# Using Deep Generative Priors For Inverse Problems And Bayesian Optimisation

**Ang Ming Liang**

A report presented for Undergraduate Research Opportunities (UROP) Module

Faculty of Science, Mathematics
National University Of Singapore
Nov 2021

# Using Deep Generative Priors For Inverse Problems And Bayesian Optimisation

**Ming Liang Ang** *
Department of Mathematics
National University of Singapore
Singapore, SG 119077
angmingliang@u.edu.sg

## Abstract

Deep generative modelling has seen impressive progress in the past few years. People have also recently started sampling from the latent spaces of deep generative models for downstream tasks like reconstruction and refining samples from generative models. This report explores four directions to improve and extend the sampling process for solving inverse problems using deep generative models. We first introduce a new initialisation for Langevin dynamics, which we call MAP initialisation that we show leads to a faster mixing time. Next, we explored using the discriminator to perform importance sampling of the latent space and showed that we could improve the visual quality of samples using the discriminator. We also explored adaptive sampling using posterior sampling and showed that adaptive sampling methods reduce the number of measurements needed for tasks such as inpainting. Finally, we extend these results to the closely related area of Bayesian optimisation. We empirically demonstrated these results on multiple estimation tasks and settings for various models on the MNIST dataset.

## 1 Introduction

The goal of inverse problems is to estimate an unknown signal from a set of observations. These observations are obtained from an unknown dataset by a forward process, which is typically non-invertible. Thus reconstructing a unique solution that fits the observations is difficult without prior knowledge about the data as the problem is ill-posed. This setup is interesting because numerous imaging problems such as super-resolution, compressed sensing and denoising fit into this framework [1].

To make this description more precise, we can consider an unknown $n$-pixel image $\mathbf{x}^* \in \mathbb{R}^n$ that is observed via $m$ noisy measurements $\mathbf{y} \in \mathbb{R}^m$ according to the model

$$\mathbf{y} = A(\mathbf{x}^*) + \epsilon$$

where $A$ is the (possibly nonlinear) forward measurement operator, we focus on compressed sensing, denoising and inpainting in this report, and is the noise vector, which we typically assume is Gaussian. The goal then is to recover $x^*$ given knowledge of $A$ and $y$

The classical approach would assume some prior knowledge about $\mathbf{x}$ such as smoothness [2], sparsity in some basis [3], or other geometric properties [4] and then finds an $\mathbf{x}$ that is both a good fit to the observations $\mathbf{y}$ and likely given the prior knowledge. A regularisation function $r(x)$ measures the lack of conformity of $\mathbf{x}$ to a prior model, and $\mathbf{x}$ is selected so that $r(x)$ is as small as possible while still fitting the observed data. Recent work in machine learning has demonstrated that deep

*I would like to thank the many people helping me proofread this report.

neural networks can leverage large datasets to directly compute regularised reconstructions across a host of computational imaging tasks [5–8]. In particular, deep generative models can regularise by constraining the reconstructed image $\mathbf{x}$ to remain on a learned manifold [9].

While this is sufficient for obtaining point estimates for $\mathbf{x}^*$, we are often interested in knowing the uncertainty associated with this estimate or other possible estimates for $\mathbf{x}^*$ as well. The uncertainty of our estimates could also help in the adaptive setting where we can choose where next to sample to improve our estimate of $x^*$ to reduce the number of measurements needed to obtain a reasonable estimate. This setup is often found in applications like X-ray computed tomography and other areas where measurement is expensive. More formally, we want to estimate the following posterior distribution

$$p(\mathbf{x} \mid \mathbf{y}) \propto p(\mathbf{y} \mid \mathbf{x}) p(\mathbf{x})$$

where $\mathbf{x}$ is the image and $\mathbf{y}$ is the measurement.

This work thus explores

- how we can sample efficiently from this posterior distribution using Langevin dynamics to estimate this posterior distribution
- using the samples from the posterior distribution to adaptively sample from an image to reduce the number of measurements needed for estimation
- extending the methods for adaptive sampling to Bayesian optimisation using deep generative models.

## 2 Background

### 2.1 Deep Generative Models

Deep Generative Models (DGMs) have been successfully applied to numerous domains in recent years, from generating high-resolution photo-realistic images [10] to learning policies in reinforcement learning [11]. Among the variety of proposed DGMs, Variational Autoencoder (VAEs) [12] and Generative Adversarial Networks (GANs) [13] have received widespread attention and popularity from the machine learning community for their ability to generate high-quality samples, especially images, that resemble the training data.

VAEs in the most basic setting defines a generative model of the form

$$p_\theta(z, x) = p_\theta(z) p_\theta(x|z)$$

where $p_\theta(z)$ is usually a Gaussian, and $p_\theta(x|z)$ is usually a product of exponential family distributions (e.g., Gaussian or Bernoulli), with parameters computed by a neural network decoder, $d_\theta(z)$. The decoder parameters are fitted by maximising the marginal likelihood

$$p_\theta(x) = \int p_\theta(x \mid z) p_\theta(z) dz.$$

Unfortunately, the marginal likelihood is often intractable. Thus, we maximise the lower bound of the marginal likelihood,

$$\begin{aligned}
\log p_\theta(x) &= E_{q_\phi(z|x)}[\log p_\theta(x)] \\
&= E_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{p_\theta(z \mid x)}\right)\right] \\
&= E_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z \mid x)}\frac{q_\phi(z \mid x)}{p_\theta(z \mid x)}\right)\right] \\
&= \underbrace{E_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right)\right]}_{ELBO} + E_{q_\phi(z|x)}\left[\frac{q_\phi(z \mid x)}{p_\theta(z \mid x)}\right]
\end{aligned}$$

also known as the evidence lower bound (ELBO) instead. The approximate posterior $q_\phi(z \mid x)$ used to calculate the ELBO can be thought of as encoding the input $x$ into a stochastic latent bottleneck $z$ and then decoding it to reconstruct the input approximately. It is thus sometimes called the encoder of the VAE.

GANs are likelihood-free methods where training is formulated to solve the following minimax problem involving a generator and a discriminator.

$$\min_G \max_D V(G, D) = E_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z))]$$

In this setup, the generator seeks to generate samples similar to the real data by minimising a discrepancy determined by the discriminator between real and generated samples. After the game converges, the discriminator is thrown away, and we only keep the generator to generate new samples.

However, recent work has shown that this practice of discarding the discriminator is wasteful, as the discriminator contains valuable information about the underlying data distribution [14–18]. This insight has led to new training strategies and sampling techniques that use this information from discriminators to improve the quality of generated samples from GANs and other generative models such as Variational Autoencoders (VAEs) or Normalising Flows [19, 20]. On the other hand, the current methods used to solve inverse problems using such generative priors typically discards the discriminator from the estimation process altogether [9, 21]. This motivates an intriguing possibility to incorporate discriminator information to solve such inverse problems, which we explore further in Section 3

### 2.2 Energy-Based Models and Langevin Dynamics

An energy-based model (EBM) is a specific type of generative model that is defined by a Boltzmann distribution (otherwise known as the Gibbs distribution) $p_\theta(x) = e^{-E_\theta(x)}/Z_\theta$, where $x \in X$, $X$ is the sate space, and $E(x) : X \to R$ is the energy function. The model is then trained using maximum likelihood estimation by maximising the expected log-likelihood over the data distribution as such,

$$l(\theta) = \mathbb{E}_{x \sim p_{\text{data}(x)}} [\log p_\theta(x)]$$

which is often intractable due to the normalising constant $Z_\theta$. Nevertheless, we can still estimate the log-likelihood gradient with MCMC approaches, allowing for likelihood maximisation using stochastic gradient descent. In this report, we mainly discuss fully trained EBMs and thus focus on sampling from EBMs instead. For more information about EBMs, we refer the reader to [22].

Since $p_\theta(x)$ is intractable, samples are typically generated from $p_\theta(x)$ by an MCMC algorithm to avoid the intractable normalising constant in the denominator. One common MCMC algorithm in continuous state spaces is Langevin dynamics [23] which has the following update equations $x_{i+1} = x_i - \frac{\epsilon}{2}\nabla_x E(x_i) + \sqrt{\epsilon} n, n \sim N(0, I)$. In fact, under certain regularity conditions, such as theorem 1, we can show that Langevin dynamics samples close to the distribution with high probability.

**Theorem 1** (Informal)**.** *Suppose the Langevin chain corresponding to $p(x) \propto e^{-f(x)}$ is initialized close to a manifold M satisfying the following two properties:*

1. *(Nearness to the manifold): When initialized close to $M$, the Langevin dynamics stay in some neighborhood $D = \{X : \min_{X' \in M} \|X - X'\|_2 \leq s\}$ of $M$ up to time $T$ with high probability.*
2. *(Poincare inequality along level sets): The distribution $\tilde{p}^\Delta$ for all $\Delta \in B$ have a Poincare constant bounded by $C_{level}$]*
3. *(Poincare inequality across level sets): The distribution $q$ has a Poincare constant bounded by $C_{across}$.*
4. *(Bounded change of manifold probability): If we denote $G_\Delta : M \to M^\Delta$ the map $G_\Delta(X) = X + \phi_X(\Delta)$, for all $X \in M$ and $\Delta \in B$, the relative change (with respect to $\Delta$) in the manifold density is bounded:*

$$\left\| \frac{\nabla_B (p^\Delta(X + \phi_X(\Delta)) \det((dG_\Delta)_X))}{p^\Delta(X +_X (\Delta) \det((dG_\Delta)_X)} \right\|_2 \leq C_{change}$$

*Then Langevin chains run for time $O(max(1, C_{level} \max 1, C_{across} \max 1, C^2_{change})$ outputs a sample from a distribution that is close in total variation distance to the condition distribution of $p(X)$ restricted to $D$ with high probability.*

The formal statement and proof are in Section 4.1 and Theorem 4 in [24]. However, in practice, these regularity conditions may not hold, and thus we may not sample from the target distribution. In this report, we explore ways to initialise near the manifold to satisfy the first regularity condition in Theorem 1.

Furthermore, poor geometry of the latent space might mean the mixing time of Langevin dynamics might be very long. Hence, to avoid slow-sampling Markov Chains, one common method is to perform sampling using a carefully structured latent space [25, 26]. In this report, we also explore ways to reduce the mixing time by constructing latent spaces for efficient sampling using Langevin dynamics.

### 2.3 Upper-Confidence Bound and Thompson Sampling

One fundamental problem, especially when dealing with optimisation, is the exploitation vs exploration trade-off. To theoretically explore this trade-off, researchers typically explore the multi-armed bandit setting [27], which idealises the problem and allow researchers to focus on this trade-off. One example of the bandit setup is to imagine you are in a casino facing multiple slot machines. Each slot machine has its unknown probability of how likely you can get a reward at one play. The question is: what is the best strategy to achieve the highest rewards in the long term? There are two commonly applied algorithms for solving the multi-armed bandit problem in a tractable way in the bandit literature. The first is the Upper Confidence Bound (UCB) algorithm [28], and the second is Thompson sampling [29]. The idea of the UCB algorithm is to be optimistic about actions with high uncertainty and thus to prefer actions for which we haven't had a confident value estimation yet, thus favouring exploration of actions with a solid potential to have an optimal value. The UCB algorithm measures this potential by an upper confidence bound of the reward value, $\hat{U}_t(a)$, so that the true value is below with bound $Q(a) \leq \hat{Q}_t(a) + \hat{U}_t(a)$, where $Q(a) = \mathbb{E}[r \mid a]$, with high probability. and selects the greediest action to maximise the upper confidence bound:

$$a_t^{UCB} = \text{argmax}_{a \in A} \hat{Q}_t(a) + \hat{U}_t(a)$$

While, in the second approach, Thompson sampling, at each time step, we select action by sampling from the following posterior probability distribution as such

$$\pi(a \mid h_t) = \mathbb{E}_{R|h_t}[\mathbb{1}(a = \text{argmax}_{a \in A} Q(a))]$$

where $\pi(a \mid h_t)$ is the probability of taking action $a$ given the history $h_t$. After that, we observe the actual reward and update our posterior distribution accordingly, allowing us to balance between exploration and exploitation systematically.

## 3 Directions and Approaches Explored

As an overview, we first describe how posterior sampling can be used to perform estimation and then describe a problem faced by this approach in practice. Due to space constraints, we focus on conveying the key concepts and relegate details (e.g. proofs) to the appendix.

**Definition 1** (Posterior Sampling For Recovery)**.** Given an observation $y$, the posterior sampling recovery algorithm with respect to $P$ outputs $\hat{x}$ according to the posterior distribution $P(\cdot \mid y)$

Posterior sampling using generative priors have shown great promise in compressed sensing and other inverse problems, as shown in [21], who showed that by modelling the posterior distribution as an EBM where

$$E(\mathbf{z}) = -\frac{\|A(G(\mathbf{z})) - \mathbf{y}\|^2}{2\sigma^2/m} - \frac{\|\mathbf{z}\|^2}{2} + \log c(\mathbf{y})$$

we can prove instance optimality of the recovery algorithm for general distribution $R$ give $m$ measurements as long as $P$ and $R$ are close in Wasserstein distance.

In practice, due to the poor geometry of the latent space of DGMs, the mixing time for such algorithms can be very long. In this report, we explore two possibilities to overcome this problem; initialising near the manifold and applying an importance weight over the entire latent space to reshape the latent space into a more MCMC friendly geometry.

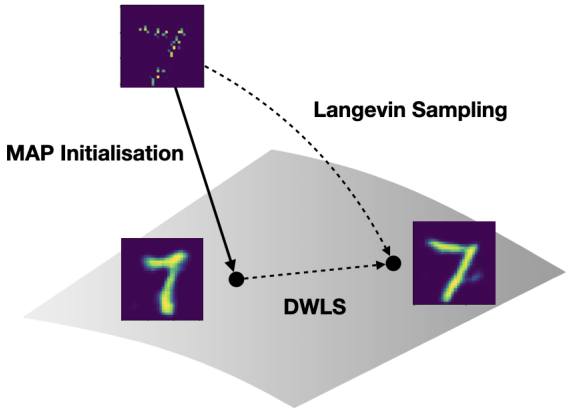


Figure 1: Illustration of MAP initialisation and DWLS sampling on an image manifold (represented as the surface)

### 3.1 Posterior and MAP Initialisation

One approach we explored to ensure quick mixing time for Langevin dynamics is to initialise near the manifold using an encoder to first encode the observations $\mathbf{y}$ before performing posterior sampling on the latent space. We call this approach posterior initialisation. This approach is beneficial if we have a trained encoder available such as in VAEs or Normalising Flows. For this method to work on operators that may not result in $\mathbf{y}$ having the same shape as $\mathbf{x}$ such as compressed sensing, we use a MAP estimate of $\mathbf{x}$ first and encode the estimate before sampling around it in the latent space. This second method we call MAP initialisation. We show that this method significantly reduces the number of steps required for Langevin dynamics to mix.

### 3.2 Discriminator Weighted Latent Sampling

Another approach we explored to reduce the mixing time for Langevin dynamics is to apply an importance weight over the entire latent space to reshape the latent space into a more MCMC friendly geometry. We propose to find this re-weighting function via the discriminator of a GAN. We call this approach Discriminator Weighted Latent Sampling (DWLS).

#### 3.2.1 Density Ratio Trick For GANs

Given a binary classifier (discriminator) $D$ that has been trained to distinguish between samples $\mu$ and $\rho$, the density-ratio $\rho(\mathbf{x})/\mu(\mathbf{x})$ can be estimated via the density ratio trick [30],

$$\frac{\rho(\mathbf{x})}{\mu(\mathbf{x})} \approx^{2} \frac{1 - D(y = 1 \mid \mathbf{x})}{D(y = 1 \mid \mathbf{x})} = \exp(-d(\mathbf{x})) \tag{1}$$

where $D(y = 1 \mid \mathbf{x})$ denotes the conditional probability of the sample $\mathbf{x}$ being from $\mu$ and $d(\mathbf{x})$ denotes the logit output of the classifier $D$.

**Lemma 1** (Latent Density Ratio)**.** *Let $g : \mathcal{Z} \to \mathcal{X}$ be a sufficiently well-behaved function where $\mathcal{Z} \subseteq \mathbb{R}^n$ and $\mathcal{X} \subset \mathbb{R}^m$ with $m > n$. Let $p_Z(\mathbf{z})$, $p_{\hat{Z}}(\hat{\mathbf{z}})$ be probability densities on and $q_X(\mathbf{x})$, $q_{\hat{X}}(\hat{\mathbf{x}})$ be the densities of the pushforward measures $g\sharp Z$, $g\sharp\hat{Z}$ respectively. Assume that $p_Z(\mathbf{z})$ and $p_{\hat{Z}}(\hat{\mathbf{z}})$ have same support, and the Jacobian matrix $\mathbf{J}_g$ has full column rank. Then, the density-ratio $p_{\hat{Z}}(\mathbf{u})/p_Z(\mathbf{u})$ at the point $\mathbf{u} \in \mathcal{Z}$ is given by*

$$\frac{p_{\hat{Z}}(\mathbf{u})}{p_Z(\mathbf{u})} = \frac{q_{\hat{X}}(g(\mathbf{u}))}{q_X(g(\mathbf{u}))}. \tag{2}$$

[2]If the discriminator is perfect, then this approximate sign becomes an equal sign.

$$\frac{p_Z(\mathbf{u})}{p_{\hat{Z}}(\mathbf{u})} = \frac{\rho(g_\theta(\mathbf{u}))}{\mu(g_\theta(\mathbf{u}))} = \exp(-d_\phi(g_\theta(\mathbf{u}))). \tag{3}$$

### 3.2.2 Importance Sampling Using Density Ratio

One way to use this density ratio is to apply the importance sampling formula directly to compute a good estimate for $z$.

$$\mathbb{E}\left[z \mid y\right] = \int z \ \frac{p_{\hat{Z}}(z \mid y)}{p_Z(z \mid y)} p_Z(z \mid y) dz$$

However, this method does not allow us to estimate the uncertainty because we do not have the posterior distribution to sample from.

### 3.2.3 Constructing Energy-Based Prior Using Density Ratio

We can instead construct the following energy-based model for the latent space of $\mathbf{z}$ to model the posterior distribution to sample from as such,

$$p_{\hat{Z}}(\mathbf{u}) = p_Z(\mathbf{u}) \exp(d(g_\theta(\mathbf{u})))/Z_\theta \tag{4}$$

Therefore,

$$\log p(\mathbf{z}) = \log p_Z(\mathbf{z}) + d(g_\theta(\mathbf{z}))) - \log Z_\theta \tag{5}$$

Hence,

$$\begin{aligned} E(\mathbf{z}) &= -\log p(\mathbf{z} \mid \mathbf{y}) \\ &= -\log p(\mathbf{y} \mid \mathbf{z}) - \log p(\mathbf{z}) + \log p(\mathbf{y}) \\ &= \frac{\left\|A(g_\theta(\mathbf{z})) - \mathbf{y}\right\|^2}{2\sigma^2/m} + \frac{\left\|\mathbf{z}\right\|^2}{2} - d_\phi(g_\theta(\mathbf{z})) + c(y) \end{aligned} \tag{6}$$

where $c(y)$ is a constant that depends only on $y$. Since we only care about the gradient of $\log p(\mathbf{z} \mid \mathbf{y})$, we can ignore this constant $c(y)$. Also, since $\sigma$ and $\eta$ (step-size) are hyperparameters we choose, we are effectively sampling from the following energy function

$$\tilde{E}(\mathbf{z}) = \left\|A(g_\theta(\mathbf{z})) - \mathbf{y}\right\|^2 + \frac{\lambda}{2}\left\|\mathbf{z}\right\|^2 - \lambda d_\phi(g_\theta(\mathbf{z}))$$

where $\lambda = \frac{2\sigma^2}{m}$.

**Algorithm 1** Discriminant weighted latent sampling

**Require:** generator ($g_\theta$), discriminator ($d_\phi$), number of update steps ($N$), step-size ($\eta$), noise factor ($\gamma$).

1: $\mathbf{z}_0 \sim p_Z(\mathbf{z})$ ▷ Sample from the prior.
2: **for** $i \leftarrow 1$ to $N$ **do**
3: $\quad \xi_i \sim \mathcal{N}(0, I)$
4: $\quad \mathbf{z}_{i+1} = \mathbf{z}_i - \eta \nabla_{\mathbf{z}} \tilde{E}(\mathbf{z}) + \sqrt{\eta}\xi_i$
5: **end for**
6: **if** denoise $Z_N$ **then** ▷ Optional denoising step, refer to appendix subsection B.1
7: $\quad$ **return** $g_\theta(\mathbf{z}_n) - \eta \nabla_{\mathbf{z}} \tilde{E}(\mathbf{z})$
8: **else**
9: $\quad$ **return** $g_\theta(\mathbf{z}_n)$

### 3.2.4 Discriminator Importance Weighting For All

One of the limitations of the construction above is that we are often unable to estimate the corresponding density ratio for $p_\theta/\mu$. Here, we propose a technique that extends our approach to refine samples from a larger class of DGMs such as VAEs and Normalizing Flows.

A naive approach is to use the crude approximation $p_\theta/\mu \approx p_\phi/\mu$, where $p_\theta$ is the density

of the samples generated by a generator $g_\theta$, $\mu$ is the density of the real data distribution and $p_\phi$ is density of the samples generated by a generator $g_\phi$ which is a different generative model $g_\phi$ trained on the same dataset as $g_\theta$. This approach works only if $p_\phi$ is close to $p_\theta$ which not always hold.

Another method proposed by [14][31], is to use train a density ratio corrector $p_\theta(x)/p_\phi(x)$ based on the samples produced from $g_\phi$ and $g_\theta$ as such

$$\frac{p_\theta(\mathbf{x})}{\mu(\mathbf{x})} = \frac{p_\phi(\mathbf{x})}{\mu(\mathbf{x})}\frac{p_\theta(\mathbf{x})}{p_\phi(\mathbf{x})} = \exp(-d_\phi(\mathbf{x})) \cdot \exp(-d_\lambda(\mathbf{x})), \tag{7}$$

Alternatively, another approach is to fine-tune the discriminator based on samples from $g_\theta$.

**Theorem 2.** *Assuming overlapping support between $p_\theta$ and $p_\theta$ and some regularity conditions such that the KL-divergence is differentiable*

$$\mathbb{E}_{z\sim P(z)}\left[\nabla_\phi D_\phi(g_\theta(z))\right] = \nabla_\phi \mathbb{KL}(p_\theta||p_\phi) \tag{8}$$

The proof is given in Section A.2. This suggests a straightforward idea for using the discriminator on a new generator, is to simply perform gradient descent on the backward KL divergence of $p_\theta$ and $p_\phi$ to minimise the difference between $p_\theta$ and $p_\phi$ i.e fine-tuning the discriminator. Then, using the new $D_\theta$ in (6) we can obtain samples from the posterior distribution.

### 3.3 Adaptive Setting

In adaptive sampling, we choose the subsequent measurements the model receives based on which pixels the model has the greatest uncertainty about to improve the model's estimate. The uncertainty can be estimated using the variance of the estimator. We can estimate the estimator's variance by taking various samples from the posterior and calculating the estimated variance. Then choose pixels with the highest measurements to be estimated as seen below in the following algorithm[3]

**Algorithm 2** Adaptive Sampling Using Deep Generative Prior For Inpainting

**for** $m \leftarrow 1$ to $M$ **do**
  $\mathbf{z}_0 \sim p_Z(\mathbf{z})$ ▷ Sample from the prior.
  **for** $i \leftarrow 1$ to $N$ **do**
    $\xi_i \sim \mathcal{N}(0, I)$
    $\mathbf{z}_{i+1} = \mathbf{z}_i - \eta\nabla_{\mathbf{z}} E(\mathbf{z} \mid \mathbf{y}_m, A_m) + \sqrt{\eta}\xi_i$
  **end for**
  $\widehat{\text{Var}(\mathbf{z})} = \frac{1}{N-l+1}\sum_{i=l}^{N}\mathbf{z}_i^2 - \left(\frac{1}{N-l+1}\sum_{i=l}^{N}\mathbf{z}_i\right)^2$
  $A_{m+1} \leftarrow \text{update}(A_m, \text{topk}(\widehat{\text{Var}(\mathbf{z})}, K))$
  $\mathbf{y}_{m+1} \leftarrow \text{update}(\mathbf{y}_m, \text{measure}(\mathbf{x}^*, \text{topk}(\widehat{\text{Var}(\mathbf{z})}, K))))$

where $E(\mathbf{z} \mid \mathbf{y}, A) = -\frac{\|A(G(\mathbf{z}))-\mathbf{y}\|^2}{2\sigma^2/m} - \frac{\|\mathbf{z}\|^2}{2} + \log c(\mathbf{y})$, $M$ is the number of adaptive steps that can be taken, $N$ is the number of samples from the posterior used to estimate the variance, $K$ is the number of measurements per adaptive step and $l$ is the burn-in period. This report focuses only on inpainting because inpainted measurements are easier to visualise and understand than noisy or compressed measurements. Furthermore, after each time step, we estimate and calculate the resulting mean squared error to evaluate our adaptive sample's performance.

### 3.4 Optimisation Setting

Finally, we explore the optimisation setting. Instead of finding the maximum variance like in the adaptive case, we are interested in finding the region with the highest mean intensity to efficiently find the maximum pixel in an image. There are two approaches we explored in the optimisation setting.

[3]The use of the update and measure function are implementation specific as they depend on how topk, $A_m$ and $y_m$ are being implemented for more details please refer to the github code at https://github.com/Neoanarika/inverse-probelms-urops.

The first is a Thomspon sampling, and the second is a UCB-type algorithm, both using DGMs as priors to sample from. The algorithms are as follows

**Algorithm 3** Thompson sampling using deep generative prior for inpainting

**for** $m \leftarrow 1$ to $M$ **do**
  $\mathbf{z}_0 \sim p_Z(\mathbf{z})$ ▷ Sample from the prior.
  **for** $i \leftarrow 1$ to $N$ **do**
    $\xi_i \sim \mathcal{N}(0, I)$
    $\mathbf{z}_{i+1} = \mathbf{z}_i - \eta \nabla_{\mathbf{z}} E(\mathbf{z} \mid \mathbf{y}_m, A_m) + \sqrt{\eta}\xi_i$
  **end for**
  $\bar{\mathbf{z}} = \frac{1}{N-l+1} \sum_{i=l}^{N} \mathbf{z}_i$
  $A_{m+1} \leftarrow \text{update}(A_m, \text{topk}(\bar{\mathbf{z}}, K))$
  $\mathbf{y}_{m+1} = \text{update}(\mathbf{y}_m, \text{measure}(\mathbf{x}^*, \text{topk}(\bar{\mathbf{z}}, K))))$

and

**Algorithm 4** UCB using deep generative prior for inpainting

**for** $m \leftarrow 1$ to $M$ **do**
  $\mathbf{z}_0 \sim p_Z(\mathbf{z})$ ▷ Sample from the prior.
  **for** $i \leftarrow 1$ to $N$ **do**
    $\xi_i \sim \mathcal{N}(0, I)$
    $\mathbf{z}_{i+1} = \mathbf{z}_i - \eta \nabla_{\mathbf{z}} E(\mathbf{z} \mid \mathbf{y}_m, A_m) + \sqrt{\eta}\xi_i$
  **end for**
  $\bar{\mathbf{z}} = \frac{1}{N-l+1} \sum_{i=l}^{N} \mathbf{z}_i$
  $\widehat{\text{Var}(\mathbf{z})} = \frac{1}{N-l+1} \sum_{i=l}^{N} \mathbf{z}_i^2 - \left( \frac{1}{N-l+1} \sum_{i=l}^{N} \mathbf{z}_i \right)^2$
  $A_{m+1} \leftarrow \text{update}(A_m, \text{topk}(\bar{\mathbf{z}} + \sigma \widehat{\text{Var}(z)}, K))$
  $\mathbf{y}_{m+1} = \text{update}(\mathbf{y}_m, \text{measure}(\mathbf{x}^*, \text{topk}(\bar{\mathbf{z}} + \sigma \widehat{\text{Var}(z)}, K)))$
**end for**=0

where $\sigma$ is hyperparameter that determines the trade off between choosing a more uncertain region vs a pixel with higher value.

## 4 Related Work

The idea of leveraging the posterior distributions to reduce the mixing time of Langevin dynamics is also found in other work [32]. In [32] the idea is to use the posterior to initialise the sample within a high-probability density region, preventing the sample from exploring low-density regions. This is similar to MAP initialisation, which tries to initialise near the image manifold where most of the images lie.

Previous work has also considered utilising the discriminator to achieve better sampling for GANs. Discriminator rejection sampling [16] and Metropolis-Hastings GANs [17] use the generator as the proposal distribution and the discriminator as the criterion of acceptance or rejection. However, these methods are inefficient as they may need to reject a lot of samples. More recently, methods like Discriminator Optimal Transport (DOT) [18], Discriminator Driven Latent Sampling (DDLS) [15], and Discriminator gradient flow (DGflow) [14] have used gradient-based approaches to improve the efficiency of this refinement process. In particular, our construction of our EBM for our prior distribution is based on the EBM construction used by DDLS. However, none of these papers has used these improved samples for downstream problems like reconstruction on an inverse problem or adaptive sampling before, and DDLS is only limited to GAN priors, but our method can generalise to use any generative models as a prior due to the fine-tuning step.

Recent work has also sought to use EBMs to structure the latent space [26, 33]. However, most of this work focused on training the EBM and base generative model concurrently, while our

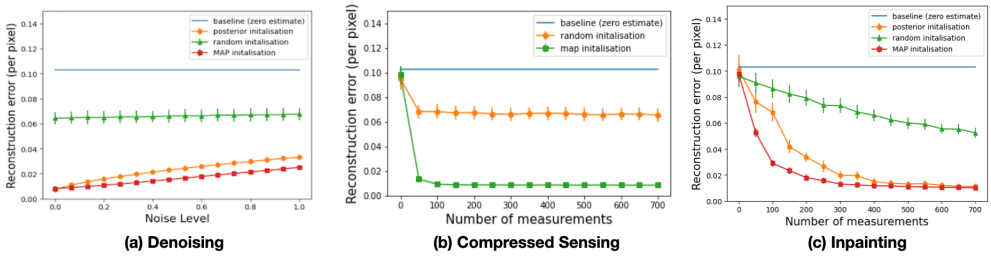


Figure 2: A plot of the per-pixel reconstruction error as we vary the number of measurements $m$ for inpainting and compressed sensing and across different noise level for denoising

work focuses on only using pre-trained discriminators and generators. More closely related to our work is the recent papers on using pre-trained generative priors [9, 21, 34] to perform either MAP estimation or posterior sampling. However, this work often discards the discriminator and only use the generator to perform sampling.

There has also been some work that proposes adaptive sampling to maximise the reconstruction quality of magnetic resonance images (MRI) given a limited sample budget [35, 36] by adaptively sampling the target. This is similar to our setup for adaptive sampling, except we focus on adaptive sampling for inpainting instead of other operators such as compressed sensing. Furthermore, we do not train a specific generator network to sample from but use a pre-trained generator and Langevin dynamics to then sample from the pre-trained network. This allows our method to leverage the wide number of pre-trained models for image generation instead of training our own.

Finally, to our knowledge, Bayesian optimisation with generative priors hasn't been widely explored. Bayesian optimisation typically uses Gaussian processes instead, rather than deep generative models [37]. While some papers have discussed using deep generative models for optimisation, such as in automated chemical design [38], these papers typically optimise through the latent spaces. Treating the generative model as an encoder and decoder to reduce the size of the search space rather than as a way we can sample a posterior distribution to aid in the optimisation.

## 5 Experiments

This section presents empirical results on various deep generative models trained on multiple synthetic and real-world datasets. Our primary goals were to determine (a) if the proposed MAP initialisation can reduce the mixing time of Langevin dynamics, (b) if DWLS is effective in improving the estimation process on a variety of different inverse problems and (c) if posterior sampling of deep generative models is helpful in the adaptive and optimisation settings.

We experimented with three different types of inverse problems, namely inpainting, denoising and compressed sensing on the MNIST dataset. The code is available online at https://github.com/Neoanarika/inverse-probelms-urops.

### 5.1 MAP Initialisation Experiments

In Figure 2, we show the performance of our proposed algorithm for inpainting on the MNIST dataset with a convolutional VAE model. The baselines we consider are random initialisation, posterior initialisation, and MAP initialisations. Random initialisation directly samples from $\mathcal{N}(0, 1)$ before performing Langevin sampling with our energy function $E(x)$. In contrast, posterior initialisation feeds the observation image $y$ into the posterior/encoder of the VAE and uses the resulting latent vector as initialisation for Langevin sampling. Lastly, MAP initialisation uses the posterior/encoder of the VAE like posterior initialisation but first perform MAP estimation before feeding the MAP estimate into the VAE encoder and using the latent vector as initialisation for Langevin sampling.

Notice that MAP initialisation outperforms both random and Posterior initialisation across any number of measurements in Figure 2. Furthermore, observe that MAP initialisation has the smallest variance and thus error bars out of the three methods. The small variance is probably because the MAP initial estimate is closer to the manifold, so the region of exploration is more restricted than posterior or

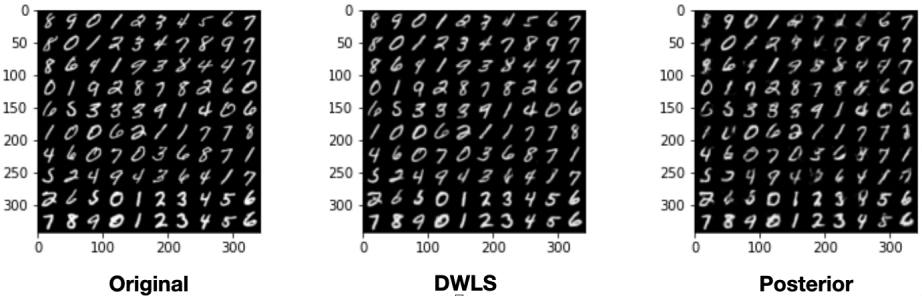


Figure 3: Reconstructed Samples From DWLS vs Posterior Sampling For Compressed Sensing

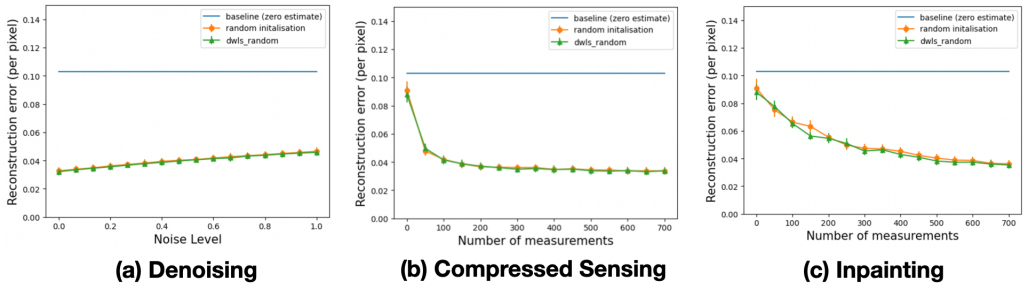


Figure 4: Reconstructed Samples From DWLS vs Posterior Sampling For Compressed Sensing

random initialisation resulting in the smaller variance. We obtain similar results for compressed sensing and denoising, as shown in Figure 2, further supporting our claims that MAP initialisation is helpful for posterior sampling.

### 5.2 Discriminator Weighted Latent Sampling (DWLS) Experiments

In Figure 3, we show that DWLS slightly improves the samples obtained compared to posterior sampling. However, we do not observe any significant improvements when comparing sampling with DWLS versus sampling without DLWS in both the GAN and VAE settings, as seen in Figures 4 and 5 regardless of initialisation or fine-tuning.

One possible reason that Figure 4 and 5 do not show any difference in the MSE is that MSE does not sufficiently capture the differences in the images. Using another more informative metric such as structural similarity index measure (SSIM) might be beneficial when evaluating the samples. Another possible reason is that what the discriminator penalises is what MSE and the Gaussian prior already penalise. The discriminator does not add much new information, and hence the MSE does

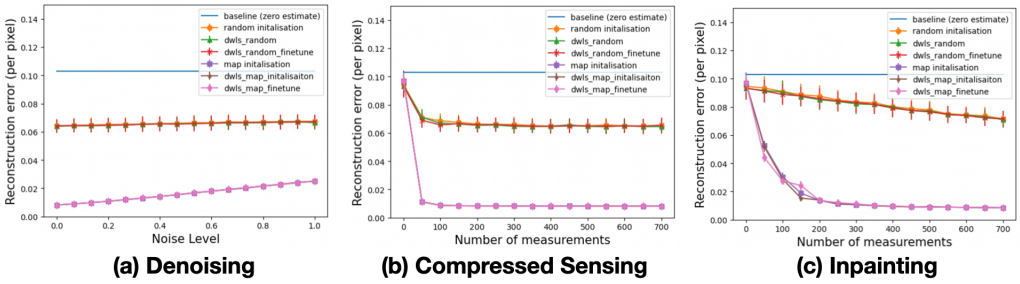


Figure 5: Comparing sampling with and without DWLS and with and without fine-tuning of the discriminator for VAES

not look significantly different or different at all in Figures 4 and 5 . This also partially explains why fine-tuning does not seem to have any effect on the MSE.

### 5.3 Adaptive Sampling Experiments

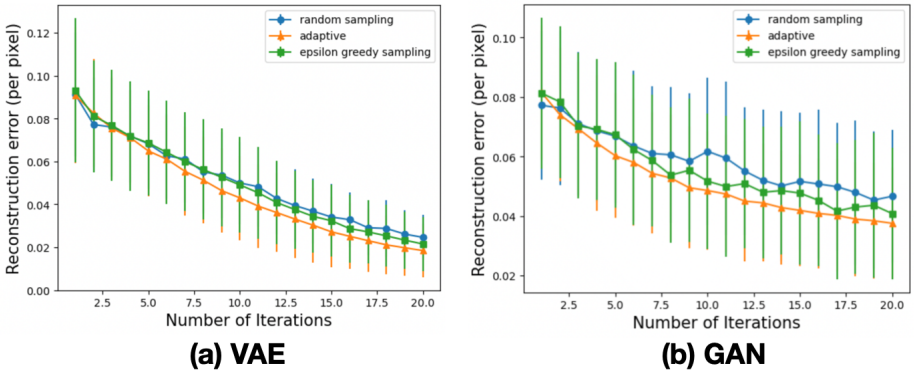


Figure 6: Adaptive Sampling For MNIST dataset

In the adaptive setting, we initially observed no pixels from MNIST. Then we choose which pixels to sample next based on 2 and sample the pixels repeating this till we run out of samples we can take. In this adaptive sampling setup, both the GAN and VAE adaptive sampling slightly outperforms random sampling and epsilon-greedy sampling using the same models as can be seen from figure 6. This indicates that samples chosen based on how much variance they produce during sampling can help in improving the estimates of the GAN and VAE models. The large variance for both VAE and GAN adaptive sampling is because the internal sampling during each time step is unlikely to fully converge due to the limited number of steps resulting in the higher variance. The higher variance and MSE for GANs compared to VAEs is because they take longer to converge when sampling from their posterior, resulting in more variation and poorer estimates when we stop sampling before Langevin can converge.

### 5.4 Optimisation Experiments

We augmented the MNIST dataset by adding a gradient filter over the original images; for more information, refer to Section B.2 in the appendix. The goal is to estimate where the maximum pixel is, and the performance, i.e. error term's, is measured by the absolute difference between the maximum and the estimated maximum pixel. For our baseline, we randomly pick images in the training set and choose their maximum pixels as the estimated maximum pixels of our target image. In this setup, we found that using VAE as the base model, UCB and Thompson sampling converged slower at the same rate or possibly even slower than our baseline across 25 randomly chosen images. While for GANs, both UCB and Thompson sampling converged faster than our baseline across the same 25 randomly chosen images as shown in figure 7. This difference might be because VAEs are poorer at capturing the gradient filter applied across the images than the GAN model. The difference in the generator ability can be seen in Figure 8. This suggests, unsurprisingly, that a good generator is required for effective optimisations.

## 6 Conclusion and Future Work

This report explores four different directions. The first direction we explore is MAP initialisation, which allows us to sample the posterior near the underlying manifold, resulting in faster convergence than random initialisation. Another direction we explored is using the discriminator to produce importance weights for the latent space. We also showed how to apply the technique to commonly used deep generative models: GANs and VAEs. However, we did not manage to show that importance sampling significantly improves the accuracy nor convergence of the sampling process. We also explored the adaptive sampling setting, where we found that adaptive sampling works better than random sampling for both GANs and VAEs. Finally, in the optimisation setting, UCB and Thompson

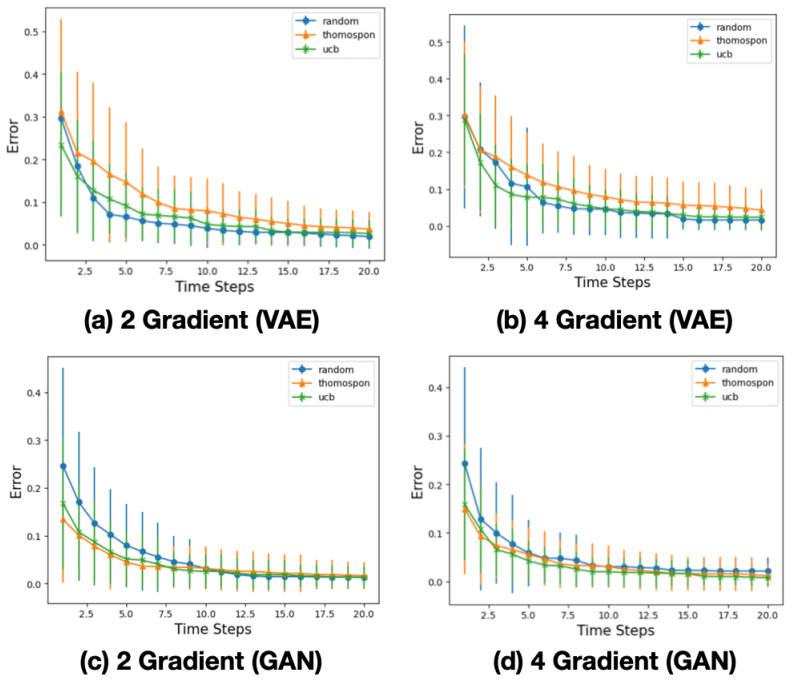


Figure 7: Optimisation results on multi-gradient MNIST dataset

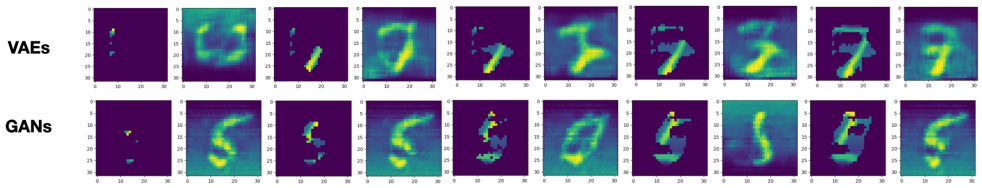


Figure 8: Thompson samples from GAN and VAE for 2-Gradient MNIST dataset

sampling did not do better than our baseline for VAEs but did better for GANs. The difference in performance is likely due to GANs learning the data distribution more accurately than VAEs and thus better capturing the gradient. Moving forward, we are considering exploring using more powerful models and architectures such as score-based generative models [39], or Normalising flows [19] to test our results on larger image datasets. Similarly, we can explore more complex task for the optimisation setting as well, such as automatic chemical design [38] where the goal is to optimise specific target chemical properties in molecules. Lastly, another direction that could be interesting is to sample across various layers of the generator. This idea is based on iterative layer optimisation [40], which progressively changes the input layer, obtaining more expressive generators successively instead of optimising only over the initial latent code.

## References


[1] Harrison H Barrett and Kyle J Myers. *Foundations of image science*. John Wiley & Sons, 2013.

[2] Andrej Nikolaevič Tihonov and Vasilij Jakovlevič Arsenin. *Solutions of ill-posed problems*. Winston, 1977.

[3] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696, 2009.

[4] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[5] Gregory Ongie, Ajil Jalal, Christopher A. Metzler, Richard G. Baraniuk, Alexandros G. Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging, 2020.

[6] Ali Mousavi, Ankit B Patel, and Richard G Baraniuk. A deep learning approach to structured signal recovery. In *2015 53rd annual allerton conference on communication, control, and computing (Allerton)*, pages 1336–1343. IEEE, 2015.

[7] Yan Yang, Jian Sun, Huibin Li, and Zongben Xu. Deep admm-net for compressive sensing mri. In *Proceedings of the 30th international conference on neural information processing systems*, pages 10–18, 2016.

[8] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.

[9] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed sensing using generative models, 2017.

[10] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks, 2021.

[11] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning, 2016.

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[14] Abdul Fatir Ansari, Ming Liang Ang, and Harold Soh. Refining deep generative models via discriminator gradient flow, 2021.

[15] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. Your gan is secretly an energy-based model and you should use discriminator driven latent sampling, 2021.

[16] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. Discriminator rejection sampling, 2019.

[17] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatci, and Jason Yosinski. Metropolis-hastings generative adversarial networks, 2019.

[18] Akinori Tanaka. Discriminator optimal transport, 2019.

[19] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

[20] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[21] Ajil Jalal, Sushrut Karmalkar, Alexandros G Dimakis, and Eric Price. Instance-optimal compressed sensing via posterior sampling. *arXiv preprint arXiv:2106.11438*, 2021.

[22] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

[23] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[24] Ankur Moitra and Andrej Risteski. Fast convergence for langevin diffusion with manifold structure. *arXiv preprint arXiv:2002.05576*, 2020.

[25] Matthew Hoffman, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. Neutra-lizing bad geometry in hamiltonian monte carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

[26] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. *arXiv preprint arXiv:2006.08205*, 2020.

[27] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[28] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[29] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[30] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[31] Benjamin Rhodes, Kai Xu, and Michael U Gutmann. Telescoping density-ratio estimation. *arXiv preprint arXiv:2006.12204*, 2020.

[32] Meng Qu, Tianyu Gao, Louis-Pascal Xhonneux, and Jian Tang. Few-shot relation extraction via bayesian meta-learning on relation graphs. In *International Conference on Machine Learning*, pages 7867–7876. PMLR, 2020.

[33] Michael Arbel, Liang Zhou, and Arthur Gretton. Generalized energy based models, 2020.

[34] Jonas Adler and Ozan Öktem. Deep bayesian inversion. *arXiv preprint arXiv:1811.05910*, 2018.

[35] Kyong Hwan Jin, Michael Unser, and Kwang Moo Yi. Self-supervised deep active accelerated mri. *arXiv preprint arXiv:1901.04547*, 2019.

[36] Thomas Sanchez, Igor Krawczuk, Zhaodong Sun, and Volkan Cevher. Closed loop deep bayesian inversion: Uncertainty driven acquisition for fast mri. 2019.

[37] Apoorv Agnihotri and Nipun Batra. Exploring bayesian optimization. *Distill*, 5(5):e26, 2020.

[38] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[39] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[40] Giannis Daras, Joseph Dean, Ajil Jalal, and Alexandros G. Dimakis. Intermediate layer optimization for inverse problems using deep generative models, 2021.

[41] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *CoRR*, abs/2006.09011, 2020.

[42] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

# A Proofs

## A.1 Proof of Lemma 2 to prove Theorem 2

**Lemma 2.**

$$\mathbb{E}_{x\sim P_\theta(x)}\left[\log\frac{p_\phi(x)}{\mu(x)}\right] = \mathbb{KL}(p_\theta||\mu) - \mathbb{KL}(p_\theta||p_\phi)$$

*Proof.*

$$\begin{aligned}
\mathbb{E}_{x\sim P_\theta(x)}\left[\log\frac{p_\phi(x)}{\mu(x)}\right] &= \int p_\theta(x)\log\frac{p_\phi(x)}{\mu(x)}dx \\
&= \int p_\theta(x)\log\frac{p_\theta(x)}{\mu(x)}\frac{p_\phi(x)}{p_\theta(x)}dx \\
&= \int p_\theta(x)\log\frac{p_\theta(x)}{\mu(x)} + p_\theta(x)\log\frac{p_\phi(x)}{p_\theta(x)}dx \\
&= \int p_\theta(x)\log\frac{p_\theta(x)}{\mu(x)} - p_\theta(x)\log\frac{p_\theta(x)}{p_\phi(x)}dx \\
&= \mathbb{KL}(p_\theta||\mu) - \mathbb{KL}(p_\theta||p_\phi)
\end{aligned}$$

□

## A.2 Proof of Theorem 2

*Proof.*

$$\begin{aligned}
\nabla_\phi\mathbb{E}_{x\sim P_\theta(x)}\left[\log\frac{p_\phi(x)}{\mu(x)}\right] &= \nabla_\phi\mathbb{KL}(p_\theta||\mu) - \mathbb{KL}(p_\theta||p_\phi) \\
&= -\nabla_\phi\mathbb{KL}(p_\theta||p_\phi) \qquad (9)
\end{aligned}$$

Plugging (9) to (1), we see that

$$\mathbb{E}_{x\sim P_\theta(x)}\left[\nabla_\phi d_\phi(x)\right] = \nabla_\phi\mathbb{KL}(p_\theta||p_\phi)$$

Therefore,

$$\mathbb{E}_{z\sim P(z)}\left[\nabla_\phi d_\phi(g_\theta(z))\right] = \nabla_\phi\mathbb{KL}(p_\theta||p_\phi)$$

□

# B Experimental Notes

## B.1 Usefulness of the denoising step

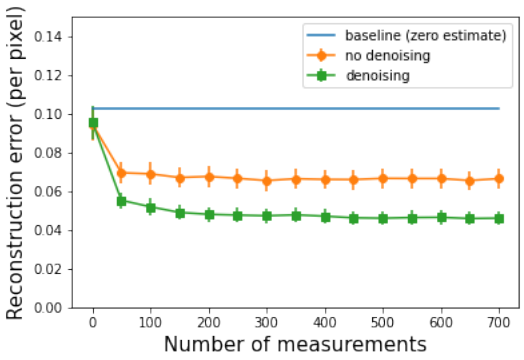


Figure 9: Denoising ($\sigma = 0.1$), vs no-denoising

Several papers [21][41] have reported using the denoising step as a final stage in the estimation procedure. This is based on the Tweedie formula from Bayesian statistics [42]. An intuitive explanation for why the denoising step is used is that we typically take the average of several samples, which might make the resulting estimate "blur". One way to combat this is to denoise it. As seen in Figure 9, we found that using this final denoising step helps improve the estimates when the hyperparameters $\sigma$ is chosen appropriately. In this work, we did not use this denoising step in any of the experiments we reported. Thus, the experimental results we obtain in the report can be easily improved with this final denoising step if an appropriate $\sigma$ is chosen.

## B.2 Multi-gradient Augmentation

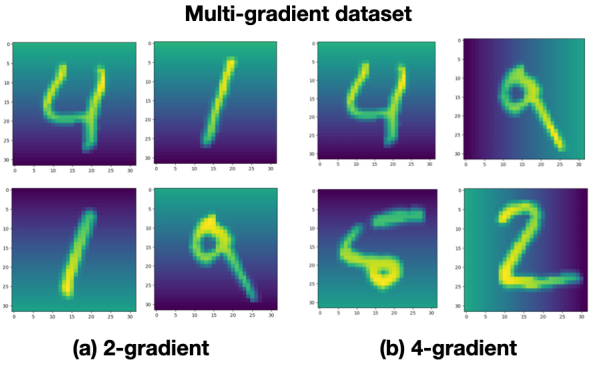


Figure 10: Multi-gradient dataset

We augmented the MNIST dataset by adding a gradient filter over the original images which is randomly applied throughout the dataset such that each the different filters have the same chance of occurring on any image. There are two types of augmentations available 2-gradient or 4-gradient. In the 2-gradient setup we have two directions: top and down. While in the 4-gradient setup we have 4-directions: top, down, left and right. The goal is to try to sample from the image to eventually find the maximum pixel. We evaluate the performance on this dataset by taking the difference between the maximum pixel of the image and the estimated maximum pixel.