MULTISCALE INVERTIBLE GENERATIVE NETWORKS FOR HIGH-DIMENSIONAL BAYESIAN INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

High-dimensional Bayesian inference problems cast a long-standing challenge in generating samples, especially when the posterior has multiple modes. For a wide class of Bayesian inference problems equipped with the *multiscale structure* that low-dimensional (coarse-scale) surrogate can approximate the original high-dimensional (fine-scale) problem well, we propose to train a Multiscale Invertible Generative Network (MsIGN) for sample generation. A novel prior conditioning layer is designed to bridge networks at different resolutions, enabling coarse-to-fine multi-stage training. Jeffreys divergence is adopted as the training objective to avoid mode dropping. On two high-dimensional Bayesian inverse problems, MsIGN approximates the posterior accurately and clearly captures multiple modes, showing superior performance compared with previous deep generative network approaches. On the natural image synthesis task, MsIGN achieves the superior performance in bits-per-dimension compared with our baseline models and yields great interpret-ability of its neurons in intermediate layers.

1 INTRODUCTION

Bayesian inference provides a powerful framework to blend prior knowledge, data generation process and (possibly small) data for statistical inference. With some prior knowledge ρ (distribution) for the quantity of interest $\boldsymbol{x} \in \mathbb{R}^d$, and some (noisy) measurement $\boldsymbol{y} \in \mathbb{R}^{d_y}$, it casts on $\boldsymbol{x}$ a posterior

$$q(\mathbf{x}|\boldsymbol{y}) \propto \rho(\mathbf{x}) L(\boldsymbol{y}|\mathbf{x}) \,, \quad \text{where} \quad L(\boldsymbol{y}|\mathbf{x}) = \mathcal{N}(\boldsymbol{y} - \mathcal{F}(\mathbf{x}); \mathbf{0}, \boldsymbol{\Gamma}_{\boldsymbol{\varepsilon}}) \,. \tag{1}$$

where $L(\boldsymbol{y}|\mathbf{x})$ is the likelihood that compares the data $\boldsymbol{y}$ with system prediction $\mathcal{F}(\mathbf{x})$ from the candidate $\mathbf{x}$, here $\mathcal{F}$ denotes the forward process. We can use different distributions to model the mismatch $\boldsymbol{\varepsilon} = \boldsymbol{y} - \mathcal{F}(\mathbf{x})$, and for illustration simplicity, we assume Gaussian in Equation 1. For example, Bayesian deep learning generates model predicted logits $\mathcal{F}(\mathbf{x})$ from model parameters $\mathbf{x}$, and compares it with discrete labels $\boldsymbol{y}$ through binomial or multinomial distribution.

Sampling or inferring from q is a long-standing challenge, especially for high-dimensional (high-d) cases. An arbitrary high-d posterior can have its importance regions (also called "modes") anywhere in the high-d space, and finding these modes requires computational cost that grows exponentially with the dimension d. This intrinsic difficulty is the consequence of "the curse of dimensionality", which all existing Bayesian inference methods suffer from, e.g., MCMC-based methods (Neal et al., 2011; Welling & Teh, 2011; Cui et al., 2016), SVGD-type methods (Liu & Wang, 2016; Chen et al., 2018; 2019a), and generative modeling (Morzfeld et al., 2012; Parno et al., 2016; Hou et al., 2019).

In this paper, we focus on Bayesian inference problems with multiscale structure and exploit this structure to sample from a high-d posterior. While the original problem has a high spatial resolution (fine-scale), its low resolution (coarse-scale) analogy is computationally attractive because it lies in a low-dimension (low-d) space. A problem has the multiscale structure if such coarse-scale low-d surrogate exists and gives good approximation to the fine-scale high-d problem, see Section 2.1. Such multiscale property is very common in high-d Bayesian inference problems. For example, inferring 3-D permeability field of subsurface at the scale of meters is a reasonable approximation of itself at the scale of centimeters, while the problem dimension is 10^6-times fewer.

We propose a Multiscale Invertible Generative Network (MsIGN) to sample from high-d Bayesian inference problems with multiscale structure. MsIGN is a flow-based generative network that can both generate samples and give density evaluation. It consists of multiple scales that recursively

lifts up samples to a finer-scale (higher-resolution), except that the coarsest scale directly samples from a low-d (low resolution) distribution. At each scale, a fixed prior conditioning layer combines coarse-scale samples with some random noise according to the prior to enhance the resolution, and then an invertible flow modifies the samples for better accuracy, see Figure 1. The architecture of MsIGN makes it fully invertible between the final sample and random noise at all scales.

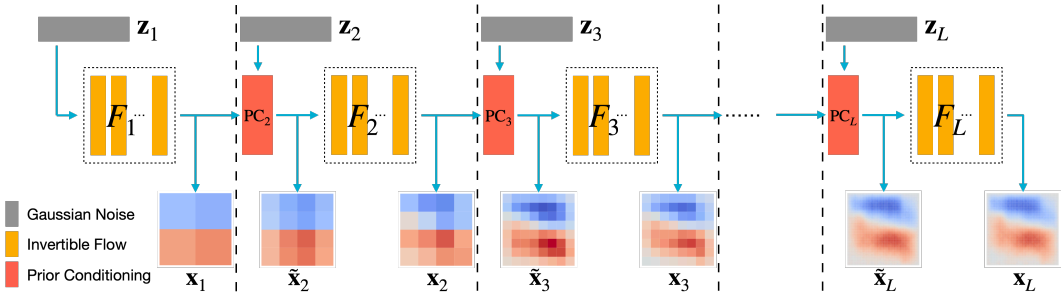

Figure 1: MsIGN generates samples from coarse to fine scale. Each scale, as separated by vertical dash lines, takes in feature $\mathbf{x}_{l-1}$ from the coarser scale and Gaussian noise $\mathbf{z}_l$, and outputs a sample $\mathbf{x}_l$ of finer scale. The prior conditioning layer PC_l lifts up the coarser-scale sample $\mathbf{x}_{l-1}$ to a finer scale $\tilde{\mathbf{x}}_l$, which is the best guess of $\mathbf{x}_l$ given its coarse-scale value $\mathbf{x}_{l-1}$ and the prior. An invertible flow F_l further modifies $\tilde{\mathbf{x}}_l$ to better approximate $\mathbf{x}_l$. See Section 2.1 for detailed explanation.

MsIGN undergoes a multi-stage training that learns a hierarchy of distributions with dimensions growing from the lowest to the highest (the target posterior). Each stage gives a good initialization to the next stage thanks to the multiscale property. To capture multiple modes, we choose Jeffreys divergence $D_{\mathrm{J}}(p\|q)$ as the training objective at each stage, which is defined as

$$D_{\mathrm{J}}(p\|q) = D_{\mathrm{KL}}(p\|q) + D_{\mathrm{KL}}(q\|p) = \mathbb{E}_{\mathbf{x}\sim p}\left[\log\left(p(\mathbf{x})/q(\mathbf{x})\right)\right] + \mathbb{E}_{\mathbf{x}\sim q}\left[\log\left(q(\mathbf{x})/p(\mathbf{x})\right)\right] . \quad (2)$$

Jeffreys divergence removes bad local minima of single-sided Kullback-Leibler (KL) divergence to avoid mode missing. We build an unbiased estimation of it by leveraging prior conditioning layer in importance sampling. Proper loss function and good initialization from multi-stage training solve the non-convex optimization stably and capture multi-modes of the high-d distribution.

In summary, we claim four contributions in this work. First, we propose a Multiscale Invertible deep Generative Network (MsIGN) with a novel prior conditioning layer, which can be trained in a coarse-to-fine scale manner. Second, Jeffreys divergence is used as the objective function to avoid mode collapse, and is estimated by importance sampling based on the prior conditioning layer. Third, when applied to two Bayesian inverse problems, MsIGN clearly captures multiple modes in the high-d posterior and approximates the posterior accurately, demonstrating its superior performance compared with previous methods via the generative modeling approach. Fourth, we also apply MsIGN to image synthesis tasks, where it achieves superior performance in bits-per-dimension among our baseline models, like Glow (Kingma & Dhariwal, 2018), FFJORD (Grathwohl et al., 2018), Flow++ (Ho et al., 2019), i-ResNet (Behrmann et al., 2019), and Residual Flow (Chen et al., 2019b). MsIGN also yields great interpret-ability of its neurons in intermediate layers.

2 METHOLOGY

We will abbreviate $q(\mathbf{x}|\boldsymbol{y})$ in Equation 1 as $q(\mathbf{x})$ for simplicity in the following context, because $\boldsymbol{y}$ *only* plays the role of defining the target distribution $q(\mathbf{x})$ in MsIGN. In Section 2.1, we discuss the multiscale structure in detail of the posterior $q(\mathbf{x})$ and derive a scale decoupling that can be utilized to divide and conquer the high-d challenge of Bayesian inference.

As a flow-based generative model like in Dinh et al. (2016), MsIGN models a bijective that maps Gaussian noise $\mathbf{z}$ to a sample $\mathbf{x}$ whose distribution is denoted as $p_\theta(\mathbf{x})$, where θ is the network parameters. MsIGN allows fast generation of samples $\mathbf{x}$ and density evaluation $p_\theta(\mathbf{x})$, so we train our working distribution $p_\theta(\mathbf{x})$ to approximate the target distribution $q(\mathbf{x})$. We present the architecture of MsIGN in Section 2.2 and the training algorithm in Section 2.3.

2.1 MULTISCALE STRUCTURE AND SCALE DECOUPLING

We say a Bayesian inference problem has *multiscale structure* if the associated coarse-scale likelihood L_c approximates the original likelihood L well:

$$L(\boldsymbol{y}|\mathbf{x}) \approx L_c(\boldsymbol{y}|\mathbf{x}_c)\,, \quad \text{where} \quad L_c(\boldsymbol{y}|\mathbf{x}_c) := \mathcal{N}(\boldsymbol{y} - \mathcal{F}_c(\mathbf{x}_c); \mathbf{0}, \boldsymbol{\Gamma}_{\boldsymbol{\varepsilon}})\,. \tag{3}$$

Here $\mathbf{x}_c \in \mathbb{R}^{d_c}$ is a coarse-scale version of the fine-scale quantity $\mathbf{x} \in \mathbb{R}^d$ ($d_c < d$), given by a deterministic pooling operator $\mathcal{A} : \mathbf{x}_c = \mathcal{A}(\mathbf{x})$. The map $\mathcal{F}_c : \mathbb{R}^{d_c} \to \mathbb{R}^{d_y}$ is a forward process that gives system prediction based on the coarse-scale information $\mathbf{x}_c$. A popular case of the multiscale structure is when $\mathcal{A}$ is the average pooling operator, and $\mathcal{F}(\mathbf{x}) \approx \mathcal{F}_c(\mathbf{x}_c)$, meaning that the system prediction mainly depends on the lower-resolution information $\mathbf{x}_c$. Equation 3 motivates us to define a surrogate distribution $\tilde{q}(\mathbf{x}) \propto \rho(\mathbf{x})L_c(\boldsymbol{y}|\mathcal{A}(\mathbf{x}))$ that approximates the target posterior $q(\mathbf{x})$ well[1]:

$$\tilde{q}(\mathbf{x}) = \rho(\mathbf{x})L_c(\boldsymbol{y}|\mathcal{A}(\mathbf{x})) = \rho(\mathbf{x})L_c(\boldsymbol{y}|\mathbf{x}_c) \approx \rho(\mathbf{x})L(\boldsymbol{y}|\mathbf{x}) = q(\mathbf{x})\,. \tag{4}$$

We also notice that the prior ρ allows an exact scale decoupling. To generate a sample $\mathbf{x}$ from ρ, one can first sample its coarse-scale version $\mathbf{x}_c = \mathcal{A}(\mathbf{x})$, and then replenish missing fine-scale details without changing the coarse-scale structure by sampling from the conditional distribution $\rho(\mathbf{x}|\mathbf{x}_c) = \rho(\mathbf{x}|\mathcal{A}(\mathbf{x}) = \mathbf{x}_c)$. Using ρ_c to denote the distribution of $\mathbf{x}_c = \mathcal{A}(\mathbf{x})$, the conditional probability calculation summarizes this scale decoupling process as $\rho(\boldsymbol{x}) = \rho(\boldsymbol{x}|\boldsymbol{x}_c)\rho_c(\boldsymbol{x}_c)$.

Combining the scale effect in the likelihood and the scale decoupling in the prior, we decouple the surrogate $\tilde{q}(\mathbf{x}) = \rho(\mathbf{x})L_c(\boldsymbol{y}|\mathcal{A}(\mathbf{x}))$ into the prior conditional distribution $\rho(\mathbf{x}|\mathbf{x}_c)$ and a coarse-scale posterior, defined as $q_c(\mathbf{x}_c) := \rho_c(\mathbf{x}_c)L(\boldsymbol{y}|\mathbf{x}_c)$. The decoupling goes as

$$\tilde{q}(\mathbf{x}) = \rho(\mathbf{x})L_c(\boldsymbol{y}|\mathbf{x}_c) = \rho(\mathbf{x}|\mathbf{x}_c)\rho_c(\mathbf{x}_c)L_c(\boldsymbol{y}|\mathbf{x}_c) = \rho(\mathbf{x}|\mathbf{x}_c)q_c(\mathbf{x}_c)\,, \tag{5}$$

The prior conditional distribution $\rho(\mathbf{x}|\mathbf{x}_c)$ bridges the coarse-scale posterior $q_c(\mathbf{x}_c)$ and the surrogate $\tilde{q}(\mathbf{x})$, which in turn approximates the original fine-scale posterior $q(\mathbf{x})$. Parno et al. (2016) proposed a similar scale decoupling relation, and we leave the discussion and comparison to Appendix A.

Figure 1 shows the integrated sampling strategy. To sample an $\mathbf{x}$ from q, we start with an $\mathbf{x}_c$ from q_c. The prior conditioning layer then performs random upsampling from the prior conditional distribution $\rho(\cdot|\mathbf{x}_c)$, and the output will be a sample $\tilde{\mathbf{x}}$ of the surrogate $\tilde{q}$. Due to the approximation $\tilde{q} \approx q$ from Equation 4, we stack multiple invertible blocks for the invertible flow F to modify the sample $\tilde{\mathbf{x}} \sim \tilde{q}$ to a sample $\mathbf{x} \sim q$: $\mathbf{x} = F(\tilde{\mathbf{x}})$. F is initialized as an identity map in training. Finally, to obtain the $\mathbf{x}_c$ from q_c, we apply the above procedure recursively until the dimension of the coarsest scale is small enough so that q_c can be easily sampled by a standard method.

2.2 MULTISCALE INVERTIBLE GENERATIVE NETWORK: ARCHITECTURE

Our proposed MsIGN has multiple levels to recursively apply the above strategy. We denote L the number of levels, $\mathbf{x}_l \in \mathbb{R}^{d_l}$ the sample at level l, and $\mathcal{A}_l : \mathbb{R}^{d_l} \to \mathbb{R}^{d_{l-1}}$ the pooling operator from level l to $l-1$: $\mathbf{x}_{l-1} = \mathcal{A}_l(\mathbf{x}_l)$. Following the idea in Section 2.1, we can define the l-th level target $q_l(\mathbf{x}_l)$ and surrogate $\tilde{q}_l(\tilde{\mathbf{x}}_l)$, and the last-level target q_L is our original target q in Equation 1. The l-th level of MsIGN uses a prior conditioning layer PC_l and an inverse transform F_l to capture q_l.

Prior conditioning layer. The prior conditioning layer PC_l at level l lifts a coarse-scale sample $\mathbf{x}_{l-1} \in \mathbb{R}^{d_{l-1}}$ up to a *random* fine-scale one $\mathbf{x}_l \in \mathbb{R}^{d_l}$ following the conditional distribution $\rho(\mathbf{x}_l|\mathbf{x}_{l-1})$. The difference in dimension is compensated by a Gaussian noise $\mathbf{z}_l \in \mathbb{R}^{d_l - d_{l-1}}$, which is the source of randomness: $\mathbf{x}_l = \mathrm{PC}_l(\mathbf{x}_{l-1}, \mathbf{z}_l)$. PC_l depends only on the prior conditional distribution $\rho(\mathbf{x}_l|\mathbf{x}_{l-1})$, and thus can be pre-computed *independently for different levels* regardless of the likelihood L. When the prior is Gaussian and the pooling operators are linear (e.g., average pooling), the prior conditional distribution is still Gaussian with moments specified as follows.

Lemma 2.1 *Suppose that $\rho(\mathbf{x}_l) = \mathcal{N}(\mathbf{x}_l; \mathbf{0}, \boldsymbol{\Sigma}_l)$, and $\mathcal{A}_l(\mathbf{x}_l) = \boldsymbol{A}_l\mathbf{x}_l$ for some $\boldsymbol{A}_l \in \mathbb{R}^{d_{l-1}\times d_l}$, then with $\boldsymbol{U}_{l-1} := \boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}$ and $\boldsymbol{\Sigma}_{l|l-1} := \boldsymbol{\Sigma}_l - \boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l$, we have*

$$\rho(\mathbf{x}_l|\mathbf{x}_{l-1} = \boldsymbol{A}_l\mathbf{x}_l) = \mathcal{N}(\mathbf{x}_l; \boldsymbol{U}_{l-1}\mathbf{x}_{l-1}, \boldsymbol{\Sigma}_{l|l-1})\,.$$

[1] We omit normalizing constants. Equivalence and approximation are up to normalization in the following.

With the Cholesky decomposition (or eigen-decomposition) $\mathbf{\Sigma}_{l|l-1} = \boldsymbol{B}_l \boldsymbol{B}_l^T$, we design the prior conditioning layer PC_l as below, which is invertible between $\mathbf{x}_l$ and $(\mathbf{x}_{l-1}, \mathbf{z}_l)$:

$$\mathbf{x}_l = \mathrm{PC}_l(\mathbf{x}_{l-1}, \mathbf{z}_l) := \boldsymbol{U}_{l-1}\mathbf{x}_{l-1} + \boldsymbol{B}_l \mathbf{z}_l \,, \quad \mathbf{z}_l \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_l - d_{l-1}}) \,. \tag{6}$$

We refer readers to Appendix B for proof of Lemma 2.1 and the invertibility in Equation 6.

When the prior is non-Gaussian or the pooling operators are nonlinear, there exists a *nonlinear* invertible prior conditioning operator $\mathbf{x}_l = \mathrm{PC}_l(\mathbf{x}_{l-1}, \mathbf{z}_l)$ such that $\mathbf{x}_l$ follows the prior conditional distribution $\rho(\mathbf{x}_l|\mathbf{x}_{l-1})$ given $\mathbf{x}_{l-1}$ and $\mathbf{z}_l \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_l - d_{l-1}})$. We can pre-train an invertible network to approximate this sampling process, and fix it as the prior conditioning layer.

Invertible flow. The invertible flow F_l at level l modifies the surrogate $\tilde{q}_l$ towards the target q_l. The more accurate the multiscale structure in Equation 3 is, the better $\tilde{q}_l$ approximates q_l, and the closer F_l is to the identity map. Therefore, we parameterize F_l by some flow-based generative model and initialize it as an identity map. In practice, we utilize the invertible block of Glow (Kingma & Dhariwal, 2018), which consists of actnorm, invertible 1×1 convolution, and affine coupling layer, and stack several blocks as the inverse flow F_l in MsIGN.

Overall model. MsIGN is a bijective map between random noise inputs at different scales $\{\mathbf{z}_l\}_{l=1}^L$ and the finest-scale sample $\mathbf{x}_L$. The forward direction of MsIGN maps $\{\mathbf{z}_l\}_{l=1}^L$ to $\mathbf{x}_L$ as below:

$$\begin{aligned} & \mathbf{x}_1 = F_1(\mathbf{z}_1) \,, \\ \tilde{\mathbf{x}}_l = \mathrm{PC}_l(\mathbf{x}_{l-1}, \mathbf{z}_l) \,, \quad & \mathbf{x}_l = F_l(\tilde{\mathbf{x}}_l) \,, \quad 2 \le l \le L \,. \end{aligned} \tag{7}$$

As a flow-based generative model, sample generation as in Equation 7 and density evaluation $p_\theta(\mathbf{x})$ by the change-of-variable rule is accessible and fast for MsIGN. In scenarios when certain bound needs enforcing to the output, we can append element-wise output activations at the end of MsIGN. For example, image synthesis can use the sigmoid function so that pixel values lie in $[0, 1]$. Such activations should be bijective to keep the invertible relation between random noise to the sample.

2.3 Multiscale Invertible Generative Network: Training

Since the prior conditioning layer PC is pre-computed and the output activation G is fixed, only the inverse flow F contains trainable parameters in MsIGN. We train MsIGN with the following strategy so that the distribution p_θ of its output samples, where θ is the network parameter, can approximate the target distribution q defined in Equation 1 well.

Multi-stage training and interpret-ability. The multiscale strategy in construction of MsIGN enables a coarse-to-fine multi-stage training. At stage l, we target at capturing q_l, and only train invertible flows before or at this level: $F_{l'}, l' \le l$. Equation 4 implies that q_l can be well approximated by the surogate $\tilde{q}_l$, which is the conditional upsampling from q_{l-1} as in Equation 5. So we use $\tilde{q}_l$ to initialize our model by setting $F_{l'}, l' < l$ as the trained model at stage $l-1$ and setting F_l as the identity map. Our experiments demonstrate such multi-stage strategy significantly stabilizes training and improves final performance.

Figure 1 and Equation 7 imply that intermediate activations, i.e., $\tilde{\mathbf{x}}_l$ and $\mathbf{x}_l$, who are samples of predefined posterior distributions at the coarse scales (see Equation 5), are semantically meaningful and interpret-able. This is different from Glow (Kingma & Dhariwal, 2018), whose intermediate activations are not interpret-able due to the loss of spatial relation.

Jeffreys divergence and importance sampling with the surrogate. The KL divergence is easy to compute, and thus is widely used as the training objective. However, its landscape could admit local minima that don't favor the optimization. Nielsen & Nock (2009) suggests that $D_{\mathrm{KL}}(p_\theta \| q)$ is zero-forcing, meaning that it enforces p_θ be small whenever q is small. As a consequence, mode missing can still be a local minimum, see Appendix C. Therefore, we turn to the Jeffreys divergence defined in Equation 2 which penalizes mode missing much and can remove such local minima.

Estimating the Jeffreys divergence requires computing an expectation with respect to the target q, which is normally prohibited. Since MsIGN constructs a good approximation $\tilde{q}$ to q, and $\tilde{q}$ can be constructed from coarser levels in multi-stage training, we do importance sampling with the

surrogate $\tilde{q}$ for the Jeffreys diveregence and its derivative (see Appendix D for detailed derivation):

$$D_{\mathrm{J}}(p_\theta \| q) = \mathbb{E}_{\mathbf{x}\sim p_\theta}\left[\log \frac{p_\theta(\mathbf{x})}{q(\mathbf{x})}\right] + \mathbb{E}_{\mathbf{x}\sim \tilde{q}}\left[\frac{q(\mathbf{x})}{\tilde{q}(\mathbf{x})}\log \frac{q(\mathbf{x})}{p_\theta(\mathbf{x})}\right]. \tag{8}$$

$$\frac{\partial}{\partial\theta} D_{\mathrm{J}}(p_\theta \| q) = \mathbb{E}_{\mathbf{x}\sim p_\theta}\left[\left(1+\log \frac{p_\theta(\mathbf{x})}{q(\mathbf{x})}\right)\frac{\partial \log p_\theta(\mathbf{x})}{\partial\theta}\right] - \mathbb{E}_{\mathbf{x}\sim \tilde{q}}\left[\frac{q(\mathbf{x})}{\tilde{q}(\mathbf{x})}\frac{\partial \log p_\theta(\mathbf{x})}{\partial\theta}\right]. \tag{9}$$

With the derivative estimate given above, we optimize the Jeffreys divergence by stochastic gradient descent. We remark that $\partial \log p_\theta(\mathbf{x})/\partial\theta$ is computed by the backward propagation of MsIGN.

3 Related Work

Invertible generative models (Deco & Brauer, 1995) are powerful exact likelihood models with efficient sampling and inference. They have achieved great success in natural image synthesis, see, e.g., Dinh et al. (2016); Kingma & Dhariwal (2018); Grathwohl et al. (2018); Ho et al. (2019); Chen et al. (2019b), and variational inference in providing a tight evidence lower bound (ELBO), see, e.g, Rezende & Mohamed (2015). In this paper, we propose a new multiscale invertible generative network (MsIGN) structure, which utilizes the invertible block in Glow (Kingma & Dhariwal, 2018) as building piece for the invertible flow at each scale. The Glow block can be replaced by any other invertible blocks, without any algorithmic changes. Different from Glow, different scales of MsIGN can be trained separately, and thus features in its intermediate layers can be interpreted as low-resolution approximation of the final high-resolution output. This novel multiscale structure enables better explain-ability of its hidden neurons and makes training much more stable.

Different from the image synthesis task where large amount of samples from target distribution are available, in Bayesian inference problems only an unnormalized density is available and i.i.d. samples from the posterior are the target. This paper's main goal is to train MsIGN to approximate certain high-d Bayesian posteriors. Various kinds of parametric distributions have been proposed to approximate posteriors before, such as polynomials (El Moselhy & Marzouk, 2012), non-invertible generative networks (Feng et al., 2017; Hou et al., 2019), invertible networks (Rezende & Mohamed, 2015; Ardizzone et al., 2018; Kruse et al., 2019) and certain implicit maps (Chorin & Tu, 2009; Morzfeld et al., 2012). Generative modeling approach has the advantage that i.i.d. samples can be efficiently obtained by evaluating the model in the inference stage. However, due to the tricky non-convex optimization problem, this approach for both invertible (Chorin & Tu, 2009; Kruse et al., 2019) and non-invertible (Hou et al., 2019) generative models becomes increasingly challenging as the dimension grows. To overcome this difficulty, we propose (1) to use the Jeffreys divergence as loss function, which has fewer shallow local minima and better landscape compared with the commonly-used KL divergence (see Appendix C for a concrete example), and (2) to train MsIGN in a coarse-to-fine manner with coarse-scale solution serving as an initialization to fine-scale optimization problem. In Kruse et al. (2019), authors list some recent models for low-d inverse problems. We remark that their formulation of posterior assumes no observation or model error in Equation 1, and is different from ours. See Appendix J for detailed discussion and experimental comparison.

Other than the generative modeling, various Markov Chain Monte Carlo (MCMC) methods have been the most popular in Bayesian inference, see, e.g., Beskos et al. (2008); Neal et al. (2011); Welling & Teh (2011); Chen et al. (2014; 2015); Cui et al. (2016). Particle-optimization-based sampling is a recently developed effective sampling technique with Stein variational gradient descent (SVGD) (Liu & Wang, 2016)) and many related works, e.g., Liu (2017); Liu & Zhu (2018); Chen et al. (2018; 2019a). The intrinsic difficulty of Bayesian inference displays itself as highly correlated samples, leading to undesired low sample efficiency, especially in high-d cases. The multiscale structure and multi-stage strategy proposed in this paper can also benefit these particle-based methods, as we can observe that they benefit the amortized-SVGD (Feng et al., 2017; Hou et al., 2019) in Section 4.1.3. We leave a more thorough study of this topic as a future work.

Works in Parno et al. (2016); Matthies et al. (2016) utilize the multiscale structure in Bayesian inference and build generative models with polynomials. They suffer from exponential growth of parameter number for high-d polynomial basis. The Markov property (Spantini et al., 2018) is used to alleviate this exponential growth. Different from these works, we leverage the great capacity of invertible generative networks to parametrize the high-d distribution, and we design novel network architecture to make use of the multiscale structure. The multiscale structure is a more general

structure than commonly-used intrinsic low-d structure (Spantini, 2017; Cui et al., 2016; Chen et al., 2019a), which assumes that the density of high-d posterior concentrates in a low-d subspace.

In the image synthesis task, this multiscale idea incorporates with various generative models. For example, Denton et al. (2015); Odena et al. (2017); Karras et al. (2017); Xu et al. (2018) uses it in generative adversarial networks (GANs) to grow a high-resolution image from low-resolution ones. But the lack of invertibility in these models makes it difficult for them to apply to Bayesian inference problems. Invertible generative models like Dinh et al. (2016); Kingma & Dhariwal (2018); Ardizzone et al. (2019) adopted this multiscale idea, but their multiscale strategy is not in the spatial sense: the intermediate neurons are not semantically interpret-able, as we show in Figure 6.

4 EXPERIMENT

We study two high-d Bayesian inverse problems (BIPs) known to have at least two equally important modes in Section 4.1 as test beds for distribution approximation and multi-mode capture: one with true samples available in Section 4.1.1; one without true samples but close to real-world applications in subsurface flow in Section 4.1.2. We also report the ablation study of MsIGN in Section 4.1.3. In addition, we apply MsIGN to the image synthesis task to benchmark with flow-based generative models and demonstrate its interpret-ability in Section 4.2. We adopt the invertible block in Glow (Kingma & Dhariwal, 2018) as the building piece, and stack several of them to build our invertible flow F. We utilize average pooling with kernel size 2 and stride 2 as our pooling operator $\mathcal{A}$.

4.1 BAYESIAN INVERSE PROBLEMS

Sample $\mathbf{x}$ of our target posterior distribution q is a vector on a 2-D uniform 64×64 lattice, which means the problem dimension d is 4096. Every $\mathbf{x}$ is equivalent to a piece-wise constant function on the unit disk: $\mathrm{x}(s)$ for $s \in \Omega = [0,1]^2$, and we don't distinguish between them thereafter. We place a centered Gaussian with a Laplacian-type covariance as the prior: $\mathcal{N}\left(0, \beta^2(-\boldsymbol{\Delta})^{-1-\alpha}\right)$, which is very popular in geophysics and electric tomography. See Appendix E for problem settings in detail.

The key to guarantee the multi-modality of our posteriors is the symmetry. Combining properties of the prior defined above and the likelihood defined afterwards, the posterior is mirror-symmetric: $q(\mathrm{x}(s_1, s_2)) = q(\mathrm{x}(s_1, 1-s_2))$. We carefully select the prior and the likelihood so that our posterior q has at least two modes. They are mirror-symmetric to each other and possess equal importance.

As in Figure 1, we plan to learn our 4096-D posteriors at the end of $L = 6$ levels, and set problem dimension at each level as $d_l = 2^l * 2^l = 4^l$. The training follows our multi-stage strategy, and the first stage $l = 1$ is initialized by minimizing the Jeffreys divergence without importance sampling, because samples to q_1 is available since $d_1 = 4$ is relatively small. See Appendix E for details.

We compare MsIGN with representatives of major approaches: amortized-SVGD (short as A-SVGD) (Feng et al., 2017) and Hamilton Monte Carlo (short as HMC) (Neal et al., 2011), for high-d BIPs, see our discussion in Section 3. We measure the computational cost by the number of forward simulations (nFSs), because running the forward simulation $\mathcal{F}$ occupies most training time, especially in Section 4.1.2. We budget a same nFS for all methods for fair comparison.

4.1.1 SYNTHETIC BAYESIAN INVERSE PROBLEMS

This problem allows access to ground-truth samples so the comparison is clear and solid. The forward process is given by $\mathcal{F}(\mathbf{x}) = \langle \boldsymbol{\varphi}, \mathbf{x} \rangle^2 = (\int_\Omega \varphi(s)\mathrm{x}(s)\mathrm{d}s)^2$, where $\varphi(s) = \sin(\pi s_1)\sin(2\pi s_2)$. Together with the prior, our posterior can be factorized into one-dimensional sub-distributions, namely $q(\mathbf{x}) = \prod_{k=1}^{d} q_k(\langle \boldsymbol{w}_k, \mathbf{x} \rangle)$ for some orthonormal basis $\{\boldsymbol{w}_k\}_{k=1}^d$. This property gives us access to true samples via inversion cumulative function sampling along each direction $\boldsymbol{w}_k$. Furthermore, these 1-D sub-distributions are all single modal except that there's one, denoted as q_{k^*}, with two symmetric modes. In other words, the marginal distribution along $\boldsymbol{w}_{k^*}$ is double-model and the rest are uni-model. This confirms our construction of two equally important modes. See Appendix E for more details in problem settings. The computation budget is fixed at 8×10^5 nFSs.

Multi-mode capture. To visualize mode capture, we plot the marginal distribution of generated samples along the critical direction $\boldsymbol{w}_{k^*}$, which by construction is the source of double-modality of the posterior. The (visually) worst one in three independent experiments is shown in Figure 2(a).

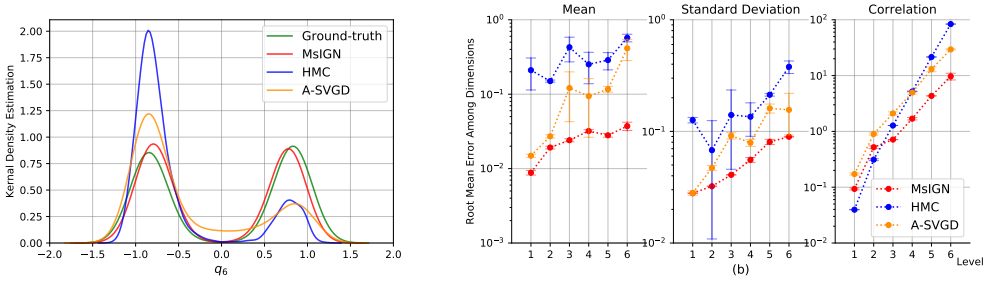

Figure 2: Results of the synthetic BIP. (a): Distribution of 2500 samples along the critical direction $\boldsymbol{w}_{k^*}$. MsIGN is more robust in capturing both modes, its samples are more balanced. (b): Error mean and its 95% confidence interval. MsIGN is more accurate in distribution approximation, especially at finer scale when the problem dimension is high. The margin is statistical significant as shown by the confidence interval. For more experimental results, please refer to Appendix F.

Distribution approximation. To measure distribution approximation, we report the error of mean, variance and correlation at or between all sub-distributions, as well as the Jeffreys divergence. Thanks to the factorization property, we compare the mean, variance and correlation estimate with theoretical ground-truths, and report the root mean square of error at all dimensions in Figure 2(b). For MsIGN and A-SVGD that gives access to not only samples but also density, we also report the Monte Carlo estimates of the Jeffreys divergence with the target posterior in Table 1. We can see that MsIGN has superior accuracy in approximating the target distribution.

Method	MsIGN	A-SVGD (Feng et al., 2017)
Error	56.77±0.15	3372±21

Table 1: Distribution approximation error by Jeffreys divergence with the target posterior in three independent runs

4.1.2 Elliptic Bayesian Inverse Problems

This problem originates from geophysics and fluid dynamics. The forward model is given by linear measurement of the solution to an elliptic partial differential equation associated with $\mathbf{x}$. We define

$$\mathcal{F}(\mathrm{x}) = \left[\int_\Omega \varphi_1(s)\mathrm{u}(s)\mathrm{d}s \quad \int_\Omega \varphi_2(s)\mathrm{u}(s)\mathrm{d}s \quad \dots \quad \int_\Omega \varphi_m(s)\mathrm{u}(s)\mathrm{d}s\right]^T ,$$

where φ_k are fixed measurement functions, and $\mathrm{u}(s)$ is the solution of

$$-\nabla \cdot \left(e^{\mathrm{x}(s)}\nabla \mathrm{u}(s)\right) = f(s)\,, s \in \Omega\,, \quad \text{with boundary condition } \mathrm{u}(s) = 0\,, s \in \partial\Omega\,. \tag{10}$$

This model appears frequently in real applications. For example, x, u can be seen as permeability field and pressure in geophysics. However, there is no known access to true samples of q. Again the trick of symmetry introduced in Section 4.1 and explained in Appendix E guarantees at least two equally important modes in the posterior. We put a 5×10^5-nFS budget on our computation cost.

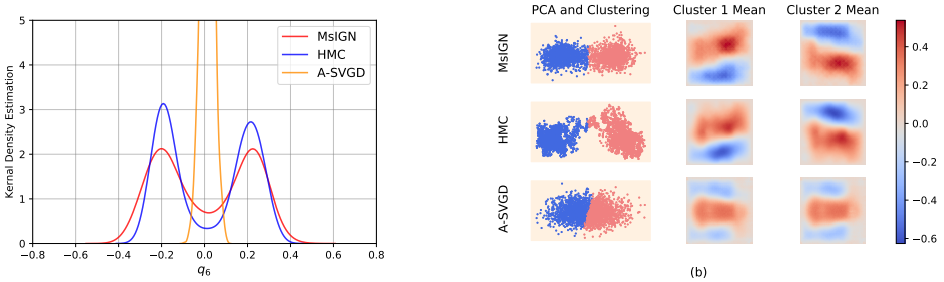

Figure 3: Results of the elliptic BIP. (a): Distribution of 2500 samples along a critical direction. MsIGN and HMC capture two modes in this marginal distribution, but A-SVGD fails. (b): Clustering result of 2500 samples. Samples of MsIGN are more balanced between two modes. The similarity of the cluster means of MsIGN and HMC implies that they both are likely to capture the correct modes. For more experimental results, please refer to Appendix I.

Multi-mode capture. Due to lack of true samples, we check the marginal distribution of the posterior along eigen-vectors of the prior, and pick a particular one to demonstrate that we can capture double modes in Figure 3(a). We also confirm the capture of multiple modes by embedding samples

by Principle Component Analysis (PCA) to a 2-D space. We report the clustering (by K-means) result and means of each cluster in Figure 3(b), where we can see that A-SVGD failed to capture the two symmetric modes, while MsIGN has a more balanced capture of the symmetric posterior.

4.1.3 Ablation Study of Architecture Design and Training Strategy

We run extensive experiments to study the effectiveness of the network architecture and training stragtegy of MsIGN, see Figure 4. We refer to Appendix G for details in setting and more results.

Network architecture. We replace the prior conditioning layer by two direct alternatives: a stochastic nearest-neighbor upsampling layer (model denoted as "MsIGN-SNN"), or the split and squeeze layer in Glow design (now the model is essentially Glow, so we also denote it as "Glow").

Figure 4(a) shows that the prior conditioning layer design is crucial to the performance of MsIGN on both problems, because neither "MsIGN-SNN" nor "Glow" has a successful mode capture.

Training strategy. We study the effectiveness of the Jeffreys divergence objective and multi-stage training. We try substituting the Jeffreys divergence objective (no extra marks) with the KL divergence (model denoted with a string "-KL") or kernelized Stein discrepancy (which resumes A-SVGD algorithm, model denoted with a string "-AS"), and switching between multi-stage (no extra marks) or single-stage training (model denoted with a string "-S"). We remark that single-stage training using Jeffreys divergence is infeasible because of the difficulty to estimate $D_{\mathrm{KL}}(q\|p_\theta)$.

Figure 4(b) and (c) show that, all models trained in the single-stage manner ("MsIGN-KL-S", "MsIGN-AS-S") will face mode collapse. We also observe that our multi-stage training strategy can benefit training with other objectives, see "MsIGN-KL" and "MsIGN-AS".

We also notice that the Jeffreys divergence leads to a more balanced samples for these symmetric problems, especially for the complicated elliptic BIP in Section 4.1.2.

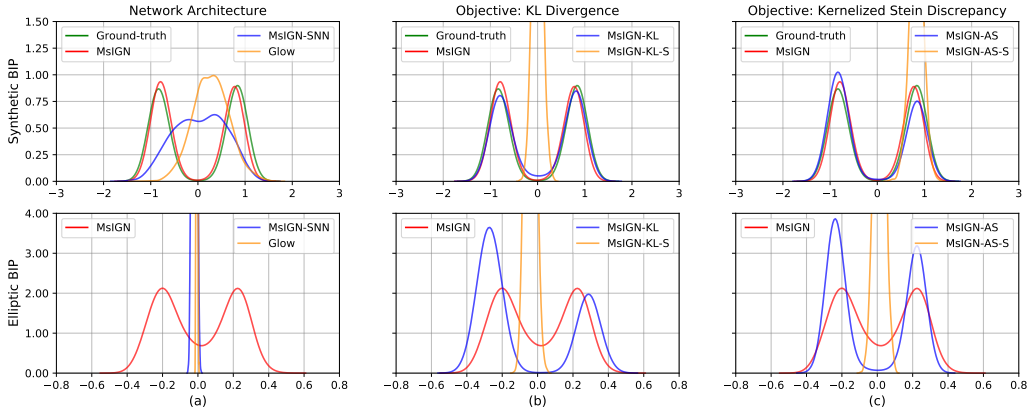

Figure 4: Ablation study of the network architecture and training strategy. "MsIGN" means our default setting: training MsIGN network with Jeffreys divergence and multi-stage strategy. Other models are named by a base model (MsIGN or Glow), followed by strings indicating its variance from the default setting. For example, "MsIGN-KL" refers to training MsIGN network with single KL divergence in a multi-stage way, while "MsIGN-KL-S" means traininng in a single-stage way.

4.2 Image Synthesis Task

We train our MsIGN architecture with maximum likelihood estimation to benchmark with other flow-based generative models. The prior conditional distribution $\rho(\mathbf{x}|\mathbf{x}_c)$ is modeled by a simple Gaussian with a scalar matrix as its covariance and is learned from a training set. We refer readers to Appendix H for more experimental details, and to Appendix I for additional results.

We report the bits-per-dimension value with our baseline models of flow-based generative networks in Table 2. Our MsIGN is superior in number and also is more efficient in parameter size: for example, MsIGN uses 24.4% fewer parameters than Glow for CelebA 64, and uses 37.4% fewer parameters than Residual Flow for ImageNet 64.

In Figure 5, we show synthesized images of MsIGN from CelebA 64 dataset, and linear interpolation of real images in the latent feature space. In Figure 6, we visualize internal activations at checkpoints of the invertible flow at different scales which demonstrates the interpret-ability of MsIGN.

Table 2: Bits-per-dimension value comparison with baseline models of flow-based generative networks. All models in this table do not use "variational dequantization" in Ho et al. (2019). *: Score obtained by our own reproducing experiment.

Model	MNIST	CIFAR-10	CelebA 64	ImageNet 32	ImageNet 64
Real NVP(Dinh et al., 2016)	1.06	3.49	3.02	4.28	3.98
Glow(Kingma & Dhariwal, 2018)	1.05	3.35	2.20*	4.09	3.81
FFJORD(Grathwohl et al., 2018)	0.99	3.40	–	–	–
Flow++(Ho et al., 2019)	–	3.29	–	–	–
i-ResNet(Behrmann et al., 2019)	1.05	3.45	–	–	–
Residual Flow(Chen et al., 2019b)	0.97	**3.28**	–	**4.01**	3.76
MsIGN (Ours)	**0.93**	**3.28**	**2.15**	4.03	**3.73**

Figure 5: Left: Synthesized CelebA 64 images with temperature 0.9. Right: Linear interpolation in latent space shows MsIGN's parameterization of natural image manifold is semantically meaningful.

Figure 6: Visualization of internal activation shows the interpret-ability of MsIGN hidden neurons. From left to right, we show how MsIGN progressively generates new samples in high resolution by taking snapshots at internal checkpoints. See Appendix I for details.

5 Conclusion

For high-dimensional Bayesian inference problems with multiscale structure, we propose Multiscale Invertible Generative Networks (MsIGN) and associated training algorithms to approximate the high-dimensional posterior. In this paper, we demonstrate the capability of this approach in high-dimensional (up to 4096 dimensions) Bayesian inference problems with spatial multiscale structure, leaving several important directions as future work. The network architecture also achieves the state-of-the-art performance in various image synthesis tasks. We plan to apply this methodology to other Bayesian inference problems, for example, Bayesian deep learning with multiscale structure in model width or depth (e.g., Chang et al. (2017); Haber et al. (2018)) and data assimilation problem with multiscale structure in the temporal variation (e.g., Giles (2008)). We also plan to develop some theoretical guarantee of the posterior approximation performance for MsIGN.

REFERENCES

Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W Pellegrini, Ralf S Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*, 2018.

Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019.

Jens Behrmann, Will Grathwohl, Ricky TQ Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pp. 573–582, 2019.

Alexandros Beskos, Gareth Roberts, Andrew Stuart, and Jochen Voss. Mcmc methods for diffusion bridges. *Stochastics and Dynamics*, 8(03):319–350, 2008.

Bo Chang, Lili Meng, Eldad Haber, Frederick Tung, and David Begert. Multi-level residual networks from dynamical systems view. *arXiv preprint arXiv:1710.10348*, 2017.

Changyou Chen, Nan Ding, and Lawrence Carin. On the convergence of stochastic gradient mcmc algorithms with high-order integrators. In *Advances in Neural Information Processing Systems*, pp. 2278–2286, 2015.

Changyou Chen, Ruiyi Zhang, Wenlin Wang, Bai Li, and Liqun Chen. A unified particle-optimization framework for scalable bayesian sampling. *arXiv preprint arXiv:1805.11659*, 2018.

Peng Chen, Keyi Wu, Joshua Chen, Tom O'Leary-Roseberry, and Omar Ghattas. Projected stein variational newton: A fast and scalable bayesian inference method in high dimensions. In *Advances in Neural Information Processing Systems*, pp. 15104–15113, 2019a.

Tian Qi Chen, Jens Behrmann, David K Duvenaud, and Jörn-Henrik Jacobsen. Residual flows for invertible generative modeling. In *Advances in Neural Information Processing Systems*, pp. 9913–9923, 2019b.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pp. 1683–1691, 2014.

Alexandre J Chorin and Xuemin Tu. Implicit sampling for particle filters. *Proceedings of the National Academy of Sciences*, 106(41):17249–17254, 2009.

Tiangang Cui, Kody JH Law, and Youssef M Marzouk. Dimension-independent likelihood-informed mcmc. *Journal of Computational Physics*, 304:109–137, 2016.

Gustavo Deco and Wilfried Brauer. Nonlinear higher-order statistical decorrelation by volume-conserving neural architectures. *Neural Networks*, 8(4):525–535, 1995.

Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pp. 1486–1494, 2015.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Tarek A El Moselhy and Youssef M Marzouk. Bayesian inference with optimal maps. *Journal of Computational Physics*, 231(23):7815–7850, 2012.

Yihao Feng, Dilin Wang, and Qiang Liu. Learning to draw samples with amortized stein variational gradient descent. *arXiv preprint arXiv:1707.06626*, 2017.

Michael B Giles. Multilevel monte carlo path simulation. *Operations research*, 56(3):607–617, 2008.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Eldad Haber, Lars Ruthotto, Elliot Holtham, and Seong-Hwan Jun. Learning across scales—multiscale methods for convolution neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. *arXiv preprint arXiv:1902.00275*, 2019.

Thomas Y Hou, Ka Chun Lam, Pengchuan Zhang, and Shumao Zhang. Solving bayesian inverse problems from the perspective of deep generative networks. *Computational Mechanics*, 64(2): 395–408, 2019.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

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

Jakob Kruse, Lynton Ardizzone, Carsten Rother, and Ullrich Köthe. Benchmarking invertible architectures on inverse problems. ICML, 2019.

Chang Liu and Jun Zhu. Riemannian stein variational gradient descent for bayesian inference. In *Thirty-second aaai conference on artificial intelligence*, 2018.

Qiang Liu. Stein variational gradient descent as gradient flow. In *Advances in neural information processing systems*, pp. 3115–3123, 2017.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, pp. 2378–2386, 2016.

Hermann G Matthies, Elmar Zander, Bojana V Rosić, Alexander Litvinenko, and Oliver Pajonk. Inverse problems in a bayesian setting. In *Computational Methods for Solids and Fluids*, pp. 245–286. Springer, 2016.

Matthias Morzfeld, Xuemin Tu, Ethan Atkins, and Alexandre J Chorin. A random map implementation of implicit filters. *Journal of Computational Physics*, 231(4):2049–2066, 2012.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11), 2011.

Frank Nielsen and Richard Nock. Sided and symmetrized bregman centroids. *IEEE transactions on Information Theory*, 55(6):2882–2904, 2009.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pp. 2642–2651, 2017.

Matthew Parno, Tarek Moselhy, and Youssef Marzouk. A multiscale strategy for bayesian inference using transport maps. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):1160–1190, 2016.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2015.

Alessio Spantini. *On the low-dimensional structure of Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2017.

Alessio Spantini, Daniele Bigoni, and Youssef Marzouk. Inference via low-dimensional couplings. *The Journal of Machine Learning Research*, 19(1):2639–2709, 2018.

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)*, pp. 681–688, 2011.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1316–1324, 2018.

APPENDIX

A MULTISCALE STRUCTURE AND SCALE DECOUPLING: RELATED WORK

In Parno et al. (2016), a similar notion of multiscale structure is defined as follows. A likelihood function has the *Parno et al. (2016)-multiscale structure*, if there exists a coarse-scale *random variable* $\boldsymbol{\gamma}$ of dimension d_c ($d_c < d$) and a likelihood L_c such that

$$L(\boldsymbol{y}|\mathbf{x}, \boldsymbol{\gamma}) = L_c(\boldsymbol{y}|\boldsymbol{\gamma}). \tag{11}$$

Then the joint posterior distribution of the fine- and coarse-scale parameters $(\mathbf{x}, \boldsymbol{\gamma})$ can be decoupled in the following way, with normalizing constants omitted in the equivalence relations:

$$q(\mathbf{x}, \boldsymbol{\gamma}) \propto \rho(\mathbf{x}, \boldsymbol{\gamma}) L(\boldsymbol{y}|\mathbf{x}, \boldsymbol{\gamma}) \overset{(i)}{=} \rho(\mathbf{x}, \boldsymbol{\gamma}) L_c(\boldsymbol{y}|\boldsymbol{\gamma}) \overset{(ii)}{=} \rho(\mathbf{x}|\boldsymbol{\gamma})\rho(\boldsymbol{\gamma}) L_c(\boldsymbol{y}|\boldsymbol{\gamma}) \overset{(iii)}{=} \rho(\mathbf{x}|\boldsymbol{\gamma}) q_c(\boldsymbol{\gamma}), \tag{12}$$

where we use the Parno et al. (2016)-multiscale structure (Equation 11) in (i) and $\rho(\mathbf{x}, \boldsymbol{\gamma}) = \rho(\mathbf{x}|\boldsymbol{\gamma})\rho(\boldsymbol{\gamma})$ in (ii) due to standard conditional probability calculation. In (iii) we define the Parno et al. (2016)-posterior in coarse scale: $q_c(\boldsymbol{\gamma}) = \rho(\boldsymbol{\gamma}) L_c(\boldsymbol{y}|\boldsymbol{\gamma})$.

There are two important differences in these two definitions. First, our coarse-scale parameter $\mathbf{x}_c$ is a deterministic function of the fine-scale parameter $\mathbf{x}$, while in Parno et al. (2016), $\boldsymbol{\gamma}$ is a random variable that may contain extra randomness outside $\mathbf{x}$ (as demonstrated in numerical examples in Parno et al. (2016)). This difference in definition results in significant difference in modeling. Since the dimension of input Gaussian random noise in the invertible transforms or networks must agree with that of the target distribution, our invertible model has d-dimensional random noise as input to approximate exactly the target posterior $q(\mathbf{x})$, while models in Parno et al. (2016) has $d + d_\gamma$-dimensional random noise as input to approximate the joint-posterior $q(\mathbf{x}, \boldsymbol{\gamma})$. It is questionable to us whether this extra randomness in $\boldsymbol{\gamma}$ is necessary in real applications. One extra consequence is that users need to define the joint prior $\rho(\mathbf{x}, \boldsymbol{\gamma})$ in Parno et al. (2016), while in our definition the prior of $\mathbf{x}_c$ is naturally induced by the prior of $\mathbf{x}$.

Second, our multiscale structure is an approximate relation and we use invertible transform F in MsIGN to model this approximation, while in Parno et al. (2016) the multiscale structure (Equation 12) is an exact relation and authors treat the prior-upscaled solution $\rho(\mathbf{x}|\boldsymbol{\gamma})q_c(\boldsymbol{\gamma})$ (RHS of Equation 12) as the final solution. Our approximate multiscale relation and further treatment by transform F enables us to apply the method recursively in a multiscale fashion, while in Parno et al. (2016) the proposed method is essentially a two-scale method and there is not further correction based on the prior-upscaled solution $\rho(\mathbf{x}|\boldsymbol{\gamma})q_c(\boldsymbol{\gamma})$ at the fine-scale. As we show in Appendix F, the true posterior could still be far away from the prior-upscaled solution, especially in the first few coarse scales.

Finally, as we discussed in related work (Section 3), the invertible model in Parno et al. (2016) is polynomials, which suffer from the exponential growth of polynomial coefficients as dimension grows. In this work, the invertible model is deep generative networks, whose number of parameters are independent of the problem dimension.

B PRIOR CONDITIONING: FORMULATION AND INVERTIBILITY

we first prove Lemma 2.1 which gives closed-form formulation for the prior conditional distribution. Then we prove a powerful tool of partition of unity in Lemma B.1, and use it to prove Theorem B.1 that gives the invertibility between $\mathbf{x}_l$ and $(\mathbf{x}_{l-1}, \mathbf{z}_l)$ as in Equation 6. We adopt the notations in Section 2.2 here.

Lemma 2.1 *Suppose that the prior $\rho(\mathbf{x}_l) = \mathcal{N}(\mathbf{x}_l; \mathbf{0}, \boldsymbol{\Sigma}_l)$, and that the pooling operator is linear, i.e., $\mathcal{A}_l(\mathbf{x}_l) = \boldsymbol{A}_l\mathbf{x}_l$ for some matrix $\boldsymbol{A}_l \in \mathbb{R}^{d_{l-1} \times d_l}$, then:*

$$\rho(\mathbf{x}_l|\mathbf{x}_{l-1} = \boldsymbol{A}_l\mathbf{x}_l) = \mathcal{N}(\mathbf{x}_l; \boldsymbol{U}_{l-1}\mathbf{x}_{l-1}, \boldsymbol{\Sigma}_{l|l-1}),$$

where $\boldsymbol{U}_{l-1} = \boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}$ and $\boldsymbol{\Sigma}_{l|l-1} = \boldsymbol{\Sigma}_l - \boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l$.

Proof: By the Bayes rule, the conditional density has the form:

$$\rho(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)=\frac{\rho(\mathbf{x}_l)}{\int_{\{\mathbf{x}_l':\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l'\}}\rho(\mathbf{x}_l')\mathrm{d}\mathbf{x}_l'}$$

Given $\mathbf{x}_{l-1}$, the denominator of the above equation is a constant, so we may write:

$$\log\rho(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)=\log\rho(\mathbf{x}_l)-\log\left(\int_{\{\mathbf{x}_l':\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l'\}}\rho(\mathbf{x}_l')\mathrm{d}\mathbf{x}_l'\right)=-\frac{1}{2}\mathbf{x}_l\boldsymbol{\Sigma}_l^{-1}\mathbf{x}_l+C\,,$$

where C is a constant that only depends on $\mathbf{x}_{l-1}$. Since $\log\rho(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)$ is a quadratic function of $\mathbf{x}_l$, then $\rho(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)$ is a Gaussian distribution. To determine this distribution we only needs to calculate $\mathbb{E}[\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]$ and $\mathrm{Cov}(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)$.

With $\boldsymbol{U}_{l-1}=\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}$, we will decompose $\mathbf{x}_l=(\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l)+\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l$. We will prove that $\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l$ is independent from $\boldsymbol{A}_l\mathbf{x}_l$, and therefore,

$$\begin{aligned}\mathbb{E}[\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]&=\mathbb{E}[(\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l)+\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]\\&=\mathbb{E}[\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]+\mathbb{E}[\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]\\&=0+\boldsymbol{U}_{l-1}\mathbf{x}_{l-1}=\boldsymbol{U}_{l-1}\mathbf{x}_{l-1}\,.\end{aligned}$$

To prove that $\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l=(\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)\mathbf{x}_l$ is independent from $\boldsymbol{A}_l\mathbf{x}_l$, we notice that they are both linear transformation of the Gaussian variable $\mathbf{x}_l$, so their joint distributions should also be a Gaussian, and their covariance can be computed as

$$\begin{aligned}\mathrm{Cov}((\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)\mathbf{x}_l,\boldsymbol{A}_l\mathbf{x}_l)&=(\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T=\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T\\&=\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T\\&=\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T=\mathbf{0}\,.\end{aligned}$$

And therefore, $\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l=(\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)\mathbf{x}_l$ is independent from $\boldsymbol{A}_l\mathbf{x}_l$.

And finally, since $\mathbb{E}[\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]=\boldsymbol{U}_{l-1}\mathbf{x}_{l-1}$, we calculate

$$\mathrm{Cov}(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)=\mathrm{Cov}[\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l]$$

Because $\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l=(\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)\mathbf{x}_l$ is independent from $\boldsymbol{A}_l\mathbf{x}_l$, we can drop the condition and write:

$$\begin{aligned}\mathrm{Cov}(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)&=\mathrm{Cov}[\mathbf{x}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\mathbf{x}_l]=(\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)\boldsymbol{\Sigma}_l(\boldsymbol{I}_{d_l}-\boldsymbol{U}_{l-1}\boldsymbol{A}_l)^T\\&=\boldsymbol{\Sigma}_l-\boldsymbol{U}_{l-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T\boldsymbol{U}_{l-1}^T+\boldsymbol{U}_{l-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T\boldsymbol{U}_{l-1}^T\\&=\boldsymbol{\Sigma}_l-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{\Sigma}_l\boldsymbol{A}_l\\&+\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{\Sigma}_l\boldsymbol{A}_l\\&=\boldsymbol{\Sigma}_l-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{\Sigma}_l\boldsymbol{A}_l\\&+\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{\Sigma}_l\boldsymbol{A}_l\\&=\boldsymbol{\Sigma}_l-\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l=\boldsymbol{\Sigma}_{l|l-1}\,.\end{aligned}$$

So now we can claim that $\rho(\mathbf{x}_l|\mathbf{x}_{l-1}=\boldsymbol{A}_l\mathbf{x}_l)=\mathcal{N}(\mathbf{x}_l;\boldsymbol{U}_{l-1}\mathbf{x}_{l-1},\boldsymbol{\Sigma}_{l|l-1})$. □

Now we introduce the following lemma, also called the partition of unity.

Since $\boldsymbol{A}_l\in\mathbb{R}^{d_{l-1}\times d_l}$ ($d_{l-1}<d_l$), we can always find a matrix $\tilde{\boldsymbol{A}}_l\in\mathbb{R}^{(d_l-d_{l-1})\times d_l}$, such that $\boldsymbol{A}_l\tilde{\boldsymbol{A}}_l^T=\mathbf{0}$. In fact, let $V\in\mathbb{R}^{d_l}$ be the row space of $\boldsymbol{A}_l$, then $\dim(V)=d_{l-1}<d_l$, so its orthogonal complement $V^\perp$ is non-trivial: $\dim(V^\perp)=d_l-d_{l-1}>0$. Collect a basis of $V^\perp$ and pack in the rows, and we have a matrix $\tilde{\boldsymbol{A}}_l\in\mathbb{R}^{(d_l-d_{l-1})\times d_l}$. By construction we know $\boldsymbol{A}_l\tilde{\boldsymbol{A}}_l^T=\mathbf{0}$.

Lemma B.1 *Since $\boldsymbol{A}_l\in\mathbb{R}^{d_{l-1}\times d_l}$, $\tilde{\boldsymbol{A}}_l\in\mathbb{R}^{(d_l-d_{l-1})\times d_l}$, $\boldsymbol{A}_l\tilde{\boldsymbol{A}}_l^T=\mathbf{0}$ and $\boldsymbol{\Sigma}_l$ is symmetric positive definite, we have the following decomposition of the identity matrix $\boldsymbol{I}_{d_l}\in\mathbb{R}^{d_l\times d_l}$:*

$$\boldsymbol{I}_{d_l}=\boldsymbol{\Sigma}_l^{\frac{1}{2}}\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l^{\frac{1}{2}}+\boldsymbol{\Sigma}_l^{-\frac{1}{2}}\tilde{\boldsymbol{A}}_l^T(\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-1}\tilde{\boldsymbol{A}}_l^T)^{-1}\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-\frac{1}{2}}\tag{13}$$

Proof: Consider the following matrix $\boldsymbol{\Omega}_l \in \mathbb{R}^{d_l \times d_l}$:

$$\boldsymbol{\Omega}_l = [\boldsymbol{\Sigma}_l^{\frac{1}{2}} \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \quad \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}}]$$

Since $\boldsymbol{A}_l \tilde{\boldsymbol{A}}_l^T = \boldsymbol{0}$ and the covariance matrix $\boldsymbol{\Sigma}_l$ is symmetric: $\boldsymbol{\Sigma}_l = \boldsymbol{\Sigma}_l^T$, we have

$$\begin{aligned}
\boldsymbol{\Omega}_l^T \boldsymbol{\Omega}_l &= \begin{bmatrix} (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \boldsymbol{A}_l \boldsymbol{\Sigma}_l^{\frac{1}{2}} \\ (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \end{bmatrix} \left[\boldsymbol{\Sigma}_l^{\frac{1}{2}} \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \quad \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \right] \\
&= \begin{bmatrix} (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} & (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \boldsymbol{A}_l \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \\ (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} & (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{I}_{d_{l-1}} & \\ & \boldsymbol{I}_{d_l - d_{l-1}} \end{bmatrix} = \boldsymbol{I}_{d_l} .
\end{aligned}$$

Therefore, $\boldsymbol{\Omega}_l$ is a $d_l \times d_l$ orthonormal matrix, and $\boldsymbol{\Omega}_l \boldsymbol{\Omega}_l^T = \boldsymbol{I}_{d_l}$, which means

$$\begin{aligned}
\boldsymbol{I}_{d_l} = \boldsymbol{\Omega}_l \boldsymbol{\Omega}_l^T &= \left[\boldsymbol{\Sigma}_l^{\frac{1}{2}} \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \quad \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \right] \begin{bmatrix} (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-\frac{1}{2}} \boldsymbol{A}_l \boldsymbol{\Sigma}_l^{\frac{1}{2}} \\ (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \end{bmatrix} \\
&= \boldsymbol{\Sigma}_l^{\frac{1}{2}} \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-1} \boldsymbol{A}_l \boldsymbol{\Sigma}_l^{\frac{1}{2}} + \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-1} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-\frac{1}{2}} .
\end{aligned}$$

And this proves Equation 13. □

Now we prove the invertibility between $\mathbf{x}_l$ and $(\mathbf{x}_{l-1}, \mathbf{z}_l)$ with the help of Lemma B.1.

Theorem B.1 *The relation between $\mathbf{x}_l$ and $(\mathbf{x}_{l-1}, \mathbf{z}_l)$ as introduced by $\mathbf{x}_l = \boldsymbol{U}_l \mathbf{x}_{l-1} + \boldsymbol{\Sigma}_{l|l-1} \mathbf{z}_l$ in Equation 6 is invertible. In other words, $\mathbf{x}_l$ and $(\mathbf{x}_{l-1}, \mathbf{z}_l)$ can uniquely determine each other.*

Proof: Since we already have $\mathbf{x}_l = \boldsymbol{U}_l \mathbf{x}_{l-1} + \boldsymbol{\Sigma}_{l|l-1} \mathbf{z}_l$, so of course $(\mathbf{x}_{l-1}, \mathbf{z}_l)$ can uniquely determine $\mathbf{x}_l$. We only need to prove the opposite.

We notice that

$$\begin{aligned}
\boldsymbol{\Sigma}_{l|l-1} &= \boldsymbol{\Sigma}_l - \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-1} \boldsymbol{A}_l \boldsymbol{\Sigma}_l \\
&= \boldsymbol{\Sigma}_l^{\frac{1}{2}} \left(\boldsymbol{I}_{d_l} - \boldsymbol{\Sigma}_l^{\frac{1}{2}} \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-1} \boldsymbol{A}_l \boldsymbol{\Sigma}_l^{\frac{1}{2}} \right) \boldsymbol{\Sigma}_l^{\frac{1}{2}}
\end{aligned}$$

Use the identity decomposition in Lemma B.1, we have

$$\begin{aligned}
\boldsymbol{\Sigma}_{l|l-1} &= \boldsymbol{\Sigma}_l^{\frac{1}{2}} \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-1} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-\frac{1}{2}} \boldsymbol{\Sigma}_l^{\frac{1}{2}} \\
&= \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-1} \tilde{\boldsymbol{A}}_l .
\end{aligned}$$

Since $\boldsymbol{\Sigma}_{l|l-1} = \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-1} \tilde{\boldsymbol{A}}_l$, we must have that its Cholesky decomposition (or eigen-decomposition) $\boldsymbol{\Sigma}_{l|l-1} = \boldsymbol{B}_l \boldsymbol{B}_l^T$ admits the form $\boldsymbol{B}_l = \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \boldsymbol{P}_l$ for some deterministic orthonormal matrix $\boldsymbol{P}_l \in \mathbb{R}^{(d_l - d_{l-1}) \times (d_l - d_{l-1})}$.

Now we claim that $\mathbf{x}_{l-1} = \boldsymbol{A}_l \mathbf{x}_l$ and $\boldsymbol{z}_l = \boldsymbol{P}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \mathbf{x}_l$ gives the inversion. We just need to plug them back into the right hand side of Equation 6 to see if it holds.

We compute

$$\begin{aligned}
\boldsymbol{U}_{l-1} \mathbf{x}_{l-1} + \boldsymbol{B}_l \boldsymbol{z}_l &= \boldsymbol{U}_{l-1} \boldsymbol{A}_l \mathbf{x}_l + \boldsymbol{B}_l \boldsymbol{P}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \mathbf{x}_l \\
&= (\boldsymbol{U}_{l-1} \boldsymbol{A}_l + \boldsymbol{B}_l \boldsymbol{P}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1}) \mathbf{x}_l .
\end{aligned}$$

We will show $\boldsymbol{U}_{l-1} \boldsymbol{A}_l + \boldsymbol{B}_l \boldsymbol{P}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} = \boldsymbol{I}_{d_l}$ to complete the proof.

Since $\boldsymbol{U}_{l-1} = \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-1}$ and $\boldsymbol{B}_l = \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \boldsymbol{P}_l$, we have

$$\begin{aligned}
\boldsymbol{U}_{l-1} \boldsymbol{A}_l &= \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-1} \boldsymbol{A}_l \\
\boldsymbol{B}_l \boldsymbol{P}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} &= \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \boldsymbol{P}_l \boldsymbol{P}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \\
&= \tilde{\boldsymbol{A}}_l^T (\tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1} \tilde{\boldsymbol{A}}_l^T)^{-1} \tilde{\boldsymbol{A}}_l \boldsymbol{\Sigma}_l^{-1}
\end{aligned}$$

Therefore,

$$
\begin{aligned}
&\boldsymbol{U}_{l-1}\boldsymbol{A}_l + \boldsymbol{B}_l\boldsymbol{P}_l^T(\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-1}\tilde{\boldsymbol{A}}_l^T)^{-\frac{1}{2}}\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-1} \\
=&\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l + \tilde{\boldsymbol{A}}_l^T(\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-1}\tilde{\boldsymbol{A}}_l^T)^{-1}\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-1} \\
=&\boldsymbol{\Sigma}_l^{\frac{1}{2}}\left(\boldsymbol{\Sigma}_l^{\frac{1}{2}}\boldsymbol{A}_l^T(\boldsymbol{A}_l\boldsymbol{\Sigma}_l\boldsymbol{A}_l^T)^{-1}\boldsymbol{A}_l\boldsymbol{\Sigma}_l^{\frac{1}{2}} + \boldsymbol{\Sigma}_l^{-\frac{1}{2}}\tilde{\boldsymbol{A}}_l^T(\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-1}\tilde{\boldsymbol{A}}_l^T)^{-1}\tilde{\boldsymbol{A}}_l\boldsymbol{\Sigma}_l^{-\frac{1}{2}}\right)\boldsymbol{\Sigma}_l^{-\frac{1}{2}} \\
\overset{(i)}{=}&\boldsymbol{\Sigma}_l^{\frac{1}{2}}\boldsymbol{I}_{d_l}\boldsymbol{\Sigma}_l^{-\frac{1}{2}} = \boldsymbol{I}_{d_l}\,.
\end{aligned}
$$

Here in step (i) we use the identity decomposition (Equation 13) from Lemma B.1. □

In the end, Theorem B.1 suggests that the prior conditioning layer (Equation 6) is invertible. Notice that, all matrices ($\boldsymbol{U}_{l-1}, \boldsymbol{B}_l, \boldsymbol{P}_l, \tilde{\boldsymbol{A}}_l$, and etc) only depends on our prior distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_l)$ and pooling operator $\boldsymbol{A}_l$, and thus can be pre-computed.

C Comparison of the KL divergence and Jeffreys Divergence

The KL divergence sometimes can be inefficient in capture multi-modes: the single-sided KL divergence could be easily trapped by a local minimum that misses some modes or is far from the ground-truth. We support our claim by a concrete example.

Given $\sigma > 0$, let q be a 1-D Gaussian mixture model, with parameters μ_1 and μ_2 unknown but fixed:

$$
q(x) = \frac{1}{2}\left(\mathcal{N}(x; \mu_1, \sigma^2) + \mathcal{N}(x; \mu_2, \sigma^2)\right)\,.
$$

Our model p is also a 1-D Gaussian mixture model with parameter m_1 and m_2:

$$
p_{m_1,m_2}(x) = \frac{1}{2}\left(\mathcal{N}(x; m_1, \sigma^2) + \mathcal{N}(x; m_2, \sigma^2)\right)\,.
$$

When we set $\mu_1 = -\mu_2 = 1.5$, and $\sigma = 0.25$, we plot the landscape of single-sided KL divergences $D_{\mathrm{KL}}(p_{m_1,m_2}\|q)$ and $D_{\mathrm{KL}}(q\|p_{m_1,m_2})$, and the Jeffreys divergence $D_{\mathrm{J}}(p_{m_1,m_2}\|q) = D_{\mathrm{KL}}(p_{m_1,m_2}\|q) + D_{\mathrm{KL}}(q\|p_{m_1,m_2})$ as functions of m_1, m_2 in Figure 7.

It is now clear that the single-sided KL divergence $D_{\mathrm{KL}}(p\|q)$ alone might guide the training towards the local minima around $(1.5, 1.5)$ or $(-1.5, -1.5)$, where only one mode of q is captured. We explain this phenomenon as $D_{\mathrm{KL}}(p\|q) = \mathbb{E}_{\mathrm{x}\sim p}[\log(p(\mathrm{x})/q(\mathrm{x}))]$ becomes small as long as p is close to zero wherever q close to zero. Nielsen & Nock (2009) describes this property as "zero-forcing", and observes that $D_{\mathrm{KL}}(p\|q)$ will be small when high-density region of p is covered by that of q. However, it doesn't strongly enforce p to capture all high-density region of q. In our example, when $(m_1, m_2) = (1.5, 1.5)$ or $(-1.5, -1.5)$, the only high-density region of p is a strict subset of high-density region of q, and thus it attains a small value of $D_{\mathrm{KL}}(p\|q)$.

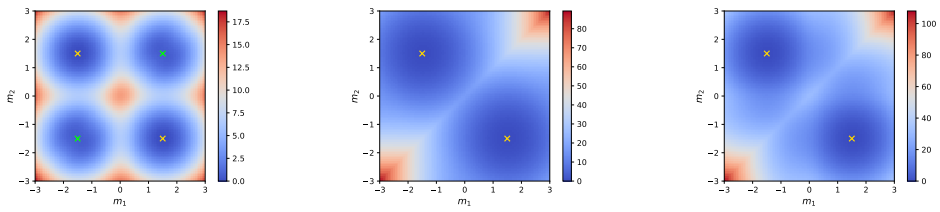

Figure 7: Landscape of $D_{\mathrm{KL}}(p\|q)$ (left), $D_{\mathrm{KL}}(q\|p)$ (middle), and $D_{\mathrm{J}}(p\|q)$ (right). We mark the global minima (ground-truth) by golden cross, and other local minima by green cross.

We also argue here that the other single-sided KL divergence $D_{\mathrm{KL}}(q\|p)$ alone faces the risk as well. Similar as discussed above, $D_{\mathrm{KL}}(q\|p) = \mathbb{E}_{\mathrm{x}\sim q}[\log(q(\mathrm{x})/p(\mathrm{x}))]$ becomes small as long as q is close to zero wherever p is close to zero. Thus if p captures all modes in q but also contains some extra modes, described as "zero-avoiding" in Nielsen & Nock (2009), we could also observe a small value of $D_{\mathrm{KL}}(q\|p)$. Therefore, we choose to use the Jeffreys divergence as a robust learning objective to capture multi-modes.

D Optimization of the Jeffreys Divergence

In Equation 8, we introduced the importance sampling approach to estimate the Jeffreys divergence. When training, we use Monte Carlo method to estimate its gradient, and use stochastic gradient descent to optimize the Jeffreys divergence. More specifically, we recall that MsIGN models a working distribution p_θ, where θ is the network parameter, we calculate its gradient as follows

$$\begin{aligned}
\frac{\partial}{\partial\theta}D_{\mathrm{J}}(p_\theta\|q) &= \frac{\partial}{\partial\theta}\left(D_{\mathrm{KL}}(p_\theta\|q)+D_{\mathrm{KL}}(q\|p_\theta)\right)\\
&= \frac{\partial}{\partial\theta}\left(\mathbb{E}_{\mathrm{x}\sim p_\theta}\left[\log\frac{p_\theta(\mathrm{x})}{q(\mathrm{x})}\right]+\mathbb{E}_{\mathrm{x}\sim\tilde{q}}\left[\frac{q(\mathrm{x})}{\tilde{q}(\mathrm{x})}\log\frac{q(\mathrm{x})}{p_\theta(\mathrm{x})}\right]\right)\\
&= \frac{\partial}{\partial\theta}\left(\int p_\theta(\mathbf{x})\log\frac{p_\theta(\mathbf{x})}{q(\mathbf{x})}\mathrm{d}\mathbf{x}+\int\tilde{q}(\mathbf{x})\frac{q(\mathbf{x})}{\tilde{q}(\mathbf{x})}\log\frac{q(\mathbf{x})}{p_\theta(\mathbf{x})}\mathrm{d}\mathbf{x}\right)\\
&= \int\left(\frac{\partial p_\theta(\mathbf{x})}{\partial\theta}\log\frac{p_\theta(\mathbf{x})}{q(\mathbf{x})}+p_\theta(\mathbf{x})\frac{\partial\log p_\theta(\mathbf{x})}{\partial\theta}\right)\mathrm{d}\mathbf{x}-\int\tilde{q}(\mathbf{x})\frac{q(\mathbf{x})}{\tilde{q}(\mathbf{x})}\frac{\partial\log p_\theta(\mathbf{x})}{\partial\theta}\mathrm{d}\mathbf{x}\,.
\end{aligned}$$

Now since $\frac{\partial}{\partial\theta}\log p_\theta(\mathbf{x})=\frac{1}{p_\theta(\mathbf{x})}\frac{\partial}{\partial\theta}p_\theta(\mathbf{x})$, we have $\frac{\partial}{\partial\theta}p_\theta(\mathbf{x})=p_\theta(\mathbf{x})\frac{\partial}{\partial\theta}\log p_\theta(\mathbf{x})$. So we continue

$$\begin{aligned}
\frac{\partial}{\partial\theta}D_{\mathrm{J}}(p_\theta\|q) &= \frac{\partial}{\partial\theta}\left(D_{\mathrm{KL}}(p_\theta\|q)+D_{\mathrm{KL}}(q\|p_\theta)\right)\\
&= \int\left(p_\theta(\mathbf{x})\frac{\partial\log p_\theta(\mathbf{x})}{\partial\theta}\log\frac{p_\theta(\mathbf{x})}{q(\mathbf{x})}+p_\theta(\mathbf{x})\frac{\partial\log p_\theta(\mathbf{x})}{\partial\theta}\right)\mathrm{d}\mathbf{x}\\
&\quad-\int\tilde{q}(\mathbf{x})\frac{q(\mathbf{x})}{\tilde{q}(\mathbf{x})}\frac{\partial\log p_\theta(\mathbf{x})}{\partial\theta}\mathrm{d}\mathbf{x}\\
&= \mathbb{E}_{\mathrm{x}\sim p_\theta}\left[\left(1+\log\frac{p_\theta(\mathrm{x})}{q(\mathrm{x})}\right)\frac{\partial\log p_\theta(\mathrm{x})}{\partial\theta}\right]-\mathbb{E}_{\mathrm{x}\sim\tilde{q}}\left[\frac{q(\mathrm{x})}{\tilde{q}(\mathrm{x})}\frac{\partial\log p_\theta(\mathrm{x})}{\partial\theta}\right]\,.
\end{aligned}$$

We comment that samples and (unnormalized) density to p_θ are accessible, because p_θ is modeled by MsIGN, who is essentially a flow-based generative network. We can compute the gradient $\frac{\partial}{\partial\theta}\log p_\theta$ by backward propagation. Samples and density to $\tilde{q}$ is also accessible because $\tilde{q}$ is essentially the MsIGN model at the last level. Therefore, the Jeffreys divergence and its gradient can be estimated by Monte Carlo method with samples of p_θ and $\tilde{q}$.

E Bayesian Inverse Problems Experimental Details

E.1 Experiment Settings

We place a Gaussian distribution with a Laplacian-type covariance $\mathcal{N}\left(\mathbf{0},\beta^2(-\boldsymbol{\Delta})^{-1-\alpha}\right)$ for both of our Bayesian inverse problem examples. Here the Laplacian operator $\boldsymbol{\Delta}$ can be understood as a graph Laplacian when we consider $\mathbf{x}$ as a vector on a 2-D uniform lattice, or it can be understood as the operator gives the divergence of the gradient when we consider $\mathrm{x}=\mathrm{x}(s)$ as a function on the unit disk $\Omega=[0,1]^2\subset\mathbb{R}^2$ (as introduced in Section 4.1, we don't distinguish these two interpretation for $\mathbf{x}$ or x). We choose zero Dirichlet boundary condition for $\boldsymbol{\Delta}$. As for the distribution to model noise (error) as in Equation 1, we set $\boldsymbol{\Gamma}_{\boldsymbol{\varepsilon}}=\sigma_{\boldsymbol{\varepsilon}}^2\boldsymbol{I}$. Specifically, we set $(\alpha,\beta,\sigma_\varepsilon)=(0.1,2.0,0.2)$ for the synthetic Bayesian inverse problem, and set $(\alpha,\beta,\sigma_\varepsilon)=(0.5,2.0,0.02)$ for the elliptic Bayesian inverse problem.

The synthetic Bayesian inverse problem sets its ground-truth for x as $\mathrm{x}(s)=\sin(\pi s_1)\sin(2\pi s_2)$, and generates its observation by the formula $\boldsymbol{y}=\mathcal{F}(\mathbf{x})+\boldsymbol{\varepsilon}$, with $\mathcal{F}(\mathbf{x})=\langle\boldsymbol{f},\mathbf{x}\rangle^2=(\int_\Omega\varphi(s)\mathrm{x}(s)\mathrm{d}s)^2$ is a nonlinear measurement, where $\varphi(s)=\sin(\pi s_1)\sin(2\pi s_2)$.

The elliptic Bayesian inverse problem also sets its ground-truth for x as $\mathrm{x}(s)=\sin(\pi s_1)\sin(2\pi s_2)$, and generates its observation by the formula $\boldsymbol{y}=\mathcal{F}(\mathbf{x})+\boldsymbol{\varepsilon}$. The forward process $\mathcal{F}$ is given by

linear measurement of the solution to an elliptic partial differential equation (PDE) associate with x. $\mathcal{F}$ is given by

$$\mathcal{F}(\mathrm{x}) = \left[\int_\Omega \varphi_1(s)\mathrm{u}(s)\mathrm{d}s \quad \int_\Omega \varphi_2(s)\mathrm{u}(s)\mathrm{d}s \quad \cdots \quad \int_\Omega \varphi_m(s)\mathrm{u}(s)\mathrm{d}s\right]^T , \tag{14}$$

where φ_k are fixed measurement functions, and u is the solution of an elliptic PDE in below:

$$\begin{aligned} -\nabla \cdot \left(e^{\mathrm{x}(s)}\nabla \mathrm{u}(s)\right) &= f(s)\,, \quad \text{for } s \in \Omega\,, \\ \mathrm{u}(s) &= 0\,, \quad \text{for } s \in \partial\Omega\,. \end{aligned} \tag{15}$$

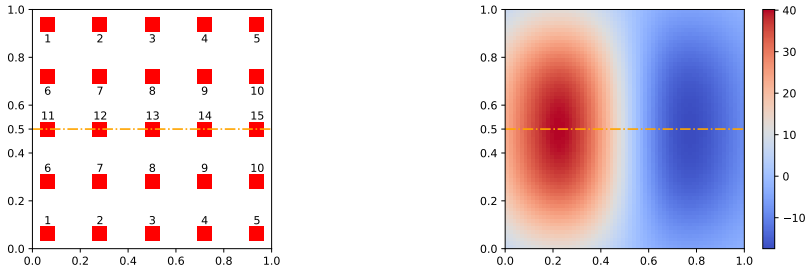

Figure 8: Left: 15 measurement functions in Equation 14. Here we plotted non-zero patches of φ_k, $k = 1, \ldots, 15$, with k labeled next to them. φ_k has a constant non-zero value on its patch(es) and is zero else where. The constant value here is chosen so that we have $\|\varphi_k\|_{L^2} = 1$. Right: The force term (Equation 16) of the elliptic PDE (Equation 15). We remark that both measurement functions and the force term are symmetric along the s_2 direction (the orange dash line).

We have $m = 15$ measurement functions φ_k that all detects local properties of u, and are all symmetry along the s_2 direction, see Figure 8. As for the force term of the elliptic PDE, we set

$$f(s) = \frac{100}{\pi} e^{-10\|s-f_1\|^2} + \frac{100}{\pi} e^{-10\|s-f_2\|^2} - \frac{50}{\pi} e^{-10\|s-f_3\|^2} - \frac{50}{\pi} e^{-10\|s-f_4\|^2}\,, \tag{16}$$

where $f_1 = (0.25, 0.3)$, $f_2 = (0.25, 0.7)$, $f_3 = (0.7, 0.3)$, $f_4 = (0.7, 0.3)$, and $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^2$. The force term is also symmetric along the s_2 direction, see Figure 8.

By the symmetric design, our posterior q has the property $q(\mathrm{x}) = q(\tilde{\mathrm{x}})$, where $\tilde{\mathrm{x}}$ is a function that satisfies $\tilde{\mathrm{x}}(s_1, 1 - s_2) = \mathrm{x}(s_1, s_2)$. We carefully choose our hyper-parameters $\alpha, \beta, \sigma_\varepsilon$, such that it ends up to be a symmetric double-modal posterior distribution. To certify the multi-modality, we run multiple gradient ascent searching of the posterior, starting from different initial points. They all converge to two mutually symmetric points x^* and $\tilde{\mathrm{x}}^*$. Visualization of the 1D landscape profile of the posterior q on the line passing through x^* and $\tilde{\mathrm{x}}^*$ also shows a clear double-modal feature.

To simulate the forward process $\mathcal{F}$, we solve the PDE (Equation 15) by the Finite Element Method with mesh size $1/64$. We remark here that no matter how the dimension of the posterior q changes, we always solve the elliptic PDE (Equation 15) using $1/64$ mesh size, because at this resolution the solution is well resolved numerically, and the computational cost is also moderate.

When counting the number of forward simulations (nFSs) as our indicator for computational cost, we notice that A-SVGD requires not only the log posterior $\log q(\mathbf{x})$ but also its gradient: $\partial_{\mathbf{x}} \log q(\mathbf{x})$. Thanks to the adjoint method, the gradient can be computed with only one extra forward simulation.

E.2 Training Details

In Table 3 we report our hyperparameter for network setting in the Bayesian inverse problem examples. Our multi-stage training of MsIGN enables the usage of the Jeffreys divergence as objective in all levels but the bottom one. For the bottom level $l = 1$, we still use the Jeffreys divergence $D_{\mathrm{J}}(p\|q) = \mathbb{E}_{\mathrm{x}\sim p}[\log(p(\mathrm{x})/q(\mathrm{x}))] + \mathbb{E}_{\mathrm{x}\sim q}[\log(q(\mathrm{x})/p(\mathrm{x}))]$ as training objective, but this time it is directly estimated by the Monte Carlo method with samples from distribution p and q. p samples

come from the model itself and q samples come from an HMC chain. We remark that at $l = 1$, the posterior lies in 4-D space, which is relatively a low-d problem, so an HMC run can approximate the target distribution q_1 well. Our present solution of seeking help from HMC can be replaced by some other strategies, like other MCMC methods, and deep generative networks.

For A-SVGD, we choose Glow (Kingma & Dhariwal, 2018) as its network design, with the same network hyperparameter in Table 3. Due to the fact that MsIGN is more parameter-saving than Glow with the same hyperparameter, A-SVGD model has more trainable parameters than our MsIGN model, reducing the possibility that that its network is not expressive enough to capture the modes.

As for our training of HMC, we grid search its hyperparameters, and use curves of acceptance rate and autocorrelation as evidence of mixing. We consider our HMC chain mixing successfully if the acceptance rate stabilizes and falls somewhere between $30\% - 75\%$, and the autocorrelation decays fast with respect to lag.

Table 3: Hyperparameter setting for MsIGN in Section 4.1. The meaning of terms can be found in Kingma & Dhariwal (2018).

Data Set	Minibatch Size	Levels (L)	Number of Glow Block	Hidden Channels
Synthetic BIP	100	6	16	32
Elliptic BIP	100	6	32	64

F Bayesian Inverse Problems Additional Results

In this section we provide more results on the Bayesian inverse problems examples in Section 4.1.

F.1 Synthetic Bayesian Inverse Problem

In Figure 9 we provide marginal distribution comparison at intermediate levels $l = 1, \ldots, 5$, and the learning curve of multi-stage MsIGN at $l = 6$. We can see from Figure 9 that as the dimension increases, A-SVGD starts to collapse to one mode, and HMC becomes imbalanced between modes. In other words, when the dimension goes beyond 64 (the dimension of q_3), HMC requires more computational budget for convergence. We remark here that in q_1, A-SVGD failed to capture both modes as it did to q_2. This phenomenon might be caused by the aliasing effect. Very rough resolution at this level pushes the prior to penalize the smoothness much, and also adds the sensitivity to likelihood because entries of $\mathbf{x}$ can easily influence its global behavior. Therefore, there is a larger log density gap between modes in the posterior q_1 than other levels, which adds up to the difficulty of multi-mode capture. A similar effect is observed in the elliptic example as in the next section.

The learning curve in Figure 9 shows the effectiveness of our multi-stage training of MsIGN. As we can see, the training process at $l = 6$ did improve the model, with the Jeffreys divergence dropped from 252 to 56.8. Rather than simply refining the resolution, our multi-stage training strategy does improve our approximation to the distribution when entering the next level. We will show more evidence about this in the next section.

F.2 Elliptic Bayesian Inverse Problem

In Figure 10 marginal comparison at intermediate levels $l = 1, \ldots, 5$ are presented. Again, for this complicated posterior we observe that A-SVGD failed in detecting all modes, and could even get stuck in the middle. In this testbed, HMC seems to capture both modes well. However we will point out that its samples can't be treated like a reference solution. The failure of HMC at q_1 is due to the aliasing effect: the prior penalizes fluctuation in spatial directions heavily, and the likelihood is also very strong. As a consequence, the posterior q_1 is highly twisted, and the log density gap between two modes becomes significant.

In Figure 11, we also show the necessity of training after prior conditioning. In other words, q_l is not the same as the prior-conditioned surrogate $\tilde{q}_{l-1}$, though they are similar. We plot one of the modes we detected by our models for $l = 4, 5, 6$. Comparing left figures and right figures of Figure 11, we can see the location, shape and scale of bumps and caves are different, which means the learned q_l is

different from the prior-conditioned surrogate $\tilde{q}_{l-1}$, who serves as its initialization. Our multi-stage training does learn more information at each level, rather than simply scale up the resolution.

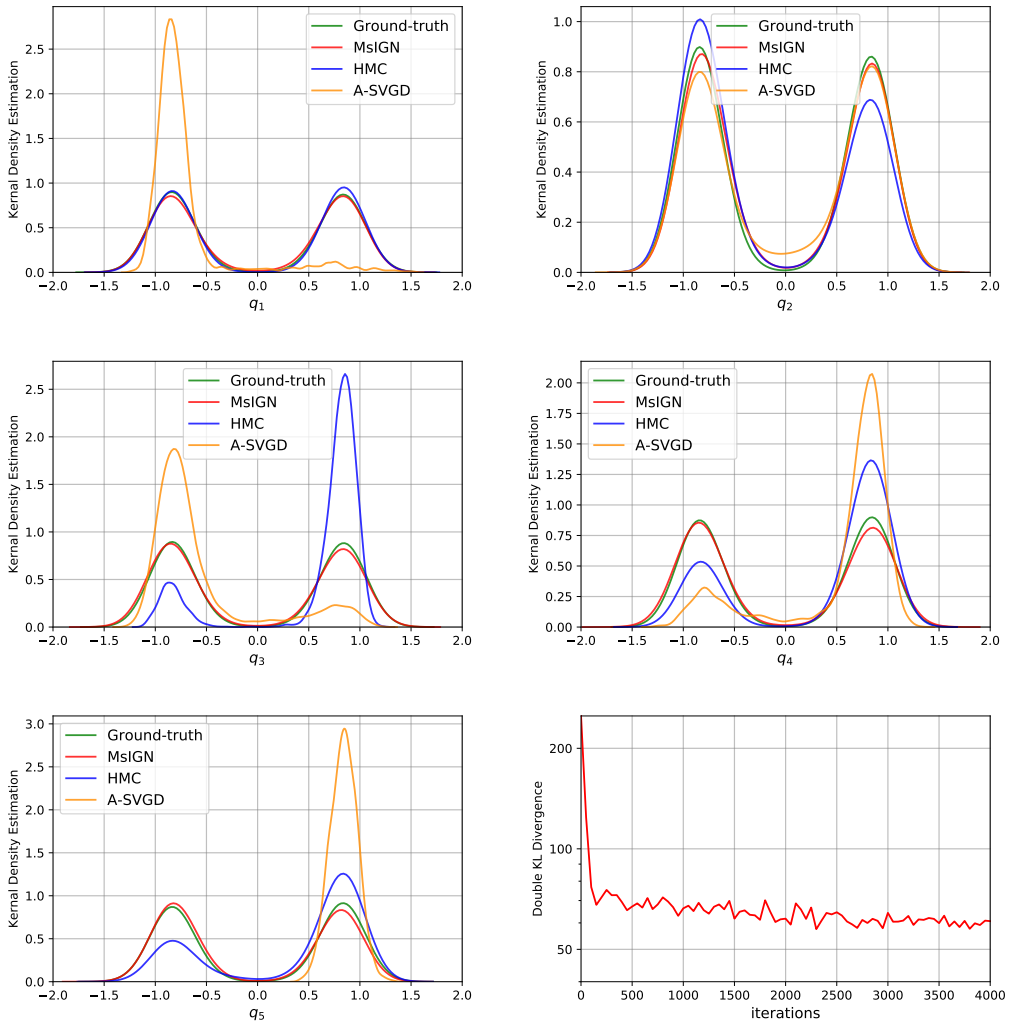

Figure 9: Synthetic BIP. Marginal comparison at the intermediate levels $l = 1, \ldots, 5$, and the learning curve of multi-stage MsIGN at $l = 6$. As the dimension increases, A-SVGD starts to collapse to one mode, and HMC becomes imbalanced between modes. The learning curve shows that the model distribution is constantly getting closer to the target distribution in the last stage of training, supporting the necessity of training after prior conditioning.

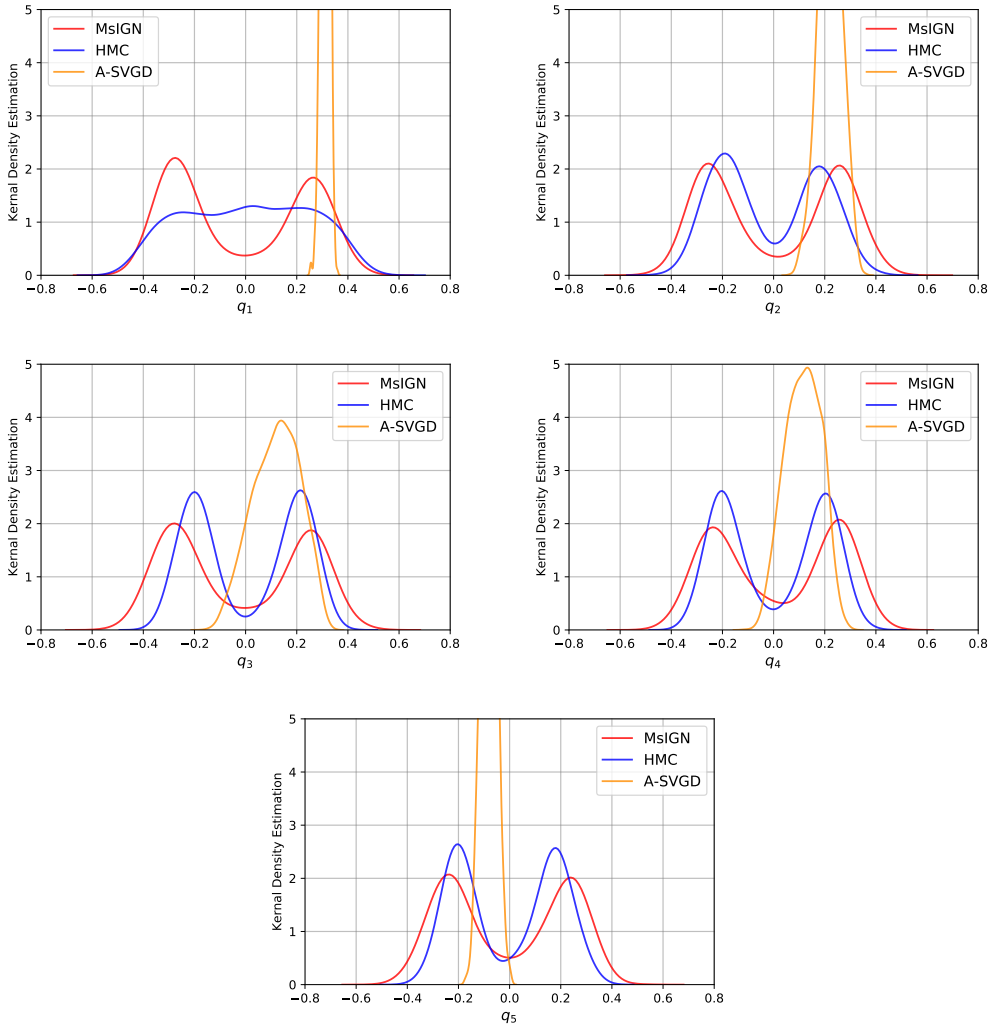

Figure 10: Marginal comparison in the intermediate levels $l = 1, \ldots, 5$. Amortized-SVGD fails in detecting modes at all levels. HMC has acceptable performance, but still suffers from imbalanced modes at some levels.

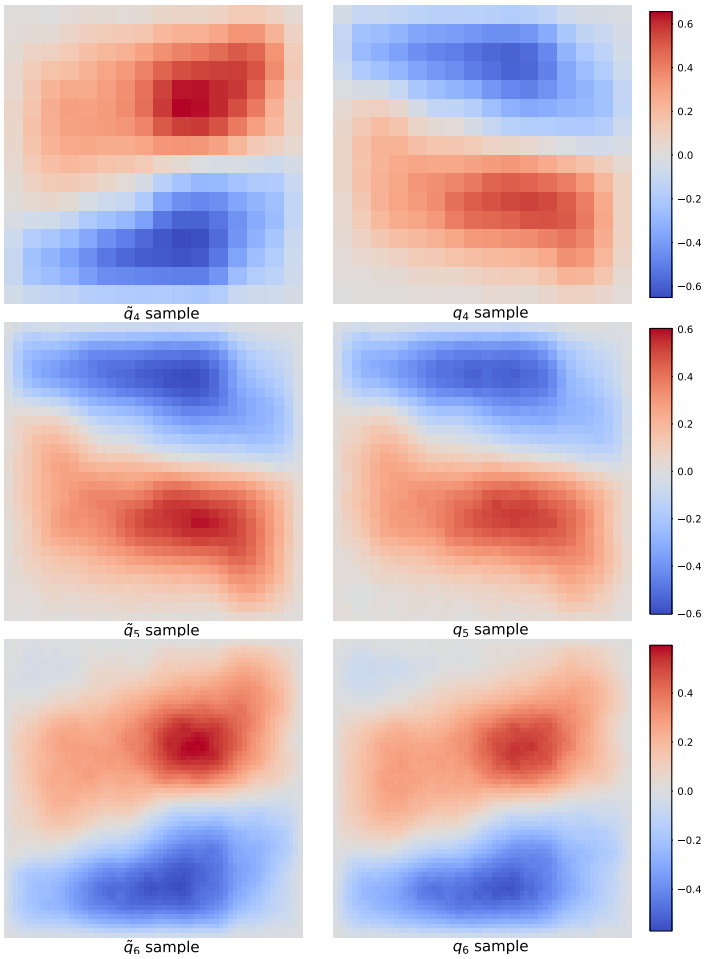

Figure 11: Comparison of the modes captured by the prior conditioned untrained model and the trained model. Bumps and caves in the left images are different from the right ones in shape, position, and scale. Therefore, prior conditioning provides a good initial guess, but training is still necessary.

G Ablation Study Additional Materials

We provide more experimental details and results on our ablation study in Section 4.1.3.

G.1 Experiment Setting

In Figure 4, all models involved Glow or MsIGN adopt network hyperparameters as shown in Table 3. As for the training details, we list them as follows:

- It is not straightforward to design multi-stage strategy for Glow models, because their channel size increases with l. So for models with different number of levels L, there is no direct way to initialize one model with another. Therefore for methods using Glow, we don't consider multi-stage training.
- As introduced in Appendix F.2, for the elliptic problem, the posterior at $l = 1$ is ill-posed, and MsIGN variants (like MsIGN trained by the KL divergence) can hardly capture its two modes, as shown in Table 4 and Figure 12. We report that in general it is unlikely for multi-stage training to pick up the missing mode. Therefore, to make more convincing comparison, for models with multi-stage training, we use pretrained MsIGN model at $l = 1$ (who captures q_1 well) as their initialization for $l = 2$.

G.2 More Results and Discussions

In Figure 4 we compared different variants of MsIGN and its training strategy at the level $l = 6$. In Figure 12 we plot the same comparison at all intermediate levels $l = 1, \ldots, 5$. Since the curves overlap each other heavily in Figure 12, we conclude their results of mode capturing (together with Figure 4) in Table 4.

Table 4: Table for mode capturing results by eye ball norm. T demotes the successful capturing of two modes, F denotes mode collapse, while I denotes biased, not well-separated modes capturing. For results marked with I, we refer readers to Figure 12 for detail information. Upper: synthetic Bayesian inverse problem; Lower: elliptic Bayesian inverse problem. *: we initialize the $l = 2$ model by our MsIGN $l = 1$ pretrained model.

Level	$l=1$	$l=2$	$l=3$	$l=4$	$l=5$	$l=6$
Glow	T	F	F	F	F	F
MsIGN-SNN	T	T	T	T	I	I
MsIGN-KL-S	T	F	F	F	I	F
MsIGN-KL*	T	T	T	T	T	T
MsIGN-AS-S	T	F	F	F	F	F
MsIGN-AS*	T	T	T	I	I	I
MsIGN	**T**	**T**	**T**	**T**	**T**	**T**
Level	$l=1$	$l=2$	$l=3$	$l=4$	$l=5$	$l=6$
Glow	F	F	F	F	F	F
MsIGN-SNN	F	F	F	F	F	F
MsIGN-KL-S	F	F	F	F	F	F
MsIGN-KL*	F	I	I	I	I	I
MsIGN-AS-S	F	F	F	F	F	F
MsIGN-AS*	F	I	I	T	I	I
MsIGN	**T**	**T**	**T**	**T**	**T**	**T**

We can see from Table 4 that our framework and strategy outperforms all its variants in these two Bayesian inverse problems, which proved the necessity of our prior conditioning layer, multi-stage training strategy, Jeffreys divergence, and our network design. In particular, the experiment of MsIGN-SNN supports our prior conditioning layer design, the experiment of MsIGN-KL supports our use of the Jeffreys divergence and MsIGN-KL-S supports our use of multi-stage training strategy.

Besides that, we can also see that multi-stage training also benefits other models like MsIGN with KL divergence objective or A-SVGD with MsIGN. By carefully comparing the marginals plotted in Figure 12, we can also conclude that Jeffreys divergence can help capture more balanced modes than KL divergence.

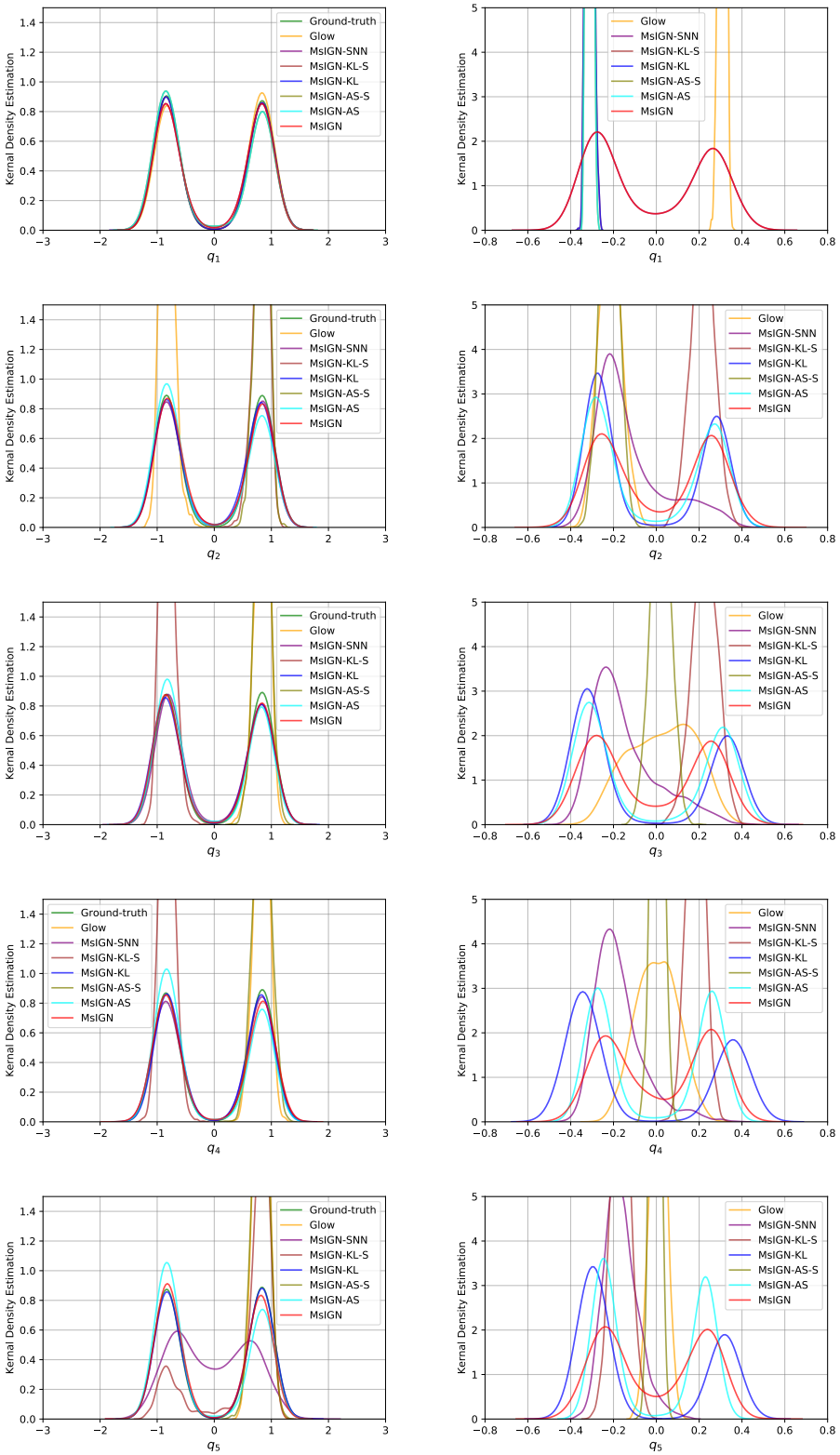

Figure 12: Ablation study at intermediate levels $l = 1, \ldots, 5$. Left: Synthetic Bayesian inverse problem; Right: Elliptic Bayesian inverse problem. For MsIGN-AS and MsIGN-KL, we initialize their $l = 2$ models by our MsIGN $l = 1$ pretrained model.

H IMAGE SYNTHESIS TASK EXPERIMENTAL DETAILS

We use the invertible block introduced in Kingma & Dhariwal (2018) as our model for the invertible flow. For our numbers in Table 2, we report our hyperparameter settings in Table 5. And we treat samples from those data sets as 8-bit images. For all experiments we use Adam optimizer with $\alpha = 0.001$ and default choice of β_1, β_2. For models here that requires hierarchical training, each upper level will be pretrained for only 125 epochs.

Table 5: Hyperparameter setting for results in Table 2. Here the meaning of terms can be found in Kingma & Dhariwal (2018). The size of hidden channels in the Glow block is fixed at 512.

Data Set	Minibatch Size	Levels (L)	Depth per level (K)	Epochs
MNIST	400	2	32	2000
CIFAR-10	400	3	32	2000
CelebA 64	200	3	32	1000
ImageNet 32	400	3	32	400
ImageNet 64	200	3	32	200

As for the prior conditioning layer in this image application, we let the upscale operator $\boldsymbol{A}_l$ from level l to level $l-1$ be average pooling, and thus the downscale operator $\boldsymbol{U}_{l-1}$ will be nearest upsampling, $l \geq 2$. We further assume the covariance $\boldsymbol{\Sigma}_l$ at each level be a scalar matrix, i.e. a diagonal matrix with equal diagonal elements.

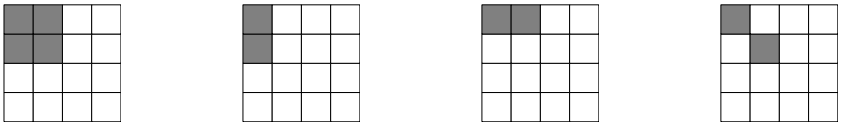

Figure 13: Leftmost: example row of A_l; The rest: example rows of H_l. They (with some unplotted ones) form the Haar basis, and can be expressed as local convolution operation.

Since $\boldsymbol{A}_l \in \mathbb{R}^{d_{l-1} \times d_l}$ is the average pooling operator, its rows, which give average of each local pool, is a subset of the Haar basis, see Figure 13. We can collect the rest Haar basis as $\boldsymbol{H}_l \in \mathbb{R}^{(d_l - d_{l-1}) \times d_l}$. Due to the orthogonality of Haar basis, there exists a constant λ_l such that

$$\begin{bmatrix} \boldsymbol{A}_l \boldsymbol{A}_l^T & \\ & \boldsymbol{H}_l \boldsymbol{H}_l^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_l \\ \boldsymbol{H}_l \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_l \\ \boldsymbol{H}_l \end{bmatrix}^T = \lambda_l \boldsymbol{I}_{d_l} = \begin{bmatrix} \boldsymbol{A}_l \\ \boldsymbol{H}_l \end{bmatrix}^T \begin{bmatrix} \boldsymbol{A}_l \\ \boldsymbol{H}_l \end{bmatrix} = \boldsymbol{A}_l^T \boldsymbol{A}_l + \boldsymbol{H}_l^T \boldsymbol{H}_l \,.$$

Since we assume the covariance $\boldsymbol{\Sigma}_l$ is a scalar matrix, we can find a scalar c_l such that $\boldsymbol{\Sigma}_l = c_l \boldsymbol{I}_{d_l}$. Now followed from Lemma 2.1 we can find an explicit form for $\boldsymbol{\Sigma}_{l|l-1}$, $l \geq 2$:

$$\begin{aligned} \boldsymbol{\Sigma}_{l|l-1} &= \boldsymbol{\Sigma}_l - \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{\Sigma}_l \boldsymbol{A}_l^T)^{-1} \boldsymbol{A}_l \Sigma_l \\ &= c_l \boldsymbol{I}_{d_l} - c_l \boldsymbol{A}_l^T (\boldsymbol{A}_l \boldsymbol{A}_l^T)^{-1} \boldsymbol{A}_l \\ &= c_l \boldsymbol{I}_{d_l} - c_l \boldsymbol{A}_l^T (\lambda_l \boldsymbol{I}_{d_{l-1}})^{-1} \boldsymbol{A}_l \\ &= \frac{c_l}{\lambda_l} \lambda_l \boldsymbol{I}_{d_l} - \frac{c_l}{\lambda_l} \boldsymbol{A}_l^T \boldsymbol{A}_l \\ &= \frac{c_l}{\lambda_l} (\boldsymbol{A}_l^T \boldsymbol{A}_l + \boldsymbol{H}_l^T \boldsymbol{H}_l) - \frac{c_l}{\lambda_l} \boldsymbol{A}_l^T \boldsymbol{A}_l \\ &= \frac{c_l}{\lambda_l} \boldsymbol{H}_l^T \boldsymbol{H}_l \,. \end{aligned}$$

Therefore we obtain the Cholesky decomposition of $\boldsymbol{\Sigma}_{l|l-1}$ for free as $\boldsymbol{B}_l = \mu_l \boldsymbol{H}_l$, with $\mu_l = \sqrt{\frac{c_l}{\lambda_l}}$. And we are only left to estimate the scalar μ_l for each $l \geq 2$.

We choose to estimate μ_l statistically. In fact we have accessible to different resolutions of images from the data set when we perform pooling operation. Since we hope at each level l, $\mathbf{x}_l$ looks like a natural image at its resolution, we set those images at that resolution as samples of $\mathbf{x}_l$, and estimate μ_l according to Equation 6:

$$\mathbf{x}_l = \boldsymbol{U}_{l-1}\mathbf{x}_{l-1} + \boldsymbol{B}_l\boldsymbol{z}_l = \boldsymbol{U}_{l-1}\mathbf{x}_{l-1} + \mu_l\boldsymbol{H}_l\boldsymbol{z}_l, \quad \boldsymbol{z}_l \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_l - d_{l-1}}) \,.$$

Using 10000 sample images from each data set, we report our estimates of μ_l in Table 6.

Table 6: Estimate of μ_l for different data sets and l.

Data Set	μ_2	μ_3	μ_4
MNIST	0.67	0.61	–
CIFAR-10	0.48	0.46	0.46
CelebA 64	0.22	0.3	0.38

I IMAGE SYNTHESIS TASK ADDITIONAL RESULTS

We attach more synthesized images by MsIGN from MNIST and CIFAR-10 in Figure 14, 15. For the CelebA data set, we made use of our multiscale design and trained our MsDGN for a higher resolution 128. In this case, the number of levels $L = 4$, and we set the hyperparameters for the first 3 levels the same as we use for the $64 * 64$ resolution model. For the last level $l = 4$, due to memory limitation, we set $K = 32$ and hidden channels 128. We show our synthesized $128 * 128$ results in Figure 16.

We also use this 4-level model to show the interpret-ability of our internal neurons in Figure 17. We snapshot internal neurons every 8 invertible blocks, resulting a chain of length $K * L/8 = 32 * 4/8 = 16$ for every generated image. We can see our MsIGN generates global features at the beginning levels and starts to add more local details at higher levels.

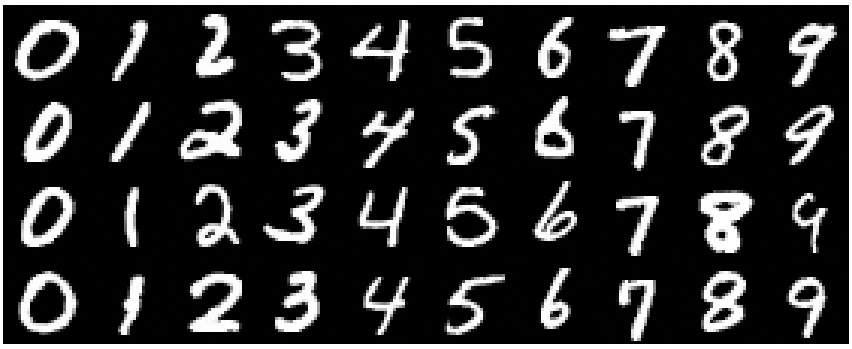

Figure 14: Synthesized images from MsIGN on the MNIST data set, temperature $= 1.0$. We show 4 samples for each digit.

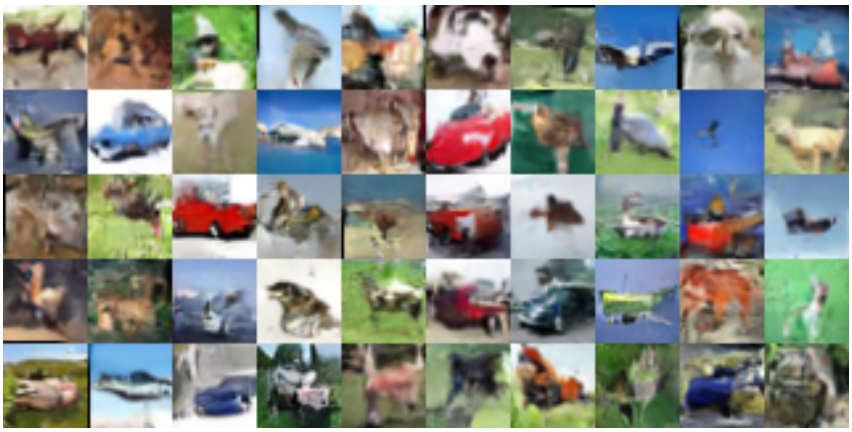

Figure 15: Synthesized images from MsIGN on the CIFAR-10 data set, temperature $= 1.0$.

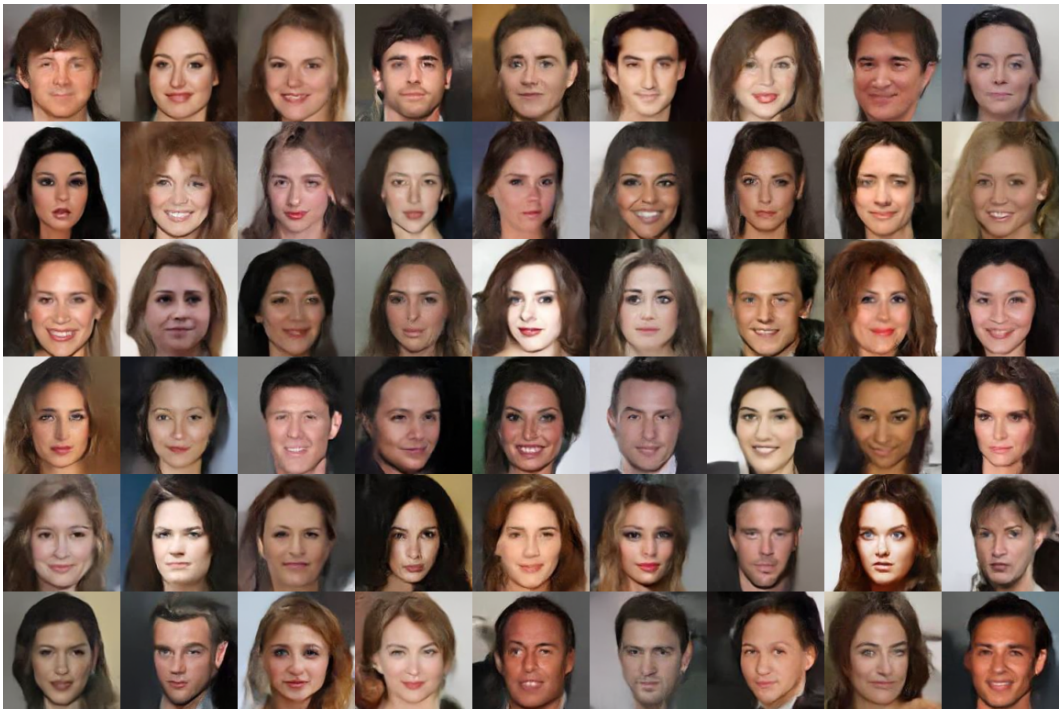

Figure 16: Synthesized images from MsIGN on the CelebA 128*128 data set, temperature $= 0.8$.

Figure 17: Visualization of internal neurons of MsIGN on CelebA 128*128 data set. Snapshots (from top to bottom) are taken every 8 invertible blocks at each level, resulting 16 checkpoints for every image generated. Each level will have 4 snapshots, and we separate them with snapshots from other levels. Left: when recovering images from the data set; Right: when synthesizing new images from random noise.

J POSTERIOR FORMULATION COMPARISON

We provide a detailed comparison of the definition of posterior with Ardizzone et al. (2018).

Authors in Ardizzone et al. (2018) purposed another approach for Bayesian inference that is different from ours in Equation 1. They framework can be simplified as finding an invertible map $[\boldsymbol{y}, \mathbf{z}] = T(\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^{d_x}$ is the quantity of interest, $\boldsymbol{y} \in \mathbb{R}^{d_y}$ is observation, and $\mathbf{z} \in \mathbb{R}^{d_z}$, $d_z = d_x - d_y$ is a latent variable. They write $T \in \mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$ as $T = [T_y, T_z]$ so that $\boldsymbol{y} = T_y(\mathbf{x})$, $\mathbf{z} = T_z(\mathbf{x})$ and require the following of T:

1. $T_y = \mathcal{F}$, i.e. T_y recovers the forward process,
2. When we impose the prior ρ on $\mathbf{x}$, its push-forward[2] $T_\sharp \rho$ can be decomposed as $p_y \otimes p_z$, the Cartesian product of its marginals in $\boldsymbol{y}$ and $\mathbf{z}$. And p_z is a standard multi-variate Gaussian distribution.

If such invertible map T is found, then they obtain their posterior sample of $\mathbf{x}$ by sampling $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_z})$ and feeding $(\boldsymbol{y}, \mathbf{z})$ to T^{-1}: $\mathbf{x} = T^{-1}(\boldsymbol{y}, \mathbf{z})$, where $\boldsymbol{y}$ is your observation. And naturally, authors in Ardizzone et al. (2018) use invertible neural networks (INN) to model, parameterize the map T, and introduce training strategy based on the above two requirements.

To avoid confusion, when we are given one observation $\boldsymbol{y}$, we use q_y to denote the posterior defined in Ardizzone et al. (2018), and use $q(\mathbf{x}|\boldsymbol{y})$ to denote our definition in Equation 1. As a reminder, we rewrite our definition below:

$$q(\mathbf{x}|\boldsymbol{y}) = \frac{1}{Z}\rho(\mathbf{x})L(\boldsymbol{y}|\mathbf{x}) = \frac{1}{Z}\rho(\mathbf{x})\rho_\varepsilon(\boldsymbol{y} - \mathcal{F}(\mathbf{x}))\,, \tag{17}$$

where Z is a normalizing constant, ρ_ε is the density function of the Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}_{\boldsymbol{\varepsilon}})$. As for q_y, from the definition above we can write (which is also proved in Appendix 1 of Ardizzone et al. (2018))

$$q_y = T_\sharp^{-1}(\delta_y \otimes p_z)\,, \tag{18}$$

where δ_y is a delta distribution placed at the observation $\boldsymbol{y}$. We remark here that in the definition of q_y there is no appearance of noise ε. The model in Ardizzone et al. (2018) assumes no error in measurement.

We state the following theorem which characterize the difference between $q(\mathbf{x}|\boldsymbol{y})$ and q_y:

Theorem J.1 *Suppose T and T^{-1} are differentiable, then $q(\mathbf{x}|\boldsymbol{y})$ is related to q_y by*

$$q(\mathbf{x}|\boldsymbol{y}) \propto \int q_{y'}(\mathbf{x})\rho_\varepsilon(\boldsymbol{y} - \boldsymbol{y}')p_y(\boldsymbol{y}')\mathrm{d}\boldsymbol{y}'\,. \tag{19}$$

Theorem J.1 shows how $q(\mathbf{x}|\boldsymbol{y})$ is different from $q_y(\mathbf{x})$. Besides the difference caused by noise modeling, there is a non-trivial distribution $p_y = T_{y\sharp}\rho = \mathcal{F}_\sharp\rho$.

Proof: In the following we will heavily make use of the change-of-variable formula, stating as follows:

$$T_\sharp^{-1}p(\mathbf{x}) = p(T(\mathbf{x}))|\log\det \mathrm{J}T(\mathbf{x})|\,,$$

for invertiable and differentiable T, where $\mathrm{J}T(\mathbf{x})$ is the Jacobian of T at $\mathbf{x}$.

We proceed by computing

$$\begin{aligned} q_{y'}(\mathbf{x})\rho_\varepsilon(\boldsymbol{y} - \boldsymbol{y}')p_y(\boldsymbol{y}') &= T_\sharp^{-1}(\delta_{y'} \otimes p_z)\rho_\varepsilon(\boldsymbol{y} - \boldsymbol{y}')p_y(\boldsymbol{y}') \\ &= \delta_{y'}(T_y(\mathbf{x}))p_z(T_z(\mathbf{x}))\rho_\varepsilon(\boldsymbol{y} - \boldsymbol{y}')p_y(\boldsymbol{y}')|\log\det \mathrm{J}T(\mathbf{x})|\,. \end{aligned}$$

[2]For $\mathbf{x} \sim \rho$ and a map T, the push-forward $T_\sharp\rho$ is a distribution defined by $T_\sharp\rho(A) := \rho(T^{-1}(A))$ for arbitrary set A. In other words, if $\mathbf{x} \sim \rho$, then $T(\mathbf{x}) \sim T_\sharp\rho$.

Therefore,

$$\int q_{y'}(\mathbf{x})\rho_\varepsilon(\boldsymbol{y}-\boldsymbol{y}')p_y(\boldsymbol{y}')\mathrm{d}\boldsymbol{y}' = \int \delta_{y'}(T_y(\mathbf{x}))p_z(T_z(\mathbf{x}))\rho_\varepsilon(\boldsymbol{y}-\boldsymbol{y}')p_y(\boldsymbol{y}')|\log\det \mathrm{J}T(\mathbf{x})|\mathrm{d}\boldsymbol{y}'$$
$$= p_z(T_z(\mathbf{x}))|\log\det \mathrm{J}T(\mathbf{x})| \int \delta_{y'}(T_y(\mathbf{x}))\rho_\varepsilon(\boldsymbol{y}-\boldsymbol{y}')p_y(\boldsymbol{y}')\mathrm{d}\boldsymbol{y}' \,,$$

and the property of delta function further simplifies the expression to

$$\begin{aligned}\int q_{y'}(\mathbf{x})\rho_\varepsilon(\boldsymbol{y}-\boldsymbol{y}')p_y(\boldsymbol{y}')\mathrm{d}\boldsymbol{y}' &= p_z(T_z(\mathbf{x}))|\log\det \mathrm{J}T(\mathbf{x})|\rho_\varepsilon(\boldsymbol{y}-T_y(\mathbf{x}))p_y(T_y(\mathbf{x})) \\ &= p_y(T_y(\mathbf{x}))p_z(T_z(\mathbf{x}))|\log\det \mathrm{J}T(\mathbf{x})|\rho_\varepsilon(\boldsymbol{y}-T_y(\mathbf{x})) \\ &= T_\sharp^{-1}(p_y\otimes p_z)(\mathbf{x})\rho_\varepsilon(\boldsymbol{y}-\mathcal{F}(\mathbf{x}))\,,\end{aligned}$$

where we use the change-of-variable formula reversely and the fact $T_y = \mathcal{F}$. And by definition we have $T_\sharp^{-1}(p_y \otimes p_z)(x) = \rho(\mathbf{x})$, so

$$\int q_{y'}(\mathbf{x})\rho_\varepsilon(\boldsymbol{y}-\boldsymbol{y}')p_y(\boldsymbol{y}')\mathrm{d}\boldsymbol{y}' = \rho(\mathbf{x})\rho_\varepsilon(\boldsymbol{y}-\mathcal{F}(\mathbf{x}))\,.$$

Finally we have proved Equation 19 according to the definition of $q(\mathbf{x}|\boldsymbol{y})$ in Equation 17. □

Theorem J.1 gives detailed explanation of the difference in posteriors compared to Ardizzone et al. (2018). Despite this difference, their target q_y must also exhibit double-modal property in the synthetic Bayesian inverse problem in Section 4.1.1 due to our symmetric design. Thus we also try training our MsIGN under the framework of Ardizzone et al. (2018), where they use the l^2 loss for the match of T_y and $\mathcal{F}$, and use maximum mean discrepancy loss for the match of $T_\sharp\rho$ and $p_y \otimes p_z$. We also adopt the bi-directional training technique. In Figure 18 we show the marginal distribution of models trained by INN. We can see INN seems to capture only one mode of the distribution. We also remark here that in Ardizzone et al. (2018) authors design INN mostly for low-d inverse problem.

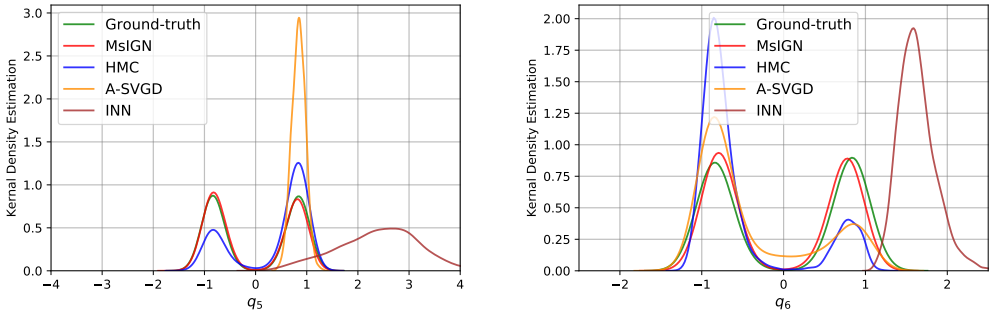

Figure 18: Comparison of INN and other methods in q_5 and q_6. INN seems to capture only one mode of their target distribution, which should also be double-modal due to our symmetric design. For other levels $l = 1, 2, 3, 4$, INN behaves similarly.