compare the results across different methods, we use Peak Signal-to-Noise Ratio (PSNR) and Learned Perceptual Image Patch Similarity (LPIPS) as our evaluation metrics. The PSNR measures the reconstruction quality relative to the ground truth image by comparing the pixel values between the two. It is defined as

$$PSNR = 10 \cdot \log_{10}(\frac{MAX_I^2}{MSE}),$$

where MAX_I is the maximum possible pixel value of the image, and MSE is the mean squared error between the reconstructed image and ground truth. A greater value indicates a better reconstruction.

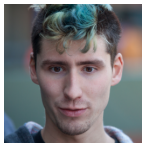
LPIPS calculates the perceptual similarity between the reconstructed image and the ground truth image by computing the distance between the activations of the images on a pre-trained network. LPIPS often provided a measure closer to human perception. For LPIPS, a lower value indicates a better reconstruction.

4.1 Baseline Diffusion Model

For the baseline diffusion model, we experiment with unconditional image generation and analyze the generated images qualitatively. The diffusion model is pre-trained on the Flickr-Faces-HQ (FFHQ) dataset [19] and uses $T = 1000$ iterations for unconditional sampling. Each run of the sampling process produces a different image because the starting point is randomly sampled from Gaussian-distributed noise. As shown in figure 1, the visual quality of the images is fairly high, clearly resembling human faces, as expected given the dataset the model was trained on.

Fig. 1: Images generated through unconditional sampling.

We also experiment with estimating the denoised image at varying starting times, which is later used in the implementation for DPS. Specifically, we first apply a partial forward diffusion to the image and then estimate the original image in a single denoising step, rather than recovering it iteratively over many steps. Examples of this process are shown in Figure 2. As expected, applying more of the forward diffusion process (i.e., starting from a larger time t) reduces the accuracy of the estimate, causing the predicted image to become oversmoothed as the original features are increasingly lost by noise.



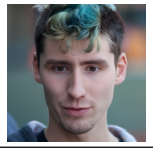
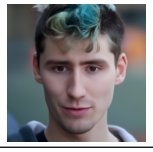
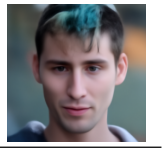
	$t = 30$	$t = 100$	$t = 300$
x_t			
Denoised			
PSNR	37.49	32.58	27.34
LPIPS	0.0325	0.0872	0.2019

Fig. 2: A visualization of the one-step denoising approximation. The top row shows the noisy image x_t at various timesteps, while the bottom row shows the model's prediction of the clean image from that state.

4.2 Inverse Problems

To compare the performance of the three methods on inverse problems, we evaluate them on two tasks: inpainting and deconvolution. In the inpainting task, a square region of an image is masked, and the goal is to reconstruct the missing portion to recover the original image. In the deconvolution task, the image is blurred, and the objective is to recover a sharp version of the image that approximates the original.

For both tasks, a small amount of Gaussian noise ($\sigma = 0.05$) is also added to the corrupted image to ensure the measurement robustness of the model.

(a) Inpainting Task Input (b) Deconvolution Task Input (c) Ground Truth

Fig. 3: The inputs to the models for each task.

4.2.1 SDEdit

SDEdit provides a simple workflow for recovering the original image. We investigate the effects of varying the extent of forward diffusion on the image prior to performing the reverse diffusion process. Specifically, we compare the results between the start times $T = \{300, 500, 700\}$, indicating the number of steps applied using the forward diffusion process. The results in Figure 4 show that using a larger starting time causes the reconstructed images to look less faithful to the original, likely because most of the original features are lost to noise. At the same time, however, the images tend to appear more realistic, resembling a proper human face. This indicates a tradeoff between realism and faithfulness by tuning the hyperparameter T. A good balance point seemed to be at around $T = 500$ for both the PSNR and LPIPS metrics, shown in Table 1.

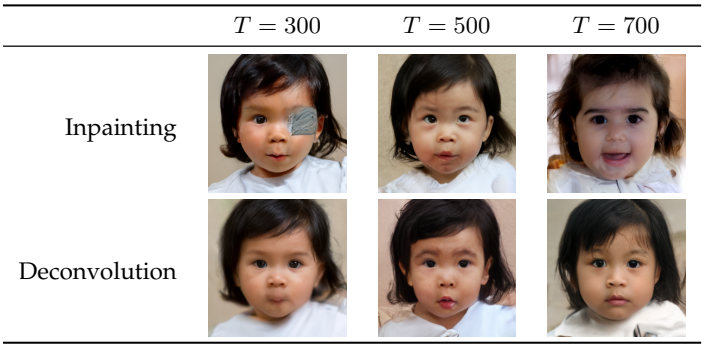


Fig. 4: Results of using SDEdit for inpainting and deconvolution tasks across varying starting levels of forward diffusion.

4.2.2 ScoreALD

ScoreALD uses a time-varying annealing factor γ_t to modulate the weight of the guiding term in the score function. We investigate how using a linear annealing schedule with varying start and end values affects the results. The linear schedule is defined as $\gamma_t = [\gamma_0, \gamma_T]$, where γ_0 is the annealing factor at $t = 0$ and γ_T is the annealing factor at $t = T$ (where $T = 1000$ is the total number of iterations used in the reverse diffusion process). We keep the width of the interval consistent across runs, but vary the endpoints as $\gamma_t = \{[5, 10], [10, 15], [15, 20]\}$.

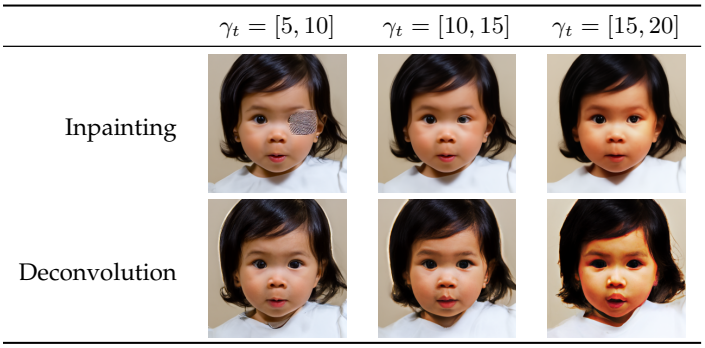


Fig. 5: Results of using ScoreALD for inpainting and deconvolution tasks across varying schedules for the annealing factor.

Figure 5 shows that ScoreALD produces results that are significantly better, both quantitatively and qualitatively, compared to SDEdit. The figure also shows the effects of changing the annealing factor. Using a larger γ_t effectively lessens the weight of the guidance term in the posterior sampling process, reducing the effects of the conditioning based on the measurement. This results in an image that is less faithful to the measurement, which in this case manifests as increased saturation. Theoretically, as $\gamma_t \to \infty$, the sampling process approaches unconditional sampling. Conversely, reducing γ_t too much forces the sampling to rely heavily on an unreliable estimate of the likelihood gradient, producing artifacts that reflect the mask from the original measurement in the case of the inpainting task. For the inpainting task, a γ_t schedule of $[10, 15]$ yielded the best results, whereas for the deconvolution task, a schedule of $[5, 10]$ performed best, both of which are also supported quantitatively in Table 1.

4.2.3 DPS

Much like the ScoreALD method, DPS also uses a scaling factor ζ to weigh the guidance term during the posterior sampling process. For our experiments, we use $\zeta = \{0.3, 0.7, 1.0\}$ to compare the effects of the estimated likelihood function.

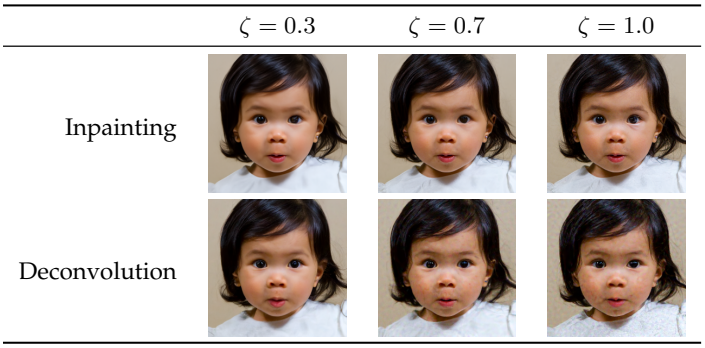


Fig. 6: Results of using DPS for inpainting and deconvolution tasks across varying scale values.

Visually, the images produced by DPS are the most faithful to the original, retaining the majority of the original face's features compared to all other methods tested so far.

	SDEdit			ScoreALD			DPS		
	T	PSNR	LPIPS	γ_t	PSNR	LPIPS	ζ	PSNR	LPIPS
Inpainting	300	22.59	0.1553	[5, 10]	24.95	0.0987	0.3	35.27	0.0257
	500	20.60	0.1895	[10, 15]	28.64	0.0572	**0.7**	**36.54**	**0.0083**
	700	15.95	0.3364	[15, 20]	23.18	0.1132	1.0	34.82	0.0199
Deconvolution	300	23.33	0.1971	[5, 10]	24.84	0.1084	**0.3**	**28.36**	**0.0584**
	500	20.39	0.2077	[10, 15]	22.29	0.1334	0.7	28.49	0.0801
	700	17.06	0.2684	[15, 20]	18.23	0.2657	1.0	27.65	0.1134

TABLE 1: A comparison of PSNR and LPIPS for SDEdit, ScoreALD, and DPS.

Quantitatively, DPS mostly outperforms the prior methods in both PSNR and LPIPS regardless of the choice in the hyperparameter ζ, shown in Table 1. Because the scaling factor ζ appears in the numerator of the guidance term, its effect on scaling is opposite to that of ScoreALD. As shown in Figure 6, increasing ζ introduces artifacts in the deconvolution task, where the generation amplifies the noise present in the original measurement. However, the overall perceptual quality of the images, particularly their faithfulness to the expected result, appears to be much more robust to changes in the scaling factor compared to ScoreALD.

5 Conclusion

Our experiments show that diffusion priors are effective not only for realistic image synthesis but also for restoration tasks such as inpainting and deconvolution. Among the conditioned methods, DPS produced the most reliable reconstructions in both perceptual quality and fidelity, while ScoreALD was more sensitive to tuning and could introduce artifacts under aggressive updates. SDEdit provided a simple and intuitive editing baseline, but generally offered weaker measurement consistency than gradient-based posterior sampling methods.

Overall, the results support diffusion models as a strong framework for inverse problems when combined with principled conditioning. The main limitations remain computational cost from iterative sampling and sensitivity to the training distribution and hyperparameter settings. Future work could focus on faster samplers, adaptive step-size schedules, and methods that improve robustness across operators, resolutions, and out-of-distribution inputs.

Acknowledgments

The authors would like to thank Professor Gorden Wetzstein and TA Sonia Kim for their amazing work in preparing and teaching materials of the Stanford University EE367 course, Winter 2026.

References

[1] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html

[2] 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. [Online]. Available: https://openreview.net/forum?id=aBsCjcPu_tE

[3] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. I. Tamir, "Robust compressed sensing mri with deep generative priors," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021, pp. 14 938–14 954. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/7d6044e95a16761171b130dcb476a43e-Abstract.html

[4] 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. [Online]. Available: https://openreview.net/forum?id=OnD9zGAGT0k

[5] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, 2014. [Online]. Available: https://arxiv.org/abs/1312.6114

[6] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014, pp. 2672–2680. [Online]. Available: https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html

[7] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *International Conference on Machine Learning (ICML)*, 2016, pp. 1747–1756. [Online]. Available: https://proceedings.mlr.press/v48/oord16.html

[8] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," in *International Conference on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1605.08803

[9] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching," *Journal of Machine Learning Research*, vol. 6, pp. 695–709, 2005. [Online]. Available: https://jmlr.org/papers/v6/hyvarinen05a.html

[10] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference on Machine Learning (ICML)*, 2015, pp. 2256–2265. [Online]. Available: https://proceedings.mlr.press/v37/sohl-dickstein15.html

[11] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/3001ef257407d5a371a96dcd947c7d93-Abstract.html

[12] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International Conference on Machine Learning (ICML)*, 2021, pp. 8162–8171. [Online]. Available: https://proceedings.mlr.press/v139/nichol21a.html

[13] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations (ICLR)*, 2021. [Online]. Available: https://openreview.net/forum?id=St1giarCHLP

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

on Learning Representations (ICLR), 2021. [Online]. Available: https://openreview.net/forum?id=PxTIG12RRHS
[15] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1–4, pp. 259–268, 1992.
[16] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013, pp. 945–948.
[17] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," in *International Conference on Machine Learning (ICML)*, 2017, pp. 537–546. [Online]. Available: https://proceedings.mlr.press/v70/bora17a.html
[18] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 9446–9454.
[19] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, p. 4217–4228, Dec. 2021. [Online]. Available: https://doi.org/10.1109/TPAMI.2020.2970919

APPENDIX

Here, we provide several derivations used in the implementation of the models described in this paper.

1

First, we prove the one-step forward noise model to obtain the image $\boldsymbol{x}_t$ depending only on x_0 using the variance-preserving formulation of diffusion models. Specifically, we want to show that

$$\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_{t-1}, \quad t = 1, 2, \ldots, T,$$

is equivalent to

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z},$$

where $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, and $\mathbf{z} \sim \mathcal{N}(0, I)$ (i.i.d. Gaussian random variables).

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_{t-1} \\
&= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1} \\
&= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\mathbf{z}_{t-2}) + \sqrt{1-\alpha_t}\mathbf{z}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\mathbf{z}_{t-2} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1}
\end{aligned}$$

Because the random variables are i.i.d. Gaussian with a zero mean, their sum is also Gaussian with a zero mean and a variance equal to the sum of the individual variances. The variance of $\mathbf{z}_{t-1}$ is $1-\alpha_t$, and the variance of $\mathbf{z}_{t-2}$ is $\alpha_t(1-\alpha_{t-1})$. Combined, their variance is $1-\alpha_t+\alpha_t(1-\alpha_{t-1}) = 1-\alpha_t\alpha_{t-1}$, which gives:

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\mathbf{z}_{t-2} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\mathbf{z}'
\end{aligned}$$

Following this pattern to $t = 0$ gives:

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{\prod_{i=1}^{t}\alpha_i}\boldsymbol{x}_0 + \sqrt{1-\prod_{i=1}^{t}\alpha_i}\mathbf{z} \\
&= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z}
\end{aligned}$$

2

Next, we prove that the update function used in the algorithm for the baseline DDPM sampling process is equivalent to the formulation used in the DPS sampling algorithm that is dependent on $\hat{\boldsymbol{x}}_0$. Specifically, we want to show that

$$\hat{\boldsymbol{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$$

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

is equivalent to

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t))$$

Substituting the equation, we get:

$$\begin{aligned}
\boldsymbol{x}_{t-1} &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t \\
&\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\left(\frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\boldsymbol{x}_t, t))\right) \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\bar{\alpha}_t}}\boldsymbol{x}_t \\
&\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)(1-\bar{\alpha}_t)\mathbf{s}_\theta(\boldsymbol{x}_t, t)}{(1-\bar{\alpha}_t)\sqrt{\bar{\alpha}_t}}
\end{aligned}$$

Using the fact that $\frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}} = \frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\alpha_t\bar{\alpha}_{t-1}}} = \frac{1}{\sqrt{\alpha_t}}$, we get

$$\begin{aligned}
\boldsymbol{x}_{t-1} &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\boldsymbol{x}_t \\
&\quad + \frac{(1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \\
&= \frac{\alpha_t(1-\bar{\alpha}_{t-1})}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\boldsymbol{x}_t + \frac{1-\alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\boldsymbol{x}_t \\
&\quad + \frac{(1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \\
&= \frac{\alpha_t(1-\bar{\alpha}_{t-1}) + 1 - \alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\boldsymbol{x}_t + \frac{(1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \\
&= \frac{1-\alpha_t\bar{\alpha}_{t-1}}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\boldsymbol{x}_t + \frac{(1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \\
&= \frac{1-\bar{\alpha}_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\boldsymbol{x}_t + \frac{(1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \\
&= \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t + \frac{(1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \\
&= \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t))
\end{aligned}$$

3

Lastly, we want to prove that the update rule for the DDPM sampling process using the score function formulation is equivalent to the one used in the original DDPM paper based on the noise-prediction network $\boldsymbol{\epsilon}_\theta$. Specifically, we want to show that

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t))$$

is equivalent to

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t))$$

Tweedie's formula gives the following expression for $\boldsymbol{x}_0$

$$\begin{aligned}\boldsymbol{x}_0 &= \frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{x_t}\log p_t(\boldsymbol{x_t})) \\ &= \frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t))\end{aligned}$$

Plugging this into the variance-preserving formulation of the forward diffusion model:

$$\begin{aligned}\boldsymbol{x}_t &= \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \\ &= \sqrt{\bar{\alpha}_t}(\frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t))) + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \\ &= \boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}\end{aligned}$$

We can rearrange this to give the following relationship:

$$\begin{aligned}\boldsymbol{x}_t &= \boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \\ \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) &= -\frac{\sqrt{1-\bar{\alpha}_t}}{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \\ \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) &= -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\end{aligned}$$

Plugging this into the update function using the score function,

$$\begin{aligned}\boldsymbol{x}_{t-1} &= \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)) \\ &= \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t + (1-\alpha_t)\frac{-1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}) \\ &= \frac{1}{\sqrt{\alpha_t}}(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon})\end{aligned}$$

where $\boldsymbol{\epsilon}$ is predicted by the network, making $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$.