Outlier-Robust Diffusion Solvers for Inverse Problems

Yang Zheng[1] Jiahua Liu[1]
Tongyao Pang[2] Wen Li[1] Zhaoqiang Liu[1,*]
[1]University of Electronic Science and Technology of China [2]Tsinghua University
202511081645@std.uestc.edu.cn 202421081114@std.uestc.edu.cn
typang@tsinghua.edu.cn liwenbnu@gmail.com zqliu12@gmail.com

Abstract

Methods based on diffusion models (DMs) for solving inverse problems (IPs) have recently achieved remarkable performance. However, DM-based methods typically struggle against outliers, which are common in real-world measurements. In this work, to tackle IPs with outliers, we first refine the measurement via explicit noise estimation to mitigate the effect of noise. Subsequently, we formulate an iteratively reweighted least squares objective based on the Huber loss to address the outliers. We propose a method utilizing gradient descent to approximately solve the corresponding optimization problem for the robust objective. To avoid delicate tuning of the learning rate required by the gradient descent method, we further employ the conjugate gradient method with an efficient strategy for updating. Extensive experiments on multiple image datasets for linear and nonlinear tasks under various conditions demonstrate that our proposed methods exhibit robustness to outliers and outperform recent DM-based methods in most cases.

1. Introduction

Inverse problems (IPs) encompass a broad class of tasks focused on estimating underlying signals from degraded and noisy observations. IPs are crucial in numerous domains, including audio signal processing [43, 50], remote sensing [63], and image restoration [2, 67]. The typical objective of an IP is to recover an unknown signal $\boldsymbol{x}_0^* \in \mathbb{R}^n$ from a noisy observation $\boldsymbol{y} \in \mathbb{R}^m$, modeled as [19, 49]:

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}_0^*) + \boldsymbol{\nu}, \tag{1}$$

where $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^m$ denotes a forward operator and $\boldsymbol{\nu} \in \mathbb{R}^m$ represents unknown stochastic noise.

Recent advances in diffusion models (DMs) have catalyzed the development of a wide range of DM-based approaches for solving IPs. These methods can be broadly classified into two paradigms. The first paradigm comprises methods tailored to specific IPs, such as super-resolution [20, 52], inpainting [10, 40], and deblurring [48, 51], which are trained in a supervised fashion using paired datasets of degraded observations and corresponding ground-truth signals. While these methods often produce high-quality reconstructions, their dependence on task-specific training and large volumes of labeled data introduces computational overhead and limits their generalizability. The second paradigm circumvents these limitations by leveraging pretrained DMs to solve IPs without requiring additional fine-tuning [6, 8, 16, 27, 45, 56, 58, 61, 64, 65]. These methods implicitly assume that the target signal is close to the generative manifold learned by a pretrained DM and exploit the learned prior of the DM to facilitate signal reconstruction. The methods of the second paradigm have recently gained popularity as they do not require retraining for specific tasks and avoid the need for paired datasets.

However, recent DM-based methods typically only consider how to address the noise contained in the observation, but neglect the effect of outliers. In practical scenarios, the observation is often affected not only by stochastic Gaussian noise but also by outliers, such as those encountered in image deblurring tasks [3, 13, 14, 47], and other IPs [11, 23, 59]. To model these outliers, we employ the arbitrary corruption model proposed in [23, 59]. This model stipulates that a fraction $\rho \in [0, 1)$ of the observation are arbitrarily replaced by an outlier vector $\boldsymbol{\xi} \in \mathbb{R}^m$, such that:

$$y_i = \begin{cases} \xi_i, & \text{if } i \in \mathcal{C} \\ [\mathcal{A}(\boldsymbol{x}_0^*)]_i + \nu_i, & \text{if } i \notin \mathcal{C} \end{cases}, \tag{2}$$

where $\mathcal{C}$ is the unknown set of corrupted indices and its elements are randomly selected from the indices of the measurement $\boldsymbol{y}$ with the probability ρ. ξ_i denotes the i-th entry of the outlier vector. Motivated by the limitations of recent DM-based methods on IPs with outliers, this work proposes robust DM-based methods for IPs with outliers.

*Corresponding author.

1.1. Related work

We discuss related work in two parts: 1) Inverse problems with diffusion models and 2) methods for outlier handling.

Inverse problems with diffusion models: Since the pioneering advancements in DMs [22, 55, 60], extensive research has aimed to enhance both the computational efficiency and the quality of generated outputs [26, 38, 57, 70]. Furthermore, DMs have emerged as powerful tools for addressing IPs [9, 15, 33, 41, 44, 58, 64], achieving remarkable performance across diverse applications.

DM-based methods for IPs reconstruct underlying signals through different frameworks. For instance, MCG [7] and DPS [9] employ Tweedie's formula to estimate the reconstructed signal, supplementing this with gradient-based posterior corrections. DiffPIR [71] adopts a proximal update scheme to approximate the conditional posterior mean. DCPS [24] introduces divide-and-conquer posterior sampling to leverage the inherent structure of DMs. REDdiff [41] leverages variational inference by introducing a tractable surrogate distribution to approximate the real posterior. DAPS [68] decouples the diffusion sampling trajectory to enable early-stage error correction.

Although recent DM-based approaches yield promising results for IPs under Gaussian noise, these methods often struggle to accurately recover the underlying signal when the measurements contain outliers, highlighting the need for robust DM-based methods against IPs with outliers.

Methods for outlier handling: Outliers are common in real-world scenarios due to faulty sensor readings or brief interference during data transmission [23, 59]. The central idea behind methods for outlier handling lies in constructing a robust loss function to mitigate the influence of the outliers. To achieve this mitigation, various robust methods have been proposed.

One class of methods achieves robustness through a sample selection strategy, such as trimmed loss (TL), which explicitly discards the data points that contribute the highest loss [34, 35, 53]. Another class involves robust estimators, such as the median-of-means (MOM) method, which leverages the resistance of median to extreme values [23, 31]. MOM estimates the expected risk by partitioning the data into blocks and taking the median of the block means, thereby filtering out groups that are heavily biased by outliers. However, both TL and MOM methods discard the information of data points flagged as potential outliers, thus failing to fully utilize the information in the measurement data. Alternatively, the Huber loss achieves robustness by leveraging all measurement information and differentially penalizing the data points [11, 17, 59]. Specifically, the Huber loss achieves robustness by applying a quadratic penalty to data points with small discrepancies and a linear penalty to those with large discrepancies, thus limiting the loss contribution of outliers without discarding any measurement information.

Additionally, methods for outlier handling typically incorporate prior knowledge regarding the underlying signal. This is often achieved either through manually designed regularization terms [5, 37, 47, 59] or by leveraging generative models, such as generative adversarial models [23, 66]. Given the strong generative capabilities of DMs, in this paper, we utilize pre-trained DMs coupled with methods for outlier handling to solve IPs contaminated with outliers.

1.2. Contributions

Our main contributions are summarized as follows:

- We first introduce an optimization objective employing the Huber loss coupled with explicit noise estimation to mitigate the influence of outliers. We approximately solve the corresponding optimization problem via gradient descent, and the approach is referred to as Robust-GD.
- To circumvent the need for delicate learning rate tuning in Robust-GD, we substitute gradient descent with the conjugate gradient method and introduce an efficient update strategy tailored for this optimization.The corresponding approach is referred to as Robust-CG.
- We conduct extensive empirical evaluations across diverse image datasets and under various conditions. The results show that our algorithms achieve superior performance compared to recent DM-based methods in most cases, confirming the effectiveness of our methods.

2. Preliminaries

DM comprises a forward stochastic process that gradually corrupts data into noise and a corresponding reverse process that aims to reconstruct the data from the noise. The forward dynamics are described by the following stochastic differential equation (SDE):

$$\mathrm{d}\boldsymbol{x}_t = f(t)\,\boldsymbol{x}_t\,\mathrm{d}t + g(t)\,\mathrm{d}\boldsymbol{w}_t, \quad \boldsymbol{x}_0 \sim p_0, \tag{3}$$

where $f(t)$ and $g(t)$ are time-dependent drift and diffusion coefficients, and $\boldsymbol{w}_t \in \mathbb{R}^n$ is a standard Wiener process. Let p_t be the marginal distribution of $\boldsymbol{x}_t$. For $t \in [0, T]$, the conditional distribution of $\boldsymbol{x}_t|\boldsymbol{x}_0$ is Gaussian with $\boldsymbol{x}_t|\boldsymbol{x}_0 \sim \mathcal{N}(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I}_n)$, where α_t and σ_t are differentiable, non-negative and monotonic functions with bounded derivatives, such that the signal-to-noise ratio α_t^2/σ_t^2 is decreasing in t. Functions α_t and σ_t are chosen that the marginal distribution at the terminal time p_T approximates $\mathcal{N}(\mathbf{0}, \tilde{\sigma}^2 \boldsymbol{I}_n)$ for some $\tilde{\sigma} > 0$.

To match the SDE in Eq. (3) with the conditional distribution of $\boldsymbol{x}_t|\boldsymbol{x}_0$ specified above, the coefficients $f(t)$ and $g(t)$ need to satisfy the following [38]:

$$f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}, \quad g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\,\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\,\sigma_t^2. \tag{4}$$

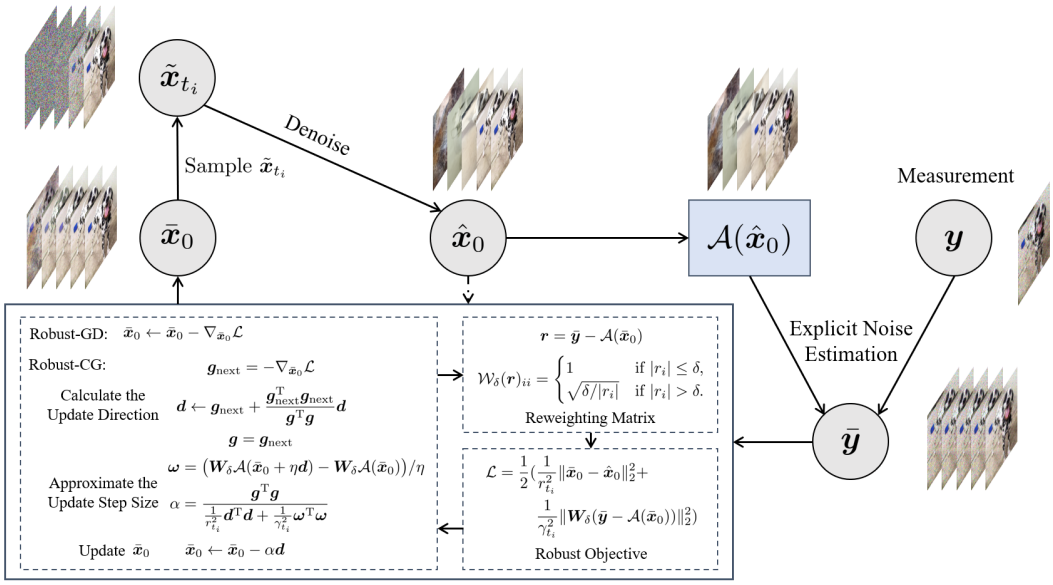

Figure 1. Overview of our proposed Robust-GD and Robust-CG methods. At each timestep t_i, we first calculate the signal estimate $\hat{\boldsymbol{x}}_0$. Next, we refine the measurement $\boldsymbol{y}$ to $\bar{\boldsymbol{y}}$ using explicit noise estimation. We then formulate a robust objective function designed to handle outliers and approximately solve the corresponding optimization problem using either the gradient descent method (Robust-GD) or the conjugate gradient method (Robust-CG). After obtaining the result $\bar{\boldsymbol{x}}_0$, we sample $\tilde{\boldsymbol{x}}_{t_{i-1}} \sim \mathcal{N}(\alpha_{t_{i-1}}\bar{\boldsymbol{x}}_0, \sigma_{t_i}^2\boldsymbol{I}_n)$. After N iterations, we use $\tilde{\boldsymbol{x}}_{t_0}$ as the final reconstructed image.

Following [60], the forward SDE admits a corresponding time-reversed diffusion process as follows:

$$\mathrm{d}\boldsymbol{x}_t = \big(f(t)\,\boldsymbol{x}_t - g^2(t)\,\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)\big)\,\mathrm{d}t + g(t)\,\mathrm{d}\bar{\boldsymbol{w}}_t, \quad (5)$$

where $\bar{\boldsymbol{w}}_t$ denotes a reverse-time Wiener process and $\nabla_{\boldsymbol{x}_t}\log p_t$ is the score function.

As shown in [60], a corresponding deterministic process with the same marginal distribution is defined by the following ordinary differential equation (ODE):

$$\mathrm{d}\boldsymbol{x}_t = \big(f(t)\,\boldsymbol{x}_t - \tfrac{1}{2}g^2(t)\,\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)\big)\,\mathrm{d}t. \quad (6)$$

Sampling can thus be performed via numerical solvers of this ODE, given that the unknown score function $\nabla_{\boldsymbol{x}_t}\log p_t$ is approximated by a certain neural network function. Concretely, the score function $\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)$ can be approximated by $(\alpha_t\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_t)/\sigma_t^2$ [29, 39], where $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ is a data prediction network.

Let the time interval $[0, T]$ be partitioned as $0 = t_0 < t_1 < \cdots < t_N = T$. Then, for instance, the DDIM [57] sampling scheme can be regarded as a first-order numerical solution to Eq. (6), and can be expressed as follows:

$$\tilde{\boldsymbol{x}}_{t_{i-1}} = \frac{\sigma_{t_{i-1}}}{\sigma_{t_i}}\tilde{\boldsymbol{x}}_{t_i} + \sigma_{t_{i-1}}\left(\frac{\alpha_{t_{i-1}}}{\sigma_{t_{i-1}}} - \frac{\alpha_{t_i}}{\sigma_{t_i}}\right)\boldsymbol{x}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_{t_i}, t_i). \quad (7)$$

3. Methods

This section introduces our proposed methods, referred to as Robust-GD and Robust-CG, two novel DM-based approaches for solving IPs with outliers. In Section 3.1, we present the concept of explicit noise estimation to mitigate the effect of noise in the measurement. In Section 3.2, we formulate an iteratively reweighted least squares objective based on the Huber loss to address the outliers. We approximately solve the corresponding optimization problem via gradient descent, which we summarize as the Robust-GD algorithm. To avoid the delicate fine-tuning of the learning rate required by Robust-GD, in Section 3.3, we replace the gradient descent method with the conjugate gradient method. We also propose an efficient updating strategy, summarizing the complete algorithm as Robust-CG. We visualize the entire sampling process of our proposed Robust-GD and Robust-CG methods in Figure 1.

3.1. Explicit noise estimation

Following previous work [68, 71], the estimation of the underlying signal at time step t using DMs is typically formulated as the following optimization problem:

$$\min_{\bar{\boldsymbol{x}}_0}\frac{1}{2r_t^2}\|\bar{\boldsymbol{x}}_0 - \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)\|_2^2 + \lambda\|\boldsymbol{y} - \mathcal{A}(\bar{\boldsymbol{x}}_0)\|_2^2, \quad (8)$$

where $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ estimates the underlying signal given the noisy latent variable $\tilde{\boldsymbol{x}}_t$. Here, $r_t > 0$ is a time-dependent hyperparameter, and $\lambda > 0$ is a hyperparameter related to the measurement noise. Various DM-based approaches utilize different strategies for estimating the underlying signal. For instance, DiffPIR uses Tweedie's formula to obtain the estimate $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t) = \boldsymbol{x}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}_t, t)$, whereas DAPS obtains $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ through a multi-step reverse process starting from

time t. In this paper, our approach is similar to DAPS, as we also obtain $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ through a multi-step reverse process starting from time t.

Recent DM-based methods typically perform well when the measurements are only slightly perturbed, for example, under Gaussian noise with a low noise level. However, the work [4] suggests that the second squared ℓ_2 fidelity term, $\|\boldsymbol{y} - \mathcal{A}(\bar{\boldsymbol{x}}_0)\|_2^2$, may be problematic when the measurement $\boldsymbol{y}$ is heavily contaminated, as the large discrepancy between the noisy measurement $\boldsymbol{y}$ and the noiseless measurement $\mathcal{A}(\boldsymbol{x}_0^*)$ can destroy the reconstruction process. Hence, we consider refining the measurement $\boldsymbol{y}$. We first mitigate the effect of noise. Instead of training an additional model to generate a pseudo condition as in [4], we explicitly estimate the additive noise $\boldsymbol{\nu}$ to refine the measurement. Assuming the additive noise follows $\mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}_m)$, we formulate the following optimization problem:

$$\min_{\bar{\boldsymbol{x}}_0, \bar{\boldsymbol{\nu}}} \frac{1}{2r_t^2}\|\bar{\boldsymbol{x}}_0 - \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)\|_2^2 + \frac{1}{2\sigma^2}\|\bar{\boldsymbol{\nu}}\|_2^2 + \frac{1}{2\gamma_t^2}\|\boldsymbol{y} - \mathcal{A}(\bar{\boldsymbol{x}}_0) - \bar{\boldsymbol{\nu}}\|_2^2, \tag{9}$$

where γ_t is a time-dependent hyperparameter that balances the guidance strength of the fidelity term $\|\boldsymbol{y} - \mathcal{A}(\bar{\boldsymbol{x}}_0) - \bar{\boldsymbol{\nu}}\|_2^2$. As suggested in DAPS [68], the sensitivity of the estimation $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ to its noisy latent input $\tilde{\boldsymbol{x}}_t$ is time-dependent. Since $\tilde{\boldsymbol{x}}_t$ is derived from the result of the optimization problem in the previous step, noise accumulated in the intermediate estimate $\tilde{\boldsymbol{x}}_0$ can affect the estimation $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ in subsequent steps. Specifically, $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ is robust to perturbations in $\tilde{\boldsymbol{x}}_t$ during early timesteps, allowing for stronger guidance from the measurement $\bar{\boldsymbol{y}}$ (i.e., a smaller γ_t) and tolerance of potential noise. Conversely, in later timesteps, $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$ becomes more sensitive, necessitating a larger hyperparameter γ_t to ensure that the estimated $\bar{\boldsymbol{x}}_0$ remains consistent with the estimation $\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)$, thereby preserving the naturalness of the restoration. We set the hyperparameter $\gamma_t = 1/\sigma_t$ empirically, where $1/\sigma_t$ satisfies the requirement of having a low value in early timesteps and a high value in later timesteps.

We then proceed to solve the optimization problem in Eq. (9) iteratively. Since we aim to refine the measurement, we first estimate the additive noise as:

$$\tilde{\boldsymbol{\nu}} = \arg\min_{\bar{\boldsymbol{\nu}}} \frac{1}{2\sigma^2}\|\bar{\boldsymbol{\nu}}\|_2^2 + \frac{1}{2\gamma_t^2}\|\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)) - \bar{\boldsymbol{\nu}}\|_2^2 \tag{10}$$

$$= \frac{\sigma^2}{\gamma_t^2 + \sigma^2}\left(\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t))\right). \tag{11}$$

Substituting Eq. (11) into Eq. (9) to refine the measurement $\boldsymbol{y}$, the optimization problem for $\bar{\boldsymbol{x}}_0$ becomes:

$$\min_{\bar{\boldsymbol{x}}_0} \frac{1}{2}\left(\frac{1}{r_t^2}\|\bar{\boldsymbol{x}}_0 - \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)\|_2^2 + \frac{1}{\gamma_t^2}\|\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0)\|_2^2\right), \tag{12}$$

where $\bar{\boldsymbol{y}} = \frac{1}{\gamma_t^2 + \sigma^2}(\gamma_t^2 \boldsymbol{y} + \sigma^2 \mathcal{A}(\hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)))$.

3.2. Robust objective for outliers

To further address IPs with outliers, we follow [59] to replace the squared ℓ_2 fidelity term, $\|\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0)\|_2^2$, with a sum of element-wise Huber loss functions. For the threshold $\delta > 0$, the univariate Huber loss operator $\mathcal{H}_\delta(\cdot)$ is defined as:

$$\mathcal{H}_\delta(r) = \begin{cases} r^2, & \text{if } |r| \le \delta, \\ 2\delta|r| - \delta^2, & \text{if } |r| > \delta. \end{cases} \tag{13}$$

The robust data fidelity term is applied to the residual vector $\boldsymbol{r} = \bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0)$ and is defined by summing the Huber loss over all entries:

$$\mathcal{H}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0)) = \sum_{i=1}^{m} \mathcal{H}_\delta((\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))_i). \tag{14}$$

We follow [30] and express the Huber loss term $\mathcal{H}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))$ in a quadratic form, $\tilde{\mathcal{H}}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))$. This new form shares the same gradient with respect to $\bar{\boldsymbol{x}}_0$ as the original term and enables the use of an iteratively reweighted least squares scheme:

$$\tilde{\mathcal{H}}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0)) = \|\mathcal{W}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))\|_2^2, \tag{15}$$

where $\mathcal{W}_\delta(\cdot)$ is a diagonal operator defined as:

$$\mathcal{W}_\delta(\boldsymbol{r})_{ii} = \begin{cases} 1, & \text{if } |r_i| \le \delta, \\ \sqrt{\delta/|r_i|}, & \text{if } |r_i| > \delta. \end{cases} \tag{16}$$

Note that although $\mathcal{W}_\delta(\boldsymbol{r})$ is computed using the current estimate $\bar{\boldsymbol{x}}_0$, it is detached from the gradient computation with respect to $\bar{\boldsymbol{x}}_0$ to ensure $\nabla_{\bar{\boldsymbol{x}}_0}\mathcal{H}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0)) = \nabla_{\bar{\boldsymbol{x}}_0}\tilde{\mathcal{H}}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))$. The modified optimization objective incorporates the Huber loss to achieve robustness against outliers and is given by:

$$\min_{\bar{\boldsymbol{x}}_0} \frac{1}{2}\left(\frac{1}{r_t^2}\|\bar{\boldsymbol{x}}_0 - \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)\|_2^2 + \frac{1}{\gamma_t^2}\|\boldsymbol{W}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))\|_2^2\right), \tag{17}$$

where we use $\boldsymbol{W}_\delta \in \mathbb{R}^{m \times m}$ to denote $\mathcal{W}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0))$ for brevity. We can approximately solve the optimization problem in Eq. (17) via gradient descent, initialized at $\tilde{\boldsymbol{x}}_0$ and using an empirically set learning rate η_x:[1]

$$\bar{\boldsymbol{x}}_0^{(j+1)} = \bar{\boldsymbol{x}}_0^{(j)} - \frac{\eta_x}{2}\nabla_{\bar{\boldsymbol{x}}_0^{(j)}}\Big(\frac{1}{r_t^2}\|\bar{\boldsymbol{x}}_0^{(j)} - \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)\|_2^2 + \frac{1}{\gamma_t^2}\|\boldsymbol{W}_\delta^{(j)}(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)}))\|_2^2\Big). \tag{18}$$

The corresponding algorithm is presented in Algorithm 1.

[1] Experiments with varying learning rate η_x for Robust-GD revealed that the performance of Robust-GD is sensitive to the selection of the learning rate. The detailed results are presented in the supplementary material.

Algorithm 1 Robust Diffusion Solver using Gradient Descent Method (Robust-GD)

Require: Data prediction network $\boldsymbol{x}_{\boldsymbol{\theta}}(\cdot,\cdot)$, measurement vector $\boldsymbol{y}$, forward operator $\mathcal{A}(\cdot)$, noise schedule $\{\sigma_{t_i}\}_{i=0}^{N}$, $\{\alpha_{t_i}\}_{i=0}^{N}$, time dependent parameters $\{r_{t_i}\}_{i=0}^{N}$, $\{\gamma_{t_i}\}_{i=0}^{N}$, Gaussian noise level σ, learning rate η_x, total number of sample steps and iterations N, J, the Huber loss threshold parameter δ

Sample $\tilde{\boldsymbol{x}}_{t_N} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$
for $i = N, \dots, 1$ **do**
  $\bar{\boldsymbol{x}}_0^{(0)} = \hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_{t_i}, t_i)$
  $\bar{\boldsymbol{y}} = \frac{1}{\gamma_{t_i}^2+\sigma^2}(\gamma_{t_i}^2 \boldsymbol{y} + \sigma^2 \mathcal{A}(\hat{\boldsymbol{x}}_0))$
  for $j = 0, \dots, J-1$ **do**
    $\boldsymbol{W}_\delta^{(j)} = \mathcal{W}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)}))$
    $\mathcal{L} = \frac{1}{2}\big(\frac{1}{r_{t_i}^2}\|\bar{\boldsymbol{x}}_0^{(j)} - \hat{\boldsymbol{x}}_0\|_2^2 + \frac{1}{\gamma_{t_i}^2}\|\boldsymbol{W}_\delta^{(j)}(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)}))\|_2^2\big)$
    $\bar{\boldsymbol{x}}_0^{(j+1)} = \bar{\boldsymbol{x}}_0^{(j)} - \eta_x \nabla_{\bar{\boldsymbol{x}}_0^{(j)}} \mathcal{L}$
  end for
  Sample $\tilde{\boldsymbol{x}}_{t_{i-1}} \sim \mathcal{N}(\alpha_{t_{i-1}} \bar{\boldsymbol{x}}_0^{(J)}, \sigma_{t_{i-1}}^2 \boldsymbol{I}_n)$
end for
return $\tilde{\boldsymbol{x}}_{t_0}$

3.3. Solve via conjugate gradient method

Using gradient descent to solve Eq. (17) typically requires delicate tuning of the learning rate. We instead employ the conjugate gradient (CG) method to mitigate the issue. At the j-th iteration, given the current gradient $\boldsymbol{g}_j$, the update direction $\boldsymbol{d}_j$, and the current result $\bar{\boldsymbol{x}}_0^{(j)}$, the next step is to calculate the update step size α_j. This step is formulated as the line search problem:

$$\alpha_j = \arg\min_{\alpha} \frac{1}{2}\big(\frac{1}{r_t^2}\|(\bar{\boldsymbol{x}}_0^{(j)} + \alpha \boldsymbol{d}_j) - \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_t, t)\|_2^2 + \frac{1}{\gamma_t^2}\|\boldsymbol{W}_\delta^{(j)}(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)} + \alpha \boldsymbol{d}_j))\|_2^2\big). \tag{19}$$

When the forward operator is linear, specifically, the forward operator is $\mathcal{A}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$, where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is the measurement matrix, the objective function is quadratic, which allows us to efficiently obtain the closed-form optimal update step size α_j as:

$$\alpha_j = \frac{\boldsymbol{g}_j^{\mathrm{T}} \boldsymbol{d}_j}{\frac{1}{r_t^2}\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{d}_j + \frac{1}{\gamma_t^2}(\boldsymbol{W}_\delta^{(j)}\boldsymbol{A}\boldsymbol{d}_j)^{\mathrm{T}}(\boldsymbol{W}_\delta^{(j)}\boldsymbol{A}\boldsymbol{d}_j)}. \tag{20}$$

Furthermore, we adapt the $\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{g}_j$ form for the numerator, which is commonly used in non-linear conjugate gradient methods and is known to be effective in line searches [46]. Empirically, this choice also shows superior performance compared to using the standard $\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{d}_j$ term,[2] yielding the closed-form expression for the optimal step size α_j:

$$\alpha_j = \frac{\boldsymbol{g}_j^{\mathrm{T}} \boldsymbol{g}_j}{\frac{1}{r_t^2}\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{d}_j + \frac{1}{\gamma_t^2}(\boldsymbol{W}_\delta^{(j)}\boldsymbol{A}\boldsymbol{d}_j)^{\mathrm{T}}(\boldsymbol{W}_\delta^{(j)}\boldsymbol{A}\boldsymbol{d}_j)}. \tag{21}$$

For the nonlinear IP, we cannot directly calculate the optimal step size. We first linearize the nonlinear operator $\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)} + \alpha \boldsymbol{d}_j)$ via a first-order Taylor expansion around the current estimate $\bar{\boldsymbol{x}}_0^{(j)}$:

$$\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)} + \alpha \boldsymbol{d}_j) \approx \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)}) + \alpha \boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})\boldsymbol{d}_j, \tag{22}$$

where $\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})$ is the Jacobian matrix of $\mathcal{A}(\cdot)$ evaluated at $\bar{\boldsymbol{x}}_0^{(j)}$. By substituting this linear approximation, the optimal step size is obtained in the same form as the linear case:

$$\alpha_j = \frac{\boldsymbol{g}_j^{\mathrm{T}} \boldsymbol{g}_j}{\frac{1}{r_t^2}\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{d}_j + \frac{1}{\gamma_t^2}(\boldsymbol{W}_\delta^{(j)}\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})\boldsymbol{d}_j)^{\mathrm{T}}(\boldsymbol{W}_\delta^{(j)}\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})\boldsymbol{d}_j)}. \tag{23}$$

However, calculating $\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})$ explicitly is also computationally expensive. To avoid explicitly computing the Jacobian matrix $\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})$, we utilize a finite difference approximation for the Jacobian-vector product $\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})\boldsymbol{d}_j$ [28, 42]:

$$\boldsymbol{J}(\bar{\boldsymbol{x}}_0^{(j)})\boldsymbol{d}_j \approx (\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)} + \eta \boldsymbol{d}_j) - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)}))/\eta, \tag{24}$$

where η is a small, experimentally determined hyperparameter. By substituting this approximation into the derivative of Eq. (19) and setting it to zero, we obtain the calculation for α_j under nonlinear scenarios:

$$\boldsymbol{\omega}_j = \left(\boldsymbol{W}_\delta^{(j)}\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)} + \eta \boldsymbol{d}_j) - \boldsymbol{W}_\delta^{(j)}\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)})\right)/\eta, \tag{25}$$

$$\alpha_j = \frac{\boldsymbol{g}_j^{\mathrm{T}} \boldsymbol{g}_j}{\frac{1}{r_t^2}\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{d}_j + \frac{1}{\gamma_t^2}\boldsymbol{\omega}_j^{\mathrm{T}}\boldsymbol{\omega}_j}. \tag{26}$$

To ensure the conjugacy property for the search direction, we employ the Fletcher-Reeves formula [18]. The complete Robust-CG algorithm is detailed in Algorithm 2.

4. Experimental results

We evaluate the two proposed Robust-GD and Robust-CG methods on three 256×256 validation datasets, namely CelebA [36], FFHQ [25] and ImageNet [12].[3] From each dataset, we randomly sample 100 validation images for image reconstruction, following the experimental setting adopted in recent DM-based methods for IPs [64, 68, 71]. To assess reconstruction quality and demonstrate that our methods balance the perception-distortion tradeoff [1], we report both distortion and perceptual metrics. Specifically,

[2] Empirical results demonstrate that Robust-CG utilizing the conjugate condition $\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{g}_j$ achieves superior performance compared to the version employing $\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{d}_j$. Detailed experimental results are presented in the supplementary material.

[3] Due to the page limit, the experimental results on FFHQ are presented in the supplementary material.

Algorithm 2 Robust Diffusion Solver using Conjugate Gradient Method (Robust-CG)

Require: Data prediction network $\boldsymbol{x}_{\boldsymbol{\theta}}(\cdot,\cdot)$, measurement vector $\boldsymbol{y}$, forward operator $\mathcal{A}(\cdot)$, noise schedule $\{\sigma_{t_i}\}_{i=0}^{N}$, $\{\alpha_{t_i}\}_{i=0}^{N}$, time dependent parameters $\{r_{t_i}\}_{i=0}^{N}$, $\{\gamma_{t_i}\}_{i=0}^{N}$, Gaussian noise level σ, finite-difference approximation parameter η, total number of sample steps and iterations N, J, the Huber loss threshold parameter δ

Sample $\tilde{\boldsymbol{x}}_{t_N} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$
for $i = N, \dots, 1$ **do**
  $\bar{\boldsymbol{x}}_0^{(0)} = \hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0(\tilde{\boldsymbol{x}}_{t_i}, t_i)$
  $\bar{\boldsymbol{y}} = \frac{1}{\gamma_{t_i}^2+\sigma^2}(\gamma_{t_i}^2 \boldsymbol{y} + \sigma^2 \mathcal{A}(\hat{\boldsymbol{x}}_0))$
  $\boldsymbol{W}_\delta^{(0)} = \mathcal{W}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(0)}))$
  $\mathcal{L} = \frac{1}{2}\big(\frac{1}{r_{t_i}^2}\|\bar{\boldsymbol{x}}_0^{(0)} - \hat{\boldsymbol{x}}_0\|_2^2 + \frac{1}{\gamma_{t_i}^2}\|\boldsymbol{W}_\delta^{(0)}(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(0)}))\|_2^2\big)$
  $\boldsymbol{d}_0 = \boldsymbol{g}_0 = -\nabla_{\bar{\boldsymbol{x}}_0^{(0)}}\mathcal{L}$
  for $j = 0, \dots, J-1$ **do**
    $\boldsymbol{\omega}_j = \big(\boldsymbol{W}_\delta^{(j)}\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)} + \eta \boldsymbol{d}_j) - \boldsymbol{W}_\delta^{(j)}\mathcal{A}(\bar{\boldsymbol{x}}_0^{(j)})\big)/\eta$
    $\alpha_j = (\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{g}_j)/\big(\frac{1}{r_{t_i}^2}\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{d}_j + \frac{1}{\gamma_{t_i}^2}\boldsymbol{\omega}_j^{\mathrm{T}}\boldsymbol{\omega}_j\big)$
    $\bar{\boldsymbol{x}}_0^{(j+1)} = \bar{\boldsymbol{x}}_0^{(j)} + \alpha_j \boldsymbol{d}_j$
    $\boldsymbol{W}_\delta^{(j+1)} = \mathcal{W}_\delta(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j+1)}))$
    $\mathcal{L} = \frac{1}{2}\big(\frac{1}{r_{t_i}^2}\|\bar{\boldsymbol{x}}_0^{(j+1)} - \hat{\boldsymbol{x}}_0\|_2^2 + \frac{1}{\gamma_{t_i}^2}\|\boldsymbol{W}_\delta^{(j+1)}(\bar{\boldsymbol{y}} - \mathcal{A}(\bar{\boldsymbol{x}}_0^{(j+1)}))\|_2^2\big)$
    $\boldsymbol{g}_{j+1} = -\nabla_{\bar{\boldsymbol{x}}_0^{(j+1)}}\mathcal{L}$
    $\boldsymbol{d}_{j+1} = \boldsymbol{g}_{j+1} + \frac{\boldsymbol{g}_{j+1}^{\mathrm{T}}\boldsymbol{g}_{j+1}}{\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{g}_j}\boldsymbol{d}_j$
  end for
  Sample $\tilde{\boldsymbol{x}}_{t_{i-1}} \sim \mathcal{N}(\alpha_{t_{i-1}}\bar{\boldsymbol{x}}_0^{(J)}, \sigma_{t_{i-1}}^2 \boldsymbol{I}_n)$
end for
return $\tilde{\boldsymbol{x}}_{t_0}$

we use the peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) to evaluate distortion, and learned perceptual image patch similarity (LPIPS) [69] to measure perceptual quality. We also report the Fréchet Inception Distance (FID) [21] for the main experiments to quantify perceptual quality. We use **bold** to indicate the best performance, underline for the second-best.

For linear IPs, we compare our methods with baseline DM-based methods, including DPS [9], DiffPIR [71], DCPS [24], RED-diff [41], and DAPS [68]. For nonlinear deblurring tasks, we compare our methods with DPS, RED-Diff, and DAPS. Experiments on all main tasks are conducted under Gaussian noise with the noise level $\sigma = 0.05$. To validate the robustness of our methods under different contamination levels, we set the contamination factor $\rho = 0.02$ or 0.10 following [23, 59]. For all tasks, we set the value of all entries of the outlier vector ξ_i to -1, which is consistent with the measurement boundary $\boldsymbol{y} \in [-1, 1]$. All experiments are run on a single NVIDIA GeForce RTX 4090 GPU.

4.1. Inpainting and super-resolution

For linear IPs, we first evaluate the performance of our methods on two tasks, namely the inpainting (random 70% masking) and super-resolution ($4\times$ downscaling) tasks. The settings for both tasks follow those established in previous works [9, 64, 68]. For the inpainting task, the threshold for Huber loss δ is set to 0.01 in Robust-GD and 0.02 in Robust-CG. For the super-resolution task, δ is set to 0.02 in Robust-GD and 0.005 in Robust-CG. The finite difference approximation parameter η in Robust-CG is set to 10^{-3} for both tasks.[4] The results presented in Table 1 demonstrate that Robust-CG performs the best across almost all metrics. Robust-GD also performs well compared to other recent DM-based methods and exhibits robustness against outliers. We visualize the experimental results for the inpainting and super-resolution tasks in Figures 2 and 3, respectively. The results demonstrate that our approaches yield better performance against outliers and reconstruct images that are visually more similar to the reference image.

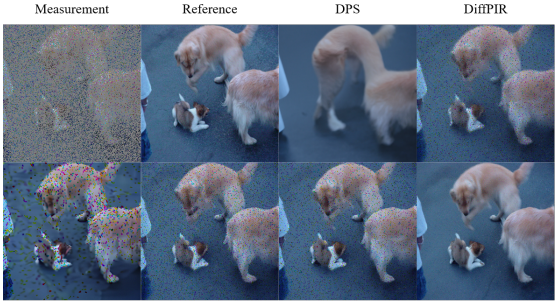

Figure 2. Visualization results of our methods and other DM-based approaches for the inpainting task, with Gaussian noise ($\sigma = 0.05$) and a contamination fraction of $\rho = 0.10$.

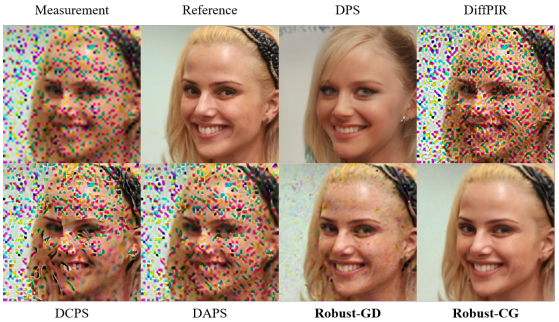

Figure 3. Visualization results of our methods and other DM-based approaches for the super-resolution task, with Gaussian noise ($\sigma = 0.05$) and a contamination fraction of $\rho = 0.10$.

4.2. Linear image deblurring

We also evaluate the performance of our methods on two deblurring tasks for linear IPs, namely Gaussian deblurring and motion deblurring. For Gaussian and motion de-

[4] More detailed setup for the two proposed algorithms is presented in the supplementary material.

		CelebA (256×256)				**ImageNet** (256×256)				**CelebA** (256×256)				**ImageNet** (256×256)			
		PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
ρ	Methods	**Super-resolution** ($4\times$)								**Inpainting (random** 70%**)**							
0.02	DPS	23.99 ± 1.40	0.655 ± 0.075	0.184 ± 0.057	66.96	20.86 ± 3.48	0.495 ± 0.189	0.447 ± 0.154	181.15	26.68 ± 1.32	0.748 ± 0.058	0.148 ± 0.048	63.49	23.49 ± 3.48	0.620 ± 0.171	0.361 ± 0.155	144.56
	DiffPIR	20.75 ± 1.68	0.561 ± 0.049	0.486 ± 0.073	239.75	18.66 ± 2.24	0.389 ± 0.087	0.591 ± 0.094	248.36	25.83 ± 1.40	0.695 ± 0.039	0.316 ± 0.088	150.54	23.49 ± 2.65	0.651 ± 0.064	0.325 ± 0.104	133.38
	DCPS	20.64 ± 2.09	0.591 ± 0.081	0.370 ± 0.103	193.03	18.00 ± 2.71	0.395 ± 0.144	0.613 ± 0.143	238.36	30.70 ± 1.37	**0.865** ± 0.031	0.103 ± 0.037	68.34	23.63 ± 4.11	0.727 ± 0.110	0.244 ± 0.102	117.91
	RED-diff	22.23 ± 1.76	0.679 ± 0.028	0.472 ± 0.067	192.54	20.81 ± 2.38	0.594 ± 0.077	0.544 ± 0.086	229.77	24.69 ± 1.15	0.656 ± 0.030	0.392 ± 0.086	152.93	22.49 ± 2.37	0.599 ± 0.047	0.423 ± 0.095	185.72
	DAPS	22.51 ± 1.98	0.685 ± 0.048	0.403 ± 0.087	236.29	20.97 ± 1.90	0.633 ± 0.062	0.519 ± 0.110	241.92	25.57 ± 1.98	0.691 ± 0.048	0.298 ± 0.087	195.00	23.20 ± 1.90	0.645 ± 0.062	0.277 ± 0.110	182.83
	Robust-GD	29.41 ± 1.46	0.803 ± 0.035	0.145 ± 0.043	72.29	24.02 ± 3.26	0.623 ± 0.088	0.484 ± 0.130	165.10	30.47 ± 1.18	0.818 ± 0.021	0.089 ± 0.027	62.62	24.50 ± 2.95	0.710 ± 0.071	0.211 ± 0.099	83.10
	Robust-CG	**29.67** ± 1.62	**0.831** ± 0.040	**0.125** ± 0.038	**64.04**	**24.90** ± 4.02	**0.697** ± 0.129	**0.375** ± 0.133	**146.05**	**31.20** ± 1.52	0.855 ± 0.023	**0.071** ± 0.021	**51.42**	**26.55** ± 3.96	**0.785** ± 0.076	**0.120** ± 0.040	**55.24**
0.10	DPS	20.95 ± 1.40	0.607 ± 0.079	0.217 ± 0.061	70.76	18.78 ± 2.85	0.450 ± 0.188	0.496 ± 0.146	192.57	22.29 ± 1.28	0.679 ± 0.068	0.185 ± 0.055	68.02	20.19 ± 2.59	0.538 ± 0.179	0.459 ± 0.165	177.36
	DiffPIR	14.76 ± 1.71	0.290 ± 0.085	0.671 ± 0.081	328.01	13.40 ± 2.56	0.203 ± 0.092	0.769 ± 0.104	302.80	19.72 ± 1.80	0.438 ± 0.096	0.599 ± 0.142	191.35	18.90 ± 2.39	0.433 ± 0.111	0.561 ± 0.168	193.35
	DCPS	15.74 ± 1.51	0.371 ± 0.103	0.578 ± 0.093	317.65	14.87 ± 2.38	0.324 ± 0.101	0.714 ± 0.092	284.75	24.26 ± 1.58	0.807 ± 0.039	0.136 ± 0.042	78.22	16.59 ± 3.13	0.411 ± 0.156	0.644 ± 0.206	212.46
	RED-diff	16.10 ± 1.92	0.449 ± 0.068	0.639 ± 0.083	264.86	18.96 ± 2.31	0.412 ± 0.086	0.610 ± 0.147	271.98	19.86 ± 1.67	0.439 ± 0.081	0.623 ± 0.134	182.13	15.99 ± 2.54	0.425 ± 0.076	0.672 ± 0.088	243.29
	DAPS	16.63 ± 2.45	0.428 ± 0.103	0.582 ± 0.101	333.06	15.84 ± 2.17	0.433 ± 0.079	0.660 ± 0.114	274.08	19.15 ± 2.45	0.437 ± 0.103	0.596 ± 0.101	261.25	18.02 ± 2.17	0.408 ± 0.079	0.567 ± 0.114	104.99
	Robust-GD	26.52 ± 1.27	0.716 ± 0.042	0.264 ± 0.068	110.21	20.48 ± 2.17	0.488 ± 0.079	0.649 ± 0.114	233.46	23.28 ± 0.87	0.575 ± 0.057	0.374 ± 0.086	169.96	19.02 ± 2.49	0.453 ± 0.120	0.491 ± 0.185	171.40
	Robust-CG	**28.96** ± 1.56	**0.819** ± 0.041	**0.129** ± 0.039	**67.41**	**24.28** ± 3.80	**0.671** ± 0.128	**0.416** ± 0.136	**159.32**	**29.74** ± 1.29	**0.809** ± 0.022	**0.093** ± 0.029	**58.48**	**25.57** ± 3.60	**0.720** ± 0.076	**0.143** ± 0.046	**62.32**

Table 1. (**Linear IPs**) **Super-resolution** ($4\times$), **inpainting (random** 70%) with additive Gaussian noise ($\sigma = 0.05$) and contamination fraction $\rho = 0.02$ or 0.10.

blurring, we used kernels of size 61×61 with standard deviations of 3.0 and 0.5, respectively. All task settings follow those established in previous works [9, 68]. For both Robust-GD and Robust-CG, the threshold δ is set to 0.02 and the finite difference approximation parameter η in Robust-CG is set to 10^{-4}. The results are presented in Table 2, demonstrating that our methods achieve the best performance across almost all metrics. We also visualize the experimental results for the Gaussian deblurring and motion deblurring tasks in Figures 4 and 5, respectively. Compared to other methods, the results demonstrate that our approaches yield better performance against outliers and produce images that retain more details and are visually more similar to the reference images.

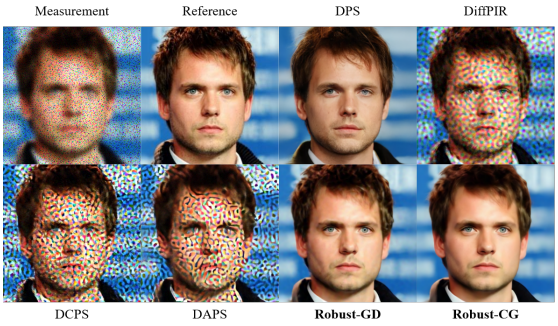

Figure 4. Visualization results of our methods and other DM-based approaches for the Gaussian deblurring task, with Gaussian noise ($\sigma = 0.05$) and a contamination fraction of $\rho = 0.10$.

4.3. Nonlinear deblurring

For nonlinear deblurring, we utilize the learned blurring operators from [62] with a known Gaussian-shaped kernel, following the setting in [9, 64]. The corresponding threshold parameter δ in Robust-GD and Robust-CG is set to 0.01. The finite difference approximation parameter η in Robust-CG is set to 0.0001. The results summarized in Table 3 demonstrate that our Robust-GD and Robust-CG methods achieve the best performance across all tasks, highlighting the effectiveness of our methods.

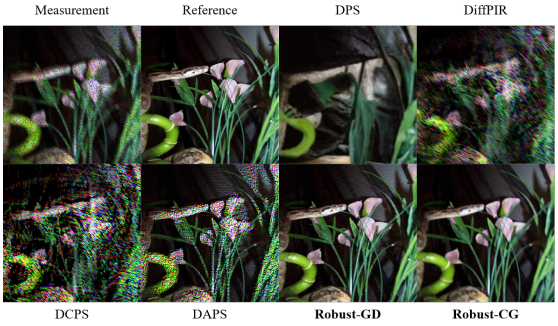

Figure 5. Visualization results of our methods and other DM-based approaches for the motion deblurring task, with Gaussian noise ($\sigma = 0.05$) and a contamination fraction of $\rho = 0.10$.

4.4. High Gaussian noisy condition

We follow [4, 27, 54] and validate the robustness of our proposed Robust-GD and Robust-CG methods under high Gaussian noisy conditions. We conduct experiments on the CelebA dataset, using 100 randomly selected validation images for super-resolution and inpainting tasks. We set the noise level at $\sigma = 0.5$ as in [54], and the contamination fraction at $\rho = 0.02$. The settings for these two tasks are identical to those detailed in Section 4.1. The results presented in Table 4 demonstrate that our methods achieve the best performance compared to all other DM-based approaches, confirming the robustness of our methods against high Gaussian noise.

4.5. Ablation study

We validate the effect of the Huber loss threshold parameter δ in our algorithms and the finite-difference approximation parameter η in the Robust-CG method. Additionally, we present the performance of Robust-GD with varying learning rates η_x and compare the use of $\boldsymbol{g}^{\mathrm{T}}\boldsymbol{g}$ versus $\boldsymbol{g}^{\mathrm{T}}\boldsymbol{d}$ terms for calculating α_j in Robust-CG (referencing Eq. (21)) in the supplementary material.

4.5.1. Huber loss threshold parameter

To evaluate the effect of the Huber loss threshold parameter δ, we test the Robust-CG method on 100 validation

		CelebA (256×256)				ImageNet (256×256)				CelebA (256×256)				ImageNet (256×256)			
		PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
ρ	Methods	Gaussian deblurring								Motion deblurring							
0.02	DPS	25.14 ± 1.48	0.680 ± 0.074	0.152 ± 0.044	**62.67**	21.82 ± 3.73	0.532 ± 0.184	**0.358** ± 0.137	**139.39**	23.44 ± 1.43	0.633 ± 0.078	0.182 ± 0.053	68.37	20.34 ± 3.31	0.480 ± 0.181	0.425 ± 0.142	180.41
	DiffPIR	22.45 ± 1.00	0.482 ± 0.074	0.493 ± 0.084	211.40	19.85 ± 1.90	0.367 ± 0.101	0.676 ± 0.109	250.67	20.48 ± 0.95	0.362 ± 0.070	0.608 ± 0.082	274.28	17.13 ± 1.58	0.250 ± 0.091	0.703 ± 0.139	286.64
	DCPS	22.45 ± 1.83	0.563 ± 0.097	0.307 ± 0.087	172.98	18.49 ± 2.62	0.356 ± 0.144	0.593 ± 0.146	224.90	21.65 ± 3.99	0.585 ± 0.184	0.351 ± 0.240	148.53	14.22 ± 3.16	0.245 ± 0.148	0.765 ± 0.238	265.70
	RED-diff	27.99 ± 1.56	0.751 ± 0.051	0.273 ± 0.067	85.72	23.80 ± 3.68	0.603 ± 0.136	0.558 ± 0.144	169.24	24.80 ± 1.42	0.573 ± 0.066	0.381 ± 0.086	150.30	21.66 ± 3.03	0.459 ± 0.095	0.565 ± 0.095	269.02
	DAPS	23.73 ± 2.11	0.586 ± 0.098	0.362 ± 0.092	202.42	20.35 ± 2.11	0.450 ± 0.098	0.619 ± 0.092	251.12	23.74 ± 3.15	0.621 ± 0.121	0.330 ± 0.173	**63.44**	19.27 ± 3.15	0.447 ± 0.121	0.544 ± 0.173	232.58
	Robust-GD	**30.00** ± 1.75	**0.825** ± 0.044	**0.132** ± 0.040	63.44	**25.96** ± 3.90	**0.679** ± 0.120	0.424 ± 0.131	159.84	**29.87** ± 1.86	**0.801** ± 0.049	**0.116** ± 0.045	81.80	**25.80** ± 3.79	**0.696** ± 0.106	**0.300** ± 0.139	**115.26**
	Robust-CG	29.48 ± 1.81	0.820 ± 0.051	0.150 ± 0.047	63.50	24.79 ± 4.34	0.673 ± 0.155	0.458 ± 0.162	169.11	29.27 ± 2.36	0.793 ± 0.065	0.136 ± 0.057	87.09	24.94 ± 4.13	0.665 ± 0.124	0.376 ± 0.154	152.85
0.10	DPS	22.06 ± 1.42	0.646 ± 0.077	0.178 ± 0.051	63.49	19.70 ± 2.89	0.490 ± 0.185	**0.405** ± 0.131	**160.82**	20.90 ± 1.37	0.596 ± 0.082	0.208 ± 0.058	**68.96**	18.60 ± 2.80	0.437 ± 0.186	0.455 ± 0.130	192.53
	DiffPIR	18.07 ± 1.02	0.355 ± 0.083	0.622 ± 0.083	308.93	16.60 ± 1.67	0.274 ± 0.098	0.740 ± 0.088	292.01	17.11 ± 0.93	0.259 ± 0.071	0.712 ± 0.078	340.27	14.59 ± 1.52	0.180 ± 0.081	0.781 ± 0.117	317.89
	DCPS	15.46 ± 1.65	0.302 ± 0.124	0.575 ± 0.095	331.91	12.89 ± 2.54	0.193 ± 0.123	0.702 ± 0.092	305.81	12.60 ± 3.45	0.174 ± 0.176	0.927 ± 0.292	338.22	10.29 ± 1.92	0.112 ± 0.084	0.953 ± 0.183	332.85
	RED-diff	22.96 ± 1.70	0.626 ± 0.076	0.391 ± 0.090	139.05	20.76 ± 2.68	0.497 ± 0.118	0.645 ± 0.129	219.67	17.36 ± 2.52	0.311 ± 0.109	0.836 ± 0.185	303.61	16.71 ± 3.08	0.270 ± 0.116	0.807 ± 0.171	347.46
	DAPS	16.18 ± 2.26	0.356 ± 0.122	0.561 ± 0.104	297.13	14.17 ± 2.26	0.253 ± 0.122	0.720 ± 0.104	308.08	14.99 ± 3.01	0.331 ± 0.130	0.718 ± 0.178	292.81	12.45 ± 3.01	0.226 ± 0.130	0.823 ± 0.178	305.09
	Robust-GD	29.27 ± 1.68	0.817 ± 0.044	**0.138** ± 0.041	65.59	**24.90** ± 3.71	0.660 ± 0.114	0.453 ± 0.130	171.81	**29.44** ± 1.76	**0.789** ± 0.048	**0.123** ± 0.046	84.79	**25.27** ± 3.55	**0.669** ± 0.103	**0.330** ± 0.134	**128.60**
	Robust-CG	**29.38** ± 1.78	**0.819** ± 0.051	0.151 ± 0.047	**62.52**	24.73 ± 4.29	**0.672** ± 0.155	0.450 ± 0.163	167.56	28.91 ± 2.16	0.777 ± 0.062	0.144 ± 0.057	91.22	24.57 ± 3.80	0.638 ± 0.119	0.401 ± 0.151	165.45

Table 2. (**Linear IPs**) **Gaussian deblurring** and **motion deblurring** with additive Gaussian noise ($\sigma = 0.05$) and contamination fraction $\rho = 0.02$ or 0.10.

		CelebA (256×256)				ImageNet (256×256)			
ρ	Methods	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
0.02	DPS	23.42 ± 1.54	0.632 ± 0.076	0.203 ± 0.055	69.71	21.01 ± 3.53	0.493 ± 0.174	0.439 ± 0.148	191.88
	RED-diff	22.49 ± 1.61	0.490 ± 0.064	0.568 ± 0.116	169.33	18.53 ± 2.31	0.289 ± 0.067	0.727 ± 0.111	305.67
	DAPS	20.90 ± 1.81	0.428 ± 0.113	0.658 ± 0.177	237.47	20.22 ± 1.81	0.457 ± 0.113	0.530 ± 0.177	202.20
	Robust-GD	**28.25** ± 1.55	**0.766** ± 0.037	**0.136** ± 0.040	**68.55**	**26.40** ± 3.61	**0.741** ± 0.084	**0.214** ± 0.100	**83.52**
	Robust-CG	26.89 ± 1.59	0.733 ± 0.055	0.192 ± 0.060	85.30	23.66 ± 3.80	0.634 ± 0.137	0.411 ± 0.147	201.28
0.10	DPS	21.26 ± 1.45	0.609 ± 0.077	0.211 ± 0.052	**70.11**	19.19 ± 2.80	0.455 ± 0.179	0.463 ± 0.147	183.46
	RED-diff	16.37 ± 1.96	0.274 ± 0.088	0.922 ± 0.167	288.09	14.95 ± 2.39	0.166 ± 0.087	0.998 ± 0.185	336.14
	DAPS	15.89 ± 2.17	0.232 ± 0.110	1.036 ± 0.174	439.06	15.21 ± 2.17	0.228 ± 0.110	0.947 ± 0.174	316.56
	Robust-GD	**27.06** ± 1.29	0.711 ± 0.037	**0.156** ± 0.044	80.23	**24.55** ± 3.07	**0.636** ± 0.084	**0.253** ± 0.093	**108.10**
	Robust-CG	26.80 ± 1.54	**0.728** ± 0.058	0.188 ± 0.061	89.73	23.67 ± 3.75	0.632 ± 0.135	0.404 ± 0.143	191.30

Table 3. **Nonlinear deblurring** with additive Gaussian noise ($\sigma = 0.05$) and contamination fraction $\rho = 0.02$ or 0.10.

	Super-resolution ($4\times$)			Inpainting (random 70%)		
Methods	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
DPS	20.55	0.546	0.246	21.90	0.604	0.212
DiffPIR	15.68	0.192	0.643	20.31	0.471	0.382
DCPS	18.77	0.476	0.315	22.54	0.620	0.202
RED-diff	12.88	0.109	0.793	17.47	0.186	0.873
DAPS	14.29	0.143	0.784	16.99	0.181	0.901
Robust-GD	22.16	0.590	0.279	23.91	**0.690**	**0.179**
Robust-CG	**23.04**	**0.632**	**0.245**	**24.62**	0.689	0.200

Table 4. **Super-resolution** ($4\times$) and **inpainting (random** 70%**)** with additive Gaussian noise ($\sigma = 0.5$) and contamination fraction $\rho = 0.02$.

images from the CelebA dataset for Gaussian and motion deblurring tasks, using a contamination factor of $\rho = 0.10$ and a Gaussian noise level of $\sigma = 0.05$. The corresponding experimental results for Robust-GD are presented in the supplementary material. We choose the value of δ within $\{0.005, 0.01, 0.02, 0.04\}$. The results presented in Table 5 indicate that Robust-CG performs similarly across these different threshold values, demonstrating the robustness of our method to the selection of the threshold parameter.

4.5.2. Finite-difference approximation parameter

We validate the effect of the finite-difference approximation parameter η in the Robust-CG method on two tasks, namely Gaussian deblurring and the nonlinear deblurring task. We use 100 validation images from the CelebA dataset with a contamination factor of $\rho = 0.10$ and a Gaussian noise level of $\sigma = 0.05$. We choose three values of η within $\{0.0010, 0.0005, 0.0001\}$. The results presented in Table 6 indicate that the choice of η does not affect the reconstruction results for Gaussian deblurring and has a slight effect on the results for the nonlinear deblurring task.

	Gaussian Deblurring			Motion Deblurring		
Threshold δ	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
0.005	28.75	0.804	0.164	27.91	0.786	0.161
0.010	29.04	0.811	0.157	28.55	0.793	0.146
0.020	29.38	0.819	0.151	28.91	0.777	0.144
0.040	29.64	0.823	0.145	27.67	0.681	0.220

Table 5. Performance of Robust-CG with different value of the Huber loss threshold δ on **Gaussian deblurring** and **motion deblurring** with additive Gaussian noise ($\sigma = 0.05$) and contamination fraction 0.10.

	Gaussian Deblurring			Nonliear Deblurring		
Parameter η	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
0.0010	29.38	0.819	0.151	25.97	0.681	0.212
0.0005	29.38	0.819	0.151	26.37	0.703	0.195
0.0001	29.38	0.819	0.151	26.80	0.728	0.188

Table 6. Performance of Robust-CG with different value of finite-difference approximation parameter η on **Gaussian deblurring** and **nonlinear deblurring** with additive Gaussian noise ($\sigma = 0.05$) and contamination fraction 0.10.

5. Conclusion

In this paper, we present Robust-GD and Robust-CG, two novel DM-based methods to address IPs contaminated with outliers. Numerical results demonstrate the effectiveness of our proposed methods in mitigating the influence of outliers, thereby confirming their superior robustness compared to existing DM-based methods.

References

[1] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In CVPR, 2018. 5

[2] Emmanuel J Candès and Terence Tao. Decoding by linear programming. IEEE Transactions on Information Theory, 2005. 1

[3] Liang Chen, Faming Fang, Jiawei Zhang, Jun Liu, and Guixu Zhang. OID: Outlier identifying and discarding in blind image deblurring. In ECCV, 2020. 1

[4] Xin Chen, Gillian Dobbie, Xinyu Wang, Feng Liu, Di Wang, and Jingfeng Zhang. Robust learning of diffusion models with extremely noisy conditions. arXiv preprint arXiv:2510.10149, 2025. 4, 7

[5] Sunghyun Cho, Jue Wang, and Seungyong Lee. Handling outliers in non-blind image deconvolution. In ICCV, 2011. 2

[6] Hyungjin Chung, Eun Sun Lee, and Jong Chul Ye. MR image denoising and super-resolution using regularized reverse diffusion. IEEE Transactions on Medical Imaging, 2022. 1

[7] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In NeurIPS, 2022. 2

[8] Hyungjin Chung, Jeongsol Kim, Sehui Kim, and Jong Chul Ye. Parallel diffusion models of operator and image for blind inverse problems. In CVPR, 2023. 1

[9] Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In ICLR, 2023. 2, 6, 7

[10] Ciprian Corneanu, Raghudeep Gadde, and Aleix M Martinez. Latentpaint: Image inpainting in latent space with diffusion models. In CVPR, 2024. 1

[11] Arnak Dalalyan and Philip Thompson. Outlier-robust estimation of a sparse linear model using ℓ_1-penalized Huber's M-estimator. In NeurIPS, 2019. 1, 2

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In CVPR, 2009. 5

[13] Jiangxin Dong and Jinshan Pan. Deep outlier handling for image deblurring. IEEE Transactions on Image Processing, 2021. 1

[14] Jiangxin Dong, Jinshan Pan, Zhixun Su, and Ming-Hsuan Yang. Blind image deblurring with outlier handling. In ICCV, 2017. 1

[15] Hongkun Dou, Zeyu Li, Jinyang Du, Lijun Yang, Wen Yao, and Yue Deng. Hybird regularization improves diffusion-based inverse problem solving. In ICLR, 2025. 2

[16] Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In ICLR, 2024. 1

[17] Tommaso d'Orsi, Gleb Novikov, and David Steurer. Consistent regression when oblivious outliers overwhelm. In ICML, 2021. 2

[18] Reeves Fletcher and Colin M Reeves. Function minimization by conjugate gradients. The Computer Journal, 1964. 5

[19] Simon Foucart and Holger Rauhut. A mathematical introduction to compressive sensing. Springer New York, 2013. 1

[20] Sicheng Gao, Xuhui Liu, Bohan Zeng, Sheng Xu, Yanjing Li, Xiaoyan Luo, Jianzhuang Liu, Xiantong Zhen, and Baochang Zhang. Implicit diffusion models for continuous super-resolution. In CVPR, 2023. 1

[21] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In NeurIPS, 2017. 6

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In NeurIPS, 2020. 2

[23] Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alexandros G Dimakis, and Jon Tamir. Robust compressed sensing MRI with deep generative priors. In NeurIPS, 2021. 1, 2, 6

[24] Yazid Janati, Badr Moufad, Alain Durmus, Eric Moulines, and Jimmy Olsson. Divide-and-conquer posterior sampling for denoising diffusion priors. In NeurIPS, 2024. 2, 6

[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In CVPR, 2019. 5

[26] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In NeurIPS, 2022. 2

[27] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In NeurIPS, 2022. 1, 7

[28] Carl T Kelley. Iterative methods for linear and nonlinear equations. SIAM, 1995. 5

[29] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In NeurIPS, 2021. 3

[30] Christian Kümmerle, Claudio Mayrink Verdun, and Dominik Stöger. Iteratively reweighted least squares for basis pursuit with global linear convergence rate. In NeurIPS, 2021. 4

[31] Pierre Laforgue, Guillaume Staerman, and Stephan Clémençon. Generalization bounds in the presence of outliers: A median-of-means study. In ICML, 2021. 2

[32] Ji Li and Chao Wang. Integrating reweighted least squares with plug-and-play diffusion priors for noisy image restoration. In AAAI, 2026. 1

[33] Xiang Li, Soo Min Kwon, Shijun Liang, Ismail R Alkhouri, Saiprasad Ravishankar, and Qing Qu. Decoupled data consistency with diffusion purification for image restoration. arXiv preprint arXiv:2403.06054, 2024. 2

[34] Liu Liu, Tianyang Li, and Constantine Caramanis. High dimensional robust M-estimation: Arbitrary corruption and heavy tails. arXiv preprint arXiv:1901.08237, 2019. 2

[35] Liu Liu, Yanyao Shen, Tianyang Li, and Constantine Caramanis. High dimensional robust sparse regression. In ICAIS, 2020. 2

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In ICCV, 2015. 5

[37] Po-Ling Loh and Martin J Wainwright. Regularized M-estimators with nonconvexity: Statistical and algorithmic theory for local optima. The Journal of Machine Learning Research, 2015. 2

[38] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In NeurIPS, 2022. 2

[39] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. arXiv preprint arXiv:2211.01095, 2022. 3

[40] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In CVPR, 2022. 1

[41] Morteza Mardani, Jiaming Song, Jan Kautz, and Arash Vahdat. A variational perspective on solving inverse problems with diffusion models. In ICLR, 2023. 2, 6

[42] James Martens et al. Deep learning via hessian-free optimization. In ICML, 2010. 5

[43] Eloi Moliner, Jaakko Lehtinen, and Vesa Välimäki. Solving audio inverse problems with a diffusion model. In ICASSP, 2023. 1

[44] Badr Moufad, Yazid Janati, Lisa Bedin, Alain Durmus, Randal Douc, Eric Moulines, and Jimmy Olsson. Variational diffusion posterior sampling with midpoint guidance. In ICLR, 2025. 2

[45] Naoki Murata, Koichi Saito, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Yuki Mitsufuji, and Stefano Ermon. GibbsDDRM: A partially collapsed Gibbs sampler for solving blind inverse problems with denoising diffusion restoration. In ICML, 2023. 1

[46] Jorge Nocedal and Stephen J Wright. Numerical optimization. Springer, 2006. 5

[47] Jinshan Pan, Zhouchen Lin, Zhixun Su, and Ming-Hsuan Yang. Robust kernel estimation with outliers handling for image deblurring. In CVPR, 2016. 1, 2

[48] Mengwei Ren, Mauricio Delbracio, Hossein Talebi, Guido Gerig, and Peyman Milanfar. Multiscale structure guided diffusion for image deblurring. In CVPR, 2023. 1

[49] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In SIGGRAPH, 2022. 1

[50] Koichi Saito, Naoki Murata, Toshimitsu Uesaka, Chieh-Hsin Lai, Yuhta Takida, Takao Fukui, and Yuki Mitsufuji. Unsupervised vocal dereverberation with diffusion-based generative models. In ICASSP, 2023. 1

[51] Yash Sanghvi, Yiheng Chi, and Stanley H Chan. Kernel diffusion: An alternate approach to blind deconvolution. In ECCV, 2025. 1

[52] Shuyao Shang, Zhengyang Shan, Guangxing Liu, LunQian Wang, XingHua Wang, Zekai Zhang, and Jinglin Zhang. ResDiff: Combining CNN and diffusion model for image super-resolution. In AAAI, 2024. 1

[53] Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In ICML, 2019. 2

[54] Shirin Shoushtari, Edward P Chandler, Yuanhao Wang, M Salman Asif, and Ulugbek S Kamilov. Unsupervised detection of distribution shift in inverse problems using diffusion models. In arXiv preprint arXiv:2505.11482, 2025. 7

[55] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In ICML, 2015. 2

[56] Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency. In ICLR, 2024. 1

[57] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In ICLR, 2021. 2, 3

[58] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In ICLR, 2023. 1, 2

[59] Lingfei Song and Hua Huang. Robust image restoration with an adaptive Huber function based fidelity. International Journal of Computer Vision, 2024. 1, 2, 4, 6

[60] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In ICLR, 2021. 2, 3

[61] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. In ICLR, 2022. 1

[62] Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. In CVPR, 2021. 7

[63] Sean Twomey. Introduction to the mathematics of inversion in remote sensing and indirect measurements. Courier Dover Publications, 2019. 1

[64] Hengkang Wang, Xu Zhang, Taihui Li, Yuxiang Wan, Tiancong Chen, and Ju Sun. DMPlug: A plug-in method for solving inverse problems with diffusion models. In NeurIPS, 2024. 1, 2, 5, 6, 7

[65] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In ICLR, 2023. 1

[66] Ziyu Wang, Bin Dai, David Wipf, and Jun Zhu. Further analysis of outlier detection with deep generative models. In NeurIPS, 2020. 2

[67] Yan Wu, Mihaela Rosca, and Timothy Lillicrap. Deep compressed sensing. In ICML, 2019. 1

[68] Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In CVPR, 2025. 2, 3, 4, 5, 6, 7

[69] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In CVPR, 2018. 6

[70] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In NeurIPS, 2023. 2

[71] Yuanzhi Zhu, Kai Zhang, Jingyun Liang, Jiezhang Cao, Bihan Wen, Radu Timofte, and Luc Van Gool. Denoising diffusion models for plug-and-play image restoration. In CVPR, 2023. 2, 3, 5, 6