Deconvolution using ADMM with Diffusion Denoising Prior

Zakaria Patel, Akash Haridas, Kartikaeya Kumar

Abstract—Image deconvolution is an ill-posed inverse problem which is solved with the help of a prior that encodes our understanding of the space of viable solutions. Alternating-Direction Method of Multipliers (ADMM) is an optimization algorithm that provides a natural way to encode a variety of priors for deconvolution. While previous work has shown that using a Gaussian denoiser in the z-update of ADMM produces more natural reconstructions containing less noise and patches, some high-frequency details are irreversibly lost to the blur (low-pass filter) that cannot be recovered by existing methods. We propose using Denoising Diffusion Models, a powerful class of generative models that are shown to produce high quality images from noise, as a Gaussian denoising prior in ADMM. We demonstrate higher quality reconstructions than previous methods and test our method on both in-distribution and out-of-distribution samples.

1 Introduction

Deconvolution is the problem of estimating an image from a blurry and possibly noisy measurement. Restoring a blurry image to its true form has applications ranging from everyday photography to medical and scientific imaging. Image deconvolution is often framed as an inverse problem and solved using iterative optimization techniques incorporating priors.

Alternating-Direction Method of Multipliers (ADMM) [1] is a general optimization algorithm for inverse problems. It allows us to incorporate priors based on our understanding of the characteristics of the original image. Moreover, it allows us to mix and match different image formation models (how the image was corrupted) and regularizers/priors (how we think a natural image looks) [6]. In particular, an existing image denoiser can be used as a prior to produce clean images when deblurring under Gaussian noise. Applying learned denoising priors has been shown to perform well for this task [4].

However none of these methods can recover the high-frequency details of the target image that are irreversibly lost when applying the blur kernel, which acts as a low-pass filter. The only way to "restore" them is to hallucinate them using a powerful generative model.

Denoising diffusion probabilistic models [3] (referred to simply as diffusion models) are a recent development in deep learning that are currently state-of-the-art in image generation. Diffusion models work by iteratively denoising an initial image to invert a forward noising process, and are capable of producing high-quality images starting from pure noise. In this project, we incorporate these powerful models as denoising priors in the ADMM algorithm. We demonstrate higher quality reconstructions than previous methods and test our method on both in-distribution and out-of-distribution samples.

2 Background

2.1 Deconvolution priors

The problem of deconvolution is ill-posed; that is, there are many solutions that satisfy $\mathbf{b} = \mathbf{Ax}$. A prior introduces additional information that constrains the set of solutions to those that are more likely based on our knowledge of the specific problem. Arising from maximum a-posteriori estimation, deconvolution with a prior solves the following problem,

$$\mathbf{x}_{MAP} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2\sigma^2} ||\mathbf{b} - \mathbf{Ax}||_2^2 + \psi(\mathbf{x}) \tag{1}$$

Here, $\psi(\mathbf{x}) = -\log p(\mathbf{x})$, where $p(\mathbf{x})$ is the prior distribution of $\mathbf{x}$. This term is the prior, also called a regularizer in imaging. It allows us to incorporate our understanding of the image's natural statistics into the deconvolution process. The first term is the data fidelity term, which enforces consistency between the reconstruction of $\mathbf{x}$ and the measurements $\mathbf{b}$.

2.2 Alternating-Direction Method of Multipliers

ADMM is an iterative method to solve convex optimization problems, such as Equation 1. In imaging, Equation 1 is rewritten as,

$$\underset{\mathbf{x}}{\text{minimize}} \frac{1}{2} ||\mathbf{Ax} - \mathbf{b}||_2^2 + \lambda\psi(\mathbf{x}) \tag{2}$$

Both terms depend on $\mathbf{x}$. To eliminate this shared dependency, we introduce a slack variable $\mathbf{z}$ and reformulate the optimization problem as,

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}}\ f(\mathbf{x}) + g(\mathbf{z}) \\ &\text{subject to } \mathbf{Dx} - \mathbf{z} = 0 \\ &\text{where } f(\mathbf{x}) = \frac{1}{2} ||\mathbf{Ax} - \mathbf{b}||_2^2,\ g(\mathbf{z}) = \lambda\psi(\mathbf{z}) \end{aligned} \tag{3}$$

Applying Lagrangian optimization to solve Equation 3:

$$L(\mathbf{x},\mathbf{z},\mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{D}\mathbf{x} - \mathbf{z}) \tag{4}$$

where $\mathbf{y}$ is the dual variable. In particular, we take the augmented Lagrangian where we introduce a new quadratic penalty term and a scaled dual variable, $\mathbf{u} = \frac{\mathbf{y}}{\rho}$. We can write the augmented Lagrangian as:

$$L(\mathbf{x},\mathbf{z},\mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \frac{\rho}{2}||\mathbf{D}\mathbf{x} - \mathbf{z} + \mathbf{u}||_2^2 - \frac{\rho}{2}||\mathbf{u}||_2^2 \tag{5}$$

ADMM proposes a three step iterative approach to solving this problem. In order, we update $\mathbf{x}$, $\mathbf{z}$, and then $\mathbf{u}$ using the update rules below. The x-update can be implemented efficiently in the Fourier domain as:

$$\mathbf{x} \leftarrow (\mathbf{A^T A} + \rho\mathbf{D^T D})^{-1}\mathbf{A^T b} + \rho\mathbf{D^T v} \tag{6}$$

The z- and u-updates are then given by:

$$\mathbf{z} \leftarrow \underset{\mathbf{z}}{\operatorname{argmin}}\, g(\mathbf{z}) + \frac{\rho}{2}||\mathbf{v} - \mathbf{z}||_2^2 \tag{7}$$

$$\mathbf{u} \leftarrow \mathbf{u} + \mathbf{D}\mathbf{x} - \mathbf{z} \tag{8}$$

2.2.1 *The z-update*

The form of Equation 7 is similar to that of a general Gaussian denoising problem. Using our original definition for $g(\mathbf{z})$:

$$\begin{aligned} &g(\mathbf{z}) + \frac{\rho}{2}||\mathbf{v} - \mathbf{z}||_2^2 \\ &= \psi(\mathbf{z}) + \frac{\rho}{2\lambda}||\mathbf{v} - \mathbf{z}||_2^2 \end{aligned} \tag{9}$$

In other words, we are free to treat the z-update as a general denoising where we may employ any denoiser of our choosing:

$$\mathbf{z} = \mathcal{D}\left(\mathbf{v}, \frac{\lambda}{\rho}\right) \tag{10}$$

This powerful framework allows us to experiment with different denoisers in conjunction with ADMM.

2.3 DnCNN

Denoising Convolutional Neural Network (DnCNN) [8] is a residual CNN that learns to remove the latent clean image to produce the noise present in an image, which is then subtracted from the input image to produce the denoised image. In prior work it has been used as a denoising prior in ADMM [6].

2.4 Bilateral filtering

A standard Gaussian filter only accounts for spatial proximity. The bilateral filter [5] is a Gaussian filter that also accounts for photometric proximity. The filter kernel is given by:

$$\begin{aligned} BF &= \frac{1}{W_p}\sum_{q \in S} G_{\sigma_s}(||\mathbf{p} - \mathbf{q}||)G_{\sigma_r}(I_\mathbf{p} - I_\mathbf{q})I_\mathbf{q} \\ W_p &= \sum_{\mathbf{q} \in S} G_{\sigma_s}(||\mathbf{p} - \mathbf{q}||)G_{\sigma_r}(I_\mathbf{p} - I_\mathbf{q}) \end{aligned} \tag{11}$$

where G_σ is a 2-D Gaussian kernel. The G_{σ_s} term attenuates contributions from pixels that are located further away, while the G_{σ_r} does the same for pixels whose intensities differ from the pixel of interest. The intensity term makes the bilateral filter edge-preserving. Since pixels separated by an edge will generally differ in intensity, G_{σ_r} assigns low weight to pixels across an edge, thus preventing the edge from being blurred. In effect, this filter performs similar to a Gaussian denoiser while preserving details such as edges and areas of sharp contrast.

2.5 Non-local means denoising

Denoising methods that use local filters only consider fixed neighbourhoods surrounding the pixel of interest. For instance, Equation 11 considers a neighbourhood surrounding pixel $\mathbf{p}$. In contrast, the non-local means (NLM) algorithm operates on the principle of self-similarity [2]. That is, NLM uses Gaussian neighbourhoods that may be located at an arbitrary distance from the pixel being denoised. Buades et al. [2] define the NLM algorithm as:

$$\begin{aligned} NL[v](i) &= \sum_{j \in I} w(i,j)v(j) \\ w(i,j) &= \frac{1}{Z(i)} e^{\frac{||v(\mathcal{N}_i) - v(\mathcal{N}_j)||_{2,\sigma}^2}{h^2}} \\ Z(i) &= \sum_j e^{\frac{||v(\mathcal{N}_i) - v(\mathcal{N}_j)||_{2,\sigma}^2}{h^2}} \end{aligned} \tag{12}$$

The patch similarity is computed as $||v(\mathcal{N}_i) - v(\mathcal{N}_j)||_{2,\sigma}^2$ where $v(\mathcal{N})$ is the Gaussian neighbourhood in the image v with standard deviation σ. The motivation behind this algorithm is the idea that some images contain several nearly identical patch configurations that appear redundant in the clean image, but provide information to restore one another in a noisy image (e.g. natural images). Despite its effectiveness, a major drawback of NLM is its time complexity. Given a search window of $S \times S$, a Gaussian window size of $G \times G$, and image dimensions $N \times N$, the complexity is $O(S^2G^2N^2)$.

2.6 Total variation prior

Most natural images are characterised by large patches of pixels with similar intensity values with occasional sharp change in intensities at the edges. The Total Variation prior encodes this property of natural images by encouraging sparsity of edges (or gradients). The gradients of

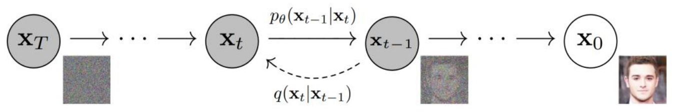
$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_0$

Fig. 1. The forward and reverse processes used in denoising diffusion models [3].

the image($D_x\mathbf{x}$ and $D_y\mathbf{x}$) are calculated by taking the first derivative of the image in the horizontal and vertical directions by convolving the finite difference kernels d_x, d_y over the image $\mathbf{x}$. The regularization term produced by the TV prior is the norm of the gradient over all points in the image.

$$TV(\mathbf{x}) = \sum_{i=1}^{N} \sqrt{((D_x\mathbf{x})_i^2) + ((D_y\mathbf{x})_i^2)} \tag{13}$$

2.7 Diffusion denoising models

Recently, diffusion denoising models have succeeded in generating high-quality images from pure Gaussian noise [3]. The training process consists of two steps: First, a forward process (called diffusion) where noise is gradually added to the image sample over T steps until it resembles pure noise, and second, a reverse process where the original image is recovered by iteratively learning the added noise which needs to be subtracted from the current image to obtain the next less-noisy image in the sequence. The resulting generative model can then be sampled by iteratively denoising either a pure Gaussian noise sample or a partially noisy image.

The forward process is a Markov chain that adds Guassian noise with variance β_t to a scaled result of the previous step in the sequence:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{14}$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{15}$$

As t goes to a large value, the $\sqrt{1-\beta_t}$ factor makes the mean gradually go from $\mathbf{x}_0$ to 0. In other words, the final image is a zero-mean Gaussian distribution with variance β_T, roughly equal to 0.

The reverse denoising process to obtain the sampled image $\mathbf{x}_0$, from the noisy image $\mathbf{x}_T$, is modelled by:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{16}$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \beta_t\mathbf{I}) \tag{17}$$

where $\mu_\theta(\mathbf{x}_t, t)$ is parameterized by a neural network.

3 METHOD

We propose using a diffusion model as a denoising prior in the ADMM algorithm to guide the deconvolution process towards better noise-free solutions. Diffusion models can be used to denoise an image by injecting it into the reverse process at the appropriate timestep according to the variance of noise present in the image.

Since the forward diffusion Markov chain consists of a sequential addition of Gaussian noise, the distribution of the noisy image at timestep t in the forward process conditioned on the initial noise-free image x_0 is given by Equation 4 in Ho et al. [3]:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_t, (1-\bar{\alpha}_t)\mathbf{I}) \tag{18}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i$

Consider an image $\mathbf{y}$ corrupted with zero-mean Gaussian noise of variance β^* to give $\bar{\mathbf{y}} = \mathbf{y} + \mathcal{N}(0, \beta^*)$, which can be rewritten as $\bar{\mathbf{y}} = \mathcal{N}(\mathbf{y}, \beta^*)$. Comparing this with the forward diffusion process $q(\mathbf{x}_t|\mathbf{x}_0)$, we compute the timestep where $(1-\bar{\alpha}_t)$ is closest to β^*. This is the timestep t^* at which the noisy image should be injected into the reverse process to produce a noise-free image.

$$t^* = \underset{t}{\operatorname{argmin}}[\beta^* - (1-\bar{\alpha}_t)] \tag{19}$$

Since the noise variance schedule β_t is fixed, it is easy to compute t^* given a diffusion model and a noisy image.

We use this diffusion denoiser as a denoising prior in ADMM by plugging it into Equation 10.

4 EXPERIMENTS

4.1 Setup

In each experiment, we start with a clean image, convolve with a blur kernel, apply synthetic Gaussian noise of variance σ^2, perform deconvolution with a given method and compare the resulting reconstruction with the original clean image. We experiment with $\sigma = \{0.01, 0.1, 0.5\}$, ranging from a low to very high noise level.

To measure the performance of each method, we use PSNR, a measure of the quality of reconstruction from a degraded image, as well as subjective visual quality. We also compare the total and per-iteration runtime required to achieve the optimal PSNR across priors.

4.1.1 Baselines

In each experiment, we produce baseline results with the DnCNN, bilateral filter, NLM and TV priors.

For each noise level σ in our experiments, we train a DnCNN on RGB images from the BSDS300 dataset that are corrupted with that noise level. Our DnCNN consists of 5 convolutional layers with kernel size 3 and 32 channels in each intermediate layer.

The ADMM update steps are performed on each channel of the image separately. In cases where the denoiser operates

TABLE 1
Parameters used in the ADMM priors

Prior	$\mathcal{G}_s$ size	$\mathcal{G}_r$ size	σ_s	σ_r	S
Bilateral	5×5	5×5	0.1	0.25	-
NLM	5×5	-	0.1	-	5×5
TV	-	-	-	-	-

on RGB images, we perform three ADMM updates on the different channels in parallel and rebuild the RGB array before performing denoising.

The remaining priors are used out of the box, and their parameter settings are listed in table 1.

4.1.2 *Diffusion denoising prior*

For the diffusion denoising prior, we use a pretrained DDPM model from the Hugging Face Diffusers library [7] trained on the CelebA-HQ-256 dataset consisting of face images. We modify the sampling code to take an initial image, compute the timestep t^* and start the reverse process (sampling) at this intermediate timestep.

4.2 Experimental scenarios

We first evaluate the performance of ADMM with diffusion denoising prior in deconvolving images belonging to the same class that the model was trained on (in-distribution). Since the model is trained on face images, we use an image of a face for testing.

We then evaluate the performance of our method on a non-face image (out-of-distribution) to test whether the diffusion model can act as a general denoising prior, as well as to test the importance of the model's learned class-specific knowledge in guiding the reconstruction towards a good solution.

4.3 Tuning

TABLE 2
Number of ADMM iterations required to achieve optimal PSNR with various denoising priors

σ	DnCNN	Bilateral	NLM	Diffusion
0.01	75	30	5	4
0.1	10	5	5	5
0.5	10	3	5	15

4.3.1 *ADMM timesteps*

The performance of a denoising prior is sensitive to the number of ADMM iterations. We observe this to varying degrees across the baselines and the diffusion prior. To this end, we compute the PSNR between the ground truth and the intermediate result of ADMM at each iteration. We tune each method by identifying the iteration at which the PSNR peaks or saturates. We assume that the general trends observed here would hold when our methods are applied on novel images. We exclude NLM from this tuning process as it is computationally costly, instead opting to keep the number of iterations fixed. We expect that its performance would improve if we were able to tune it freely.

4.3.2 *Diffusion timesteps*

In our main experiments, we computed the timestep t^* at which we inject our image in the reverse process using Equation 19. The same t^* is used at every iteration of ADMM. However, we intuit that as the ADMM iterations progress, the intermediate image contains less noise, so the noise variance β^* used to compute t^* must be reduced.

To this end, we experiment with two more schemes for scheduling β^*. First, we vary the noise variance as $\beta^*_{i+1} = \lambda_i \beta^*_i$, where $0 < \lambda < 1$, throughout the N iterations of ADMM. Second, we linearly decrease the noise variance as $\beta^*_{i+1} = \beta^*_i - c$, where $0 < c < \frac{\beta^*_i}{N}$.

Due to time constraints, we did not produce detailed results from the latter two schemes. However, a comparison of the convergence properties is presented in Figure 4.

5 Results and Analysis

Figure 2 shows the results of our first experiment, where the test image belongs to the class that the diffusion model was trained on (in-distribution). At low noise levels $\sigma = \{0.01, 0.1\}$, even though all baseline methods produce a reasonable image, our method produces the cleanest image with the least artifacts. At the high noise level $\sigma = 0.5$, our method produces an image belonging to the data distribution (faces), while all baselines fail under the extremely high level of noise.

Figure 3 shows the results of our second experiment, where the test image is out-of-distribution from what the diffusion model was trained on. Surprisingly, at low noise levels $\sigma = \{0.01, 0.1\}$ our method still produces reasonable results, although it is outperformed by the DnCNN denoiser. At the high noise level $\sigma = 0.5$, our method begins to distort the image significantly.

Table 3 shows the runtime per ADMM iteration with the different priors. Diffusion models are known to be slow and computationally expensive due to their iterative nature. Furthermore, the runtime of ADMM with the diffusion prior depends on the noise level through the timestep t^*. The baselines are much faster to use, although they present a trade-off between computational cost and quality.

5.1 Hallucination of details

Interestingly, the reconstruction metric PSNR fails to capture the visual quality of the generated images. Even though our method produces the best visual quality on the in-distribution image, it scores a lower PSNR than the DnCNN prior. This suggests that our method "hallucinates" details that were irrecoverably lost to the blur. While it is not reflected in the PSNR, this behaviour is beneficial in cases where the measured image is not highly corrupted.

Images corrupted with high noise are injected deeper into the reverse process, such that more diffusion steps are required to completely denoise it. A consequence of this is that additional unwanted details are hallucinated by the diffusion model. This is illustrated in figure 2, where the reconstructed face image from $\sigma = 0.5$ is qualitatively different from the original image.

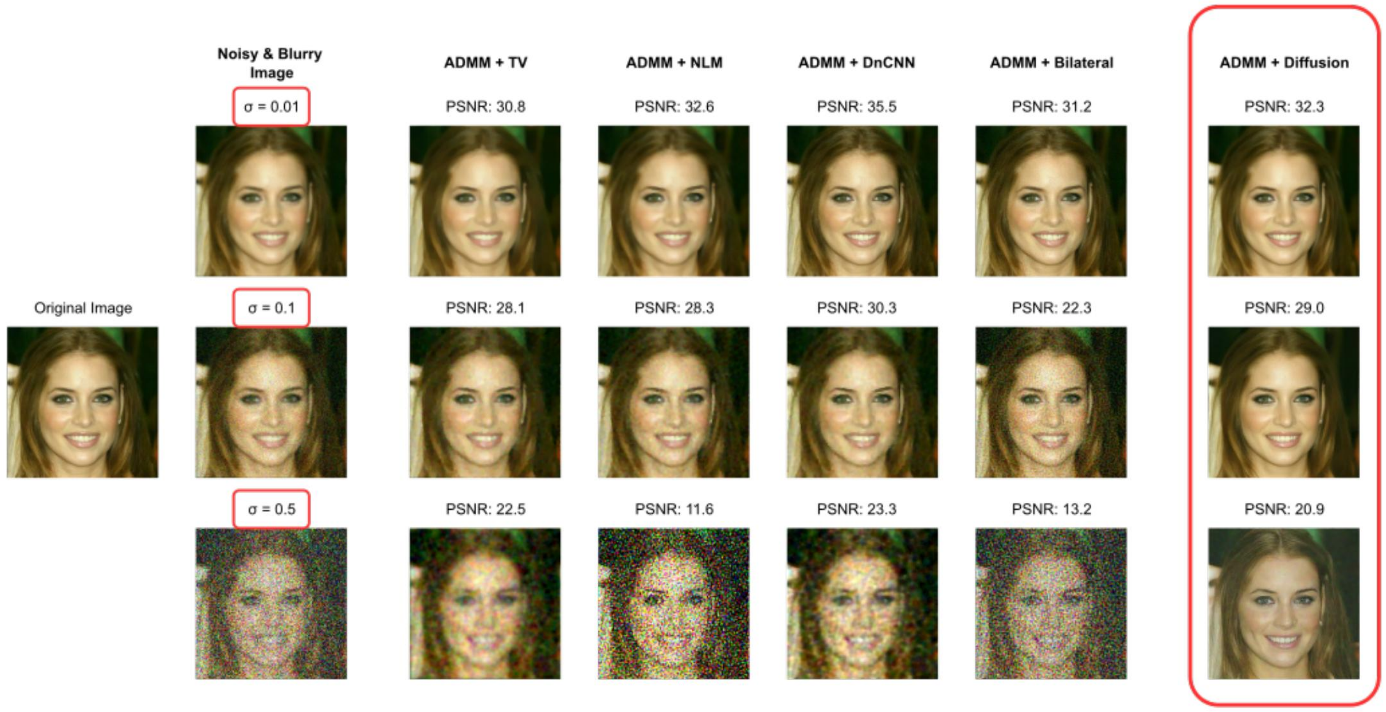

Fig. 2. In the first experiment, we perform deconvolution of an image belonging to the same class (in-distribution) that the diffusion model was trained on (faces). The image is corrupted with noise levels of $\sigma = \{0.01, 0.1, 0.5\}$. The number of iterations of ADMM is chosen for each prior such that the resulting PSNR between the reconstruction and the ground truth is as high as possible. The diffusion prior is comparable to the DnCNN in terms of PSNR, but produces the visually cleanest results.

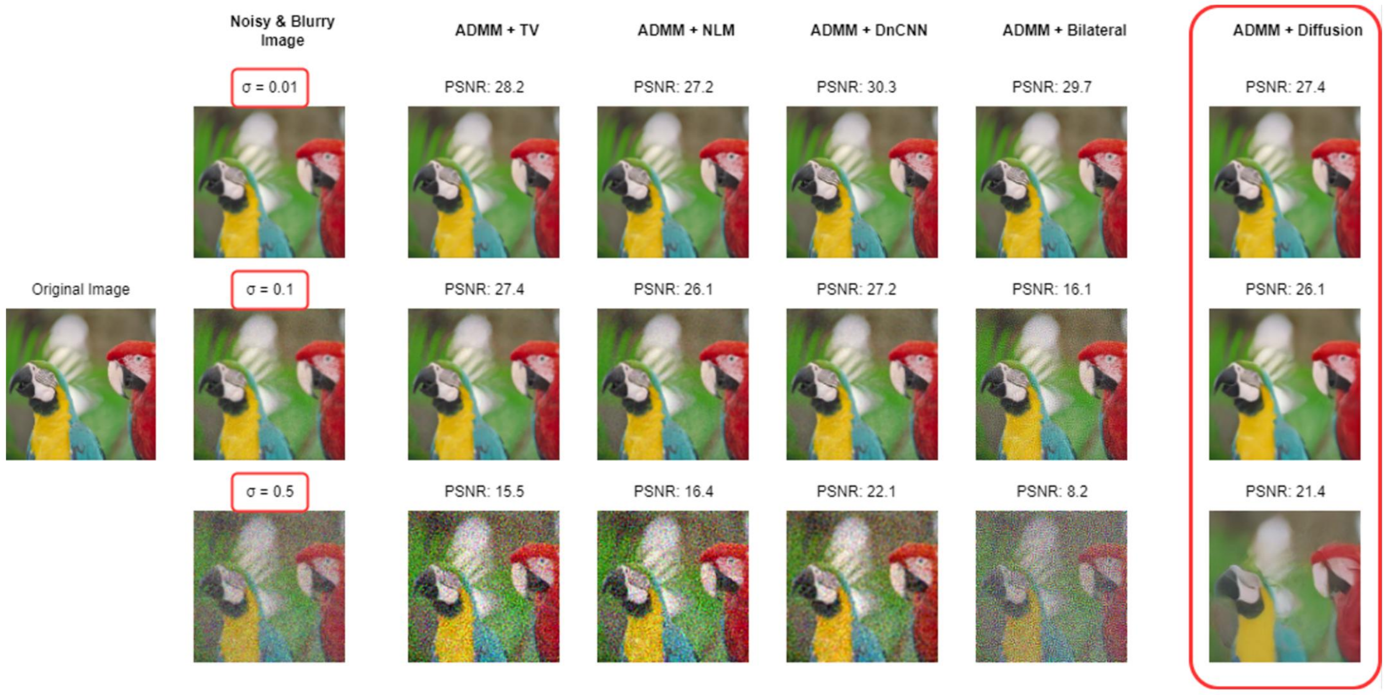

Fig. 3. In the second experiment, we perform deconvolution of an image belonging to a different class (out-of-distribution) than what the diffusion model was trained on (faces). The image is corrupted with noise levels of $\sigma = \{0.01, 0.1, 0.5\}$. At low noise, our method performs comparably to the baselines. At high noise however, our method produces a distorted result.

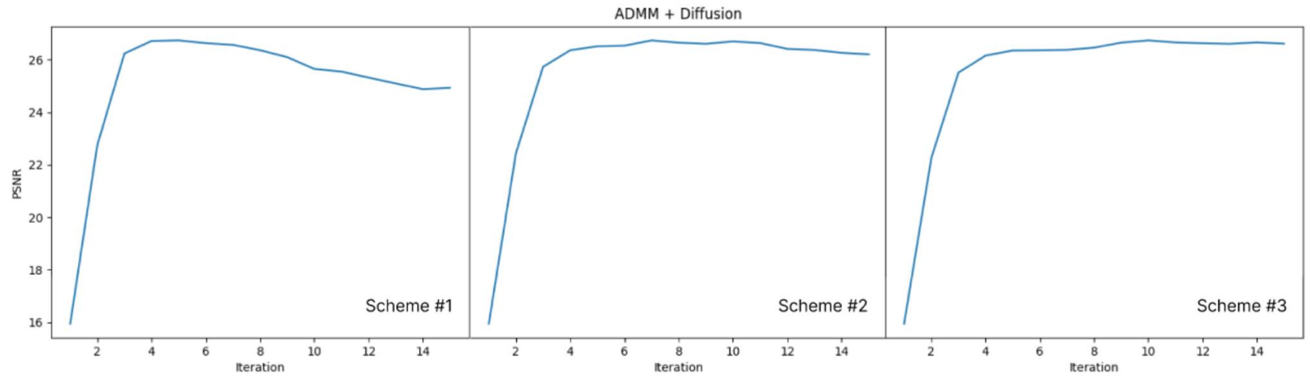

Fig. 4. The timestep t^* at which we inject our image into the reverse process depends upon the noise variance. Scheme 1 is constant noise variance for all ADMM iterations. Scheme 2 reduces it exponentially with $\lambda = 0.92$. Scheme 3 reduces it by a constant amount $c = \frac{\beta^*}{N}$ each ADMM iteration. Reducing the noise variance, either exponentially or linearly, both lead to an increase in PSNR over the ADMM iterations.

5.2 Diffusion model tuning results

The reverse diffusion process produces a noise-free image from pure noise iteratively across 1000 steps. For images with finite noise, we estimate the injection timestep t^* to be somewhere within this range. Table 4 shows t^* for different noise levels.

TABLE 3
ADMM runtime with various priors

Method	s/iteration
TV	0.013
NLM	-
DnCNN	0.058
Bilateral	5.4
Diffusion @ $\sigma = 0.01$	18.75
Diffusion @ $\sigma = 0.1$	67.36
Diffusion @ $\sigma = 0.5$	179.16

TABLE 4
Injection timestep for various noise levels

σ	t^*
0.01	27
0.1	97
0.5	258

As mentioned in section 4.3.2, we naively assume a constant noise variance β^* throughout all iterations of ADMM. However, an effective denoising prior will reduce this noise variance across iterations, such that $\beta^*_{i+1} < \beta^*_i$. Therefore we experiment with reducing the noise variance in every ADMM iteration. While we were not able to incorporate these to produce our main result due to time constraints, we examined its effect on a face image corrupted with noise level $\sigma = 0.1$. By employing either linear or exponential schemes, we can prevent degradation in the PSNR due to excess generation of fine details, as illustrated in figure 4.

6 Future work

The diffusion model used in this work can be replaced with a latent diffusion model (LDM), which has the following advantages. First, LDMs are significantly more computationally efficient. Second, LDMs pretrained on large diverse datasets have recently become available, which could enable a class-agnostic denoising prior overcoming the limitation revealed by experiment 2. Third, modern LDMs can be readily guided/conditioned on text prompts, which could enable a greater degree of control. However, an issue with using LDMs as denoisers is that it is challenging to estimate the intermediate timestep corresponding to the image noise level.

As mentioned before, our method is prone to hallucinating details, which may or may not be advantageous. Future work could explore the scheduling of diffusion timesteps over ADMM iterations to control the amount of generated detail depending on the amount of degradation in the observed image.

7 Acknowledgements

We thank Prof. David Lindell and the course staff for the excellent resources, particularly the course material which we adapted for this project. We also thank Hugging Face for the Diffusers library which makes using diffusion models easy and accessible.

References

[1] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[2] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 2, pages 60–65. Ieee, 2005.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[4] Ernest Ryu, Jialin Liu, Sicheng Wang, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pages 5546–5557. PMLR, 2019.

[5] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998.

[6] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013.

[7] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022.

[8] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.