Test-Time Adaptation Improves Inverse Problem Solving With Patch-Based Diffusion Models

Jason Hu, *Graduate Student Member, IEEE*, Bowen Song, *Student Member, IEEE*, Jeffrey A. Fessler, *Fellow, IEEE*, and Liyue Shen, *Member, IEEE*

***Abstract*—Diffusion models have achieved excellent success in solving inverse problems due to their ability to learn strong image priors, but existing approaches require a large training dataset of images that should come from the same distribution as the test dataset. In practice, the size of the available training dataset can range from nonexistent to very large. In some cases, conventional diffusion model training from limited data can lead to poor reconstruction results due to poorly learned priors. One potential improvement is to start with a diffusion model trained from available training data having a possibly mismatched distribution, and then refine the network at reconstruction time to account for the distribution mismatch. In this work, we investigate the effect of this network refining process on diffusion models trained from varying degrees of out-of-distribution data. Specifically, we use a self-supervised loss to adapt the learned diffusion network to the testing data while helping the network output maintain consistency with the measurements. We show that, both theoretically and experimentally, test-time adaptation of a patch-based diffusion prior leads to higher quality reconstructions than test-time refinement of traditional whole-image diffusion models. Extensive experiments show that across a wide range of inverse problems, test-time adaptation significantly improves image reconstruction quality when there are significant domain shifts between training and testing distributions. Interestingly, even for the in-distribution case, test-time adaptation also significantly improves reconstruction quality.**

***Index Terms*—Deep learning, diffusion models, image processing, inverse problems.**

I. Introduction

In IMAGING, inverse problems are important and consist of reconstructing an image $\boldsymbol{x}$ from a measurement $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{\epsilon}$. Here, $\mathcal{A}$ represents a forward operator and $\boldsymbol{\epsilon}$ represents random unknown noise. By Bayes' rule, $\log p(\boldsymbol{x}|\boldsymbol{y})$ is proportional to $\log p(\boldsymbol{x}) + \log p(\boldsymbol{y}|\boldsymbol{x})$, so obtaining a good prior $p(\boldsymbol{x})$ is crucial for recovering $\boldsymbol{x}$ when $\boldsymbol{y}$ contains far less information than $\boldsymbol{x}$. Diffusion models obtain state-of-the-art results for learning a strong prior and sampli ng from it, so competitive results can be obtained when using them to solve inverse problems [1], [2], [3], [4], [5], [6].

However, these diffusion inverse solvers (DIS) require well-trained diffusion models, which in turn require large amounts of clean training data [7], [8]. It may be infeasible to collect large training data sets in many applications such as medical imaging [1], [9], [10], black hole imaging [11], [12], and phase retrieval [6], [13]. For example, in practical applications such as dynamic CT reconstruction [14] and single photon emission CT [15], obtaining high quality measurements, which can lead to reconstructions closely approximating the ground truth, can be slow or potentially harmful to patients, so only very small datasets of clean images are available. Moreover, for very challenging inverse problems such as black hole imaging [11] and Fresnel phase retrieval [16], no ground truth images are known, so one must obtain a reconstruction from only a single measurement $\boldsymbol{y}$. In these practical applications with extremely limited or even nonexistent available data (measurement-only), it can be difficult or even impossible to train a diffusion model to approximate the underlying distribution well. Therefore, it is desirable to develop a method that can be applied regardless of dataset limitations. In this paper, we investigate applying a self-supervised loss at reconstruction time that allows the network to be adapted to the test data. Specifically, we investigate three different settings with various data availability ranging from nonexistent, limited, and abundant training data: (1) the *dataless* setting in which no training data is available and we are only given measurement $\boldsymbol{y}$, (2) the *small dataset* setting in which we are only given a small number of samples $\boldsymbol{x}$ that belong to the same distribution as the test dataset, (3) and the *in-distribution* setting in which we have sufficient training data of the same distribution as the test dataset. Our goal is to develop methods that can be used to adapt the trained model to testing data in these settings with varying training data availability. Recently, some previous works have aimed to address these problems by demonstrating that diffusion models have a stronger generalization ability than other deep learning methods [10], and that slight distribution mismatches between the training data and test data may not significantly degrade the reconstructed image quality. However, in cases of particularly sparse or noisy measurements, as well as when the test data is severely out of distribution (OOD) with a significant domain shift, an improper choice of training data leads to an incorrect prior that causes substantial image degradation and hallucinations [11], [17]. To address these challenges in the dataless setting, recent works first train a network

Received 21 January 2025; revised 10 May 2025; accepted 30 June 2025. Date of publication 10 July 2025; date of current version 21 July 2025. This work was supported in part by the Michigan Institute for Computational Discovery and Engineering (MICDE) and in part by KLA. The associate editor coordinating the review of this article and approving it for publication was Dr. Mauricio Delbracio. *(Corresponding author: Jason Hu.)*

The authors are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122 USA (e-mail: jashu@umich.edu; bowenbw@umich.edu; fessler@umich.edu; liyues@umich.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TCI.2025.3587407, provided by the authors.

Digital Object Identifier 10.1109/TCI.2025.3587407

on a large sample of synthetic data that may differ greatly from the test data. Then, at reconstruction time, the trained diffusion network is updated based on the measurement $\boldsymbol{y}$ [17], [18]. This adaptation aims to shift the underlying prior learned by the network towards the appropriate prior corresponding to the test data distribution. However, with the large number of network parameters, a parameter-sensitive network adaptation approach is required at test time to avoid overfitting to the measurement. Furthermore, these methods have not been tested in the small dataset setting, and in the in-distribution setting, excessive network refining yields inconsistent improvement and degradation in the reconstructed image [17], [18]. Patch-based diffusion models have shown success both for image generation [19], [20] and for inverse problem solving [21]. In particular, the method of [21] involves training networks that take in only patches of images at training and reconstruction time, learning priors of the entire images from only image patches based on positional encoding. In cases of limited training data, [21] shows that patch-based diffusion models outperform whole image models for solving certain inverse problems. These works motivate our key insight that patch-based diffusion priors potentially obtain stronger generalizability than whole-image diffusion priors for both the dataless setting and the small dataset setting, given a severe lack of data. Inspired by this, we propose to use patch-based diffusion models to tackle the challenges arising from data limitations and mismatched distributions in a unified way. We first develop a test-time adaptation method to take a network trained on patches from a mismatched distribution and adapt it on the fly at reconstruction time. We then show that this method can improve image quality consistently *in all three settings* with varying data availability ranging from abundant, to limited and nonexistent.

In summary, our contributions are as follows:

- We integrate the patch-based diffusion model with the deep image prior (DIP) framework to take a network trained on a mismatched distribution and adapt it at reconstruction time towards the test-time distribution using a self-supervised loss.
- Experimentally, we find this approach of test-time diffusion adaptation leads to improved image quality both in terms of quantitative and qualitative metrics, outperforming using whole-image diffusion models under the same setting. In particular, even when the diffusion model was trained in-distribution, test-time adaptation offers an additional improvement.
- Theoretically, we show that in certain cases, using the self-supervised loss with a patch-based prior leads to a beneficial form of data augmentation compared to whole-image diffusion prior, providing further justification for the superior generalizability of adapting the patch-based prior.

The rest of the paper is organized as follows. Section II reviews the literature of diffusion models, methods of using them to solve inverse problems, and self-supervised methods. Section III details the training and reconstruction method with our patch-based diffusion prior and the main test-time adaptation algorithm, as well as theoretical analysis of the method. Section III reports experiments that empirically verify the efficacy of the proposed methods. Section V concludes the paper.

II. Background and Related Work

Diffusion models and inverse problems: In a general framework, diffusion models involve the forward stochastic differential equation (SDE)

$$\mathrm{d}\boldsymbol{x}_t = -\frac{\beta(t)}{2}\,\boldsymbol{x}_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{w}_t, \tag{1}$$

where $t \in [0, T]$, $\boldsymbol{x}_t \in \mathbb{R}^d$, and $\beta(t)$ is the noise variance schedule of the random process $\mathrm{d}\boldsymbol{w}(t)$. This process adds noise to a clean image and ends with an image indistinguishable from Gaussian noise [7]. Thus, the distribution of $\boldsymbol{x}_0$ is the training data distribution and the distribution of $\boldsymbol{x}_T$ is (approximately) a standard Gaussian. Then the reverse SDE has the form [22]:

$$\mathrm{d}\boldsymbol{x}_t = \left(-\frac{\beta(t)}{2}\boldsymbol{x}_t - \beta(t)\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)\right)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{w}}_t. \tag{2}$$

Score-based diffusion models involve training a neural network to learn the score function $\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)$, from which one can start with noise and run the reverse SDE to obtain samples from the learned data distribution.

When solving inverse problems, the goal is to sample from the posterior distribution $p(\boldsymbol{x}_t|\boldsymbol{y})$, so the reverse SDE becomes

$$\mathrm{d}\boldsymbol{x}_t = \left(-\frac{\beta(t)}{2}\boldsymbol{x}_t - \beta(t)\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t|\boldsymbol{y})\right)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{w}}_t. \tag{3}$$

Unfortunately, the term $\log p_t(\boldsymbol{x}_t|\boldsymbol{y})$ is difficult to compute from the unconditional score $\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)$ alone. To address this problem, previous works [23], [24], [25], [26] among others proposed directly learning this conditional score $\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t|\boldsymbol{y})$ instead. In particular, [23] and [24] propose initializing the reverse SDE with the degraded image and then using a diffusion bridge to arrive at the clean data distribution. On the other hand, [25] and [26] still initialize with pure noise, but use the conditional diffusion model to implicitly enforce consistency with the measurement $\boldsymbol{y}$. These methods require paired data $(\boldsymbol{x}, \boldsymbol{y})$ between the image domain and measurement domain for training. Hence, the learned conditional score function is suitable only for the particular inverse problem for which it was trained, limiting its flexibility.

For greater generalizability, it is desirable to apply the unconditional score $\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)$ to be able to solve a wide variety of inverse problems. Thus, many works have been proposed to approximate the conditional score in terms of the unconditional one [2], [4], [27], [28]. These methods generally involve alternating between updating the image using the trained score network and updating the image to be consistent with the measurement [9]. These two steps are generally detached, so many choices of sampling methods and SDE solvers are possible for the former step while changing the method by which data fidelity is enforced. For example, [29] uses Langevin dynamics with a simple gradient descent step toward the measurement. Recognizing that the image lies on a noisy data manifold (as opposed to the

clean data manifold) during intermediate timesteps, [2] improves on this method by performing gradient descent with respect to the expectation of the clean image at each iteration. As these methods require a large number of iterations, DDIM [30] was developed as an acceleration method that relies on estimating the clean image at each iteration to guide the noisy image toward the clean image manifold. Building off this method, [4] and [28] rely on hard constraint enforcement via projection onto the manifold where data consistency is satisfied while using DDIM as the sampling method, greatly reducing the number of sampling steps needed when solving inverse problems. Finally, conjugate gradient descent has proven to be useful for enforcing data fidelity when used in conjunction with DDIM when the measurements are assumed to be compressed but noiseless [27], [31].

However, even when the number of sampling steps is large, these methods all struggle to truly sample from the posterior, as they all rely on approximations to the posterior distribution at intermediate timesteps. More recently, [32] proposed a method using resampling and Monte Carlo methods to estimate the posterior distribution at each timestep, which was theoretically shown to sample from the posterior distribution when the number of particles is large. Finally, [33] unified various diffusion inverse solvers (DIS) into two categories: the first consists of direct approximations to $p_t(\boldsymbol{y}|\boldsymbol{x}_t)$, and the second consists of first approximating $\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t, \boldsymbol{y}]$ (typically through an optimization problem balancing the prior and measurement) and then applying Tweedie's formula [34] to obtain

$$\nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) = \frac{\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t, \boldsymbol{y}] - \boldsymbol{x}_t}{\sigma_t^2}, \tag{4}$$

where σ_t is the noise level of $\boldsymbol{x}_t$. All of these methods require a large quantity of clean training data that should come from the distribution $p(\boldsymbol{x})$ whose score is to be learned, which may not be available in practice.

Methods without clean training data: When no in-distribution data is available, one approach is to use traditional methods that do not require any training data, such as total variation (TV) [35] or wavelet transform [36] regularizers that encourage image sparsity. More recently, plug and play (PnP) methods have risen in popularity [37], [38], [39], [40]; these methods use a denoiser to solve general inverse problems. Although these methods often use a trained denoiser, [41] found that using an off-the-shelf denoiser such as block matching 3D [42] can yield competitive results. Nevertheless, with the rise of deep learning in image processing applications including denoising, methods that harness the power of these learning based models attract more attention.

The deep image prior (DIP) is an extensively studied self-supervised method that is popular without requiring training data, and the reconstruction is from a single measurement $\boldsymbol{y}$. This method consists of training a network $f_{\boldsymbol{\theta}}$ using the loss function

$$L(\boldsymbol{\theta}) = \|\boldsymbol{y} - \mathcal{A}(f_{\boldsymbol{\theta}}(\boldsymbol{z}))\|_2^2, \quad \boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I}), \tag{5}$$

so that $f_{\boldsymbol{\theta}}(\boldsymbol{z})$ produces the reconstruction. Although the neural network acts as an implicit regularizer whose output tends to lie in the manifold of clean images, DIP is prone to overfitting especially with noisy measurements [43]. Various methods have been proposed involving early stopping, regularization, and network initialization [44], [45], [46]. Nevertheless, the method is very sensitive to the parameter selection and implementation, which can take a long time to train [45].

Most DIS methods learn a prior from a large collection of clean in-distribution training images, but recently [17] and [18] proposed self-supervised diffusion model methods that are based off the DIP framework. These methods involve alternating between the usual reverse diffusion update step to gradually denoise the image and a network refining step that updates the score network parameters via the loss function

$$L(\boldsymbol{\theta}) = \|\boldsymbol{y} - \mathcal{A}(\mathrm{CG}(\hat{\boldsymbol{x}}_{0|t}(\boldsymbol{x}_t; \boldsymbol{\theta})))\|_2^2 \tag{6}$$

where conjugate gradient (CG) descent is used to enforce data fidelity. This CG step consists of solving an optimization of the form

$$\arg\min_{\boldsymbol{x}} \frac{\gamma}{2}\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})\|_2^2 + \frac{1}{2}\|\boldsymbol{x} - \hat{\boldsymbol{x}}_{0|t}\|_2^2, \tag{7}$$

where γ is a trade-off parameter controlling the strength of the prior versus the measurement. Crucially, these methods introduce an additional LoRA module [47] to the network and the original network parameters are frozen when backpropagating the loss, which helps to avoid overfitting the whole-image model. Nevertheless, many technical tricks are required [18] involving noisy initializations and early stopping to obtain good results and avoid artifacts. Our patch-based model avoids this overfitting issue.

Diffusion model fine-tuning: In the small dataset setting, various fine-tuning methods exist to shift the underlying prior learned by a score network away from a mismatched distribution and toward a target distribution. Given a pretrained diffusion network on a mismatched distribution, [48], [49], and [50] among others have studied ways to fine-tune the network to the desired dataset. These methods generally involve freezing certain layers of the original network, appending extra modules that contain relatively few weights, or modifying the loss function to capture details that differ greatly between distributions. However, these methods usually still require thousands of images from the desired distribution and focus on image generation. When solving inverse problems, the reconstructed image should be consistent with the measurement $\boldsymbol{y}$, reducing the number of degrees of the freedom for the image compared to generation, so with proper fine-tuning the data requirement should be lower. Furthermore, even after fine-tuning, the network's underlying distribution may still contain some inconsistencies with the test distribution, leading to artifacts in the reconstructed image.

In summary, numerous methods have been developed for solving inverse problems using well-trained diffusion models that require large training datasets. Different methods also exist to adapt diffusion models to a new distribution under settings of limited or no data, but they vary based on data availability. Section III develops a method that can be applied regardless of the problem setting.

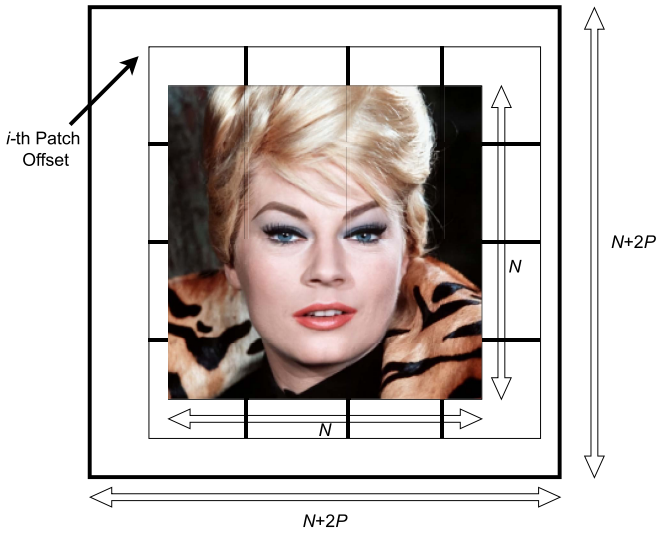


Fig. 1. Schematic for zero padding and partitioning image into patches. Each index i represents one of P^2 possible ways to choose a patch offset tuple.

III. Proposed Methods

This section extends the patch-based diffusion model framework of [21] so that it can be applied with the self-supervised loss for test-time adaptation.

A. Patch-Based Prior

We first zero pad the $N \times N$ image by an amount P on each side and model the resulting padded image $\boldsymbol{x}$. When choosing the ith patch offset tuple $(o_1, o_2) \in \{0, \ldots, P-1\}^2$ in Fig. 1, we partition $\boldsymbol{x}$ into many square patches and one bordering region consisting of all zeros. Since $k = N/P$ patches are needed in one direction to perfectly cover the image, our model for the data distribution has the form

$$p(\boldsymbol{x}) = \frac{1}{Z} \prod_{i=1}^{P^2} p_{i,B}(\boldsymbol{x}_{i,B}) \prod_{r=1}^{(k+1)^2} p_{i,r}(\boldsymbol{x}_{i,r}), \tag{8}$$

where $\boldsymbol{x}_{i,B}$ represents the bordering region of $\boldsymbol{x}$ that depends on the specific value of i, $p_{i,B}$ is the probability distribution of that region, $\boldsymbol{x}_{i,r}$ is the rth $P \times P$ patch when using the partitioning scheme corresponding to the value of i, $p_{i,r}$ is the probability distribution of that region, and Z is a normalizing factor. This model uses many possible tilings of the image, eliminating boundary artifacts that would occur if only one tiling was used.

For training, we use a neural network $D_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma_t)$ that accepts a noisy image $\boldsymbol{x}$ (or a patch of that image) and the noise level σ_t. For each patch, we define the x positional array as the 2D array consisting of the x positions of each pixel of the image, scaled between -1 and 1, and the y positional array is similarly defined for the y positions. To allow the network to learn different patch distributions at different locations in the image, we extract the corresponding patches of these positional arrays and concatenate them along the channel dimension of the noisy image patch and treat the entire array as the network input. Since we are using a patch-based prior, we perform denoising score matching on patches of an image instead of the whole image. Hence, the training loss is given by

$$\arg\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim\mathcal{U}(0,T)} \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})} \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\sigma_t^2 I)} \| D_{\boldsymbol{\theta}}(\boldsymbol{x} + \boldsymbol{\epsilon}, \sigma_t) - \boldsymbol{x} \|_2^2, \tag{9}$$

where here $\boldsymbol{x} \sim p(\boldsymbol{x})$ represents a patch drawn from a sample of the training dataset, σ_t is a predetermined noise schedule, and $\mathcal{U}$ denotes the uniform distribution. See [21] and Appendix A-C for more details about the patch-based training process.

B. Diffusion Self-Supervision

We first review how to apply the self-supervised loss in conjunction with diffusion models. For each specific measurement $\boldsymbol{y}$, the original DIP framework optimizes the network parameters $\boldsymbol{\theta}$ via the self-supervised loss (5) from the predicted reconstructed image. Diffusion models provide a prediction of the reconstructed image at each timestep: namely, the expectation of the clean image $\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]$ is approximated by the denoiser $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$ via Tweedie's formula. Then the expectation conditioned on the measurement $\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t, \boldsymbol{y}]$ can be obtained through one of many methods of enforcing the data fidelity constraint. That conditional expectation should be used in test-time adaptation of diffusion models.

We begin with the unconditional expectation by leveraging the patch-based prior. Following (8), we apply Tweedie's formula to express the denoiser of $\boldsymbol{x}$ solely in terms of denoisers of the patches of $\boldsymbol{x}$. Because the outermost product is computationally very expensive, in practice we approximate $D_{\boldsymbol{\theta}}(\boldsymbol{x})$ using only a single randomly selected value of offset index i for each denoiser evaluation:

$$D_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx D_{i,B}(\boldsymbol{x}_{i,B}) + \sum_{r=1}^{(k+1)^2} D_{i,r}(\boldsymbol{x}_{i,r}). \tag{10}$$

By definition, $D_{i,B}(\boldsymbol{x}_{i,B}) = 0$ and we compute each $D_{i,r}(\boldsymbol{x}_{i,r})$ with the network. Note that (10) provides an *unconditional* estimate of the clean image; to obtain an approximate conditional estimate $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y})$ of the clean image, we run C iterations of the conjugate gradient descent algorithm for minimizing $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2$, initialized with the unconditional estimate [27].

The image that is being reconstructed might not come from the distribution of the training images. Hence, the estimate $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y})$ may be far from the true denoised image. Thus, we use $\boldsymbol{y}$ to update the parameters of the network such that $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y})$ becomes more consistent with the measurement:

$$\boldsymbol{\theta} \leftarrow \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{y} - \boldsymbol{A}\, D_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y})\|_2^2. \tag{11}$$

Previously, additional LoRA parameters [47] were used as an injection to the network to leave the original parameters unchanged during this process [17], [18]. However, the effect of using different ranks for LoRA versus other methods of network fine-tuning on DIS has not been studied extensively, so we opt to update all the weights of the network in this step. Appendix A-A shows results from using the LoRA module.

Crucially, iterative usage of CG for computing the conditional denoiser allows for simple and efficient backpropagation through this loss function, a task that would be much more

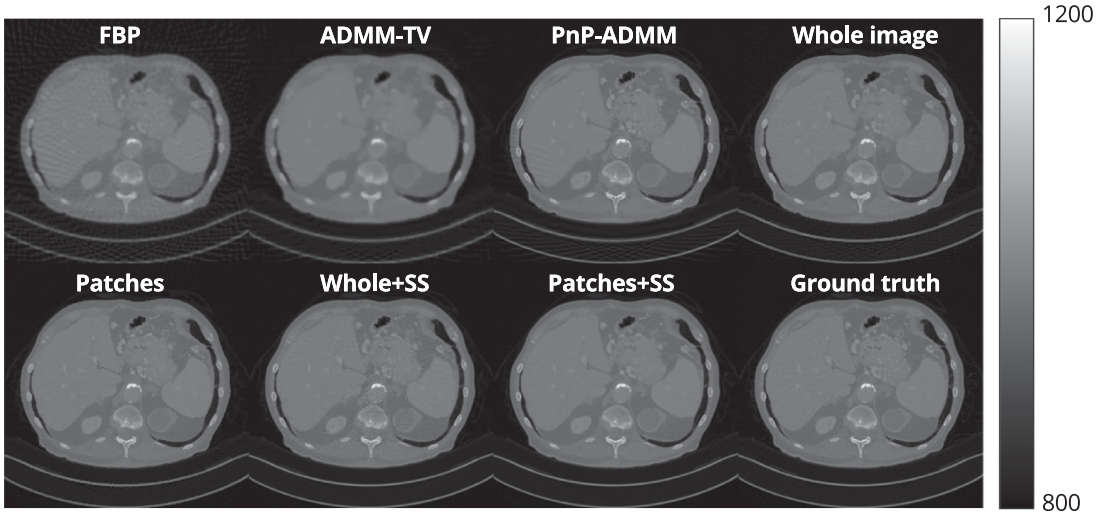


Fig. 2. Dataless setting: Results of 60 view CT reconstruction using self supervised (SS) loss. The display uses modified HU units to show more contrast between organs.

Algorithm 1: Test-Time Adapted Diffusion Inverse Solver.

Require: $\sigma_1 < \sigma_2 < \ldots < \sigma_T, \epsilon > 0, P, C, \boldsymbol{y}, K$
Initialize $\boldsymbol{x} \sim \mathcal{N}(0, \sigma_T^2 \boldsymbol{I})$
for $t = T : 1$ **do**
  if t mod $K = 0$ **then**
    Compute $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$ using (10) with a random index i
    Run C iterations of CG initialized with $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$ to obtain $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y})$
    Define $L(\boldsymbol{\theta}) = \|\boldsymbol{y} - \boldsymbol{A} D_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y})\|_2^2$
    Update $\boldsymbol{\theta}$ by backpropagating $L(\boldsymbol{\theta})$
  end if
  Sample $\boldsymbol{z} \sim \mathcal{N}(0, \sigma_t^2 \boldsymbol{I})$
  Set $\alpha_t = \epsilon \cdot \sigma_t^2$
  Compute $D(\boldsymbol{x}_t)$ using (10) with a random index i
  Run C iterations of CG for (7) initialized with $D(\boldsymbol{x}_t)$
  Set $\boldsymbol{s}_t = (D - \boldsymbol{x}_t)/\sigma_t^2$
  Set $\boldsymbol{x}_{t-1}$ to $\boldsymbol{x}_t + \frac{\alpha_t}{2}\boldsymbol{s}_t + \sqrt{\alpha_t}\boldsymbol{z}$
end for

computationally challenging if another DIS such as [2] or [4] were used. Furthermore, because the number of diffusion steps is large and the change in $\boldsymbol{x}_t$ is small between consecutive timesteps, we apply this network refining step only for certain iterations of the diffusion process, reducing the computational burden.

After this step, we apply the refined network to compute a new estimate of the score of $\boldsymbol{x}_t$ and then use it to update $\boldsymbol{x}_t$. Similar to the network refining step, we use the stochastic version of the denoiser given by (10) rather than the full version. Ref. [21] showed that for patch-based priors, Langevin dynamics [29] worked particularly well as a sampling algorithm, so we used it here in conjunction with CG steps to enforce data fidelity. Algorithm 1 summarizes the entire test-time adapted method for solving inverse problems.

C. Patch-Based Training

In the small dataset and in-distribution settings, we are provided clean training images that can be used to train the patch-based network. In the small dataset setting, the dataset is too small to directly train a diffusion model from scratch, so we initialize the training process with a checkpoint trained on a different large dataset and then fine-tune this checkpoint on the small dataset. Works such as [48] and [49] found that this fine-tuning process allows diffusion models to be trained with a much smaller dataset than would otherwise be required. On the other hand, for the in-distribution setting, we initialize the weights of the network randomly and train on the large in-distribution dataset. Ref. [19] found that training with varying patch sizes improved image generation performance compared to fixing the patch size to that used during inference. Here, we also applied a varying patch size scheme during fine-tuning as a method of data augmentation. We used the UNet architecture in [8] that can accept images of different sizes. Hence, the loss becomes

$$\arg\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim\mathcal{U}(0,T)} \mathbb{E}_{\boldsymbol{x}\sim p_d(\boldsymbol{x})} \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\sigma_t^2 I)} \|D_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\epsilon}, \sigma_t) - \boldsymbol{x}\|_2^2, \tag{12}$$

where $\boldsymbol{x} \sim p_d(\boldsymbol{x})$ represents drawing a patch of random size and location from an image belonging to the fine-tuning dataset. Appendix A-C provides full details of the training process.

D. Theoretical Analysis

This section provides a rough sketch of why Algorithm 1 should work better for the patch-based model compared to a whole-image model. Firstly, rewrite (10) as follows:

$$D(\boldsymbol{x}) = \sum_c \boldsymbol{G}_c' D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c), \tag{13}$$

where c denotes the patch location, $\boldsymbol{G}_c$ denotes a patch extracting operator that extracts the patch corresponding to location c from the whole image $\boldsymbol{x}$. Thus $\boldsymbol{G}_c'$ is an operator that takes a patch

TABLE I
COMPARISON OF QUANTITATIVE RESULTS ON FOUR DIFFERENT INVERSE PROBLEMS IN THE DATALESS SETTING WITH THE SELF-SUPERVISED LOSS (SS)

Method	CT, 20 Views		CT, 60 Views		Deblurring		Superresolution	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Baseline	24.93	0.613	30.15	0.784	23.93	0.666	25.42	0.724
ADMM-TV	26.81	0.750	31.14	0.862	27.58	0.773	25.22	0.729
PnP-ADMM [57]	30.20	0.838	36.75	0.932	28.98	0.815	27.29	0.796
PnP-RED [58]	27.12	0.682	32.68	0.876	28.37	0.793	27.73	0.809
Whole image	28.11	0.800	33.10	0.911	25.85	0.742	25.65	0.742
Patches [21]	27.44	0.719	33.97	0.934	26.77	0.782	26.12	0.759
Whole+SS [17]	33.19	0.861	40.47	0.957	29.50	0.831	27.07	0.701
Patches+SS (Ours)	**33.77**	**0.874**	**41.45**	**0.969**	**30.34**	**0.860**	**28.10**	**0.827**

Results are averages across all images in the test dataset. Best results for practical use are in bold.

and returns a whole image with the corresponding patch filled in (and the rest of the entries are zeros). Note that c is input to the patch-based network $D_{\boldsymbol{\theta}}$ through positional encoding.

Now we analyze (11) using this framework. The loss function for self-supervision is given by:

$$L(\boldsymbol{\theta}) = \left\| \boldsymbol{y} - \boldsymbol{A} \sum_c \boldsymbol{G}_c' D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y}) \right\|_2^2. \tag{14}$$

For inverse problems such as superresolution and CT reconstruction, $\boldsymbol{A}$ is a wide matrix that has full row rank. Hence, even when the measurement $\boldsymbol{y}$ is noisy, there exists some $\boldsymbol{x}_0$ with $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0$. Then we have

$$L(\boldsymbol{\theta}) = \left\| \boldsymbol{A}(\boldsymbol{x}_0 - \sum_c \boldsymbol{G}_c' D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y})) \right\|_2^2 \tag{15}$$

$$L(\boldsymbol{\theta}) \leq \|\boldsymbol{A}\|_2^2 \left\| \boldsymbol{x}_0 - \sum_c \boldsymbol{G}_c' D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y}) \right\|_2^2 \tag{16}$$

$$= \|\boldsymbol{A}\|_2^2 \left\| \sum_c \boldsymbol{G}_c' \boldsymbol{G}_c \boldsymbol{x}_0 - \sum_c \boldsymbol{G}_c' D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y}) \right\|_2^2, \tag{17}$$

where in the last step, we have used the fact that at each diffusion iteration, the patches are nonoverlapping. Thus,

$$L(\boldsymbol{\theta}) \leq \|\boldsymbol{A}\|_2^2 \left\| \sum_c \boldsymbol{G}_c' (\boldsymbol{G}_c \boldsymbol{x}_0 - D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y})) \right\|_2^2. \tag{18}$$

Now we have a sum of the form $\|\boldsymbol{z}_1 + \ldots + \boldsymbol{z}_n\|_2^2$, which [3] showed to be upper bounded by $K(\|\boldsymbol{z}_1\|_2 + \ldots + \|\boldsymbol{z}_n\|_2^2)$ for a fixed constant K. Applying this inequality and absorbing $\|\boldsymbol{A}\|_2^2$ into the constant, we have

$$L(\boldsymbol{\theta}) \leq K \sum_c \|\boldsymbol{G}_c' (\boldsymbol{G}_c \boldsymbol{x}_0 - D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y}))\|_2^2 \tag{19}$$

$$= K \sum_c \|\boldsymbol{G}_c \boldsymbol{x}_0 - D_{\boldsymbol{\theta}}(\boldsymbol{G}_c \boldsymbol{x}, c|\boldsymbol{y})\|_2^2 \tag{20}$$

A similar derivation for the whole image model shows that the loss in that case is bounded by

$$L_w(\boldsymbol{\theta}) \leq K \|\boldsymbol{x}_0 - D_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{y})\|_2^2. \tag{21}$$

Table S.5 shows that performance is improved by using more backpropagation iterations for the loss function; hence, although in practice we only perform a fixed number of iterations for speed, optimally we should aim to reduce the loss $L(\boldsymbol{\theta})$ to zero. Observe that (20) has the same form as the loss that would be used for refining the network with a whole image model (21). However, now instead of a loss of a single image, we now have individual losses of many patches of an image. For example, the experiments of Table I used 25 patches to tile each image for each diffusion iteration, so we had 25 losses. This method of data augmentation helps explain why the patch-based model obtains better performance than the whole-image model when performing test-time adaptation. We additionally note that although the positional encoding input into the network is different for each patch, the network does not separately learn a distribution for each position, as the weights are shared across these different positions. This is analogous to the analysis of [51], where a single diffusion model was trained on the 1000 classes of ImageNet with the class label of the image being included as an additional input to the network. Since each class only had around 1000 images, it would have been very difficult to train a diffusion model on only one of the classes, but by training across all the classes at once, a much better network can be trained.

IV. EXPERIMENTS

This section reports empirical results on sparse-view CT reconstruction, image deblurring, and image super-resolution. For each computational imaging application, we illustrate the benefits of using the self supervised (SS) loss in all three of the settings described in the introduction: *dataless* (zero-shot SS), *small dataset* (fine tuning followed by SS), and *in-distribution* (where one might expect SS to be counter-productive).

Experimental setup: For the CT experiments, we used the AAPM 2016 CT challenge data from [52]. We applied the same data processing methods as in [21] with the exception that we used all the XY (transaxial) slices of size 256×256 from the 9 training volumes to train the in distribution networks, yielding a total of 5936 slices. For the deblurring and superresolution experiments, we used the CelebA-HQ dataset [53] with each image having size 256×256. The test data was a randomly selected subset of 10 of the images not used for training. In all

Fig. 3. Dataless setting: Results of deblurring using test-time adaptation via the self supervised (SS) loss and several comparison methods.

cases, we report the average metrics across the test images: peak SNR (PSNR) in dB, and structural similarity metric (SSIM) [54].

In the dataless and small dataset settings, since there is insufficient data from the test distribution to train the diffusion model well, we first trained the network using generated synthetic data. This data consisted of phantom images consisting of randomly placed ellipses of different shapes and sizes. See Fig. S.7 for examples. These phantoms can be generated on the fly in large quantities. We used networks trained on grayscale phantoms for the CT experiments and networks trained on RGB phantoms for the deblurring and superresolution experiments. Appendix A-B contains precise specifications of the phantoms. Then in the small dataset setting, we fine-tuned the network (originally trained on the phantom images) on the small dataset. The small dataset consisted of 10 images randomly selected from the in-distribution training set; we also ran ablation studies using different quantities of in-distribution data in Appendix A-A. In contrast, for the in-distribution setting, there was sufficient data to directly train a diffusion model on the in-distribution data, so we initialized the network weights randomly and trained only on the in-distribution dataset.

We trained the patch-based networks with 64×64 patches and used a zero padding value of 64, so that 5 patches in both directions were used to cover the target image. We used the network architecture in [55] for both the patch-based networks and whole-image networks. All networks were trained on PyTorch using the Adam optimizer with 2 A40 GPUs.

Diffusion test-time adaptation: To evaluate the effectiveness of test-time adaptation, we applied Algorithm 1 to solve each of the inverse problems. For the forward and backward projectors in CT reconstruction, we used the implementation provided by the ODL [56]. We performed two sparse-view CT (SVCT) experiments: one using 20 projection views, and one using 60 projection views. Both of these used a parallel-beam forward projector where the detector size was 512 pixels. For the deblurring experiments, we used a uniform blur kernel of size 9×9 and added white Gaussian noise with $\sigma = 0.01$ where the clean image was scaled between 0 and 1. For the superresolution experiments, we used a scaling factor of 4 with downsampling by averaging and added white Gaussian noise with $\sigma = 0.01$.

For the comparison methods, we ran experiments that directly used the diffusion model without test-time adaptation. In particular, we used the network trained only on phantoms for the dataless setting, the fine-tuned network in the small dataset setting, and the in-distribution network in the last setting. We used the same sampling algorithm (Langevin dynamics) and inverse problem solving method (conjugate gradient descent) as Algorithm 1, but removed the test-time adaptation step. Additionally, for these diffusion model methods, we implemented both the patch-based version as well as the whole-image version. The whole-image networks were trained with the loss function in (9) and used the same network architecture as the patch-based models, but the input of the network was the entire image and did not contain positional encoding information.

We also compared with more traditional methods: applying a simple baseline, reconstructing via the total variation regularizer (ADMM-TV), and two plug and play (PnP) methods: PnP-ADMM [57] and PnP-RED [58]. For CT, the baseline was obtained by applying the filtered back-projection method to the measurement $\boldsymbol{y}$. For deblurring, the baseline was simply equal to the blurred image $\boldsymbol{y}$. For superresolution, the baseline was obtained by upsampling the low resolution image $\boldsymbol{y}$ using nearest-neighbor interpolation. The implementation of ADMM-TV is in [59]. Finally, since we assume we do not have access to a large sample of clean training data, we used the off-the-shelf denoiser BM3D [42]. Appendix A-C contains the values of all the parameters of the algorithms. Tables I, III, and IV show the main results for the dataless, small dataset, and in-distribution settings, respectively. In the latter two settings, we chose to redisplay only the results of the best baseline out of the baseline, ADMM-TV, PnP-ADMM, and PnP-RED shown in Table I and labeled that row "Best baseline." Particularly for the dataless setting, applying the test-time adaptation method yields

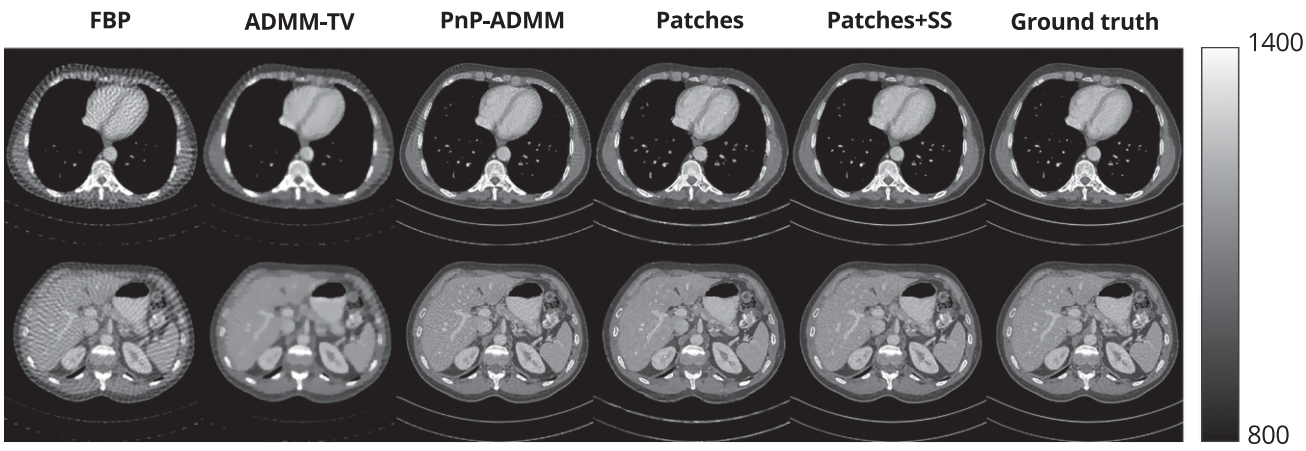


Fig. 4. Results of 60 view reconstruction on 512×512 images in dataless setting displayed in Hounsfield units.

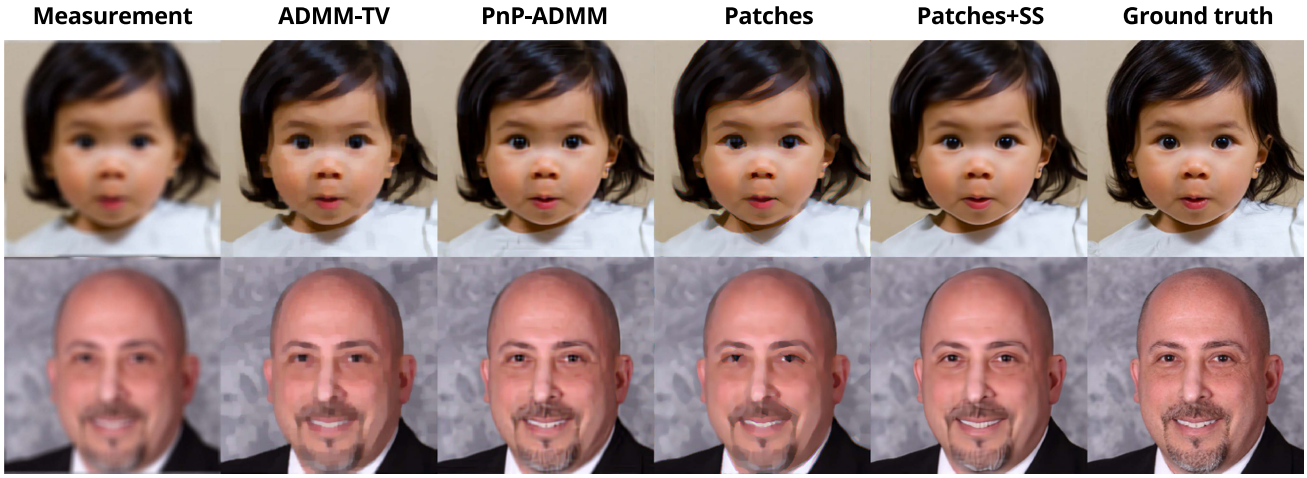


Fig. 5. Results of deblurring on 512×512 images in dataless setting.

TABLE II
RESULTS OF INVERSE PROBLEM SOLVING IN DATALESS SETTING FOR 512×512 IMAGES

Method	CT, 60 views		Deblur	
	PSNR↑	SSIM ↑	PSNR↑	SSIM ↑
Baseline	28.33	0.700	24.11	0.649
ADMM-TV	29.36	0.788	28.14	0.760
PnP-ADMM	37.48	0.910	29.77	0.812
Patch, naive	29.32	0.793	26.58	0.749
Patch, SS	**37.82**	**0.919**	**30.35**	**0.825**

much higher quantitative results when averaged across the test dataset than simply using the pretrained diffusion model in all the inverse problems. However, we observe that even when the pretrained diffusion model was trained on the large in-distribution dataset, including the test-time adaptation step still resulted in further improvement in image quality. Thus, in contrast with many other self-supervised methods such as DIP, our method can avoid overfitting to the measurement and even benefit from test-time refinement. Appendix A-A further analyzes overfitting and shows that by increasing the number of network refining iterations done per diffusion iteration, the image quality does not drop, indicating that overfitting is avoided. We further note that in all three settings, when using the self-supervised loss, the patch-based prior outperformed the whole image prior, which is consistent with our theoretical analysis of Algorithm 1. Lastly, Fig. 2 shows that some artifacts appear in the whole-image SS method that are absent in our patch SS method.

We also ran ablation studies to examine the effect of various parameters on the proposed method. [17] and [18] used the LoRA module for solving single-measurement inverse problems with diffusion models. We tested this method for CT reconstruction and deblurring with different rank adjustments and found this method to be inferior to modifying the weights of the entire network. We also ran experiments using networks with different numbers of weights. Appendix A-A shows the results of these experiments.

The initial motivation of using patch-based diffusion models was partially to solve high resolution imaging problems [21]. To show that our method scales to larger images, we ran experiments on 60 view CT reconstruction and deblurring with 512×512 images in the dataless setting. For the CT experiments, we still used the AAPM dataset [52] processed in the same way as for

TABLE III
COMPARISON OF RESULTS FOR USING SELF-SUPERVISED (SS) DIFFUSION MODELS IN SMALL DATASET SETTING

Method	CT, 20 Views		CT, 60 Views		Deblurring		Superresolution	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Best baseline	30.20	0.838	36.75	0.932	28.98	0.815	27.73	0.809
Whole image	33.09	0.875	40.54	0.964	28.41	0.812	27.29	0.775
Patches	33.44	0.875	41.21	0.965	29.25	0.840	28.10	0.827
Whole image+SS	34.57	0.881	41.34	0.962	30.16	0.852	27.72	0.814
Patches+SS	**36.43**	**0.914**	**42.42**	**0.971**	**30.56**	**0.867**	**28.60**	**0.834**

The diffusion model is first trained on ellipse phantoms and then fine-tuned with the small dataset. Best results are in bold.

TABLE IV
COMPARISON OF RESULTS FOR USING DIFFUSION MODELS TRAINED ON THOUSANDS OF IN-DISTRIBUTION IMAGES TO SOLVE INVERSE PROBLEMS

Method	CT, 20 Views		CT, 60 Views		Deblurring		Superresolution	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Best baseline	30.20	0.838	36.75	0.932	28.98	0.815	27.73	0.809
Whole image	33.99	0.886	41.67	0.969	29.87	0.851	28.33	0.801
Patches	34.02	0.889	41.70	0.967	30.12	0.865	28.49	0.835
Whole image+SS	35.38	0.897	41.68	0.966	30.31	0.854	27.96	0.797
Patches+SS	**36.82**	**0.923**	**42.33**	**0.970**	**30.78**	**0.875**	**28.72**	**0.842**

Best results are in bold. Self-supervised refinement of network weights is beneficial even in this setting.

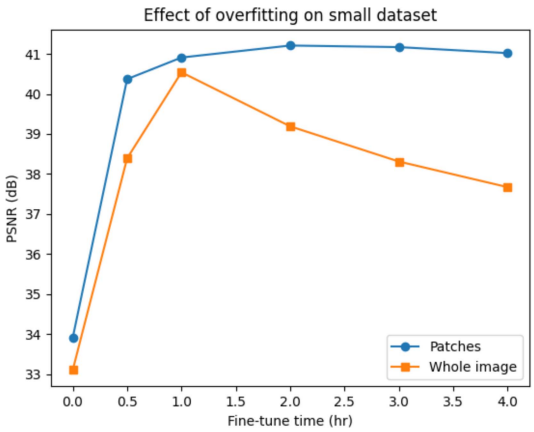


Fig. 6. Comparison of PSNR between patch-based model and whole-image model for overfitting in small dataset setting.

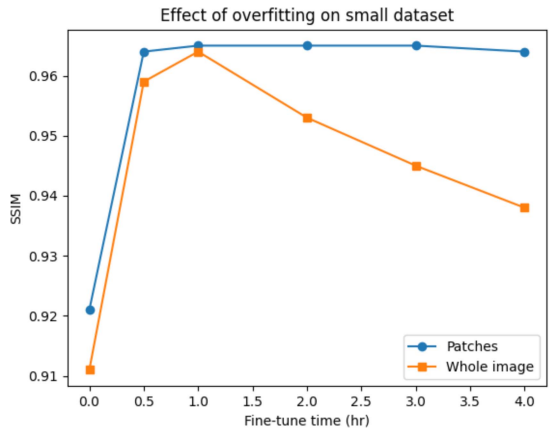


Fig. 7. Comparison of SSIM between patch-based model and whole-image model for overfitting in small dataset setting.

Table I, but kept the slices in their original size of 512×512. For deblurring, we used the FFHQ dataset [60] which contains images of size 512×512. We scaled each of the RGB channels to between 0 and 1. We used a uniform blur kernel of size 17×17 and added noise with $\sigma = 0.01$. We used the same patch-based networks trained for Table I as initializations for these out of distribution experiments. Since the patch size at reconstruction time was kept to be 64×64 as before, we used 9 patches in both directions (for 81 total) to tile each image. Table II shows results of these experiments, where our method obtained the highest quality reconstruction. Although the improvement is modest, note that we trained the patch-based model on phantoms images of size 256×256, which is extremely far out of distribution from the test dataset.

Small dataset fine-tuning: We further examined the effects of fine-tuning the networks using the small dataset with the patch-based model and the whole image model. Figs. 6 and 7 further investigate the effect of overfitting. For different amounts of training time using the small in-distribution dataset, we ran the reconstruction algorithm for 60-view CT. While the whole-image model exhibited substantial image degradation when the network was fine-tuned for too long, the patch-based model retained relatively stable performance throughout the entire training process. This illustrates that whole-image diffusion models exhibits severe overfitting problems when only a small amount of training data is unavailable, similar to the original DIP method. Furthermore, patch-based diffusion models assist greatly with this problem and the results are evident for solving inverse problems.

To look at the priors learned by the different models from fine-tuning, we unconditionally generated images from the checkpoints obtained by fine-tuning on the 10 image CT dataset. Fig. 9 shows a subset of the generated images where we used the checkpoints obtained after 4 hours of training. The top two

Fig. 8. **Small dataset setting**: Results of inverse problem solving. Top row is 60 view CT recon, middle row is deblurring, and bottom row is superresolution.

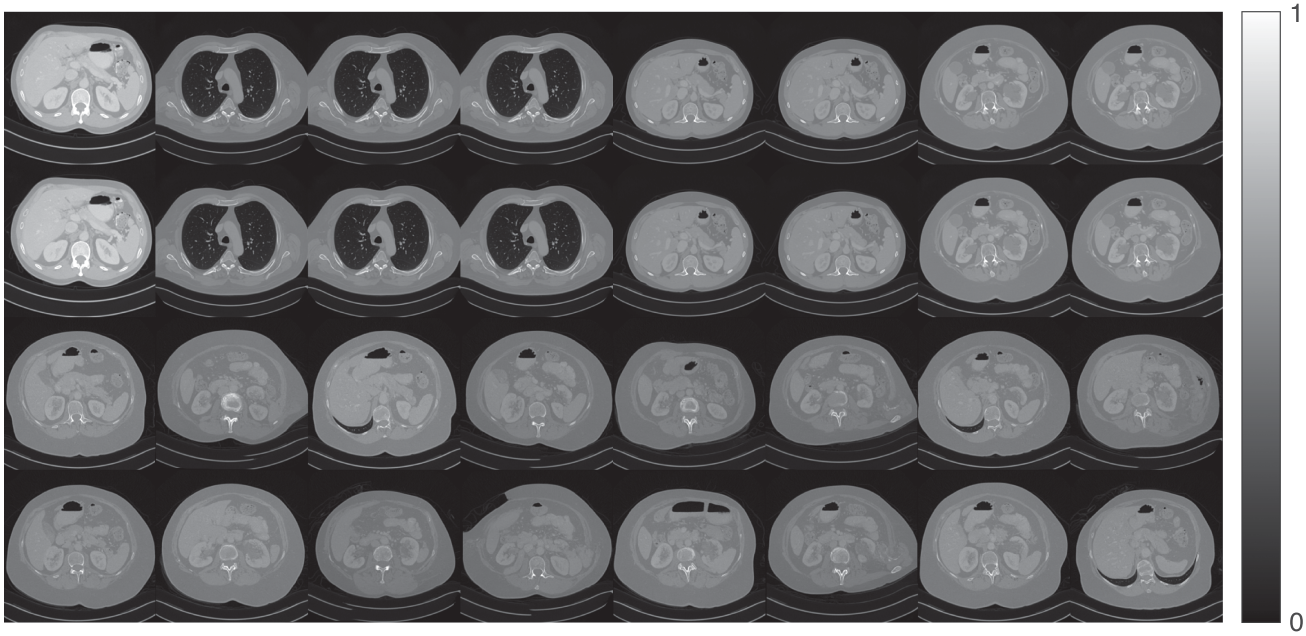


Fig. 9. Unconditional generation of CT images from networks fine-tuned in the small dataset setting. Top two rows were generated with the whole image model; bottom two rows were generated with the patch-based model.

rows consist of images generated by the whole-image model and the bottom two rows consist of images generated by the patch diffusion model. To emphasize the memorization point, we grouped together similar looking images in the top two rows: it can be seen that the images in each group look virtually identical, despite the fact that the pure white noise initializations for each sample was different. On the other hand, while the samples generated by the patch diffusion model also show some unrealistic features, they all show some distinct features, which implies that this model has much better generalization ability.

Finally, to demonstrate that our method also works well even when the mismatched distribution is closer to the true distribution, we also ran an experiment where the networks were initially trained on the LIDC-IDRI dataset of CT scans [61]. We extracted 10000 2D slices from the 3D volumes and rescaled all the images so that the pixel values were between 0 and 1. We then ran

TABLE V
CT RECONSTRUCTION RESULTS WHERE THE INITIAL CHECKPOINT WAS TRAINED ON LIDC DATASET AND REFINED ON THE FLY WITH THE AAPM MEASUREMENT

Dataset size	CT, 20 views		CT, 60 views	
	PSNR↑	SSIM ↑	PSNR↑	SSIM ↑
Whole image	33.04	0.874	40.43	0.949
Patches	33.88	0.886	40.96	0.955
Whole image+SS	35.01	0.894	41.95	0.967
Patches+SS (Ours)	36.34	0.918	42.32	0.972

Algorithm 1 to perform CT reconstruction where the test dataset was the same as the one used in Table I. Table V shows the results of this experiment. Our method achieved better quantitative results than the whole-image method and even outperformed the reconstructions using the in distribution network but without any test-time adaptation.

V. CONCLUSION

This paper presented a method of using patch-based diffusion models with test-time adaptation to solve inverse problems when the data distribution might be mismatched from the trained network. In particular, we conducted experiments in setting when no data is available, when a small dataset of training images is available, and when a large in-distribution dataset is available. In all the settings, applying the self-supervised loss improved image quality, even for a well-trained network. Furthermore, the patch-based method outperformed whole-image methods in a variety of inverse problems and we provided theoretical justifications to explain this improvement. In the future, more work could be done on using acceleration methods for faster reconstruction, exploring other less computationally expensive methods of fine-tuning the network geared toward inverse problem solving, and methods of refining the prior when a set of measurements are available [62]. Limitations of the work include a slow runtime for the test-time adapted algorithm and a lack of theoretical guarantees for dataset size requirements. Providing uncertainty quantification is also an open problem for such self-supervised methods.

REFERENCES

[1] H. Chung, B. Sim, D. Ryu, and J.C. Ye, "Improving diffusion models for inverse problems using manifold constraints," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2022, pp. 25683–25696. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/a48e5877c7bf86a513950ab23b360498-Paper-Conference.pdf

[2] H. Chung, J. Kim, M.T. Mccann, M. L. Klasky, and J.C. Ye, "Diffusion posterior sampling for general noisy inverse problems," in *Proc. 11th Int. Conf. Learn. Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=OnD9zGAGT0k

[3] B. Song, J. Hu, Z. Luo, J.A. Fessler, and L. Shen, "Diffusionblend: Learning 3D image prior through position-aware diffusion score blending for 3D computed tomography reconstruction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 89584–89611.

[4] Y. Wang, J. Yu, and J. Zhang, "Zero-shot image restoration using denoising diffusion null-space model," in *Proc. Int. Conf. Learn. Representations*, 2023. [Online]. Available: https://openreview.net/pdf?id=mRieQgMtNTQ

[5] B. Kawar, G. Vaksman, and M. Elad, "Snips: Solving noisy inverse problems stochastically," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 21757–21769. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/b5c01503041b70d41d80e3dbe31bbd8c-Paper.pdf

[6] Z. Li, J. Hu, X. Xu, L. Shen, and J.A. Fessler, "Accelerated wirtinger flow with score-based image priors for holographic phase retrieval in Poisson-Gaussian noise conditions," *IEEE Trans. Comput. Imag.*, vol. 10, pp. 1384–1399, 2024.

[7] Y. Song, J. Sohl-Dickstein, D.P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. 9th Int. Conf. Learn. Representations*, Austria, May 3–7, 2021. [Online]. Available: https://openreview.net/forum?id=PxTIG12RRHS

[8] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020, vol. 33, pp. 6840–6851. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf

[9] Y. Song, L. Shen, L. Xing, and S. Ermon, "Solving inverse problems in medical imaging with score-based generative models," in *Proc. Int. Conf. Learn. Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=vaRCHVj0uGI

[10] A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. Tamir, "Robust compressed sensing MRI with deep generative priors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 14938–14954. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/7d6044e95a16761171b130dcb476a43e-Paper.pdf

[11] B.T. Feng, J. Smith, M. Rubinstein, H. Chang, K.L. Bouman, and W.T. Freeman, "Score-based diffusion models as principled priors for inverse imaging," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 10486–10497.

[12] B.T. Feng, R. Baptista, and K.L. Bouman, "Neural approximate mirror maps for constrained diffusion models," in *Proc. Int. Conf. Learn. Representations*, 2025.

[13] Z. Wu, Y. Sun, J. Liu, and U. Kamilov, "Online regularization by denoising with applications to phase retrieval," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, 2019, pp. 3887–3895.

[14] A.W. Reed et al., "Dynamic CT reconstruction from limited views with implicit neural representations and parametric motion fields," in *Proc. Int. Conf. Comput. Vis.*, 2021, pp. 2238–2248.

[15] Z. Li, X. Xu, J. Hu, J. Fessler, and Y. Dewaraja, "Reducing spect acquisition time by predicting missing projections with single-scan self-supervised coordinate-based learning," *J. Nucl. Med.*, vol. 64, no. supplement 1, 2023, Art. no. P1014. [Online]. Available: https://jnm.snmjournals.org/content/64/supplement_1/P1014

[16] T. Gureyev, A. Pogany, D. Paganin, and S. Wilkins, "Linear algorithms for phase retrieval in the fresnel region," *Opt. Commun.*, vol. 231, no. 1, pp. 53–70, 2004.

[17] R. Barbano et al., "Steerable conditional diffusion for out-of-distribution adaptation in imaging inverse problems," *IEEE Trans. Med. Imag.*, vol. 44, no. 5, pp. 2093–2104, 2025.

[18] H. Chung and J. C. Ye, "Deep diffusion image prior for efficient OOD adaptation in 3D inverse problems," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 432–455.

[19] Z. Wang et al., "Patch diffusion: Faster and more data-efficient training of diffusion models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 72137–72154.

[20] Z. Ding, M. Zhang, J. Wu, and Z. Tu, "Patched denoising diffusion models for high-resolution image synthesis," in *Proc. Int. Conf. Learn. Representations*, 2024. [Online]. Available: https://openreview.net/pdf?id=TgSRPRz8cI

[21] J. Hu, B. Song, X. Xu, L. Shen, and J. A. Fessler, "Learning image priors through patch-based diffusion models for solving inverse problems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, vol. 38, pp. 1625–1660. [Online]. Available: https://nips.cc/virtual/2024/poster/95843

[22] B. D. Anderson, "Reverse-time diffusion equation models," *Stochastic Processes Appl.*, vol. 12, no. 3, pp. 313–326, 1982.

[23] G.-H. Liu, A. Vahdat, D.-A. Huang, E.A. Theodorou, W. Nie, and A. Anandkumar, "I ^{2} SB: Image-to-image Schrödinger bridge," in *Proc. Int.*

Conf. Mach. Learn., 2023, pp. 22042–22062. [Online]. Available: https://proceedings.mlr.press/v202/liu23ai/liu23ai.pdf

[24] H. Chung, J. Kim, and J. C. Ye, "Direct diffusion bridge using data consistency for inverse problems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 7158–7169. [Online]. Available: https://openreview.net/pdf?id=497CevPdOg

[25] O. Ozdenizci and R. Legenstein, "Restoring vision in adverse weather conditions with patch-based denoising diffusion models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 10346–10357, Aug. 2023.

[26] M. Guan et al., "DiffWater: Underwater image enhancement based on conditional denoising diffusion probabilistic model," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 17, pp. 2319–2335, 2024.

[27] H. Chung, S. Lee, and J.C. Ye, "Decomposed diffusion sampler for accelerating large-scale inverse problems," in *Proc. Int. Conf. Learn. Representations*, 2024.

[28] B. Kawar, M. Elad, S. Ermon, and J. Song, "Denoising diffusion restoration models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 36, pp. 23593–23606. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/95504595b6169131b6ed6cd72eb05616-Paper-Conference.pdf

[29] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 11918–11930. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf

[30] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Representations*, 2021.

[31] B. Song, S.M. Kwon, Z. Zhang, X. Hu, Q. Qu, and L. Shen, "Solving inverse problems with latent diffusion models via hard data consistency," in *Proc. Int. Conf. Learn. Representations*, 2023.

[32] Y. Janati, A. Durmus, E. Moulines, and J. Olsson, "Divide-and-conquer posterior sampling for denoising diffusion priors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 97408–97444. [Online]. Available: https://openreview.net/pdf/8a10688dd85040cdfeddb7d0c0035580f4b7af41.pdf

[33] X. Peng et al., "Improving diffusion models for inverse problems using optimal posterior covariance," in *Proc. Int. Conf. Mach. Learn.*, 2024.

[34] B. Efron, "Tweedie's formula and selection bias," *J. Amer. Stat. Assoc.*, vol. 106, no. 496, pp. 1602–1614, 2011.

[35] S. Li et al., "An efficient iterative cerebral perfusion CT reconstruction via low-rank tensor decomposition with spatial–temporal total variation regularization," *IEEE Trans. Med. Imag.*, vol. 38, no. 2, pp. 360–370, Feb. 2019.

[36] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA, USA: SIAM, 1992.

[37] Y. Sun, Z. Wu, X. Xu, B.E. Wohlberg, and U. Kamilov, "Scalable plug-and-play ADMM with convergence guarantees," *IEEE Trans. Comput. Imag.*, vol. 7, pp. 849–863, 2021.

[38] S. Sreehari et al., "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Trans. Comput. Imag.*, vol. 2, no. 4, pp. 408–423, Dec. 2016.

[39] T. Hong, I. Yavneh, and M. Zibulevsky, "Solving RED with weighted proximal methods," *IEEE Signal Process. Lett.*, vol. 27, pp. 501–505, 2020.

[40] T. Hong, X. Xu, J. Hu, and J.A. Fessler, "Provable preconditioned plug-and-play approach for compressed sensing MRI reconstruction," *IEEE Trans. Comput. Imag.*, vol. 10, pp. 1476–1488, 2024.

[41] E.K. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, "Plug-and-play methods provably converge with properly trained denoisers," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5546–5557. [Online]. Available: https://proceedings.mlr.press/v97/ryu19a/ryu19a.pdf

[42] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising with block-matching and 3D filtering," *Proc. SPIE*, vol. 6064, 2006, Art. no. 606414.

[43] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," *Int. J. Comput. Vis.*, vol. 128, no. 7, pp. 1867–1888, Mar. 2020.

[44] J. Liu, Y. Sun, X. Xu, and U.S. Kamilov, "Image restoration using total variation regularized deep image prior," in *Proc. 2019 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 7715–7719.

[45] Y. Jo, S. Y. Chun, and J. Choi, "Rethinking deep image prior for denoising," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5067–5076.

[46] R. Barbano et al., "An educated warm start for deep image prior-based micro CT reconstruction," *IEEE Trans. Comput. Imag.*, vol. 8, pp. 1210–1222, 2022.

[47] E.J. Hu et al., "Lora: Low-rank adaptation of large language models," in *Proc. Int. Conf. Learn. Representations*, 2022.

[48] T. Moon, M. Choi, G. Lee, J.-W. Ha, and J. Lee, "Fine-tuning diffusion models with limited data," in *Proc. NeurIPS 2022 Workshop Score-Based Methods*, 2022. [Online]. Available: https://openreview.net/forum?id=0J6afk9DqrR

[49] X. Zhang et al., "Spectrum-aware parameter efficient fine-tuning for diffusion models," 2024, *arXiv:2405.21050*.

[50] J. Zhu, H. Ma, J. Chen, and J. Yuan, "Domainstudio: Fine-tuning diffusion models for domain-driven image generation using limited data," 2024, *arXiv:2306.14153*.

[51] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 8780–8794, [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf

[52] C.H. McCollough et al., "Results of the 2016 low dose ct grand challenge," *Med. Phys.*, vol. 44, no. 10, pp. e339–e352, Oct. 2017.

[53] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2015, pp. 3730–3738.

[54] Z. Wang, A. C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[55] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the design space of diffusion-based generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 36, pp. 26565–26577,. [Online]. Available: https://openreview.net/pdf?id=k7FuTOWMOc7

[56] O.D. Team, "Odl: Operator discretization library," 2022. Accessed: Apr. 2024. [Online]. Available: https://odlgroup.github.io/odl/guide/geometry_guide.html

[57] X. Xu, J. Liu, Y. Sun, B. Wohlberg, and U.S. Kamilov, "Boosting the performance of plug-and-play priors via denoiser scaling," in *Proc. 54th Asilomar Conf. Signals, Systems, Comput.*, 2020, pp. 1305–1312.

[58] Y. Hu, J. Liu, X. Xu, and U.S. Kamilov, "Monotonically convergent regularization by denoising," in *Proc. IEEE Int. Conf. Image Process.*, 2022, pp. 426–430.

[59] T. Hong, L. Hernandez-Garcia, and J.A. Fessler, "A complex quasi-Newton proximal method for image reconstruction in compressed sensing MRI," *IEEE Trans. Comput. Imag.*, vol. 10, pp. 372–384, 2024.

[60] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4217–4228, Dec. 2021.

[61] S.G. Armato, G. McLennan, L. Bidaut, M.F. McNitt-Gray, and C.R. Meyer, "The lung image database consortium (LIDC) and image database resource initiative (IDRI): A completed reference database of lung nodules on CT scans," *Med. Phys.*, vol. 38, pp. 915–931, 2011.

[62] B. Yaman, S.A.H. Hosseini, S. Moeller, J. Ellermann, K. Ugurbil, and M. Akcakaya, "Self-supervised learning of physics-based reconstruction neural networks without fully-sampled reference data," *Mag. Res. Med.*, vol. 84, no. 6, pp. 3172–91, Dec. 2020.

SUPPLEMENT

Test-Time Adaptation Improves Inverse Problem Solving with Patch-Based Diffusion Models

A. Ablation studies

We performed four ablation studies to evaluate the impact of various parameters on the proposed methods. Similar to the main text, all quantitative results are averaged across the test dataset.

Low-rank adaptation. To avoid overfitting to the measurement when using the self-supervised loss, [17] proposed using a low-rank adaptation to the weights of the neural network, reducing the number of weights that are adjusted during reconstruction by a factor of around 100. Here we investigate the effect of using different ranks of adaptations on two inverse problems: 60 view CT reconstruction and deblurring. Consistent with [17] and [18], we only used the LoRA module for attention and convolution layers. We also allowed the biases of the network to be changed.

Tables S.1 and S.2 show the quantitative results of these experiments, where a rank of "full" represents fine-tuning all the weights of the network. In all cases, using LoRA for this fine-tuning process resulted in worse reconstructions than simply fine-tuning the entire network. The visual results are especially apparent in Figure S.2: the reconstructed image became oversmoothed when using LoRA and artifacts became present when using the whole-image model. This is likely due to the large distribution shift between the initial distribution of images and target distribution of faces: the low-rank adaptation of the mismatched network is not sufficient to represent the new distribution and thus the self-supervised loss function results in smoothed images.

TABLE S.1: Performance of 60 view CT recon using self-supervised network refining with LoRA module. Best results are in bold.

Rank	Parameters (%)	Patches		Whole image	
		PSNR↑	SSIM ↑	PSNR↑	SSIM ↑
2	1.1	40.37	0.963	39.25	0.952
4	2.0	40.32	0.963	39.10	0.951
8	3.8	40.33	0.963	39.18	0.951
16	7.2	40.32	0.963	39.33	0.953
Full	100	**41.45**	**0.966**	**40.47**	**0.957**

TABLE S.2: Performance of deblurring using self-supervised network refining with LoRA module. Best results are in bold.

Rank	Parameters (%)	Patches		Whole image	
		PSNR↑	SSIM ↑	PSNR↑	SSIM ↑
2	1.1	29.31	0.830	29.19	0.811
4	2.0	29.31	0.829	29.35	0.817
8	3.8	29.38	0.831	29.19	0.810
16	7.2	29.31	0.830	29.33	0.815
Full	100	**30.34**	**0.860**	**29.50**	**0.831**

Effect of network size. When applying the self-supervised loss, another potential method to avoid overfitting is to use a smaller network. We trained networks with differing numbers of base channels (but no other modifications) on the ellipse phantom dataset and then used Algorithm 1 to perform 60-view CT reconstruction with test-time adaptation. Table S.3 shows the quantitative results of this experiment. For both the patch-based model and the whole image model, the network with 128 base channels obtained the best result, so we used this network architecture for all the main experiments. Figure S.3 again shows evidence of overfitting in the form of artifacts in the otherwise smooth regions of the organs when using the network with 256 base channels. These artifacts are less obvious in the patch-based model.

Fine-tuning with a larger dataset. To examine the effect of fine-tuning the networks on differing sizes of in-distribution datasets, we started with the same checkpoint trained on ellipses and fine-tuned them using various sizes of datasets consisting of CT images. Each small dataset consisted of randomly selected images from the entire 5000 image AAPM dataset. Next we used these checkpoints to perform 60 view CT reconstruction (without the self supervised loss). Table S.4 shows the results of these experiments, where we also included the results of using the in-distribution network trained on the entire 5000 image

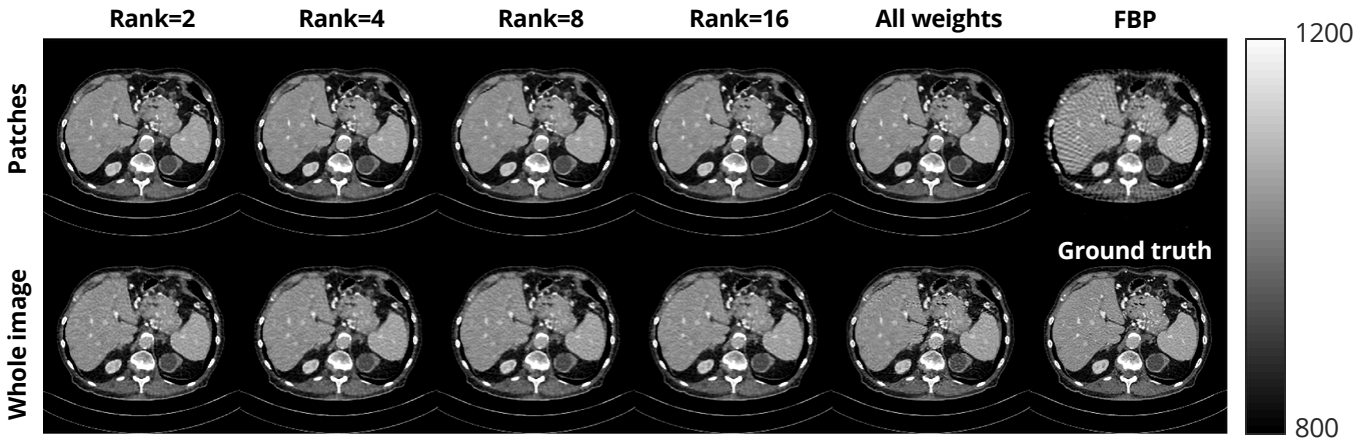


Fig. S.1: Results of using LoRA module for 60 view CT reconstruction in a single measurement setting. All weights refers to adjusting all the weights of the network at reconstruction time.

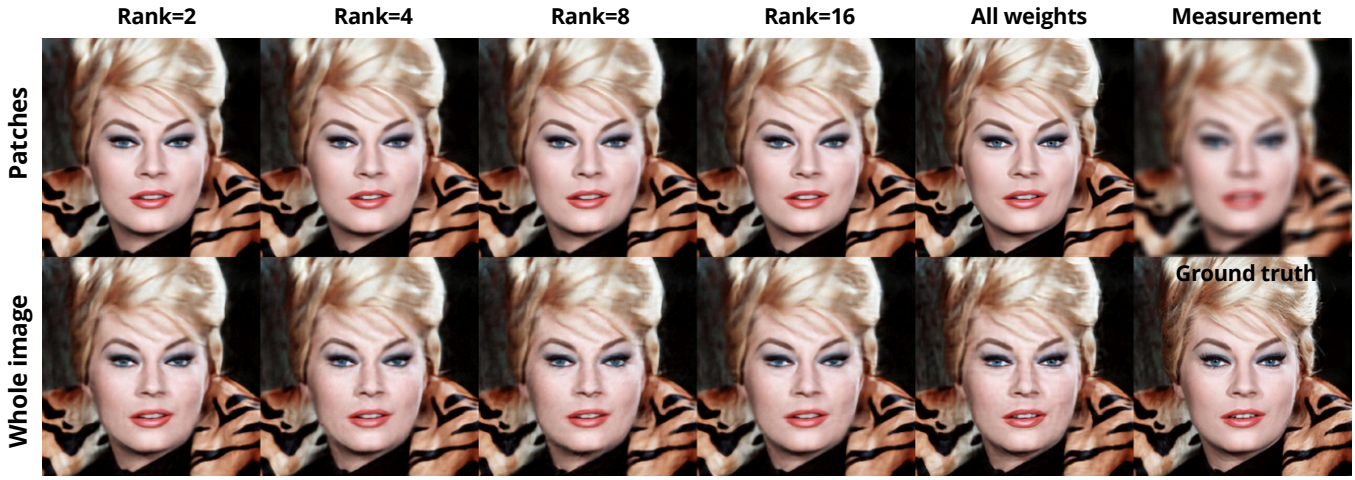


Fig. S.2: Results of using LoRA module for deblurring in a single measurement setting. All weights refers to adjusting all the weights of the network at reconstruction time.

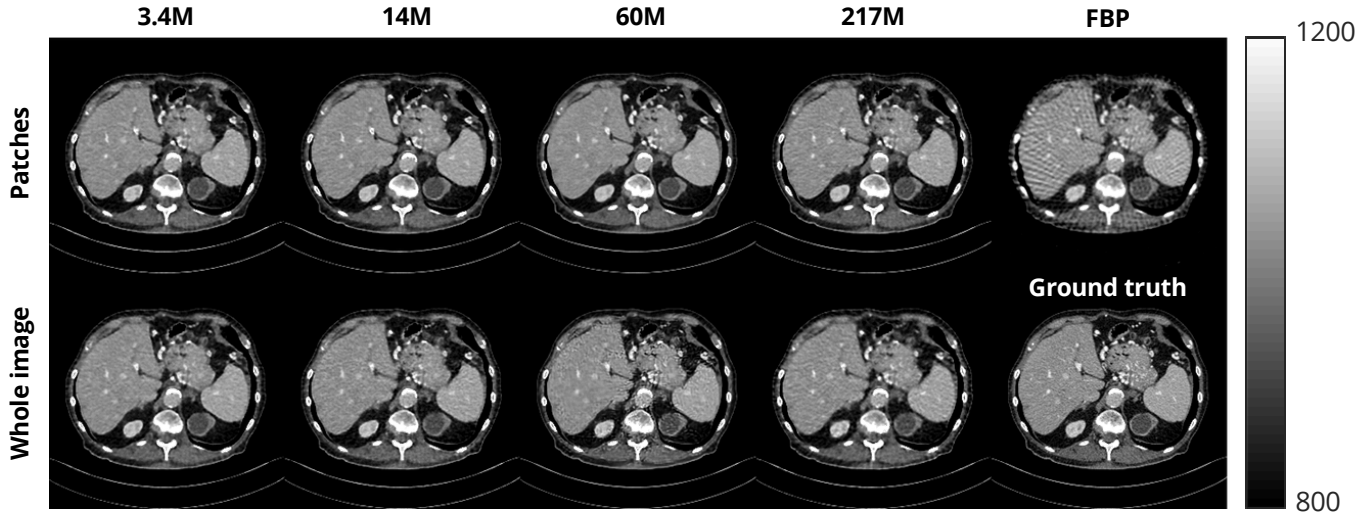


Fig. S.3: Results of 60 view CT recon using networks with different numbers of parameters in the single-measurement setting. The top numbers show the number of total parameters in the network.

TABLE S.3: Performance of 60 view CT recon using test-time adaptation with networks of different sizes. Best results are in bold.

Base Channels	Parameters (Millions)	Patches PSNR↑	Patches SSIM ↑	Whole image PSNR↑	Whole image SSIM ↑
32	3.4	39.73	0.958	39.69	0.957
64	14	40.37	0.961	40.07	**0.958**
128	60	**41.45**	**0.966**	**40.47**	0.957
256	217	40.29	0.959	39.28	0.954

dataset. This shows that for a wide range of different fine-tuning dataset sizes our proposed method obtained better metrics than the whole-image model.

We emphasize the difference between the results of [21], which showed that patch-based models outperform whole image models in cases of limited data, and the results here. Since the networks in [21] were trained from scratch, more data was required: the smallest datasets used in [21] contained 144 images. In constrast, we are able to fine-tune networks in our work using only 10 images. Consequently, the training time is also much lower for our work: Figure 6 shows that we fine-tuned a patch-based model in only about 2 hours, whereas [21] required 12-24 hours to train the patch-based models from scratch. Thus, our results complement the work of [21] by showing that, compared to whole-image models, patch-based diffusion models easier to train from scratch in settings of limited data, and they are also easier to fine-tune when data is very limited.

TABLE S.4: Performance of fine-tuning on 60 view CT using checkpoints fine-tuned from different dataset sizes. Best results are in bold.

Dataset size	Patches PSNR↑	Patches SSIM ↑	Whole image PSNR↑	Whole image SSIM ↑
3	40.93	0.964	40.45	0.964
10	41.21	0.965	40.54	0.964
30	41.31	0.966	40.66	0.967
100	41.46	0.967	40.96	0.968
5000*	**41.70**	**0.967**	**41.67**	**0.969**

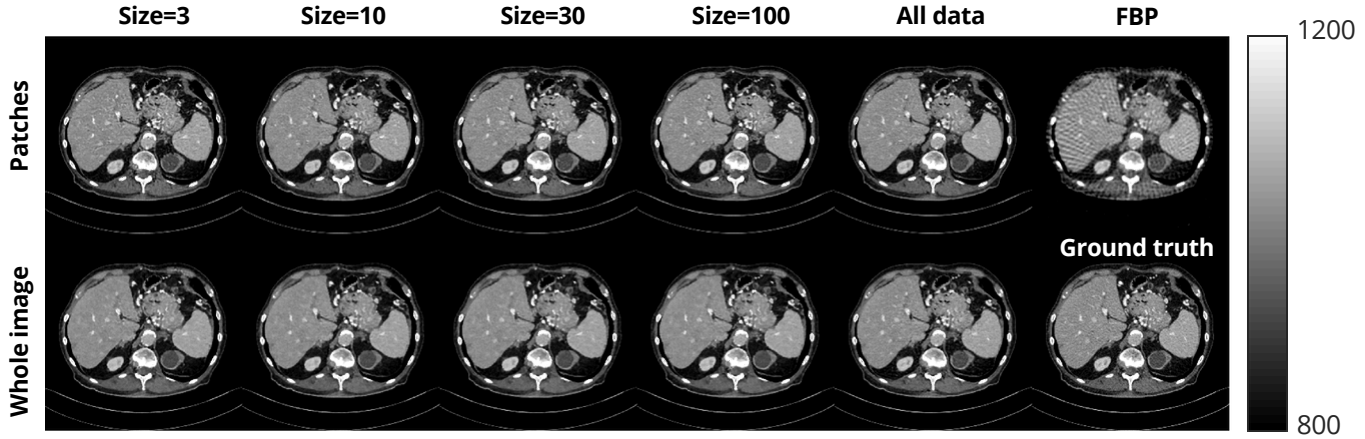


Fig. S.4: Results of 60 view CT recon in the small dataset setting where the size of the small dataset is varied.

Backpropagation iterations for the self-supervised loss. In the single measurement setting, the self-supervised loss is crucial to ensuring that the OOD network output is consistent with the measurement. Backpropagation through the network is necessary to minimize this loss, but too much network refining during this step could lead to overfitting to the measurement and image degradation. We ran experiments examining the effect of the number of backpropagation iterations during each step for the patch-based model and the whole image model. Figures S.5 and S.6 show that in both cases, performance generally improved when increasing the number of backpropagation iterations and overfitting is avoided. Additionally, the patch-based model always outperformed the whole image model and exhibited more improvement as the number of backpropagation iterations

increased. For our main experiments, we used 5 iterations as the improved performance became marginal compared to the extra runtime.

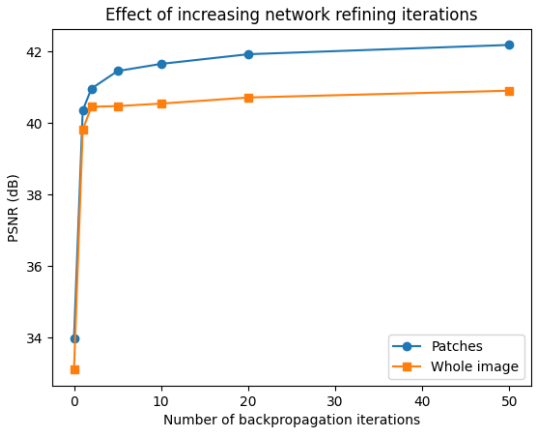


Fig. S.5: Comparison of PSNR between patch-based model and whole-image model for number of network refining iterations in single measurement setting.

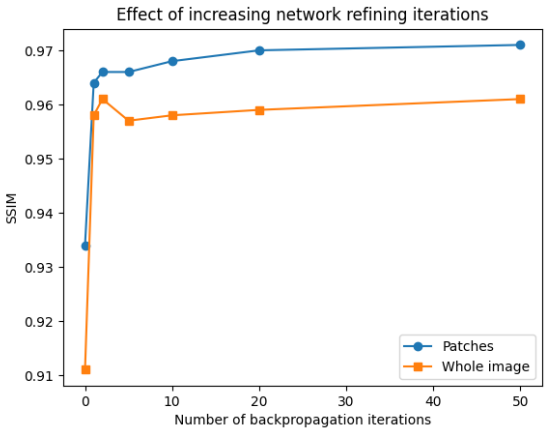


Fig. S.6: Comparison of SSIM between patch-based model and whole-image model for number of network refining iterations in single measurement setting.

TABLE S.5: Performance of Algorithm 1 for 60 view CT reconstruction in single measurement setting with different numbers of backpropagation iterations. Best results are in bold.

Backprop iterations	Patches		Whole image	
	PSNR↑	SSIM ↑	PSNR↑	SSIM ↑
0	33.97	0.934	33.10	0.911
1	40.35	0.964	39.81	0.958
2	40.96	0.966	40.45	0.961
5	41.45	0.966	40.47	0.957
10	41.65	0.968	40.54	0.958
20	41.92	0.970	40.71	0.959
50	**42.18**	**0.971**	**40.90**	**0.961**

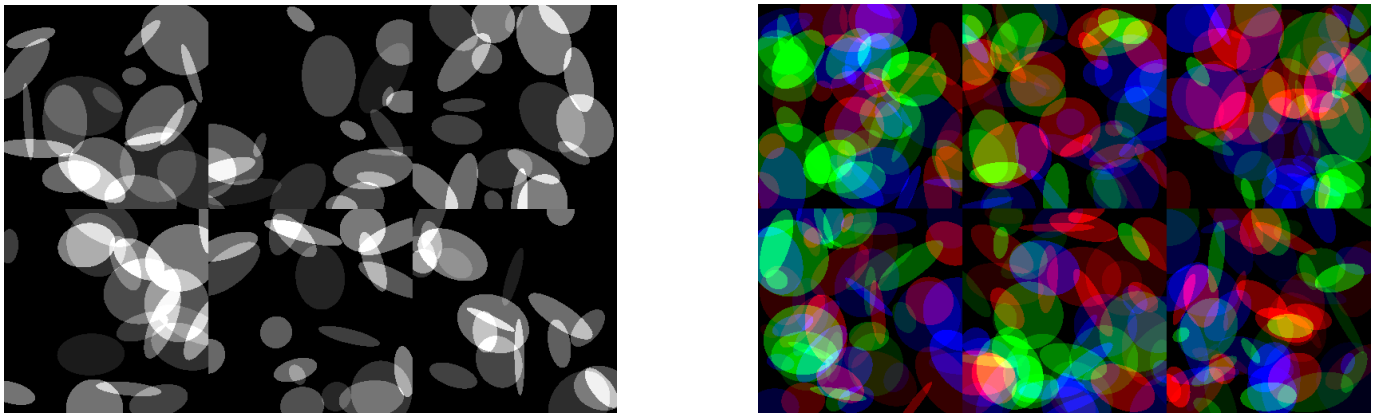
B. Phantom dataset details

We used two phantom datasets of 10000 images each: one consisting of grayscale phantoms and the other consisting of colored phantoms. The grayscale phantoms consisted of 20 ellipses with a random center within the image, each with minor and major axis having length equal to a random number chosen between 2 and 20 percent of the width of the image. The grayscale value of each ellipse was randomly chosen between 0.1 and 0.5; if two or more ellipses overlapped, the grayscale values were summed for the overlapped area with all values exceeding 1 set to 1. Finally, all ellipses were set to a random angle of rotation. The colored phantoms were generated in the same way, except the RGB values for each ellipse were set independently and then multiplied by 255 at the end. Figure S.7 shows some of the sample phantoms.

C. Experiment parameters

We applied the framework of [55] to train the patch-based networks and whole image networks. Since images were scaled between 0 and 1 for both grayscale images and RGB channels, we chose a maximum noise level of $\sigma = 40$ and a minimum noise level of $\sigma = 0.002$ for training. We used the same UNet architecture for all the networks consisting of a base channel multiplier size of 128 and 2, 2, and 2 channels per resolution for the three layers. We also used dropout connections with a probability of 0.05 and exponential moving average for weight decay with a half life of 500K images to avoid overfitting.

The learning rate was chosen to be $2{\cdot}10^{-4}$ when training networks from scratch and was $1{\cdot}10^{-4}$ for the fine-tuning experiments. For the patch-based networks, the batch size for the main patch size (64×64) was 128, although batch sizes of 256 and 512



(a) Six grayscale phantoms

(b) Six colored phantoms

Fig. S.7: Six sample grayscale phantoms and colored phantoms used to train the mismatched distribution diffusion models

were used for the two smaller patch sizes of 32×32 and 16×16. The probabilities of using these three patch sizes were $0.5, 0.3$, and 0.2 respectively. For the whole image model, we kept all the parameters the same, but used a batch size of 8.

For image generation and inverse problem solving, we used a geometrically spaced descending noise level that was fine tuned to optimize the performance for each type of problem. We used the same set of parameters for the patch-based model and whole image model. The values without the self-supervised loss are as follows:

- CT with 20 and 60 views: $\sigma_{\max} = 10, \sigma_{\min} = 0.005$
- Deblurring: $\sigma_{\max} = 40, \sigma_{\min} = 0.005$
- Superresolution: $\sigma_{\max} = 40, \sigma_{\min} = 0.01$.

The values with the self-supervised loss are as follows:

- CT with 20 and 60 views: $\sigma_{\max} = 10, \sigma_{\min} = 0.01$
- Deblurring: $\sigma_{\max} = 1, \sigma_{\min} = 0.01$
- Superresolution: $\sigma_{\max} = 1, \sigma_{\min} = 0.01$.

Finally, for generating the CT images we used $\sigma_{\max} = 40, \sigma_{\min} = 0.005$.

When running Algorithm 1, we set $K = 10$ for all experiments and $M = 5$ for CT reconstruction and $M = 1$ for deblurring and superresolution. We ran 5 iterations of network backpropagation with a learning rate of 10^{-5}. When using the LoRA module as in the ablation studies (see Tables S.2 and S.1), we ran 10 iterations of network backpropagation with a learning rate of 10^{-3}.

The ADMM-TV method for linear inverse problems consists of solving the optimization problem

$$\text{argmax}_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - A\boldsymbol{x}\|_2^2 + \lambda \,\text{TV}(\boldsymbol{x}), \tag{S.1}$$

where $\text{TV}(\boldsymbol{x})$ represents the L1 norm total variation of vx, and the problem is solved with the alternating direction method of multipliers. For CT reconstruction, deblurring, and superresolution, we chose λ to be $0.001, 0.002$, and 0.006 respectively.

The PnP-ADMM method consists of solving the intermediate optimization problem

$$\text{argmax}_{\boldsymbol{x}} f(\boldsymbol{x}) + (\rho/2)\|\boldsymbol{x} - (\boldsymbol{z} - \boldsymbol{u})\|_2^2, \tag{S.2}$$

where ρ is a constant. The values for ρ we used for CT reconstruction, deblurring, and superresolution were 0.05, 0.1, and 0.1 respectively. We used BM3D as the denoiser with a parameter representing the noise level: this parameter was set to 0.02 for 60 view CT and 0.05 for the other inverse problems. A maximum of 50 iterations of conjugate gradient descent was run per outer loop. The entire algorithm was run for 100 outer iterations at maximum and the PSNR was observed to decrease by less than 0.005dB per iteration by the end.

The PnP-RED method consists of the update step

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \mu\left(\nabla f - \lambda(\boldsymbol{x} - D(\boldsymbol{x}))\right), \tag{S.3}$$

where $D(\boldsymbol{x})$ represents a denoiser. The stepsize μ was set to 0.01 for the CT experiments and 1 for deblurring and superresolution. We set λ to 0.01 for the CT experiments and 0.2 for deblurring and superresolution. Finally, the denoiser was kept the same as the PnP-ADMM experiments with the same denoising strength.

Table S.6 shows the average runtimes of each of the implemented methods when averaged across the test dataset for 60 view CT reconstruction.

TABLE S.6: Average runtimes of different methods across images in the test dataset for 60 view CT recon.

Method	Runtime (s) $\downarrow$
Baseline	0.1
ADMM-TV	1
PnP-ADMM	73
PnP-RED	121
Whole diffusion	112
Whole SS	248
Whole LoRA	329
Patch diffusion	123
Patch SS	289
Patch LoRA	377