Composing Normalizing Flows for Inverse Problems

Jay Whang [1] Erik M. Lindgren [2] Alexandros G. Dimakis [3]

Abstract

Given an inverse problem with a normalizing flow prior, we wish to estimate the distribution of the underlying signal conditioned on the observations. We approach this problem as a task of conditional inference on the pre-trained unconditional flow model. We first establish that this is computationally hard for a large class of flow models. Motivated by this, we propose a framework for approximate inference that estimates the target conditional as a composition of two flow models. This formulation leads to a stable variational inference training procedure that avoids adversarial training. Our method is evaluated on a variety of inverse problems and is shown to produce high-quality samples with uncertainty quantification. We further demonstrate that our approach can be amortized for zero-shot inference.

1. Introduction

We are interested in solving inverse problems using a pre-trained normalizing flow prior. Inverse problems encompass a variety of tasks such as image inpainting, super-resolution and compressed sensing from linear projections. Due to this generality, the applications range from scientific and medical imaging to computational photography (Ongie et al., 2020). Inverse problems can be solved by either supervised (Pathak et al., 2016; Richardson et al., 2020; Yu et al., 2018) or unsupervised (Menon et al., 2020; Bora et al., 2017; Pajot et al., 2019) methods, see the recent survey (Ongie et al., 2020) for a unified presentation.

In this paper we focus on unsupervised image reconstruction techniques that benefit from a pre-trained deep generative prior, specifically normalizing flows. Flow models (Papamakarios et al., 2019) are a family of generative models that provide efficient sampling, likelihood evaluation, and inversion. While other types of models can outperform flow models in terms of likelihood or sample quality, flow models are often simpler to train and evaluate compared to other models.

These characteristics make normalizing flows attractive for numerous downstream tasks, including density estimation, inverse problems, semi-supervised learning, reinforcement learning, and audio synthesis (Ho et al., 2019; Asim et al., 2019; Whang et al., 2020; Atanov et al., 2019; Ward et al., 2019; Oord et al., 2018).

Even with such computational flexibility, how to perform efficient probabilistic inference on a flow model subject to observations obtained from an inverse problem remains challenging. This question is becoming increasingly important as flow models increase in size, and the computational resources necessary to train them from scratch are out of reach for many researchers and practitioners[1]. Our goal is to *re-purpose* these powerful pre-trained models for different custom inverse problems without re-training them from scratch.

Concretely, we wish to recover the distribution of the unknown image $\boldsymbol{x}$ from the observed measurements $\boldsymbol{y}^* = A(\boldsymbol{x}) +$ noise. We assume that a pre-trained flow model $p(\boldsymbol{x})$ serves as the prior for natural images we are sensing, and that the measurement function $A(\cdot)$ (also known as forward operator) is differentiable. Thus the goal is to estimate the following conditional distribution as accurately as possible:

$$p(\boldsymbol{x} \mid A(\boldsymbol{x}) = \boldsymbol{y}^*).$$

We propose a novel formulation that *composes* a new flow model with the pre-trained prior $p(\boldsymbol{x})$ to estimate the conditional distribution given observations $\boldsymbol{y}^*$. While such a composed model is in general intractable to train for latent variable models, the invertibility of the given prior leads to a tractable and stable training procedure via variational inference (VI).

Our contributions:

- We show that even though flow models are designed to

[1] Dept. of Computer Science, UT Austin, TX, USA [2] Google Research, NY, USA [3] Dept. of Electrical and Computer Engineering, UT Austin, TX, USA. Correspondence to: Jay Whang <jaywhang@utexas.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

[1] For example, Kingma & Dhariwal (2018) report that their largest model had 200M parameters and was trained on 40 GPUs for a week.

provide efficient inversion and sampling, even *approximately* sampling from the exact conditional distribution is computationally intractable for a wide class of flow models. Motivated by this, we consider the relaxation that allows approximate *conditioning*.

- We propose to estimate the relaxed target conditional distribution by composing a new flow model (the *pre-generator*) with the base model. This formulation leads to a variational inference training procedure that avoids the need for unstable adversarial training as explored in existing work (Engel et al., 2017).
- Because our method recovers the conditional distribution over $\boldsymbol{x}$ as another flow model, we can use it to efficiently generate conditional samples and evaluate conditional likelihood. This is in contrast to prior work that uses flow models for inverse problems (Asim et al., 2019), since we can obtain confidence bounds for each reconstructed pixel beyond point estimates.
- We show that our approach is comparable to MCMC baselines in terms of sample quality metrics such as Frechet Inception Distance (FID) (Heusel et al., 2017). We also qualitatively demonstrate its flexibility on various complex inference tasks with applications to inverse problems.
- We show that the pre-generator can be *amortized* over the observations to perform zero-shot inference without much degradation in sample quality.

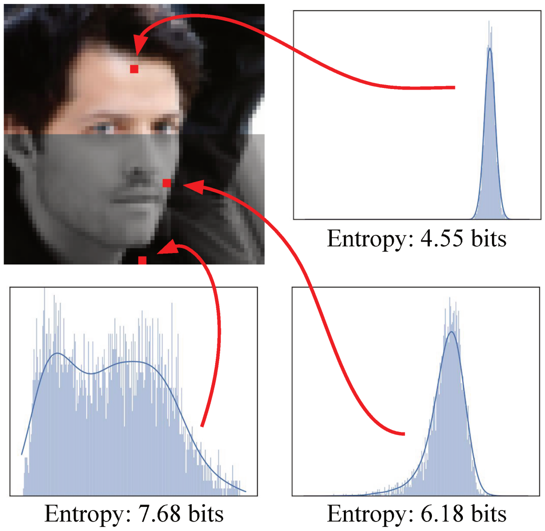

Figure 1: Uncertainty quantification highlighted at different pixel locations, obtained from our learned approximate posterior. The top pixel is observed, and thus is sharply concentrated on a single value (the small variance is due to our use of smoothing). The bimodal plot distribution in the bottom left captures the semantic ambiguity of the bottom pixel that can be part of either the neck or the background.

2. Background

2.1. Normalizing Flows

Normalizing flow models represent complex probability distributions by transforming a simple input noise $\boldsymbol{z}$ (typically standard Gaussian) through a differentiable bijection $f : \mathbb{R}^d \to \mathbb{R}^d$. Since f is invertible, we can compute the probability density of $\boldsymbol{x} = f(\boldsymbol{z})$ via the change of variables formula: $\log p(\boldsymbol{x}) = \log p(\boldsymbol{z}) + \log \left|\det \frac{df^{-1}}{d\boldsymbol{x}}(\boldsymbol{x})\right|$.

Flow models are explicitly designed so that this expression can be easily computed. This allows them to be directly trained with maximum likelihood objective on data and avoids issues such as posterior and mode collapse that plague other deep generative models.

Starting from the early works of Dinh et al. (2015) and Rezende & Mohamed (2015), there has been extensive research on invertible neural network architectures for normalizing flow. Many of them work by composing a series of invertible layers, such as in RealNVP (Dinh et al., 2016), IAF (Kingma et al., 2016), Glow (Kingma & Dhariwal, 2018), invertible ResNet (Behrmann et al., 2019), and Neural Spline Flows (Durkan et al., 2019).

One of the simplest invertible layer construction is *additive coupling layer* introduced by Dinh et al. (2015), which served as the basis for many other subsequently proposed models. In an additive coupling layer, the input variable is partitioned as $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$. The layer is parametrized by a neural network $g(\boldsymbol{x}_1) : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ used to additively transform $\boldsymbol{x}_2$. Thus the layer's output $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$ and its inverse can be computed as follows:

$$\begin{cases} \boldsymbol{y}_1 &= \boldsymbol{x}_1 \\ \boldsymbol{y}_2 &= \boldsymbol{x}_2 + g(\boldsymbol{x}_1) \end{cases} \iff \begin{cases} \boldsymbol{x}_1 &= \boldsymbol{y}_1 \\ \boldsymbol{x}_2 &= \boldsymbol{y}_2 - g(\boldsymbol{y}_1) \end{cases}$$

Notably, the determinant of the Jacobian of this transformation is always 1 for any mapping g.

2.2. Variational Inference

Variational inference (VI) is a family of techniques for estimating the conditional distribution of unobserved variables given the observed ones (Jordan et al., 1999; Wainwright et al., 2008; Blei et al., 2017). At its core, VI tries to find a tractable approximation of the intractable target posterior by solving a KL minimization problem.

Within our context of conditional inference on a joint distribution $p(\boldsymbol{x})$, we minimize the following stochastic variational inference (SVI) objective:

$$\min_{q \in \mathcal{Q}} D_{\mathrm{KL}}(q(\boldsymbol{x}) \parallel p(\boldsymbol{x} \mid \boldsymbol{y} = \boldsymbol{y}^*)), \tag{1}$$

where $\boldsymbol{y} \triangleq A(\boldsymbol{x})$ is the measurement (also called observation) that is being conditioned on, and $\boldsymbol{y}^*$ is the given realization of $\boldsymbol{y}$. The variational family $\mathcal{Q}$ must be appropriately chosen to allow efficient sampling and likelihood evaluation for all $q \in \mathcal{Q}$. Note that q is specific to the particular value of $\boldsymbol{y}^*$.

The variational posterior q can also be *amortized* over the observation (Kingma & Welling, 2013), leading to a single model trained to minimize the following amortized variational inference (AVI) objective:

$$\min_{q \in \mathcal{Q}} \mathbb{E}_{\boldsymbol{y}} \left[D_{\mathrm{KL}}(q(\boldsymbol{x} \mid \boldsymbol{y}) \parallel p(\boldsymbol{x} \mid \boldsymbol{y})) \right], \tag{2}$$

An amortized posterior has the advantage that it only needs to be trained once for all $\boldsymbol{y}$, but it generally achieves worse likelihood than SVI and often requires a more complex model (Cremer et al., 2018).

3. Hardness of Conditional Sampling

Before we present our method, we first establish a hardness result for conditional sampling for flow models. Specifically, if an algorithm is able to efficiently sample from the conditional distribution of a flow model with additive coupling layers (Dinh et al., 2015), then it can be used to solve NP-complete problems efficiently. The formal statement and the proof of the theorem can be found in Appendix A.

Theorem 1. *(Informal) Suppose we are given a flow model with additive coupling layers and wish to condition on a subset of the input dimensions. If there is an efficient algorithm that can sample from this conditional distribution, then $RP = NP$. Further, this problem remains hard even if we allow sampling to be approximate.*

Importantly, this result shows that allowing approximate *sampling* from the exact posterior does not affect the hardness of the problem, as long as we require that the conditioning is exact. Thus we are motivated to consider approximate *conditioning*, where the conditioned variable is allowed to deviate from the given observation.

We also note that flow architectures that include additive coupling layers make up a majority of existing models (e.g. most of the models in Section 2.1). Thus our hardness result applies to a variety of flow models used in practice.

4. Approximate Conditional Inference with Composed Flows

Notation. Let $p_{\boldsymbol{x}}(\boldsymbol{x})$ be the pre-trained *base model* that serves as the signal prior, parametrized by the invertible mapping $f : \boldsymbol{z} \mapsto \boldsymbol{x}$. $A(\boldsymbol{x})$ is the differentiable measurement function. We similarly define the *pre-generator* $q_{\boldsymbol{z}}(\boldsymbol{z})$ parametrized by the mapping $\hat{f} : \boldsymbol{\epsilon} \mapsto \boldsymbol{z}$, which represents a distribution in the latent space of the base model. We assume that all flow models use the standard Gaussian prior, i.e. $p_{\boldsymbol{z}}(\boldsymbol{z})$ and $q_{\boldsymbol{\epsilon}}(\boldsymbol{\epsilon})$ are $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

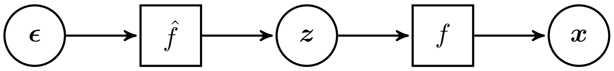

Figure 2: A flow chart of our conditional sampler. Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is mapped through the composition of our pre-generator $\hat{f}$ and the base model f to generate conditional samples.

By composing the base model and the pre-generator, we obtain the *composed model*, denoted $q_{\boldsymbol{x}}(\boldsymbol{x})$, whose samples are generated via $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \to \boldsymbol{x} = f(\hat{f}(\boldsymbol{\epsilon}))$. Figure 2 details this sampling procedure.

VI objective and smoothing.

Since our composed model $q_{\boldsymbol{x}}$ is the composition of two flow models, the VI objective in eq. (1) can be simplified further (see Appendix B.1 for derivation):

$$\min_{\hat{f}} D_{\mathrm{KL}}(q_{\boldsymbol{z}} \parallel p_{\boldsymbol{z}}) + \mathbb{E}_{q_{\boldsymbol{z}}} \left[-\log p(\boldsymbol{y} = \boldsymbol{y}^* \mid \boldsymbol{z}) \right] \tag{3}$$

Unfortunately this loss is challenging to optimize in practice when using a flow-based variational posterior. Because $\boldsymbol{y} = A(f(\boldsymbol{z}))$ is a deterministic function of $\boldsymbol{z}$, the density in the second term is zero for any $\boldsymbol{z}$ that fails to match $\boldsymbol{y}^*$ exactly. Since our pre-generator $q_{\boldsymbol{z}}$ is a flow model defined by an invertible mapping $\hat{f}$ and has full support, it would inadvertently have nonzero probability mass on invalid values of $\boldsymbol{z}$ and cause the loss to be infinity.

One simple solution to this issue is *smoothing* the observation, which turns the condition $\boldsymbol{y} = \boldsymbol{y}^*$ into a soft constraint. Notice that this is in line with the hardness result in Section 3, where we motivated the need for approximate conditioning. In the context of inverse problems, smoothing can also be viewed as the distribution for observation noise.

Concretely, we define a new random variable $\tilde{\boldsymbol{y}}$ that is allowed to deviate from $\boldsymbol{y}$ but penalized for the deviation. While there are many choices for the distribution $p(\tilde{\boldsymbol{y}} \mid \boldsymbol{y})$, we consider the following scheme. For any symmetric distance measure $d(\cdot, \cdot)$ with $d(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = 0$ iff $\boldsymbol{y} = \tilde{\boldsymbol{y}}$, we use the distribution defined by $p(\tilde{\boldsymbol{y}} \mid \boldsymbol{y}) \propto \exp(-\beta \cdot d(\tilde{\boldsymbol{y}}, \boldsymbol{y}))$. Note that we do not need to compute the normalization constant as it is constant w.r.t. $\hat{f}$, which we optimize.

This formulation includes a wide range of options for smoothing. For example, choosing ℓ_2 distance and $\beta = 1/(2\sigma^2)$ is equivalent to smoothing with Gaussian kernel $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, which leads to the following objective:

$$\begin{aligned} \mathcal{L}_{\text{ours}}(\hat{f}) &= D_{\mathrm{KL}}(q_{\boldsymbol{x}} \parallel p_{\boldsymbol{x}}(\cdot \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)) \\ &= D_{\mathrm{KL}}(q_{\boldsymbol{z}} \parallel p_{\boldsymbol{z}}) + \mathbb{E}_{q_{\boldsymbol{z}}} \left[\frac{1}{2\sigma^2} \left\| A(f(\boldsymbol{z})) - \boldsymbol{y}^* \right\|_2^2 \right] \end{aligned} \tag{4}$$

This loss function offers an intuitive interpretation. The first term tries to keep the learned distribution $q_{\boldsymbol{x}}$ close to the base distribution by pushing $q_{\boldsymbol{z}}$ to match the prior of the base model, while the second term tries to match the observation $\boldsymbol{y}^*$. This is analogous to the KL/reconstruction loss decomposition typically used in the VAE literature.

We could also choose to use a more sophisticated distance measure such as LPIPS (Zhang et al., 2018). Interestingly, our preliminary experiments showed no benefit in sample quality when using LPIPS, so we ran our experiments with Gaussian smoothing for simplicity. We leave a detailed study on the effect of different smoothing techniques for future work.

Bounding the marginal objective. An important related task is estimating the marginal distribution after conditioning. In other words, can we estimate $p(\boldsymbol{x}_2 \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)$ for some partitioning of the input $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2)$? This includes tasks such as data imputation, e.g. estimating $p(\boldsymbol{x}_2 \mid \boldsymbol{x}_1)$.

In our setup, computing $p(\boldsymbol{x}_2 \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)$ is in general intractable because we only have access to the joint distribution $p_{\boldsymbol{x}}(\boldsymbol{x}_1, \boldsymbol{x}_2)$ through the base model. Fortunately, our VI loss for the joint conditional $p_{\boldsymbol{x}}(\boldsymbol{x} \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)$ provides an upper bound (derivation in Appendix B.2):

$$\begin{aligned}\text{(Joint KL)} &= D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}) \parallel p_{\boldsymbol{x}}(\boldsymbol{x} \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)) \\ &\geq D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}_2) \parallel p_{\boldsymbol{x}}(\boldsymbol{x}_2 \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)).\end{aligned}$$

Thus we are justified in our use of eq. (4) in place of the intractable marginal KL.

Benefits of solving inverse problems distributionally. Here we explain a key benefit of recovering the conditional distribution instead of just a point estimate. Given the observation $\boldsymbol{y}^*$ generated from the underlying signal $\boldsymbol{x}^*$, suppose we wish to recover $\boldsymbol{x}^*$ with respect to the ℓ_2 loss. Then the optimal recovery function is the minimum mean square error (MMSE) estimator $\hat{\boldsymbol{x}}_{\text{MMSE}}(\boldsymbol{y}^*) = \arg\min_{\hat{\boldsymbol{x}}} \|\boldsymbol{x}^* - \hat{\boldsymbol{x}}(\boldsymbol{y}^*)\|_2^2$. Under a mild assumption, the MMSE estimator is known to be the conditional expectation: $\hat{\boldsymbol{x}}_{\text{MMSE}}(\boldsymbol{y}^*) = \mathbb{E}\left[\boldsymbol{x} \mid \boldsymbol{y}^*\right].$

Importantly, this is different from the objective employed by existing methods that produce point estimates. For example, Bora et al. (2017) minimize the reconstruction error based on a projection to the range of a GAN:

$$\hat{\boldsymbol{x}}_{\text{bora}}(\boldsymbol{y}^*) = \operatorname*{arg\,min}_{\boldsymbol{x} \in \text{range}(G)} \|A(\boldsymbol{x}) - \boldsymbol{y}^*\|_2^2,$$

and Asim et al. (2019) use an objective loosely based on a MAP estimate:

$$\hat{\boldsymbol{x}}_{\text{asim}}(\boldsymbol{y}^*) = \operatorname*{arg\,max}_{\boldsymbol{x}} p(\boldsymbol{x} \mid \boldsymbol{y}^*).$$

The issue with these objectives is that, even if these optimizations could be done perfectly, they would not produce $\hat{\boldsymbol{x}}_{\text{MMSE}}(\boldsymbol{y}^*)$ and thus lead to suboptimal recovery with respect to the ℓ_2 loss.

Instead, our approach is to recover the entire conditional distribution $p(\boldsymbol{x}|\boldsymbol{y}^*)$ and use it to obtain a Monte Carlo estimate of the conditional mean $\mathbb{E}\left[\boldsymbol{x} \mid \boldsymbol{y}^*\right]$. While MCMC methods can also be used for this purpose, they often take prohibitively long due to slow mixing and may produce correlated samples. Our approximate posterior is explicitly parametrized as a flow and can efficiently generate i.i.d. samples. As we will see in our experiments later, this has a significant performance benefit compared to the existing approaches in terms of reconstruction error and the speed of inference.

5. Related Work

Conditional generative models. There has been a large amount of work on conditional generative modeling, with varying levels of flexibility for what can be conditioned on. In the simplest case, a fixed set of observed variables can be directly fed into the model as an auxiliary conditioning input (Mirza & Osindero, 2014; Sohn et al., 2015; Ardizzone et al., 2019). Some recent works proposed to extend existing models to support conditioning on *arbitrary* subsets of variables (Ivanov et al., 2018; Belghazi et al., 2019; Li et al., 2019). This is a much harder task as there are exponentially many subsets of variables that can be conditioned on.

More relevant to our setting is (Engel et al., 2017), which studied conditional sampling from *non-invertible* latent variable generators such as VAE and GAN. It proposes to adversarially train a pre-generator, thereby avoiding the issue of intractability of VI for non-invertible models. Due to the adversarial training and the lack of invertibility of the base model, however, the learned conditional sampler lacks the computational flexibility of a flow-based posterior, such as tractable likelihood computation and inversion. The key difference of our method is that by explicitly parametrizing the conditional generator to be invertible as a composition of two flow models, we avoid the need for adversarial training.

We highlight several reasons why one might prefer our approach over the above methods: (1) the training data for the base model is not available, and only the model itself is made public (2) the conditional posterior is too costly to train from scratch (3) we wish to perform downstream tasks that require exact likelihood or inversion (4) we want to get some insight on the distribution defined by the given model.

Markov Chain Monte Carlo methods. When one is only concerned with generating samples, MCMC techniques offer a promising alternative. Unlike VI using an approximate posterior, MCMC methods come with asymptotic guarantees to generate samples from the target posterior . Though directly applying MCMC methods on complex highdimensional posteriors parametrized by a neural network often comes with many challenges in practice (Papamarkou et al., 2019), methods based on Langevin Monte Carlo have recently shown promising results (Neal et al., 2011; Welling & Teh, 2011; Song & Ermon, 2019).

The idea of leveraging the favorable geometry of the latent space of a flow model is also applicable to MCMC methods. For example, Hoffman et al. (2019) utilized the latent space of a flow model to improve mixing of Hamiltonian Monte Carlo. More recently Cannella et al. (2020) proposed PL-MCMC, a Metropolis-Hastings based sampler with transition kernel also defined in the latent space of a pre-trained flow. A similar idea was later adapted by Nijkamp et al. (2020) in the context of training energy-based models.

Inverse problems with deep generative prior. In a linear inverse problem, a vector $\boldsymbol{x} \in \mathbb{R}^d$ generates a set of measurements $\boldsymbol{y}^* = A\boldsymbol{x} \in \mathbb{R}^m$, where the number of measurements is much smaller than the dimension: $m \ll d$. The goal is to reconstruct the vector $\boldsymbol{x}$ from $\boldsymbol{y}^*$. While in general there are (potentially infinitely) many possible values of $\boldsymbol{x}$ that agree with the given measurements, it is possible to identify a unique solution when there is an additional structural assumption on $\boldsymbol{x}$.

Classically, the simplifying structure was that $\boldsymbol{x}$ is sparse (Tibshirani, 1996; Candes et al., 2006; Donoho et al., 2006; Bickel et al., 2009; Baraniuk, 2007). Recent work has considered alternative structures, such as the vector $\boldsymbol{x}$ coming from a deep generative model. Starting with Bora et al. (2017), there has been extensive work studying various settings under different priors and inference techniques (Grover & Ermon, 2019; Mardani et al., 2018; Heckel & Hand, 2019; Mixon & Villar, 2018; Pandit et al., 2019; Lucas et al., 2018; Shah & Hegde, 2018; Liu & Scarlett, 2020; Kabkab et al., 2018; Mousavi et al., 2018; Raj et al., 2019; Sun et al., 2019). In particular, we note that Asim et al. (2019) utilize a flow-based prior similar to our setting.

It is important to note that the above approaches focus on recovering a single point estimate that best matches the measurements. However, there can be many inputs that fit the measurements and thus uncertainty in the reconstruction. Due to this shortcoming, several recent works focused on recovering the signal distribution conditioned on the measurements (Tonolini et al., 2019; Zhang & Jin, 2019; Adler & Öktem, 2018; 2019).

We note that our approach differs from these, since they are learning-based methods that require access to the training data. On the contrary, our work leverages a *pre-trained* prior to produce an approximate conditional posterior, which can then be used for a variety of tasks such as generating conditional samples or estimating the MMSE recovery.

6. Experiments

We validate the efficacy of our proposed method in terms of both sample and reconstruction quality against three baselines: Langevin Monte Carlo (LMC), Ambient VI, and PL-MCMC (Cannella et al., 2020). Both LMC and Pl-MCMC are MCMC techniques that can (asymptotically) sample from the true conditional distribution our method tries to approximate. For the comparisons to be fair, we implemented both methods to run MCMC chains in the latent space of the base model, analogous what our method does for VI. Ambient VI is identical to our method, except it performs VI in the image space and is included for completeness. In addition, we also conduct our experiments on three different datasets (MNIST, CIFAR-10, and CelebA-HQ) to ensure that our method works across a range of settings.

We report four different sample quality metrics: Frechet Inception Distance (FID), Learned Perceptual Image Patch Similarity (LPIPS), and Inception Score (IS) for CIFAR-10 (Heusel et al., 2017; Zhang et al., 2018; Salimans et al., 2016). While not strictly a measure of perceptual similarity, the average mean squared error (MSE) is reported for completeness. Additionally, we also report pairwise LPIPS metric used by Zheng et al. (2019) to measure the diversity of generated samples.

For all our experiments, we use the multiscale RealNVP architecture (Dinh et al., 2016) for both the base model and the pre-generator. We use Adam optimizer (Kingma & Ba, 2014) to optimize the weights of the pre-generator using the loss defined in eq. (4). The images used to generate observations were taken from the test set and were not used to train the base models. We refer the reader to Appendix C for model hyperparameters and other details of our experiment setup.

6.1. Image Inpainting

We perform inpainting tasks using our approach, where we sample missing pixels conditioned on the visible ones. We consider three different conditioning schemes: bottom half (MNIST), top half (CelebA-HQ), and randomly chosen subpixels (CIFAR-10). For MNIST, we use the smoothing parameter value of $\sigma = 0.1$ and for CIFAR-10 and CelebA-HQ, we use $\sigma = 0.05$.

In Section 6 we see that our approach produces natural and diverse samples for the missing part of the image. The empirical pixelwise variance (normalized and averaged over the color channels) also confirms that, while the observation is not perfectly matched, most of the high-variance regions are in the unobserved parts as expected.

We also quantitatively evaluate the quality of the generated samples using widely used sample quality metrics, as shown in Table 1. As we can see, our method outperforms the base-

Table 1: Sample quality metrics for image inpainting tasks on different datasets. The best value is bolded for each metric. As shown below, our method achieves superior sample quality to all baselines.

	MNIST			CIFAR-10 (5-bit)				CelebA-HQ (5-bit)			
	FID	MSE	LPIPS	FID	IS ↑	MSE	LPIPS	FID	MSE	LPIPS	Diversity ↑
Ours (SVI)	**5.15**	**22.13**	**0.076**	**45.01**	**7.14**	9.73	**0.177**	37.24	**219.7**	**0.207**	0.450 ± 0.086
LMC	14.56	36.47	0.135	47.53	6.73	**9.31**	0.201	**30.34**	323.5	0.229	0.479 ± 0.077
Ambient VI	123.6	59.99	0.282	87.50	5.14	16.59	0.295	295.0	2084	0.738	**0.586 ± 0.186**
PL-MCMC	21.20	59.89	0.190	N/A				N/A			

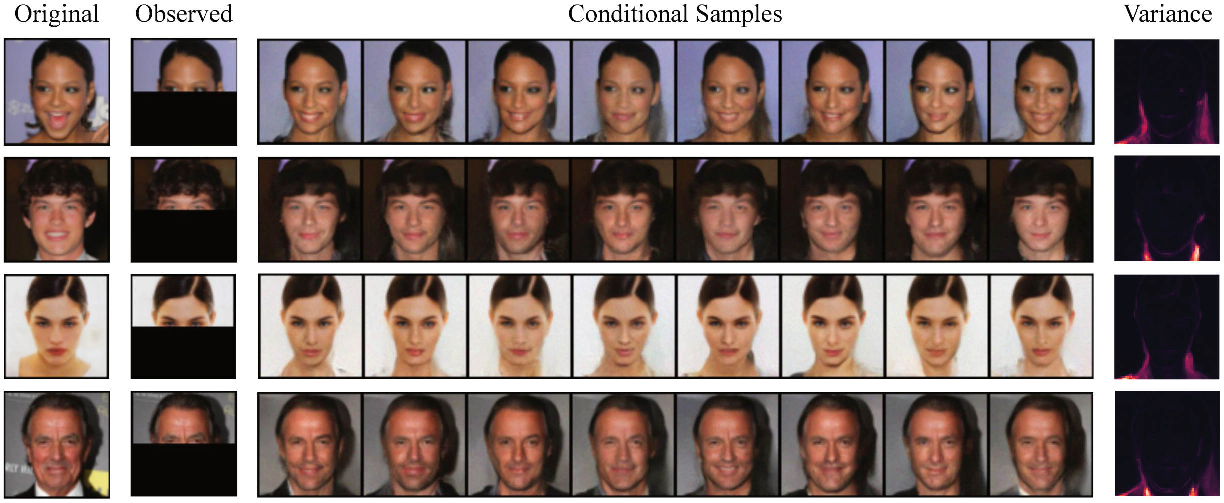

Figure 3: Conditional samples generated by our method from observing the upper half of CelebA-HQ faces. We see that our approach is able to produce diverse completions with different jaw lines, mouth shapes, and facial expression.

line methods on most of the metrics. Note that PL-MCMC results for CIFAR-10 and CelebA-HQ are omitted because it was prohibitively slow to run for hundreds of images, as each MCMC chain required over 20,000 proposals. Cannella et al. (2020) also report using 25,000 proposals for their experiments.

Although our method achieves slightly lower diversity value compared to the baselines, we point out that our method also fits the ground truth better (as evidenced by lower MSE and LPIPS to the ground truth). Measuring diversity alone has limitations, as a model could achieve high diversity by inpainting with random noise. Thus, we emphasize that in Figure 4, we observe a noticeable performance gap between using a single sample and the conditional mean (i.e. the optimal MMSE estimator) obtained by averaging multiple samples. This shows high reconstruction accuracy as well as diversity of our samples, since the samples must be **both diverse and close to the ground truth** for this gap to exist.

6.2. Compressed Sensing

We also present compressed sensing results on the CelebA-HQ dataset. We compare our method to (Bora et al., 2017) and (Asim et al., 2019), two representative techniques for solving inverse problems with a deep generative prior. We did not explicitly compare to the classical sparsity-based priors, as these papers have already demonstrated the superior performance of deep generative priors over them.

Further, we test whether using the conditional mean $\mathbb{E}[\boldsymbol{x} \mid \boldsymbol{y} = \boldsymbol{y}^*]$ helps or not by evaluating our method in two different ways. First, we compute the PSNR for individual samples averaged over 32 draws, labeled "Ours (single)". Second, we compute the PSNR using the empirical mean of the same 32 samples, labeled "Ours (MMSE)".

As shown in Figure 4, we notice a significant increase in the PSNR of the recovered signal, especially when using the MMSE estimator. The relative performance among the presented methods confirms the benefits of *distributional* approaches to inverse problems, and the importance of using the MMSE objective.

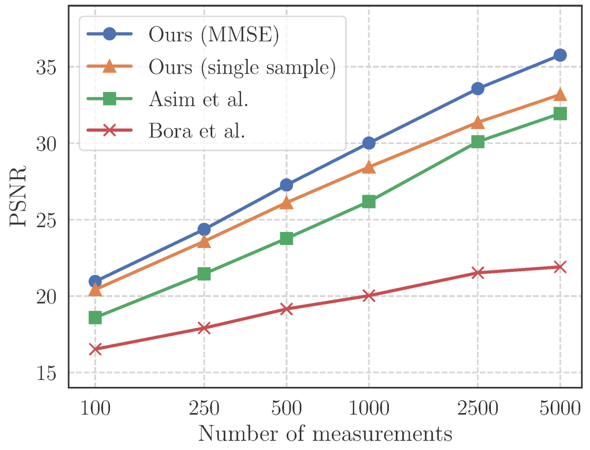

Figure 4: Compressed sensing results at varying numbers of measurements. The plot not only shows that our method outperforms existing methods on a single-sample basis, but also confirms the benefits of using the MMSE estimator.

6.3. Uncertainty Quantification

A key advantage of our approach compared to MCMC-based sampling and existing point estimate methods is the ability to efficiently sample from the learned conditional distribution. This allows us to perform uncertainty quantification by empirically estimating per-pixel variance from a large number of samples.

We demonstrate this in Figure 1. To create this figure, we took a conditional distribution from the above image completion task and generated 3200 i.i.d. samples. With our flow-based approximate posterior, this only takes 55 seconds [2]. Then we performed kernel density estimation on the histogram of pixel intensity values for each pixel position.

In the figure, we show this result on three representative pixels, each exhibiting a widely different behavior. As expected, the top pixel in the observed region is sharply concentrated around a single value, where the two unobserved pixels have higher entropy. The pixel at the bottom is a particularly interesting case, as there is some semantic ambiguity given only the top half: it could be part of either the neck or the black background. We see that our learned conditional distribution correctly captures this bimodality, as confirmed by the bottom left plot with highest entropy.

6.4. Effect of Amortization

Here we study the effect of amortizing the pre-generator over the observation. We repeat the image completion experiments from Section 6.1, except we use a pre-generator that takes in the observed half of the image as conditioning input. The architecture is similar to the conditional flow used in (Ardizzone et al., 2019), except our model is based on RealNVP instead of Glow. As a baseline, we consider Ambient VI as well as Asim et al. (2019), also known in the literature as "inference via optimization" (labeled IvOM in Table 2) (Srivastava et al., 2017; Metz et al., 2017).

The results are shown in Table 2 and Figure 5. Compared to the non-amortized version, there is some degradation in sample quality both visually and in terms of FID. However, the amortized pre-generator significantly improves the inference speed.

Table 2: Sample quality and inference speed using the amortized pre-generator. Inference speed is measured as the average time (in seconds) taken to generate a conditional sample. SVI and LMC results from the image completion experiments in Section 6.1 are provided for comparison. We can see that amortized inference is several orders of magnitude faster. We also observe that IvOM (Srivastava et al., 2017; Metz et al., 2017) performs similarly to LMC.

	MNIST			CelebA-HQ		
	FID	LPIPS	Speed	FID	LPIPS	Speed
Ours (AVI)	8.25	0.249	**0.025**	83.1	0.463	**0.046**
Ours (SVI)	5.15	0.076	13.6	37.2	0.207	70.5
LMC	14.6	0.135	3.21	30.3	0.229	88.4
Ambient VI	123	0.282	9.85	295	0.738	67.9
IvOM	24.7	0.141	1.65	95.8	0.237	88.8

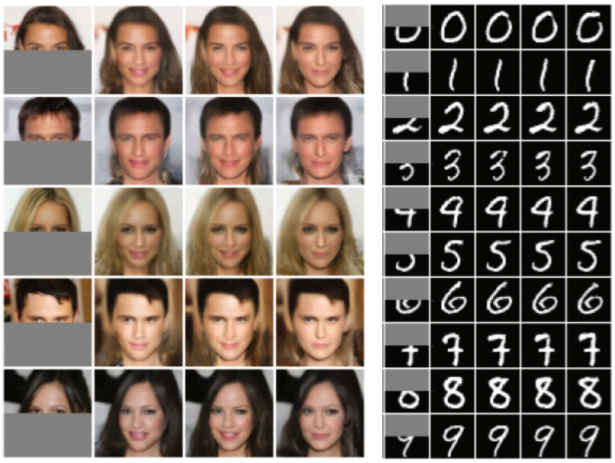

Figure 5: Samples generated using the amortized pre-generator. Each row contains samples conditioned on the masked input in the first column. We see that the amortized pre-generator produces visually plausible samples without having to perform VI for each observation separately.

[2] Measured on a single NVidia GTX 2080 GPU.

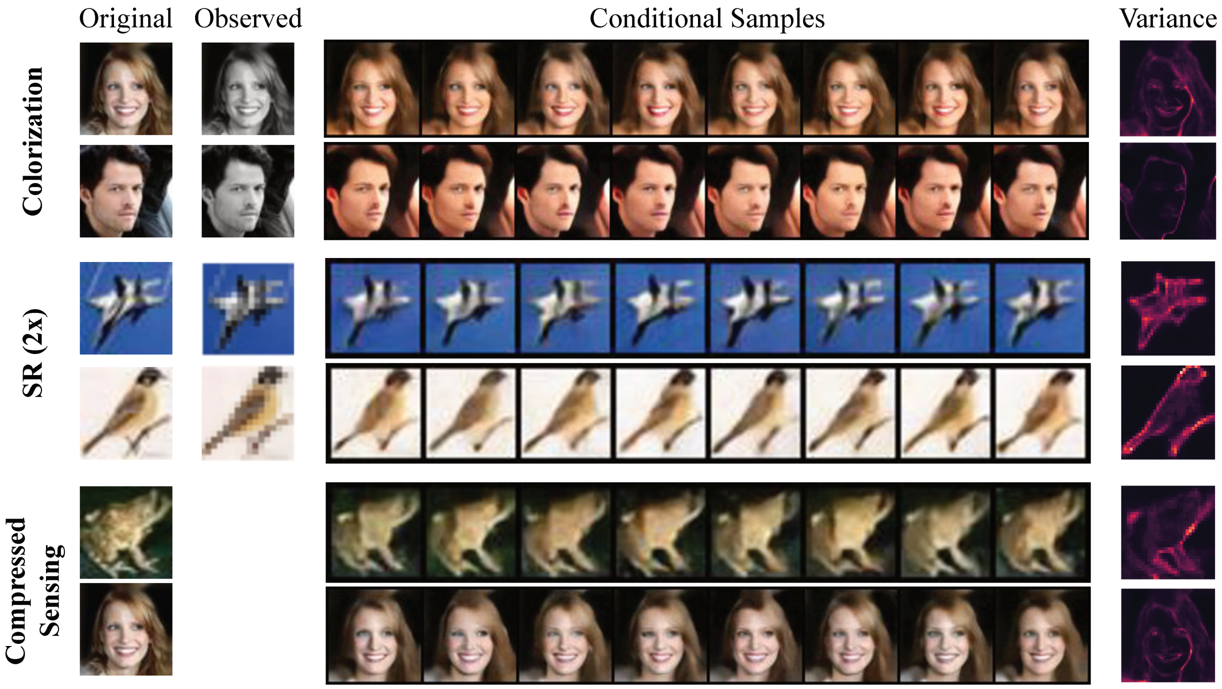

Figure 6: Results on various inverse problem tasks using our method.

6.5. Other Inverse Problems

Here we evaluate the versatility of our method on additional linear inverse problems. In Figure 6, we show the conditional samples obtained by our method on three different tasks: image colorization, super-resolution (2×), and compressed sensing with 500 random Gaussian measurements (for reference, CIFAR-10 images have 3072 dimensions). We notice that the generated samples look natural, even when they do not match the original input perfectly, again showing our method's capability to generate semantically meaningful conditional samples with diversity.

6.6. Effects of the Smoothing Parameter

The choice of variance for Gaussian smoothing in eq. (4) is an important hyperparameter, so we provide some empirical analysis on the effects of σ. As shown in Figure 7, large values of σ cause the samples to ignore the observation, whereas small values lead to unnatural samples as the learned distribution tries to match a degenerate true posterior. Visually, we achieve essentially negligible variance on the observed portion past $\sigma = 0.01$, but at the slight degradation in the sample quality. In Figure 8, we also notice that the difference between the true observation ($\boldsymbol{x}_1^*$) and generated observation ($\tilde{\boldsymbol{x}}_1$) stops improving past $\sigma = 1\text{e}{-4}$. We tried annealing σ from a large value to a small positive target value to see if that would help improve the sample quality at very small values of σ, but noticed no appreciable difference. In practice, we recommend using the largest possible σ that produces observations that are within the (task-specific) acceptable range of the true observation.

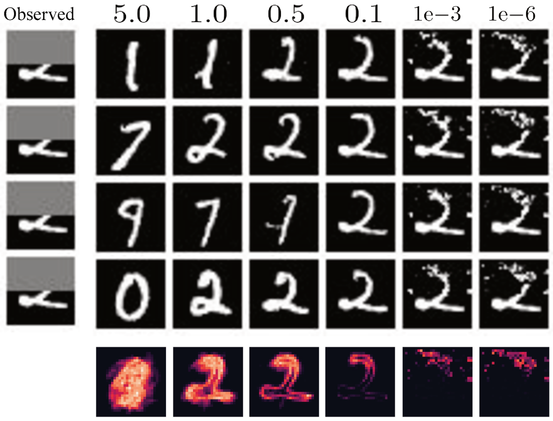

Figure 7: Each column contains samples from the learned conditional sampler at different values of σ with pixelwise variance computed using 32 samples.

7. Conclusion

We proposed a new technique for solving inverse problems with a normalizing flow prior by viewing them as conditional inference tasks. The need for approximate inference is motivated by our theoretical hardness result for exact inference. The particular parametrization of our approxi-

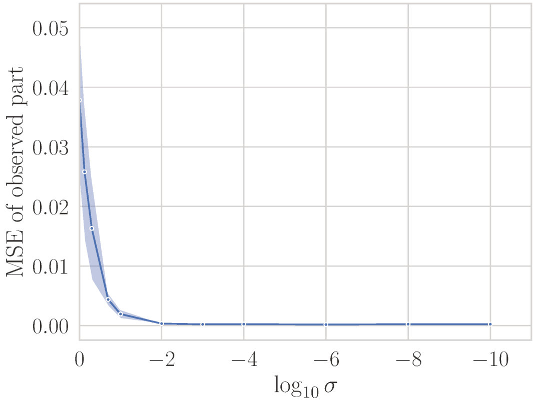

Figure 8: MSE between $\boldsymbol{x}$ and $\boldsymbol{x}^*$ at different values of σ.

mate posterior as a composition of flows is amenable to uncertainty quantification. We also presented a detailed empirical evaluation of our method with both quantitative and qualitative results on a wide range of tasks and datasets. Further, we show that our formulation can be amortized to significantly improve the inference speed without significantly sacrificing sample quality. Overall, we believe that the idea of a pre-generator creating structured noise is a useful and general method for solving inverse problems with the benefit of leveraging pre-trained models and quantifying reconstruction uncertainty.

Acknowledgements

This research has been supported by NSF Grants CCF 1934932, AF 1901292, 2008710, 2019844 the NSF IFML 2019844 award as well as research gifts by Western Digital, WNCG and MLL, computing resources from TACC and the Archie Straiton Fellowship.

References

Adler, J. and Öktem, O. Deep bayesian inversion. *arXiv preprint arXiv:1811.05910*, 2018.

Adler, J. and Öktem, O. Deep posterior sampling: Uncertainty quantification for large scale inverse problems. In *International Conference on Medical Imaging with Deep Learning–Extended Abstract Track*, 2019.

Ardizzone, L., Lüth, C., Kruse, J., Rother, C., and Köthe, U. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019.

Asim, M., Ahmed, A., and Hand, P. Invertible generative models for inverse problems: mitigating representation error and dataset bias. *arXiv preprint arXiv:1905.11672*, 2019.

Atanov, A., Volokhova, A., Ashukha, A., Sosnovik, I., and Vetrov, D. Semi-conditional normalizing flows for semi-supervised learning. *arXiv preprint arXiv:1905.00505*, 2019.

Baraniuk, R. G. Compressive sensing. *IEEE signal processing magazine*, 24(4), 2007.

Behrmann, J., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks. *International Conference on Machine Learning*, 2019.

Belghazi, M. I., Oquab, M., LeCun, Y., and Lopez-Paz, D. Learning about an exponential amount of conditional distributions. *arXiv preprint arXiv:1902.08401*, 2019.

Bickel, P. J., Ritov, Y., Tsybakov, A. B., et al. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, 37(4):1705–1732, 2009.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Bora, A., Jalal, A., Price, E., and Dimakis, A. G. Compressed sensing using generative models. In *International Conference on Machine Learning*, pp. 537–546. JMLR. org, 2017.

Candes, E. J., Romberg, J. K., and Tao, T. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

Cannella, C., Soltani, M., and Tarokh, V. Projected latent markov chain monte carlo: Conditional sampling of normalizing flows, 2020.

Cremer, C., Li, X., and Duvenaud, D. Inference suboptimality in variational autoencoders. In *International Conference on Machine Learning*, pp. 1078–1086. PMLR, 2018.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. In *International Conference on Learning Representations 2015 workshop track*, 2015.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. In *International Conference on Learning Representations*, 2016.

Donoho, D. L. et al. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. Neural spline flows. In *Neural Information Processing Systems*, 2019.

Engel, J., Hoffman, M., and Roberts, A. Latent constraints: Learning to generate conditionally from unconditional generative models. *arXiv preprint arXiv:1711.05772*, 2017.

Grover, A. and Ermon, S. Uncertainty autoencoders: Learning compressed representations via variational information maximization. In *International Conference on Artificial Intelligence and Statistics*, 2019.

Heckel, R. and Hand, P. Deep decoder: Concise image representations from untrained non-convolutional networks. In *International Conference on Learning Representations*, 2019.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pp. 6626–6637, 2017.

Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730, 2019.

Hoffman, M., Sountsov, P., Dillon, J. V., Langmore, I., Tran, D., and Vasudevan, S. Neutra-lizing bad geometry in hamiltonian monte carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

Ivanov, O., Figurnov, M., and Vetrov, D. Variational autoencoder with arbitrary conditioning. *arXiv preprint arXiv:1806.02382*, 2018.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Kabkab, M., Samangouei, P., and Chellappa, R. Task-aware compressed sensing with generative adversarial networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Neural Information Processing Systems*, pp. 10215–10224, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. In *Neural Information Processing Systems*, pp. 4743–4751, 2016.

Li, Y., Akbar, S., and Oliva, J. B. Flow models for arbitrary conditional likelihoods. *arXiv preprint arXiv:1909.06319*, 2019.

Liu, Z. and Scarlett, J. Information-theoretic lower bounds for compressive sensing with generative models. *IEEE Journal on Selected Areas in Information Theory*, 1(1): 292–303, 2020.

Lucas, A., Iliadis, M., Molina, R., and Katsaggelos, A. K. Using deep neural networks for inverse problems in imaging: beyond analytical methods. *IEEE Signal Processing Magazine*, 35(1):20–36, 2018.

Mardani, M., Sun, Q., Donoho, D., Papyan, V., Monajemi, H., Vasanawala, S., and Pauly, J. Neural proximal gradient descent for compressive imaging. In *Neural Information Processing Systems*, pp. 9573–9583, 2018.

Menon, S., Damian, A., Hu, S., Ravi, N., and Rudin, C. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. doi: 10.1109/cvpr42600.2020.00251. URL http://dx.doi.org/10.1109/cvpr42600.2020.00251.

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Mixon, D. G. and Villar, S. Sunlayer: Stable denoising with generative networks. *arXiv preprint arXiv:1803.09319*, 2018.

Mousavi, A., Dasarathy, G., and Baraniuk, R. G. A data-driven and distributed approach to sparse signal representation and recovery. In *International Conference on Learning Representations*, 2018.

Neal, R. M. et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Nijkamp, E., Gao, R., Sountsov, P., Vasudevan, S., Pang, B., Zhu, S.-C., and Wu, Y. N. Learning energy-based model with flow-based backbone by neural transport mcmc. *arXiv preprint arXiv:2006.06897*, 2020.

Ongie, G., Jalal, A., Metzler, C. A., Baraniuk, R. G., Dimakis, A. G., and Willett, R. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., Driessche, G., Lockhart, E., Cobo, L., Stimberg, F., et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pp. 3918–3926. PMLR, 2018.

Pajot, A., de Bezenac, E., and Gallinari, P. Unsupervised adversarial image reconstruction. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJg4Z3RqF7.

Pandit, P., Sahraee, M., Rangan, S., and Fletcher, A. K. Asymptotics of map inference in deep networks. *arXiv preprint arXiv:1903.01293*, 2019.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

Papamarkou, T., Hinkle, J., Young, M. T., and Womble, D. Challenges in bayesian inference via markov chain monte carlo for neural networks. *arXiv*, pp. arXiv–1910, 2019.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2536–2544, 2016.

Raj, A., Li, Y., and Bresler, Y. Gan-based projector for faster recovery with convergence guarantees in linear inverse problems. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5602–5611, 2019.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pp. 1530–1538, 2015.

Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., and Cohen-Or, D. Encoding in style: a stylegan encoder for image-to-image translation. *arXiv preprint arXiv:2008.00951*, 2020.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *Advances in neural information processing systems*, pp. 2234–2242, 2016.

Shah, V. and Hegde, C. Solving linear inverse problems using gan priors: An algorithm with provable guarantees. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4609–4613. IEEE, 2018.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. In *Neural Information Processing Systems*, pp. 3483–3491, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pp. 11918–11930, 2019.

Srivastava, A., Valkov, L., Russell, C., Gutmann, M. U., and Sutton, C. Veegan: reducing mode collapse in gans using implicit variational learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 3310–3320, 2017.

Sun, Y., Liu, J., and Kamilov, U. S. Block coordinate regularization by denoising. *NeurIPS*, 2019.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Tonolini, F., Lyons, A., Caramazza, P., Faccio, D., and Murray-Smith, R. Variational inference for computational imaging inverse problems. *arXiv preprint arXiv:1904.06264*, 2019.

Wainwright, M. J., Jordan, M. I., et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

Ward, P. N., Smofsky, A., and Bose, A. J. Improving exploration in soft-actor-critic with normalizing flows policies. *arXiv preprint arXiv:1906.02771*, 2019.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

Whang, J., Lei, Q., and Dimakis, A. G. Compressed sensing with invertible generative models and dependent noise. *arXiv preprint arXiv:2003.08089*, 2020.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5505–5514, 2018.

Zhang, C. and Jin, B. Probabilistic residual learning for aleatoric uncertainty in image restoration. *arXiv preprint arXiv:1908.01010*, 2019.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Zheng, C., Cham, T.-J., and Cai, J. Pluralistic image completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1438–1447, 2019.

A. Proof of Hardness Results

A.1. Preliminaries

A Boolean variable is a variable that takes a value in $\{-1, 1\}$. A *literal* is a Boolean variable x_i or its negation $(\neg x_i)$. A *clause* is set of literals combined with the OR operator, e.g., $(x_1 \vee \neg x_2 \vee x_3)$. A *conjunctive normal form formula* is a set of clauses joined by the AND operator, e.g. $(x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee \neg x_3 \vee x_4)$. A satisfying assignment is an assignment to the variables such that the Boolean formula is true.

The *3-SAT problem* is the problem of deciding if a conjunctive normal form formula with three literals per clause has a satisfying assignment. We will show that conditional sampling from flow models allows us to solve the 3-SAT problem.

We ignore the issue of representing samples from the conditional distribution with a finite number of bits. However the reduction is still valid if the samples are truncated to a constant number of bits.

A.2. Design of the Additive Coupling Network

Given a conjunctive normal form with m clauses, we design a ReLU neural network with 3 hidden layers such that the output is 0 if the input is far from a satisfying assignment, and the output is about a large number M if the input is close to a satisfying assignment.

We will define the following scalar function

$$\begin{aligned}\delta_\varepsilon(x) = \mathrm{ReLU}&\left(\frac{1}{\varepsilon}(x-(1-\varepsilon))\right)\\ -\mathrm{ReLU}&\left(\frac{1}{\varepsilon}(x-(1-\varepsilon))-1\right)\\ -\mathrm{ReLU}&\left(\frac{1}{\varepsilon}(x-1)\right)\\ +\mathrm{ReLU}&\left(\frac{1}{\varepsilon}(x-1)-1\right).\end{aligned}$$

This function is 1 if the input is 1, 0 if the input x has $|x-1| \geq \varepsilon$ and is a linear interpolation on $(1-\varepsilon, 1+\varepsilon)$. Note that it can be implemented by a hidden layer of a neural network and a linear transform, which can be absorbed in the following hidden layer. See Figure 9 for a plot of this function.

For each variable x_i, we create a transformed variable $\tilde{x}_i$ by applying $\tilde{x}_i = \delta_\varepsilon(x_i) - \delta_\varepsilon(-x_i)$. Note that this function is 0 on $(-\infty, -1-\varepsilon] \cup [-1+\varepsilon, 1-\varepsilon] \cup [1+\varepsilon, \infty)$, -1 at $x_i = -1$, 1 at $x_i = 1$, and a smooth interpolation on the remaining values in the domain.

Every clause has at most 8 satisfying assignments. For each satisfying assignment we will create a neuron with the following process: (1) get the relevant transformed values $\tilde{x}_i, \tilde{x}_j, \tilde{x}_k$, (2) multiply each variable by $1/3$ if it is equal to 1 in the satisfying assignment and $-1/3$ if it is equal to -1 in the satisfying assignment, (3) sum the scaled variables, (4) apply the δ_ε function to the sum.

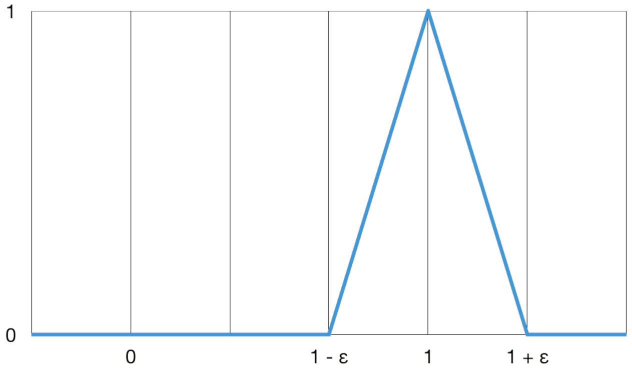

Figure 9: Plot of the scalar function used to construct an additive coupling layer that can generate samples of satisfying 3-SAT assignments.

We will then sum all the neurons corresponding to a satisfying assignment for clause C_j to get the value c_j. The final output is the value $M \times \mathrm{ReLU}(\sum_j c_j - (m-1))$, where M is a large scalar.

We say that an input to the neural network x corresponds to a Boolean assignment $x' \in \{-1, 1\}^d$ if for every x_i we have $|x_i - x'_i| < \varepsilon$. For $\varepsilon < 1/3$, if the input does not correspond to a satisfying assignment of the given formula, then at least one of the values c_j is 0. The remaining values of c_j are at most 1, so the sum in the output is at most $(m-1)$, thus the sum is at most zero, so the final output is 0. However, if the input is a satisfying assignment, then every value of $c_j = 1$, so the output is M.

A.3. Generating SAT Solutions from the Conditional Distribution

Our flow model will take in Gaussian noise $x_1, \ldots, x_d, z \sim N(0,1)$. The values $x_1, \ldots, x_d$ will be passed through to the output. The output variable y will be $z + f_M(x_1, \ldots, x_d)$, where f_M is the neural network described in the previous section, and M is the parameter in the output to be decided later.

Let A be all the valid satisfying assignments to the given formula. For each assignment a, we will define X_a to be the region $X_a = \{x \in \mathbb{R}^d : \|a - x\|_\infty \leq \varepsilon\}$, where as above ε is some constant less than $1/3$. Let $X_A = \bigcup_{a \in A} X_a$.

Given an element $x \in X_a$, we can recreate the corresponding satisfying assignment a. Thus if we have an element of X_A, we can certify that there is a satisfying assignment. We will show that the distribution conditioned on $y = M$ can generate satisfying assignments with high probability.

We have that

$$p(X_A \mid y = M) = \frac{p(y = M, X_A)}{p(y = M, X_A) + p(y = M, \overline{X}_A)}$$

If we can show that $p(y = M, \overline{X}_A) \ll p(y = M, X_A)$, then we have that the generated samples are with high probability satisfying assignments.

Note that,

$$\begin{aligned} p(y = M, \overline{X}_A) &= p(y = M \mid \overline{X}_A) P(\overline{X}_A) \\ &\leq p(y = M \mid \overline{X}_A). \end{aligned}$$

Also notice that if $x \in \overline{X}_A$, then $f_M(x) = 0$. Thus $y \sim \mathcal{N}(0, 1)$ and $P(y = M \mid \overline{X}_A) = \Theta(\exp(-M^2/2))$.

Now consider any satisfying assignment x_a. Let X'_a be the region $X'_a = \{x \in \mathbb{R}^d : \|a - x\|_\infty \leq \frac{1}{2m}\}$. Note that for every x in this region we have $f_M(x) \geq M/2$. Additionally, we have that $P(X'_a) = \Theta(m)^{-d}$. Thus for any $x \in X'_a$, we have $p(Y = M \mid x) \gtrsim \exp(-M^2/8)$. We can conclude that

$$\begin{aligned} p(y = M, X_A) &\geq p(Y = M, X'_a) \\ &= \int_{X'_a} p(Y = M \mid x) p(x)\, dx \\ &\gtrsim \exp(-M^2/8 - \Theta(d \log m)). \end{aligned}$$

For $M = O(\sqrt{d \log m})$, we have that $p(y = M, \overline{X}_A)$ is exponentially smaller than $p(y = M, X_A)$. This implies that sampling from the distribution conditioned on $y = M$ will return a satisfying assignment with high probability.

A.4. Hardness of Approximate Sampling

Definition 2. The complexity class RP is the class of decision problems with efficient random algorithms that (1) output YES with probability $1/2$ if the true answer is YES and (2) output NO with probability 1 if the true answer is NO. It is widely believed that RP is a strict subset of NP.

A simple extension of the above theorem shows that even approximately matching the true conditional distribution in terms of the total variation (TV) distance is computationally hard. TV distance is defined as $d_{\mathrm{TV}}(p, q) = \sup_E |p(E) - q(E)| \leq 1$, where E is an event. The below corollary shows that it is hard to conditionally sample from a distribution that is even slightly bounded away from 1.

Corollary 3. *The conditional sampling problem remains hard even if we only require the algorithm to sample from a distribution q such that $d_{\mathrm{TV}}(p(\cdot \mid x = x^*), q) \leq 1 - 1/\mathrm{poly}(d)$, where d is the dimension of the distribution.*

We show that the problem is still hard even if we require the algorithm to sample from a distribution q such that $d_{\mathrm{TV}}(p(x \mid y = y^*), q) \geq 1/\mathrm{poly}(d)$.

Consider the event X_A from above. We saw that $p(X_A \mid y = M) \geq 1 - \exp(-\Omega(d))$. We have that $d_{\mathrm{TV}}(p(\cdot \mid y = M), q) \geq 1 - \exp(-\Omega(d) - q(X_A))$.

Suppose that the distribution q has $q(X_A) \geq 1/\mathrm{poly}(d)$. Then by sampling a polynomial number of times from q we sample an element of X_A, which allows us to find a satisfying assignment. Thus if we can efficiently create such a distribution, we would be able to efficiently solve SAT and RP = NP. As we are assuming this is false, we must have $q(X_A) \leq 1/\mathrm{poly}(d)$, which implies $d_{\mathrm{TV}}(p(\cdot \mid y = M), q) \geq 1 - 1/\mathrm{poly}(d)$.

B. Missing Derivations

B.1. Derivation of Equation (4)

Here we present a detailed derivation of Equation (4). Note that this equality is true up to a constant w.r.t. $\hat{f}$.

$$\begin{aligned} &\mathcal{L}_{\text{ours}}(\hat{f}) \\ &\triangleq D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}) \parallel p_{\boldsymbol{x}}(\boldsymbol{x} \mid \tilde{\boldsymbol{y}} = \boldsymbol{y}^*)) \\ &= \mathbb{E}_{\boldsymbol{x} \sim q_{\boldsymbol{x}}} \left[\log q_{\boldsymbol{x}}(\boldsymbol{x}) - \log p_{\boldsymbol{x}}(\boldsymbol{x}, \tilde{\boldsymbol{y}} = \boldsymbol{y}^*) \right] + \log p_{\boldsymbol{x}}(\tilde{\boldsymbol{y}} = \boldsymbol{y}^*) \\ &\overset{A}{=} \mathbb{E}_{\boldsymbol{x} \sim q_{\boldsymbol{x}}} \left[\log q_{\boldsymbol{x}}(\boldsymbol{x}) - \log p_{\boldsymbol{x}}(\boldsymbol{x}) - \log p_{\sigma}(\tilde{\boldsymbol{y}} = \boldsymbol{y}^* \mid \boldsymbol{x}) \right] \\ &\overset{B}{=} \mathbb{E}_{\boldsymbol{x} \sim q_{\boldsymbol{x}}} \left[\log q_{\boldsymbol{x}}(\boldsymbol{x}) - \log p_{\boldsymbol{x}}(\boldsymbol{x}) \right] \\ &\quad + \mathbb{E}_{\boldsymbol{x} \sim q_{\boldsymbol{x}}} \left[- \log p_{\sigma}(\tilde{\boldsymbol{y}} = \boldsymbol{y}^* \mid \boldsymbol{y} = A(\boldsymbol{x})) \right] \\ &= D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}) \parallel p_{\boldsymbol{x}}(\boldsymbol{x})) \\ &\quad + \mathbb{E}_{\boldsymbol{x} \sim q_{\boldsymbol{x}}} \left[- \log p_{\sigma}(\tilde{\boldsymbol{y}} = \boldsymbol{y}^* \mid \boldsymbol{y} = A(\boldsymbol{x})) \right] \\ &\overset{C}{=} D_{\mathrm{KL}}(q_{\boldsymbol{z}}(\boldsymbol{z}) \parallel p_{\boldsymbol{x}}(\boldsymbol{z})) + \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{z}}} \left[\frac{1}{2\sigma^2} \| A(f(\boldsymbol{z})) - \boldsymbol{y}^* \|_2^2 \right] \end{aligned}$$

In (A), we drop $\log p_{\boldsymbol{x}}(\tilde{\boldsymbol{y}} = \boldsymbol{y}^*)$, as it is constant w.r.t. $\hat{f}$. In (B), we use the conditional independence $\tilde{\boldsymbol{y}} \perp\!\!\!\perp \boldsymbol{x} \mid \boldsymbol{y}$. In (C), we use the invariance of KL divergence under invertible transformation to rewrite it in terms of $\boldsymbol{z}$.

B.2. Joint VI vs. Marginal VI

We also provide a justification for using the joint VI loss as discussed in Section 4. Specifically, we show that the joint VI loss in eq. (4) is an upper bound to the intractable marginal VI loss. Assuming the partitioning $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2)$, we have:

$$
\begin{aligned}
&\text{(Joint KL)} \\
&= D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}) \parallel p_{\boldsymbol{x}}(\boldsymbol{x} | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*)) \\
&= \mathbb{E}_{q_{\boldsymbol{x}}}\left[\log q_{\boldsymbol{x}}(\boldsymbol{x}_1, \boldsymbol{x}_2) - \log p_{\boldsymbol{x}}(\boldsymbol{x}_1, \boldsymbol{x}_2 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*)\right] \\
&= \mathbb{E}_{q_{\boldsymbol{x}}}\Big[\log q_{\boldsymbol{x}}(\boldsymbol{x}_2) + \log q_{\boldsymbol{x}}(\boldsymbol{x}_1 \mid \boldsymbol{x}_2) \\
&\qquad - \log p_{\boldsymbol{x}}(\boldsymbol{x}_2 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*) - \log p_{\boldsymbol{x}}(\boldsymbol{x}_1 \mid \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*, \boldsymbol{x}_2)\Big] \\
&= \mathbb{E}_{q_{\boldsymbol{x}}}\left[\log q_{\boldsymbol{x}}(\boldsymbol{x}_2) - \log p_{\boldsymbol{x}}(\boldsymbol{x}_2 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*)\right] \\
&+ \mathbb{E}_{q_{\boldsymbol{x}}}\Big[\mathbb{E}_{q_{\boldsymbol{x}}(\boldsymbol{x}_1|\boldsymbol{x}_2)}\Big[\\
&\qquad \log q_{\boldsymbol{x}}(\boldsymbol{x}_1 \mid \boldsymbol{x}_2) - \log p_{\boldsymbol{x}}(\boldsymbol{x}_1 \mid \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*, \boldsymbol{x}_2)\Big]\Big] \\
&= D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}_2) \parallel p_{\boldsymbol{x}}(\boldsymbol{x}_2 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*)) \\
&\quad + \mathbb{E}_{q_{\boldsymbol{x}}(\boldsymbol{x}_2)}\left[D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}_1 \mid \boldsymbol{x}_2) \parallel p_{\boldsymbol{x}}(\boldsymbol{x}_1 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*, \boldsymbol{x}_2))\right] \\
&\geq D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}_2) \parallel p_{\boldsymbol{x}}(\boldsymbol{x}_2 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*)) \\
&= \text{(Marginal KL)},
\end{aligned}
$$

where the last inequality is due to the nonnegativity of KL. Note that equality holds when

$$D_{\mathrm{KL}}(q_{\boldsymbol{x}}(\boldsymbol{x}_1 \mid \boldsymbol{x}_2) \parallel p_{\boldsymbol{x}}(\boldsymbol{x}_1 | \tilde{\boldsymbol{x}}_1 = \boldsymbol{x}^*, \boldsymbol{x}_2)) = 0,$$

i.e. when our variational posterior matches the true conditional.

C. Experiment Details

C.1. Our Algorithm

Algorithm 1 Training the pre-generator for a given observation under transformation. We assume that $\hat{f}$ is an invertible neural network with parameters θ.

```
Input: $\boldsymbol{y}^*$: observation, $A$: differentiable measurement function.
for $i = 1 \ldots$ num_steps do
    for $j = 1 \ldots m$ do
        Sample $\boldsymbol{\epsilon}^{(j)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
        $\boldsymbol{z}^{(j)} \leftarrow \hat{f}(\boldsymbol{\epsilon}^{(j)})$    (reparametrization trick)
    end for
    $\mathcal{L} \leftarrow \frac{1}{m} \sum_{j=1}^{m} \Big[\log q_{\boldsymbol{z}}(\boldsymbol{z}^{(j)}) - \log p_{\boldsymbol{z}}(\boldsymbol{z}^{(j)}) + \frac{1}{2\sigma^2} \left\| A(f(\boldsymbol{z}^{(j)})) - \boldsymbol{y}^* \right\|_2^2 \Big]$
    $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}$    (gradient step)
end for
```

C.2. Hyperparameters: Base Model and Pre-generator

See Table 3 and Table 4 for the hyperparameters used to define the network architectures train them. For the color datasets CIFAR-10 and CelebA-HQ, we used 5-bit pixel quantization following Kingma & Dhariwal (2018). Additionally for CelebA-HQ, we used the same train-test split (27,000/3,000) of Kingma & Dhariwal (2018) and resized the images to 64 × 64 resolution. Uncurated samples from the base models are included for reference in Figure 10.

Table 3: Hyperparameters used to train the base models used in our experiments.

Base Models	MNIST	CIFAR-10	CelebA-HQ
Image resolution	28×28	32×32	64×64
Num. scales	3	6	6
Res. blocks per scale	8	12	10
Res. block channels	32	64	80
Bits per pixel	8	5	5
Batch size	128	64	32
Learning rate	0.001	0.001	0.001
Test set bits-per-dim	1.053	1.725	1.268

Table 4: Hyperparameters used to define and train the pre-generator for each of our experiments.

Base Models	MNIST	CIFAR-10	CelebA-HQ
Image resolution	28×28	32×32	64×64
Num. scales	3	4	3
Res. blocks per scale	3	4	3
Res. block channels	32	48	48
Batch size	64	32	8

C.3. Hyperparameters: Image Inpainting

We randomly chose 900/500/300 images from MNIST/CIFAR-10/CelebA-HQ test sets, applied masks defined in Section 6.1, and generated samples conditioned on the remaining parts. FID and other sample quality metrics were computed using 6 conditional samples per test image for all MNIST experiments, and 8 conditional samples for all CIFAR-10 and CelebA-HQ experiments.

For VI Methods (Ours & Ambient VI)

- Learning rate: $1\mathrm{e}{-3}$ for MNIST; $5\mathrm{e}{-4}$ for the others
- Number of training steps: 4000 for CelebA-HQ; 1000 for the others

For Langevin Dynamics

- Learning rate: $5\mathrm{e}{-4}$ for all datasets
- Length of chain: 1000 for CIFAR-10; 4000 for the others

For PL-MCMC

- Learning rate: $5\mathrm{e}{-4}$

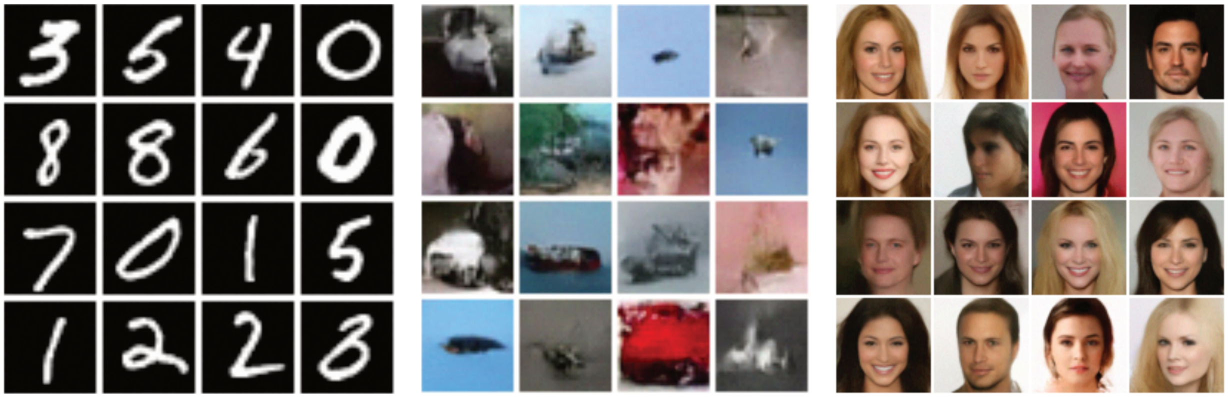

Figure 10: Unconditional samples from the base models used for our experiments. From left: MNIST, 5-bit CIFAR-10, and 5-bit CelebA-HQ models.

- Length of chain: 2000 for MNIST
- $\sigma_a = 1\text{e}{-3}$, $\sigma_p = 0.05$

C.4. Hyperparameters: Compressed Sensing

For Ours and (Asim et al., 2019)

- Learning rate: $5\text{e}{-4}$
- Number of training steps: 4000
- For (Asim et al., 2019), we used the same training objective used in their Compressed Sensing experiments: $\arg\min_{\boldsymbol{z}} \|AG(\boldsymbol{z}) - \boldsymbol{y}^*\|_2^2$

For (Bora et al., 2017)

- Learning rate: 0.02
- Regularization coefficient: $\lambda = 0.1$
- Following (Bora et al., 2017), we repeated each run three times and initialized $\boldsymbol{z}_0$ using samples from $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ where $\sigma = 0.1$. Then we used the best result out of the three runs for evaluation.

C.5. Hyperparameters: Inverse Problems

Please see Table 5.

Table 5: Hyperparameters for the extra inverse problem experiments.

	Colorize	CS	CS	SR ($2\times$)
Dataset	CelebA-HQ		CIFAR-10	
Learning rate	$5\text{e}{-4}$	$5\text{e}{-4}$	$5\text{e}{-4}$	$5\text{e}{-4}$
σ	0.05	0.05	0.05	0.05
Batch size	8	8	32	32
Number of steps	1000	2000	1000	1000