# Denoising Diffusion Restoration Models


## Authors of the Paper:

- **Bahjat Kawar** [ Department of Computer Science Technion, Haifa, Israel ]
- **Michael Elad** [ Department of Computer Science Technion, Haifa, Israel ]
- **Stefano Ermon** [ Department of Computer Science Stanford, California, USA ]
- **Jiaming Song** [ NVIDIA Santa Clara, California, USA ]

## Team Members for Project:

- Girish Madhavan Venkataramani [gmv4@illinois.edu]



## Summary :

In this project, I replicate and analyze Denoising Diffusion Restoration Models (DDRM), originally proposed by Kawar et al., for solving general inverse problems in image processing. DDRM leverages pretrained denoising diffusion probabilistic models (DDPMs) to perform image restoration tasks such as super-resolution, deblurring, and inpainting without retraining for each task. By combining the generative prior of a diffusion model with known degradation operators, DDRM offers a streamlined approach to restoration. I have reproduced the results for gaussian blurring and additional noise using OpenAI's Guided Diffusion Model. An example below shows DDRM successfully recovering image features from a degraded image.


Restored Original Degraded image

# Introduction

Many image processing tasks—such as super-resolution, deblurring, inpainting, colorization, and compressive sensing—can be formulated as linear inverse problems, where the objective is to reconstruct a clean image from noisy measurements obtained via a known linear degradation process. A common approach involves training neural networks in a supervised manner using paired data consisting of clean and degraded images. However, in real-world scenarios like medical imaging, the degradation models can be highly variable or even unknown in advance, making supervised retraining for each new setup impractical. In such cases, unsupervised methods that rely on learned priors and only utilize the degradation model at inference time are favourable. These approaches leverage general assumptions about image structure, such as learned denoisers or proximal operators, allowing effective image restoration **without the need to retrain** on each specific task.

### Limitations of Previous work & a brief overview of approach

Priors based on Deep learning-based methods have shown impressive performance in the unsupervised approaches, where inorder to recover the signal the neural network is used to learn the prior related terms and a likelihood term from the degradation model, combining these two a posterior for the signal is obtained. Now this problem can be reformulated as an optimization problem or posterior sampling problem, which would utilize gradient descent or langevin dynamics to solve. This would often require huge computational resources to train and the model would be sensitive to the set of hyperparameters used.

Coming to Denoising Diffusion Restoration models, being significantly more efficient as compared to existing methods while not sacrificing performance. In the larger picture, DDRM is a diffusion based generated model that gradually and stochastically denoises a sample to desired output, while being conditioned on measurements and degraded inputs. By using this method, the paper introduces a variational inference objective (Eg : Approximating the reverse process to that of true process) for learning the posterior of the degraded image. The paper also prove the equivalence of this logic to that of standard diffusion based generative models.

Advantages with using DDRM :

- Better performance as compared to models like Deep Generative priors, Solving Noisy Inverse Problem Stochastically (SNIPS) and Regularization by Denoising (RED)
- It outperforms neural network baselines for noiseless super-resolution and deblurring (performance measured in PSNR and KID) while being significantly compute efficient
- The performance gap gets wider when measurement noise (z) is involed, no noisy artifacts are observed as compared to iterative methods
- Moreover this method is robust to out of distribution data as well, which will be difficult to achieve with a model that learns priors

## Details of Approach : Preliminary results

**What is a linear inverse degradation ?**

$$y = Hx + z$$

Where, $y \in \mathbb{R}^m$ are the measurements observed for the signal $x \in \mathbb{R}^n$ and $H \in \mathbb{R}^{m \times n}$ is the linear degradation matrix, $z \sim \mathcal{N}(0, \sigma_y^2 I)$. Now given y and $H$, a posterior over the signal can be represented as $p_\theta(x|y) \propto p_\theta(x)p(y|x)$, where the likelihood term ($p(y|x)$) is defined by the linear degradation equation. $x$ can be recovered by sampling (iteratively) from this posterior, moreover since the prior $p_\theta(x)$ can be learned without any information about the degradation process, its an unsupervised method. Trying to perform this task in a supervised manner might not generalize well to unseen cases during training, but it can be more efficient than sampling based methods.

**DDPM :**

A generative model that would learn a model distribution, $p_\theta(x)$ to approximate data distribution $q(x)$ from samples. Diffusion models follow a markov chain structure between timesteps, i.e : $x_T \to x_{T-1} \to \cdots \to x_1 \to x_0$, with the following joint probability distribution :

$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=0}^{T-1} p_\theta(x_t|x_{t+1})$$

After obtaining this distribution, we sample from this and retain only $x_0$ as our final output for the diffusion process. The analogy here is that we destroy true $x_0$ slowly using gaussian noise for T timesteps, this becomes our forward process and is continued till we reach a standard normal distribution. Now we aim for our model to learn the process to remove this noise iteratively so that we end up with a clear image by T steps into our reverse process. [Viewing our data ($x_0$) as a combination of multiple gaussian variables with different means and variances]. Now that the analogy is clear we need some objective function to train the model against, one way to approach this is trying to compare the estimated distribution the distribution obtained from samples defined as follows :

$$q(x_{1:T}|x_0) = q(x_T|x_0) \prod_{t=0}^{T-1} q(x_t|x_{t+1}, x_0)$$

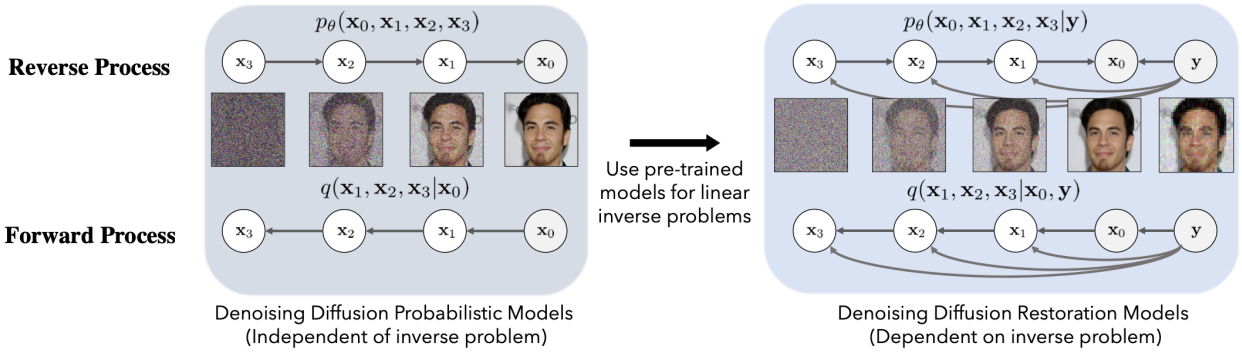
Where at each time step the $q(x_t|x_{t+1}, x_0)$ are conditional gaussians whose mean and variance are known. In order to get a tractable objective function we start with maximizing the variational loglikelihood of $p_\theta(x_0)$. Using the concave property of log function we can lower bound the loglikelihood, which upon further simplification can be written in terms of the KL-Divergence between $p_\theta$ and $q$. This variational lower bound is called : **Evidence Based Lower Bound [ELBO]** and is derived as follows :

$$\sum_{t=1}^{T} \gamma_t \mathbb{E}_{(x_0,x_t)\sim q(x_0)q(x_t|x_0)} \left[ ||x_0 - f_\theta^t(x_t)||_2^2 \right]$$

Different papers use different notations in objective function, the DDPM paper uses neural network to approximate noise ($\epsilon_\theta(x_t, t)$), here we try to predict $x_0$ at each step.

## Details of Approach : DDRM formulation

As discussed previously, current unsupervised methods are generalizable but are inefficient and supervised methods are efficient but not generalizable. Now we derive the variational objective function for DDRM, an overview of the DDRM Process is shown below :




For any linear inverse problem we can modify the DDPM reverse process to be as follows : [Won't work if its not linear inverse as the transitions would become non gaussian and spectral decomposition is not possible]

$$p_\theta(x_{0:T}|y) = p_\theta^{(T)} \prod_{t=0}^{T-1} p_\theta^{(t)}(x_t|x_{t+1}, y)$$

$$q(x_{1:T}|x_0, y) = q^{(T)}(x_T|x_0, y) \prod_{t=0}^{T-1} q^{(t)}(x_t|x_{t+1}, x_0, y)$$

Where, $x_0$ is the final diffusion output, q is the distribution we will be using while computing the variational lower bound. Before we get into proving the equivalence between DDPM and DDRM objectives, There is one more step to be considered to better grasp the derivations :

$$H = U\Sigma V^T$$

$$\bar{y} = \bar{x} + \Sigma^\dagger U^T z,\ \bar{y} = \Sigma^\dagger U^T y,\ \bar{x} = V^T x$$

Where, $H \in \mathbb{R}^{m \times n}$, $V \in \mathbb{R}^{n \times n}$, $U \in \mathbb{m} \times \mathbb{m}$ and $\Sigma \in \mathbb{R}^{m \times n}$ is the diagonal matrix created by the non zero singular values of $H$ arranged in descending order from top to bottom, $m \leq n$ zero singular values are not represented in the SVD decomposition of $H$. Hence by performing our computations in the spectral space, we can observe the values in y that are missing as compared to the original sample and use a diffusion model to generate them, moreover the conditioning on y makes sure the generated pixels are relevant to the remaining part of the image and not some random output from the diffusion model. One thing to be noted here is that the image data ($x$) here is flattened to be vector with one row, so the H matrix corresponding to it would be absolutely massive and performing SVD in it next to impossible (Even with my lab server, I couldn't process 256x256x3 images using raw svd). To overcome this the paper provides methods for efficient SVD computation for different types of degradation (One such method has been implemented on a diffusion model, will be discussed in results)

Now that all our tools are ready, we can get distributions $q$ and $p_\theta$ in the spectral space and proceed to derive the Variational lower bound, (The results below are obtained by conditioning the ddpm forward using y and expanding it)

$$q^{(T)}\left(\bar{\mathbf{x}}_T^{(i)} \mid \mathbf{x}_0, \mathbf{y}\right) = \begin{cases} \mathcal{N}\left(\bar{\mathbf{y}}^{(i)}, \sigma_T^2 - \frac{\sigma_y^2}{s_i^2}\right) & \text{if } s_i > 0 \\ \mathcal{N}\left(\bar{\mathbf{x}}_0^{(i)}, \sigma_T^2\right) & \text{if } s_i = 0 \end{cases}$$

$$q^{(t)}\left(\bar{\mathbf{x}}_t^{(i)} \mid \mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}\right) = \begin{cases} \mathcal{N}\left(\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{t+1}},\ \eta^2\sigma_t^2\right) & \text{if } s_i = 0 \\ \mathcal{N}\left(\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_y / s_i},\ \eta^2\sigma_t^2\right) & \text{if } \sigma_t < \frac{\sigma_y}{s_i} \\ \mathcal{N}\left((1-\eta_b)\bar{\mathbf{x}}_0^{(i)} + \eta_b \bar{\mathbf{y}}^{(i)},\ \sigma_t^2 - \frac{\sigma_y^2}{s_i^2}\eta_b^2\right) & \text{if } \sigma_t \geq \frac{\sigma_y}{s_i} \end{cases}$$

$$p_\theta^{(T)}(\bar{\mathbf{x}}_T^{(i)} \mid \mathbf{y}) = \begin{cases} \mathcal{N}\left(\bar{\mathbf{y}}^{(i)},\ \sigma_T^2 - \frac{\sigma_y^2}{s_i^2}\right) & \text{if } s_i > 0 \\ \mathcal{N}\left(0,\ \sigma_T^2\right) & \text{if } s_i = 0 \end{cases}$$

$$p_\theta^{(t)}(\bar{\mathbf{x}}_t^{(i)} \mid \mathbf{x}_{t+1}, \mathbf{y}) = \begin{cases} \mathcal{N}\left(\bar{\mathbf{x}}_{\theta,t}^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_{\theta,t}^{(i)}}{\sigma_{t+1}},\ \eta^2\sigma_t^2\right) & \text{if } s_i = 0 \\ \mathcal{N}\left(\bar{\mathbf{x}}_{\theta,t}^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_{\theta,t}^{(i)}}{\sigma_y / s_i},\ \eta^2\sigma_t^2\right) & \text{if } \sigma_t < \frac{\sigma_y}{s_i} \\ \mathcal{N}\left((1-\eta_b)\bar{\mathbf{x}}_{\theta,t}^{(i)} + \eta_b \bar{\mathbf{y}}^{(i)},\ \sigma_t^2 - \frac{\sigma_y^2}{s_i^2}\eta_b^2\right) & \text{if } \sigma_t \geq \frac{\sigma_y}{s_i} \end{cases}$$

Where, $i$ is the element index in the flattened signal, $\eta$ and $\eta_b$ are hyperparameters of choice that control the variance between transitions, $\sigma_t^2$ is the variance from scheduler, $\bar{x} = V^T x$ and $\bar{x}_{\theta,t}$ is the diffusion model's esitmate for $\bar{x}_0$. From proposition 3.1 in the paper, it can be seen that the variational distribution when used for computing $q(x_t|x_0)$ gives distribution as $\mathcal{N}(x_0, \sigma_t^2 I)$. This shows that the distribution is not different from a standard diffusion process when its not conditioned on y, this is also a part of the proof where we prove that retraining diffusion model is not necessary.

Ideally, after deriving distributions for $p_\theta$ and $q$ we would try to minimize log likelihood of $p(x_0)$ but this would mean retraining the problem for different H, now we prove that the ELBO objectives for DDRM and DDPM ends up being the same hence we don't need to retrain our models :

$$\begin{aligned} \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0),\, \mathbf{y} \sim q(\mathbf{y}|\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0|\mathbf{y})\right] \geq &- \mathbb{E}\left[\sum_{t=1}^{T-1} D_{\mathrm{KL}}\left(q^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}) \,\|\, p_\theta^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y})\right)\right] \\ &- \mathbb{E}\left[D_{\mathrm{KL}}\left(q^{(T)}(\mathbf{x}_T|\mathbf{x}_0, \mathbf{y}) \,\|\, p_\theta^{(T)}(\mathbf{x}_T|\mathbf{y})\right)\right] \\ &+ \mathbb{E}\left[\log p_\theta^{(0)}(\mathbf{x}_0|\mathbf{x}_1, \mathbf{y})\right] \end{aligned}$$

Here we can improve the performance of our objective by training $p_\theta$ to get close to $q$ as this term is tractable, so we will focus on this term now :

$$\begin{aligned} D_{\mathrm{KL}}\left(q^{(t)}(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}) \,\|\, p_\theta^{(t)}(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{y})\right) &= D_{\mathrm{KL}}\left(q^{(t)}(\bar{\mathbf{x}}_t \mid \mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}) \,\|\, p_\theta^{(t)}(\bar{\mathbf{x}}_t \mid \mathbf{x}_{t+1}, \mathbf{y})\right) \\ &= \sum_{i=1}^{n} D_{\mathrm{KL}}\left(q^{(t)}(\bar{\mathbf{x}}_t^{(i)} \mid \mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}) \,\|\, p_\theta^{(t)}(\bar{\mathbf{x}}_t^{(i)} \mid \mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y})\right), \end{aligned}$$

The first equality is because of $V$ being an orthogonal matrix and the second equality is expanding KL Divergence to all the elements of flattened signal (its expanded element wise so the following result can be applied) : If $p = \mathcal{N}(\mu_1, V_1)$, $q = \mathcal{N}(\mu_2, V_2)$, (univariate gaussians) then

$$D_{\mathrm{KL}}(p \,\|\, q) = \frac{1}{2} \log \frac{V_2}{V_1} + \frac{V_1 + (\mu_1 - \mu_2)^2}{2V_2} - \frac{1}{2}$$

- For cases $s_i = 0$ and $\sigma_t < \sigma_y / s_i$ it can be easily shown that the element wise KL Divergence is $\frac{(\bar{x}_{\theta,t}^{(i)} - \bar{x}_0^{(i)})^2}{2\sigma_t^2}$, through substitution in the element wise divergence
- For the case $\sigma_t \geq \sigma_y / s_i$, upon substitution we end up with $\frac{(1-\eta_b)^2}{2(\sigma_t^2 - \frac{\sigma_y^2}{s_i^2}\eta_b^2)} \cdot (\bar{x}_{\theta,t}^{(i)} - \bar{x}_0^{(i)})^2$

  Here the paper assumes the hyperparameter $\eta_b = \frac{2\sigma_t^2}{\sigma_t^2 + \sigma_y^2/s_i^2}$, Upon substitution into the simplified KL Divergence we get $\frac{(1-\eta_b)^2}{2(\sigma_t^2 - \frac{\sigma_y^2}{s_i^2}\eta_b^2)} = \frac{1}{2\sigma_t^2}$

Therefore Regardless of the case, we end up with the same objective function and rewinding to **DDPM in the Preliminary section** we can see that the **objectives are the same!** To keep the report relevant and short, we will jump into a few efficient SVD techniques

- Denoising : $H = I$ so its a trivial case and doesn't need any V matrix
- Inpainting : $H = I\Sigma P$, where $\Sigma$ is a $k \times n$ matrix with ones in major diagonal, where $k \leq n$ and $P$ is the permutation matrix. Can be stored efficiently because since P is permutations of signal and $\Sigma$ is just selecting the rows of modified signal [slicing]
- Gaussian Blur : Solved using the fact that this process can be done using kernels and 2d Convolutions become products in the frequency domain and we can bring them back together to get actual vector (implemented)
- The paper mentions a few other techniques, that I didn't have time to implement

## Results & References :

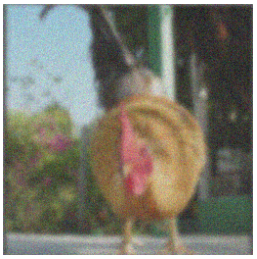
Dataset used : 1000 images from Imagenet Validation dataset
CodeBase (modification) : OpenAI Guided Diffusion Model [Used Gaussian Case]
CodeBase (results) : DDRM Official Repo

Original

Color


Inpainting 1

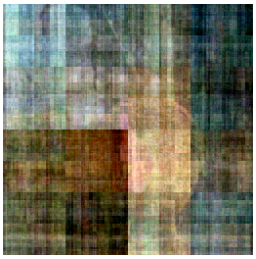
Inpainting 2



Gaussian Blur

Super Res x4

Noisy



Color Subsampling

**Restored Images :**

Original

Color

Inpainting 1

Inpainting 2

Gaussian Blur

Super Res x4

Noisy

Color Subsampling

Only selective samples have been displayed, the **PSNR** values for all restored images are as follows

| **Degradation Type** | **PSNR** |
|---|---|
| Color Subsampling x2 | 22.85 |
| Color Subsampling x4 | 18.86 |
| Inpainting (General) | 28.45 |
| Inpainting 1 | 28.65 |
| Inpainting 2 | 31.50 |
| Denoising | 33.98 |
| Color | 22.14 |

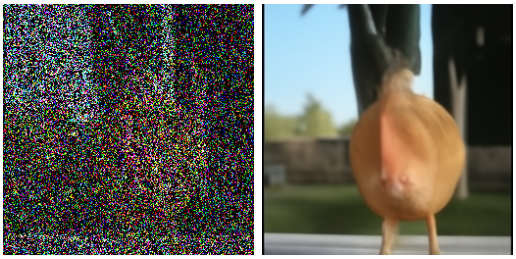
| **Degradation Type** | **PSNR** |
|---|---|
| Deblurring (Uniform) | 26.69 |
| Deblurring (Gaussian) | 28.59 |
| Deblurring (Anisotropic) | 27.92 |
| Super Resolution x2 | 29.20 |
| Super Resolution x4 | 25.97 |
| Super Resolution x8 | 23.24 |
| Super Resolution x16 | 20.87 |
| SR Bicubic x4 | 26.73 |
| SR Bicubic x8 | 24.07 |
| SR Bicubic x16 | 21.10 |

Comparing this to the PSNR values obtained for super resolution x4 task : 25.97 [DDRM] vs 22.90 [Regularization by Denoising (RED)] from Paper, where RED gave the best results as compared to DGP and SNIPS. For above results, $\sigma_y = 0.05, \eta = 0.85$ and $\eta_b = 1$

**Idea :** Start with a pretrained model like the OpenAI's Guided Diffusion Model and modify its sampling process to incorporate DDRM sampling as discussed in Pg. 5.
**What went wrong :** Even though this process seemed simple, the specific SVD formulation required for each type of degradation causes lot of errors while running and was the bottle neck in this entire process, I could code the case for Gaussian blur case alone, whose results are displayed in the first page of this report. Moreover, the paper mentions algorithms to make the sampling process faster, allowing their version of sampling to run within a minute, while my sampling process took around 5 minutes to complete for each batch size, due to time restrictions I couldn't implement this as well.

**Effect of Increasing Variance :** With increasing Noise variance ($\sigma_y^2$) it was observed that the PSNR values had a steady drop, but even with high noise the image gets restored enough to observe the features. One extreme example is shown below with noise standard deviation as 0.8.

| $\sigma_y$ | **PSNR** | $\sigma_y$ | **PSNR** |
|---|---|---|---|
| 0.01 | 28.89 | 0.30 | 25.28 |
| 0.05 | 28.45 | 0.50 | 23.94 |
| 0.10 | 27.53 | 0.80 | 22.41 |
| 0.20 | 26.28 | | |



Inpaint., $\sigma_y = 0.8$ Restored

**Out of Distribution Testing :** Inpainting psnr : 22.29, Deblurring (Gaussing) psnr : 21.96

Original Inpainting Restored Gaussian Blur Restored

We can observe that the PSNR values have fallen as compared to using DDRM on seen data, but still the quality of restoration is comparable to SOTA methods.

## Conclusions :

Overall, given that we have a linear inverse form of degradation we can use DDRM to get a restored image that is close to the original image. This method being unsupervised performs better than supervised methods, while sitting on top of a pretrained diffusion model thereby not needing any additional training. Moreover, it generalizes pretty well for different types of linear inverse problems and most importantly, from results we can see that its robust to restoring data from Out of Distribution Data. Even though DDRM is a step forward in the domain of image restoration models, from my observation it still fails to capture sharp features of images. For example, looking at the out of distribution data the restored images when zoomed in fails to capture the finer details like structures on distant buildings. Although if a Human is involved in the restoration process, it would end up being very similar to the degraded image. This would suggest that we are close in matching the logic of diffusion models to how a human/artist would generate images, but we are not there yet. Similarly running DDRM is pretty fast but its computationally intensive as well, my lab servers were using about 42 GB of VRAM during restoration of a 256 x 256 image, moving to a resolution of 512 x 512 would put considerable strain even on high performant systems like this. Finally, the bottle-neck of this problem is computing SVD and computing SVD in an efficient manner requires us to know about the type of degradation we are dealing with as different degradations have different properties for $H$ matrix that can be exploited.