Diffusion Models for Solving Inverse Imaging Problems

Aditya Vikram Gupta
adivigup@stanford.edu

Abstract—Diffusion models have recently emerged as powerful generative models capable of producing high quality images by learning the underlying data distribution. Beyond unconditional image generation, these models can also serve as strong priors for solving ill-posed inverse imaging problems. In this project, we investigate the use pre-trained diffusion models to address inverse imaging problems such image inpainting, deconvolution, and denoising. We begin by demonstrating unconditional image generation using the Denoising Diffusion Probablistic Model (DDPM) framework with a pretrained score based network. We then explore three approaches for solving inverse problems with diffusion priors. These methods are SDEdit, Score Based Annealed Langevin Dynamics (ScoreALD), and Diffusion Posterior Sampling (DPS). These methods incorporate measurement constraints during the reverse diffusion process in different ways, enabling reconstruction of images consistent with the ground truth data while remaining on the natural image manifold. We evaluate the performance of the methods using PSNR and LPIPS. Our results demonstrate that diffusion based posterior sampling methods significantly improve reconstruction quality compared to simple conditional diffusion models.

Index Terms—Computational Photography, Inverse Problems, Diffusion Models

1 Introduction

Many problems in computational imaging can be formulated as inverse problems, where the goal is to recover an unknown image from incomplete or corrupted measurements. These problems are often ill-posed , meaning that multiple images can explain the observed measurements. To obtain meaningful solutions, prior knowledge about natural images must be incorporated into the reconstruction process.

Traditional approaches rely on hand crafted priors or optimization based regularization techniques. For example, traditional state of the art methods such as BM3D [1] have been widely used for image denoising and restoration, but they often struggle to preserve fine textures and high frequency details, especially in challenging conditions with strong noise or missing data.

Recent advances in deep generative models have provided a new framework for modeling complex image distributions Among these models, diffusions models have demonstrated remarkable performance in high quality image generation. These models learn to estimate the gradient of the log probability density (score function) of the data distribution by training a neural network to iteratively remove noise from corrupted images. This learned score function implicitly captures the structure of natural image, making diffusion models a powerful prior for inverse problems.

In this project, we explore how pretrained diffusion models can be used to solve inverse imaging tasks. We first implement unconditional image generation using the DDPM sampling procedure [2] starting from pure Gaussian noise and iteratively denoising the image. We then investigate three diffusion-based approaches for solving inverse problems: SDEdit [3], Score-Based Annealed Langevin Dynamics (ScoreALD) [4], and Diffusion Posterior Sampling (DPS) [5]. These methods differ in how measurement constraints are incorporated into the reverse diffusion process. We evaluate these approaches on two representative inverse problems: image inpainting and image deconvolution, and analyze their performance using both qualitative and quantitative metrics.

Fig. 1: Examples of noisy images to restore

2 Related Work

Classical image restoration methods rely on explicit statistical priors or handcrafted regularization techniques. One widely used example is BM3D, which exploits non-local self-similarity in images to perform collaborative filtering in a transform domain. While such approaches have been successful for moderate noise levels, they often produce overly smooth reconstructions and struggle to preserve complex image structures in more challenging inverse problems.

The introduction of diffusion-based generative models provided a new approach for modeling image distributions. Early work by Sohl-Dickstein et al. [6] introduced

- *A. Gupta is with Stanford dept of CGOE (Stanford Online) and KLA, 1 Technology Drive, Milpitas, CA 95035*
  E-mail: adityavikram.gupta@kla.com or adivigup@stanford.edu
- *This is the final project for Winter 2026 Iteration of EE367.*

the concept of diffusion probabilistic models, which define a forward process that gradually adds noise to data and a reverse process that learns to recover the original data distribution. Building on this framework, Ho et al. [2] proposed Denoising Diffusion Probabilistic Models (DDPM), which significantly improved training and sampling procedures and demonstrated high-quality image generation results.

Score-based generative modeling further generalized diffusion models by directly learning the gradient of the log probability density function of the data distribution. Song and Ermon [4] showed that these score functions can be used with stochastic differential equations to generate samples through iterative denoising.

More recently, several methods have extended diffusion models to solve inverse problems by incorporating measurement constraints during sampling. SDEdit [3] introduces a simple conditional generation approach in which a corrupted observation is partially noised and then denoised using the diffusion process. While intuitive and easy to implement, this approach does not explicitly model the measurement likelihood.

To address this limitation, posterior sampling methods such as Score-based Annealed Langevin Dynamics (Score-ALD) [4] incorporate likelihood gradients during sampling, allowing the diffusion process to explore solutions that are consistent with both the data prior and the measurements. Diffusion Posterior Sampling (DPS) [5] further improves stability by normalizing likelihood gradients, enabling robust reconstruction for a wide range of inverse problems. These approaches demonstrate the effectiveness of diffusion models as flexible and powerful priors for computational imaging tasks.

3 METHOD

In this section, we describe the diffusion model framework used in this project and the algorithms used for both unconditional image generation and solving inverse imaging problems. All experiments use a pretrained diffusion model trained on the FFHQ256 dataset [7] and follow the variance-preserving (VP) [8] formulation of diffusion models.

3.1 Diffusion Model Formulation

Diffusion models define a forward process that gradually corrupts data by adding Gaussian noise over a sequence of timesteps. In the variance-preserving formulation, the forward diffusion process is defined as

$$\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{z}_{t-1}, \quad \boldsymbol{z}_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{1}$$

where β_t denotes the noise schedule. This Markov chain can be expressed directly as a function of the original clean image x_0:

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{z} \tag{2}$$

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

The reverse diffusion process iteratively denoises the image. Given a noisy image x_t, a neural network estimates the score function

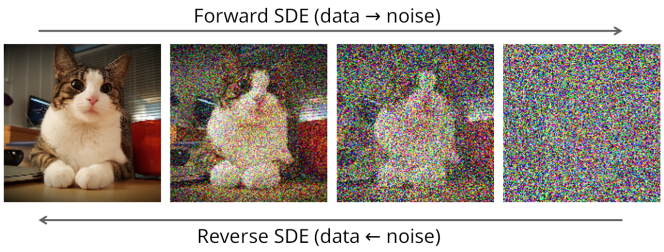


Fig. 2: Forward and Reverse Stochastic Diffrential Equation (SDE) Process in Diffusion

$$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \approx \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) \tag{3}$$

which represents the gradient of the log probability density of the data distribution. Using this score estimate, a denoised estimate of the original image can be computed as

$$\hat{\boldsymbol{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)\right) \tag{4}$$

and the incremental denoising step can be written as

$$\boldsymbol{x}_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{\boldsymbol{x}}_0 + \sigma\boldsymbol{z} \tag{5}$$

where the Gaussian noise z adds robustness and guarantees unique denoising steps when during repeated iterations of the algorithm. The score function serves as a learned prior that constrains the reconstruction process to the manifold of natural images. An alternate form can be achieved by substituting (4) in (5) denoted by (6) (proof in Appendix).

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t + (1-\alpha_t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)\right) + \sigma\boldsymbol{z}. \tag{6}$$

However, in practice (4) and (5) are used. Figure 2 gives a visual overview of these equations.

3.2 Unconditional Image Generation

Unconditional image generation is performed using the DDPM sampling procedure. The process begins with pure Gaussian noise and iteratively applies the reverse diffusion update to generate a sample from the learned data distribution.

At each timestep, the model predicts the score function, which is used to estimate the clean image and compute the next sample in the reverse diffusion chain.

The reverse diffusion process defined in Algorithm 1 uses a score estimate. We can also use a noise estimate as well. In this case, noise and score are related by the following relation:

$$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) = \frac{\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}} \tag{7}$$

See appendix for the proof. Algorithm 1 gives the details of how this is achieved.

Algorithm 1 DDPM Sampling for Unconditional Image Generation

1: Initialize $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: Sample $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\boldsymbol{z} = \mathbf{0}$
4: Compute score estimate $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$
5: Estimate clean image:

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

6: Update to previous step:

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

7: **end for**
8: **return** $\boldsymbol{x}_0$

3.3 Conditioned Image Generation

In this section we see the three methods to solve the inverse problems conditionally. In each algorithm the main idea that is built on top of unconditional generation and makes it conditional is highlighted in red.

3.3.1 SDEdit

SDEdit provides a simple approach for solving inverse problems using diffusion models. Instead of starting from pure noise, the algorithm begins with a partially noised version of the observed measurement i.e. the conditioning is on a noisy version of the observed measurement.

Let $\boldsymbol{y}$ denote the measurement (e.g., a masked or blurred image). The observation is first corrupted with noise to obtain x_T, and the reverse diffusion process is applied starting from timestep T.

This procedure allows the diffusion model to refine the measurement while preserving consistency with the observed data.

Algorithm 2 SDEdit Reconstruction

1: Initialize $x_T = \sqrt{\bar{\alpha}_T}\boldsymbol{y} + \sqrt{1 - \bar{\alpha}_T}\boldsymbol{z}, \boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: Predict score $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$
4: Estimate clean image $\hat{\boldsymbol{x}}_0$
5: Apply reverse diffusion step to obtain $\boldsymbol{x}_{t-1}$
6: **end for**
7: **return** reconstructed image x_0

The starting step T is a hyperparameter where a larger value will produce more noise which will result in a clean image with lower PSNR i.e. less faithful to original image. Smaller T will result in more faithful images.

3.3.2 Score-Based Annealed Langevin Dynamics (ScoreALD)

ScoreALD introduces measurement conditioning into the diffusion sampling process by incorporating gradients of the likelihood function. Let $\boldsymbol{y}$ denote the measurement and $\boldsymbol{A}$ the measurement operator (e.g., masking or convolution). The image formation is then $\boldsymbol{y} = \boldsymbol{A}(x) + \boldsymbol{n}$. where $\boldsymbol{n}$ is zero mean Gaussian noise. The reconstruction problem can be interpreted as sampling from the posterior distribution.

$$p(x|y) \propto p(y|x)p(x) \tag{8}$$

where $p(x|y)$ is the posterior and $p(x)$ is the prior. The likelihood of measurements is

$$p(y|x) \propto e^{-\frac{\|y - Ax\|_2^2}{2\sigma^2}} \tag{9}$$

which means the gradient of log likelihood:

$$\nabla_x \log p(y|x) = -\nabla_x \frac{\|y - Ax\|_2^2}{2\sigma^2} \tag{10}$$

however, the problem is that $\nabla_x \log p(y|x_0) \neq \nabla_x \log p_t(y|x_t)$. Which means we have to approximate this somehow.

ScoreALD then combines the score function with the gradient of the log-likelihood to guide the diffusion process toward solutions consistent with the measurements by doing the following:

$$\nabla_{x_t} \log p(y|x_0) \approx \nabla_{x_t} \log p_t(y|x_t) \approx -\frac{1}{\sigma^2 + \gamma_t^2}(A^T(y - Ax_t)) \tag{11}$$

where γ is guidance strength hyperparameter. The update step can be written as described in algorithm 3.

Algorithm 3 ScoreALD Reconstruction

1: Initialize $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: Predict score $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$
4: Estimate clean image $\hat{\boldsymbol{x}}_0$
5: Apply reverse diffusion step to obtain $\boldsymbol{x}_{t-1}$
6: Apply guidance:

$$\boldsymbol{x}_{t-1} = \boldsymbol{x}_{t-1} - \frac{1}{2(\sigma^2 + \gamma^2)} \nabla_{\boldsymbol{x}_t} \|\mathcal{A}(\boldsymbol{x}_t) - \boldsymbol{y}\|^2$$

7: **end for**
8: **return** $\boldsymbol{x}_0$

3.3.3 Diffusion Posterior Sampling (DPS)

Diffusion Posterior Sampling improves upon ScoreALD by 1) using the approximated clean image for the posterior gradient and 2)stabilizing the conditioning process using normalized likelihood gradients i.e. equation 11 is now updated to:

$$\nabla_{x_t} \log p(y|x_0) \approx \nabla_{x_t} \log p_t(y|x_0 = \mathbb{E}[x_0|x_t]) \tag{12}$$

where

$$\hat{x}_0 = \mathbb{E}[x_0|x_t] \tag{13}$$

i.e. use clean image estimate to calculate the gradient. The normalization term mentioned is defined as follows:

$$\zeta_t = \frac{\zeta}{\left\| \nabla_{x_t} \|A(\hat{x}_0) - y\|^2 \right\|} \tag{14}$$

This ζ_t is a hyperparameter.

Algorithm 4 Diffusion Posterior Sampling

1: Initialize $x_T \sim \mathcal{N}(0, I)$
2: **for** $t = T, \ldots, 1$ **do**
3: Predict score $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$
4: Estimate clean image $\hat{\boldsymbol{x}}_0$
5: Apply reverse diffusion step to obtain $\boldsymbol{x}_{t-1}$
6: Update sample using normalized gradient:

$$\boldsymbol{x}_{t-1} = \boldsymbol{x}_{t-1} - \zeta_t \nabla_{\boldsymbol{x}_t} \|\mathcal{A}(\boldsymbol{x}_0) - \boldsymbol{y}\|^2$$

7: **end for**
8: **return** reconstructed image x_0

Fig. 3: Examples of unconditional generation

4 Experimental Results

4.1 Methodology

To evaluate the results we used to forms of metrics, Peak Signal to Noise Ratio (PSNR) and Learned Perceptual Image Patch Similarity (LPIPS) [9]. PSNR is commonly used to test the clean signals to noisy signals in an image, where higher PSNR is good. LPIPS is a metric used to measure the perceptual similarity between two images, closely aligning with human visual perception. Lower is better for LPIPS. As mentioned before, the pretrained weights used for this task are from the FFHQ256 dataset [7] with a U-Net based architecture for the noise estimation model itself. The noisy and masked images were created using gaussian noise and box masks, examples of which are in 1. The different methods are tested on the same ground truth image for consistency.

4.2 Results

4.2.1 Unconditional Image generation

Here the main goal is to establish what kind of unconditional images can be generated from the pretrained diffusion model. Since these are generated from pure noise there is no value in measuring the PSNR or LPIPS. The iamges shown in Figure 3 are three different instances of random generation.

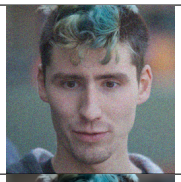
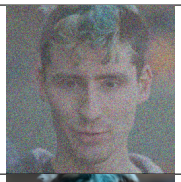
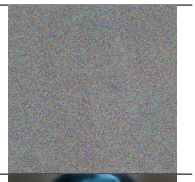
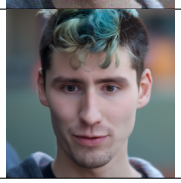
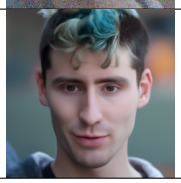
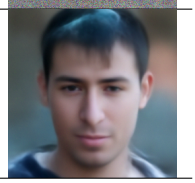
4.2.2 Single shot denoising

In this test we demonstrate the single step denoising using the pretrained model. We added noise to the test image and denoised using the pretrained model. This test demonstrates the ability of the diffusion models to reconstruct clean images by acting as the Gaussian denoisers. The results in table 1 show the denoising ability at three different noise levels by modifying the T in the forward diffusion process. The PSNRs and LPIPS are also shown with reference to the ground truth.

In table 1 we can see the result of increasing noise with T and decrease in the faithfulness and the realism of the generated image. It is still, however, impressive that despite all that noise, an image with features from the ground truth is still being generated. Also note the single shot prediction of clean image is done using (4).

Fig. 4: Ground truth.

TABLE 1: Noised and Denoised images for reference image in Figure 4 along with their PSNR and LPIPS scores

T	30	120	500
x_T			
$\hat{x}_0$			
PSNR	37.48	31.88	22.84
LPIPS	0.032	0.101	0.346

4.2.3 SDEdit

For testing SDEdit, the following time steps were used: T = 300, 500, 700. The algorithm was tested on two sets of problems 1)noise and blur 2)noise and mask. The denoised and reconstructed results show a surprising result where the image tends to be more realistic even if deviating away from the ground truth. This is consistent with how the authors of the SDEdit paper introduced the concept [3] for synthesizing images from paint strokes tasks which aimed to tune the image to be more real. As we can see in the table 2 the images for $T = 700$ are very human like, however, move away from the faithfulness to the ground truth, whereas at $T = 300$ resembles closer to the ground truth, but have features which make it appear not so human like.

4.2.4 ScoreALD

For this testing the T was fixed 1000. The hyperparameter to tune in this case of the annealing factor. In the code this is treated as a range of numbers from [lower limit, upper limit] with 1000 number in between them. Practically, this is achieved by using linspace in python. The denoised results are highly sensitive to the annealing schedule, and for this testing proved to be one of the most challenging hyperparameter tuning. The PSNR and LPIPS results are shown along with the results of the deconvolution and the

Fig. 5: Ground truth.

TABLE 2: Blur+Noise reconstruction with SDEdit for Figure 5 along with their PSNR and LPIPS scores

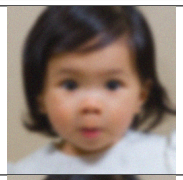
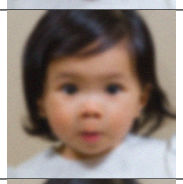
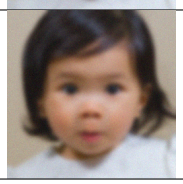
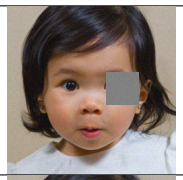
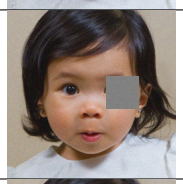
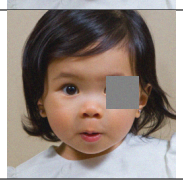
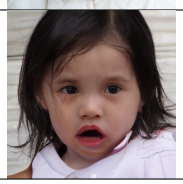
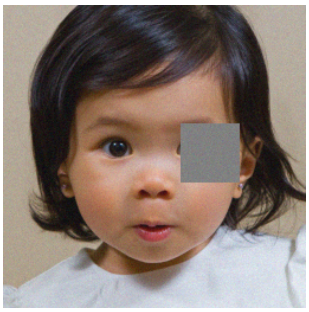
T	Blur+Noise	Recon.	PSNR	LPIPS
300			24.65	0.12
500			20.62	0.196
700			16.62	0.289

TABLE 3: Mask+Noise reconstruction with SDEdit for Figure 5 along with their PSNR and LPIPS scores

T	Blur+Noise	Recon.	PSNR	LPIPS
300			25.00	0.11
500			20.9	0.18
700			16.4	0.29

inpainting tasks. In SDEdit we can already see the quality is not quite good at T=700 so we can be sure it would definitely have been worse at T=1000. With that in mind, ScoreALD definitely shows quantitative improvements. Visually, the hyperparameter variation created either high frequency features when decreased too much and or in the case of inpainting tasks sometimes fail to inpaint. This can be attributed to the fact that a lower annealing schedule leads to the conditioning going too much towards noise, whereas too little annealing schedule leads to too much low frequency compensation. The results for the best hyperparameters are shown in table 4. It also shows the results of non-optimal hyperparameters.

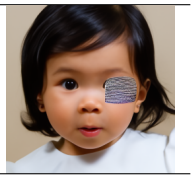
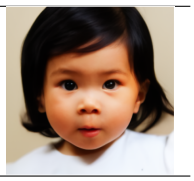
(a) Input for Deconvolution



(b) Input for Inpainting

Fig. 6: Inputs to ScoreALD and DPS

TABLE 4: Deconv. and Inpainting with ScoreALD for Figure 5 along with their PSNR and LPIPS scores

Anneal	Deconv.	Inpaint
[10, 15]		
PSNR	22.54	24.44
LPIPS	0.152	0.12
[15, 20]		
PSNR	24.12	21.95
LPIPS	0.11	0.146

4.2.5 DPS

DPS is implemented for $\zeta = 0.3, 1.0$. Arguably, this was easier to tune that scoreALD, and as such provided the best results. For all the experiments in this, T was 1000. The reconstructed images are quite close to the ground truth images. The results showed less of variation between the hyperparameters. This can be attributed to the stabilizing nature of the normalizing value we added to the gradient. The results are shown in table 5.

5 Discussion and Conclusion

In this project, we explored how pretrained diffusion models can be used as generative priors for solving inverse imaging problems. We evaluated three approaches for incorporating measurement constraints during diffusion sampling: SDEdit, ScoreALD, and Diffusion Posterior Sampling (DPS).

TABLE 5: Deconv. and Inpainting with DPS for Figure 5 along with their PSNR and LPIPS scores

ζ	Inpaint	Deconv
0.3		
PSNR	34.89	28.59
LPIPS	0.02	0.06
1.0		
PSNR	35.12	27.76
LPIPS	0.02	0.11

Our experiments highlight several important observations regarding the behavior of these methods.

First, SDEdit provides a simple and intuitive approach for conditional image reconstruction. By initializing the reverse diffusion process with a partially noised version of the measurement, the algorithm is able to refine the observation while remaining close to the natural image manifold learned by the diffusion model. However, because SDEdit does not explicitly incorporate the measurement likelihood during sampling, its reconstructions may deviate from the true measurement constraints, particularly for more challenging inverse problems such as deconvolution. This limitation is reflected in the lower PSNR values observed in our experiments.

ScoreALD addresses this limitation by explicitly incorporating likelihood gradients during the diffusion sampling process. This allows the algorithm to balance the diffusion prior with measurement consistency. In practice, we observed that ScoreALD produces reconstructions that better satisfy the measurement model compared to SDEdit. However, the method is sensitive to hyperparameters such as the annealing factor, which controls the strength of the likelihood gradient. Improper scaling can lead to unstable updates or degraded reconstruction quality.

Diffusion Posterior Sampling (DPS) further improves the stability of posterior sampling by normalizing the likelihood gradient during each sampling step. This normalization prevents large gradient magnitudes from destabilizing the reverse diffusion process. In our experiments, DPS produced the best quantitative results across both inverse problems, achieving the highest PSNR and lowest LPIPS values. These results suggest that normalized likelihood guidance provides a more stable mechanism for integrating measurement information during diffusion sampling.

Overall, our experiments demonstrate that diffusion models can serve as powerful image priors for solving inverse problems. While simple conditioning approaches such as SDEdit are easy to implement, posterior sampling methods such as ScoreALD and DPS provide more principled ways of incorporating measurement information and generally lead to improved reconstruction quality.

These results highlight the potential of diffusion models as flexible and powerful tools for computational imaging tasks. Future work could explore extensions to more complex inverse problems on different datasets, improved conditioning strategies, and more efficient sampling procedures to reduce the computational cost associated with large diffusion models.

Acknowledgments

I would like to thank the EE367 staff for giving us the opportunity to understand and work on state of the art computational imaging tasks that are highly adaptable to different scenarios across many disciplines. I would also like to thank KLA for giving me the opportunity to take this class.

References

[1] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[2] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 6840–6851.

[3] 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*, 2022. [Online]. Available: https://openreview.net/forum?id=aBsCjcPu_tE

[4] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[5] 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

[6] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 37. PMLR, 2015, pp. 2256–2265. [Online]. Available: https://proceedings.mlr.press/v37/sohl-dickstein15.html

[7] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4401–4410. [Online]. Available: https://openaccess.thecvf.com

[8] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 11 918–11 930. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf

[9] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 586–595.

Appendix A
Additional Derivations

Here we provide additional derivations related to the diffusion model formulation used in the main text.

A.1 Forward Diffusion Process

Starting from the Markov chain formulation

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

we can derive the closed-form expression

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z \tag{16}$$

where $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, $\alpha_t = 1 - \beta_t$ and $z \sim \mathcal{N}(0, I)$
Start by expanding the (15) with a recursive step:

$$x_{t-1} = \sqrt{1-\beta_{t-1}}x_{t-2} + \sqrt{\beta_{t-1}}z_{t-2} \tag{17}$$

Substitute (17) in (15)

$$x_t = \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_{t-1}}z_{t-2}) + \sqrt{(1-\alpha_t)}z_{t-1}$$

$$\implies \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}z_{t-2} + \sqrt{(1-\alpha_t)}z_{t-1}$$

...

expand till $t = 0$

$$x_t = \sqrt{\alpha_t\alpha_{t-1}...\alpha_1}x_0 + \sum_{i=1}^{t}\sqrt{(1-\alpha_i)\prod_{j=i+1}^{t}\alpha_j}z_{i-1} \tag{18}$$

Notice the coefficient of x_0 under the square root is the definition of $\bar{\alpha}_t$. Notice that z is a gaussian, which makes the second term in (18) is a linear combination of i.i.d. which is still a gaussian.

Consider $c_i = \sqrt{(1-\alpha_i)\prod_{j=i+1}^{t}\alpha_j}$.
The variance $\sigma^2 = \sum_{i=1}^{t} c_i^2$. Expanding this gives us

$$\sigma^2 = \sum_{i=1}^{t}((1-\alpha_i)\prod_{j=i+1}^{t}\alpha_j)$$

$$= \sum_{i=1}^{t}(\prod_{j=i+1}^{t}\alpha_j - \alpha_i\prod_{j=i+1}^{t}\alpha_j)$$

Expanding this yields

$$\sigma^2 = 1 - \alpha_1\alpha_2...\alpha_t = 1 - \bar{\alpha}_t \tag{19}$$

(18) then becomes

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

Notes: This proof is expanded on concept of forward process introduced in the DDPM paper as $q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$ and where $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$, $\alpha_t = 1 - \beta_t$ and $z \sim \mathcal{N}(0, I)$

A.2 Two forms of DDPM formulation

Given:

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)S_\theta(x_t, t)) \tag{20}$$

$$x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\hat{x}_0 \tag{21}$$

Prove:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + (1-\alpha_t)s_\theta(x_t, t)) \tag{22}$$

Substitute (20) in (21)

$$\begin{aligned} x_{t-1} &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\ &\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\left(\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t))\right) \\ &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\ &\quad + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)s_\theta) \end{aligned} \tag{23}$$

Consider 2nd coeff:

$$\begin{aligned} &\frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t} \cdot \frac{1}{\sqrt{\bar{\alpha}_t}} \\ = \quad &\frac{1}{\sqrt{\alpha_t}}\frac{(1-\alpha_t)}{1-\bar{\alpha}_t} \quad \because \bar{\alpha}_t = \alpha_t\bar{\alpha}_{t-1} \end{aligned}$$

Substitute back in (23)

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

Combine x_t terms, its coefficient then is:

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

Final Result:

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

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

A.3 Relation between the Score Predication and Noise Prediction and using it in the reverse diffusion step

From the forward diffusion process:

$$\begin{aligned} x_t &= \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \\ \Rightarrow x_0 &= \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}} \end{aligned}$$

This is when we know the exact noise. But practically, we do not know the true noise. It's estimated by $\epsilon_\theta(x_t, t)$.

$$\epsilon \approx \epsilon_\theta(x_t, t) \tag{24}$$

$$\Rightarrow \hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}} \tag{25}$$

Tweedie's formula states:

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

Where $\nabla_x \log p_t(x_t) \approx s_\theta(x_t, t)$ i.e. the gradient of the log probability is the score estimate. Rearranging Tweedie's formula

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

Substitute (25) for $\hat{x}_0$:

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

$$\Rightarrow s_\theta(x_t, t) = -\frac{\epsilon_\theta(x_t, t)}{\sqrt{1-\bar{\alpha}_t}} \tag{26}$$

This gives us the crucial relationship between $s_\theta(x_t, t)$ and $\epsilon_\theta(x_t, t)$. Put this in the reverse diffusion step:

$$\boxed{x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right)} \tag{27}$$