ROBUST GENERATIVE FLOWS ON RELIABLE IMAGE RECONSTRUCTION WITHOUT TRAINING DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

A key application of computational imaging is to determine the hidden information from a set of observed but sparse measurements. To fully characterize the uncertainty naturally induced by the sparse measurements, a robust inverse solver that is able to estimate the complete posterior of the unrecoverable targets is therefore important, with a potential to probabilistically interpret the observational data for decision making. In this work, we propose a deep variational framework that leverages a deep generative model to learn an approximate posterior distribution for quantifying image reconstruction uncertainty without training data. This is achieved by parameterizing the target posterior using a flow-based model and minimizing their KL divergence. To perform accurate uncertainty estimation, we propose a robust flow-based model where the stability is enhanced by adding bi-directional regularization and the expressivity is improved by using gradient boosting. We also found that the statistics of latent distribution are conservatively propagated to the posterior distribution through an invertible transformation and therefore introduce a space-filling design to achieve significant variance reduction on both latent prior space and target posterior space. We demonstrate our method on several benchmark tasks and two real-world applications (fastMRI and black hole image reconstruction) and show that it achieves a reliable and high-quality image reconstruction with robust uncertainty estimation.

1 INTRODUCTION

In computer vision and image processing, computational image reconstruction is a typical inverse problem where the goal is to learn and recover a hidden image $\mathbf{x}$ from directly measured data $\mathbf{y}$ via a forward operator $\mathcal{F}$. Such mapping $\mathbf{y} = \mathcal{F}(\mathbf{x})$, referred to as the *forward process* is often well-established. Unfortunately, the *inverse process* $\mathbf{x} = \mathcal{F}^{-1}(\mathbf{y})$, proceeds in the opposite direction, which is a nontrivial task since it is often ill-posed. A regularized optimization is formulated to recover the hidden image $\mathbf{x}^*$: $\mathbf{x}^* = \arg\min_{\mathbf{x}} \{\mathcal{L}(\mathbf{y}, \mathcal{F}(\mathbf{x})) + \lambda\, \omega(\mathbf{x})\}$, where $\mathcal{L}$ is a loss function to measure the difference between the measurement data and the forward prediction, ω is a regularization function and λ is a regularization weight. The regularization function, including ℓ_1-norm and total variation, are often used to constrain the image to a unique inverse solution in under-sampled imaging systems (Bouman & Sauer, 1993; Strong & Chan, 2003).

Recent trends have focused on using deep learning for computational image reconstruction, which does not rely on an explicit forward model or iterative updates but performs learned inversion from representative large datasets (Zhu et al., 2018; Belthangady & Royer, 2019; Tonolini et al., 2020; Wang et al., 2020), with applications in medical science, biology, astronomy and more. However, most of these existing studies in regularized optimization (Natterer & Wübbeling, 2001; Park et al., 2003) and feed-forward deep learning approaches (Ulyanov et al., 2018; Belthangady & Royer, 2019; Wang et al., 2020) mainly focus on pursuing a unique inverse solution by recovering a single point estimate. This leads to a significant limitation when working with underdetermined systems where it is conceivable that multiple inverse image solutions would be equally consistent with the measured data (Barbano et al., 2020; Sun & Bouman, 2020). Practically, in many cases, only partial and limited measurements are available which naturally leads to a *reconstruction uncertainty*. Thus, a reconstruction using a point estimate without uncertainty quantification would potentially mislead the decision making process (Beliy et al., 2019; Zhang et al., 2019; Zhou et al., 2020). Therefore, the ability to characterize and quantify reconstruction uncertainty is of paramount relevance. In principle,

Bayesian methods are an attractive route to address the inverse problems with uncertainty estimation. However, in practice, exact Bayesian treatment of complex problems is usually intractable. The common limitation is to resort to inference and sampling, typically by MCMC, which are often prohibitively expensive for imaging problems due to the *curse of dimensionality*.

For nonlinear, non-convex image reconstruction problems, a deeper architecture with enough expressivity may be required to approximate the complex posterior distribution. However, increasing model depth will result in overfitting, as well as making sampling and computing inefficient. The essential assumption of invertibility is also potentially violated along with instability issues caused by the aggregation of numerical errors. The imprecision and variation in flow-based models induce additional uncertainties. Moreover, the variation in posterior distribution caused by latent space sampling is also non-negligible for the evaluation of the total reconstruction uncertainty. These challenges make accurate uncertainty estimation to be a nontrivial task.

In this work, we achieve a reliable image reconstruction with an accurate estimation of *data* uncertainty resulting from measurement noise and sparsity. A suitable flow-based variational approach is proposed to approximate the posterior distribution of a target image *without any training data*. In this work, we propose a novel uncertainty-aware method that leverages a deep variational approach with robust generative flows and variance-reduced sampling to perform accurate characterization and quantification of reconstruction uncertainty, caused by sparse and noisy measurements. Our method minimizes the model uncertainties by building a robust flow-based model where the stability is enhanced by adding a bi-directional regularization and the flexibility is improved by leveraging gradient boosting. We observed that the statistics of latent distribution are conservatively propagated to the posterior through a deterministic invertible transformation, and therefore replace simple random sampling by using a generalized Latin Hypercube Sampling, which shows significant variance reduction on posterior approximation. We demonstrate our method on FastMRI reconstruction and interferometric imaging (EHT black hole) problems and show that it achieves a reliable and high-quality reconstruction with accurate uncertainty evaluation.

2 Related works

Bayesian deep learning. Deep learning for solving inverse problems requires uncertainty estimation to be reliable in real settings. Bayesian deep learning (Kendall & Gal, 2017; Khan et al., 2018; Wilson & Izmailov, 2020), specifically Bayesian neural networks (Hernández-Lobato & Adams, 2015; Gal) can achieve this goal while offering a computationally tractable way for recovering reconstruction uncertainty. However, exact inference in the BNN framework is not a trivial task, so several variational approximation approaches are proposed to deal with the scalability challenges. Monte Carlo dropout (Gal & Ghahramani, 2016) can be seen as a promising alternative approach that is easy to implement and evaluate. Deep ensemble (Lakshminarayanan et al., 2017) methods proposed by combining multiple deep models from different initialization have outperformed BNN. Recent methods on deterministic uncertainty quantification (Van Amersfoort et al., 2020; van Amersfoort et al., 2021) use a single forward pass but scales well to large datasets. Although these approaches show impressive performance, they rely on supervised learning with paired input-output datasets and only characterize the uncertainty conditioned on a training set.

Variational approaches. Variational methods offer a more efficient alternative approximating true but intractable posterior distribution by an optimally selected tractable distribution family (Blei et al., 2017). However, the restriction to limited distribution families fails if the true posterior is too complex. Recent advances in conditional generative models, such as conditional GANs (cGANs) (Wang et al., 2018), overcome this restriction in principle, but have limitations in satisfactory diversity in practice. Another commonly adopted option is conditional VAEs (cVAEs) (Sohn et al., 2015), which outperform cGANs in some cases, but in fact, the direct application of both conditional generative models in computational imaging is challenging because a large number of data is typically required (Tonolini et al., 2020). This introduces additional difficulties if our observations and measurements are sparse, and expensive to collect.

Deep flow-based models. Many very recent efforts have been made on solving inverse problems via deep generative models (Asim et al., 2020; Whang et al., 2021b; Bora et al., 2017; Whang et al., 2021a; Daras et al., 2021; Sun & Bouman, 2020; Kothari et al., 2021). The flow-based model is a potential alternative via learning of a nonlinear transformation between the true posterior distribution

and a simple prior distribution (Rezende & Mohamed, 2015; Dinh et al., 2016; Kingma & Dhariwal, 2018; Grathwohl et al., 2018; Wu et al., 2020; Nielsen et al., 2020). These flow-based models possess critical properties: (a) the neural architecture is *invertible*, (b) the forward and inverse mapping is efficiently *computable* and (c) the Jacobian is *tractable*, which allows explicit computation of posterior probabilities. Fully invertible neural networks (INNs) are a natural choice to satisfy these properties and can be built using coupling layers, as introduced in the RealNVP (Dinh et al., 2016) which is simple and efficient to compute. Although in principle, RealNVP layers are theoretically invertible, the actual computational of their inverse is not stable and sometimes even non-invertible due to aggravating numerical errors (Behrmann et al., 2021). Many efforts have been spent to improve the stability, invertibility, flexibility and expressivity of the flow-based models (Kong & Chaudhuri, 2020; Nielsen et al., 2020; Wu et al., 2020), which inspired us to extend for the task of computing posterior in real-world imaging reconstruction problems.

Recently, Asim *et al.* (Asim et al., 2020) focused on producing a point estimate motivated by the MAP formulation and Wang *et al.* (Whang et al., 2021b) aims at studying the full distributional recovery via variational inference. A follow-up study from Whang et al. (2021a) is to study image inverse problems with a normalizing flow prior, which is achieved by proposing a formulation that views the solution as the maximum a posteriori estimate of the image conditioned on the measurements. Our work is motivated by these recent advances but focuses on how to assess the image reconstruction (data) uncertainty with an explicitly known forward model with very sparse observations.

3 BACKGROUND

Normalizing flows. Generative models, such as GANs and VAEs, are intractable for explicitly learning the probability density function which plays a fundamental role in uncertainty estimation. Flow-based generative models overcome this difficulty with the help of *normalizing flows* (NFs), which describe the transformation from a latent density $\mathbf{z}_0 \sim \pi_0(\mathbf{z}_0)$ to a target density $\tau(\mathbf{x})$, where $\mathbf{x} = \mathbf{z}_K \sim \pi_K(\mathbf{z}_K)$ through a sequence of invertible mappings $\mathcal{T}_k : \mathbb{R}^d \rightarrow \mathbb{R}^d, k = 1, ..., K$. By using the change of variables rule $\tau(\mathbf{x}) = \pi_k(\mathbf{z}_k) = \pi_{k-1}(\mathbf{z}_{k-1}) \left|\det \left(\partial \mathcal{T}_k^{-1} / \partial \mathbf{z}_{k-1}\right)\right|$, where the target density $\pi_K(\mathbf{z}_K)$ obtained by successively transforming a random variable $\mathbf{z}_0$ through a chain of K transformations $\mathbf{z}_K = \mathcal{T}_K \circ \cdots \circ \mathcal{T}_1(\mathbf{z}_0)$ is $\log \tau(\mathbf{x}) = \log \pi_K(\mathbf{z}_K) = \log \pi_0(\mathbf{z}_0) - \sum_{k=1}^{K} \log \left|\det \left(\partial \mathcal{T}_k / \partial \mathbf{z}_{k-1}\right)\right|$ where each transformation $\mathcal{T}_k$ must be sufficiently *expressive* while being theoretically *invertible* and efficient to compute the Jacobian determinant. Affine coupling functions (Dinh et al., 2016; Kingma & Dhariwal, 2018) are often used because they are simple and efficient to compute. However, these benefits come at the cost of expressivity and flexibility; many flows must be stacked to learn a complex representation.

Density estimation. Assuming that samples $\{\boldsymbol{x}_i\}_{i=1}^{M}$ drawn from a probability density $p(\mathbf{x})$ are available, our goal is to learn a flow-based model $\tau_\phi(\mathbf{x})$ parameterized by the vector ϕ through a transformation $\mathbf{x} = \mathcal{T}(\mathbf{z})$ of a latent density $\pi_0(\mathbf{z})$ with $\mathcal{T} = \mathcal{T}_K \circ \cdots \circ \mathcal{T}_1$ as a K-step flow. This is achieved by minimizing the KL-divergence $\mathrm{D_{KL}} = \mathrm{KL}(p(\mathbf{x}) \parallel \tau_\phi(\mathbf{x}))$, which is equivalent to MLE.

Variational inference. The goal is to approximate the posterior distribution p through a variational distribution π_K encoded by a flow-based model $\tau_\phi(\mathbf{x})$, which is tractable to compute and draw samples. This is achieved by minimizing the KL-divergence $D_{\mathrm{KL}} = \mathrm{KL}(\pi_K \parallel p)$, which is equivalent to maximizing an evidence lower bound $\mathcal{V}_\phi(\mathbf{x}) = \mathbb{E}_{\mathbf{z}_K \sim \pi_K(\mathbf{z}_K)} \left[-\log p(\mathbf{x}, \mathbf{z}_K) + \log \pi_K(\mathbf{z}_K|\mathbf{x})\right]$.

Evaluation metrics for deep generative models. Designing indicative evaluation metrics for generative models and samples remains a challenge. A commonly used metric for measuring the similarity between real and generated images has been the Fréchet Inception Distance (FID) score (Heusel et al., 2017) but it fails to separate two critical aspects of the quality of generative models: *fidelity* that refers to the degree to which the generated samples resemble the real ones, and *diversity*, which measures whether the generated samples cover the full variability of the real samples. Sajjadi *et al.* (Sajjadi et al., 2018) proposed the two-value metrics (*precision* and *recall*) to capture the two characteristics separately. Recently, Naeem *et al.* (Naeem et al., 2020) introduce two reliable metrics (*density* and *coverage*) to evaluate the quality of the generated posterior samples and measure the difference between them and ground truth. *density* improves upon the precision metric by dealing with the overestimation issue and *coverage* instead of recall metric is to better measure the diversity by building the nearest neighbour manifolds around the true samples.

4 METHODOLOGY

4.1 DEEP VARIATIONAL FRAMEWORK

Our goal is to build a deep variational framework to accurately estimate the data uncertainty quantified by an approximation of the posterior distribution. The regularized optimization for solving inverse problems can be written in terms of data fidelity (data fitting loss) and regularity:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \{\mathcal{L}_{\mathrm{D}}(\mathbf{y}, \mathcal{F}(\mathbf{x})) + \lambda\omega(\mathbf{x})\} = \arg\min_{\mathbf{x}} \parallel \mathbf{y} - \mathcal{F}(\mathbf{x}) \parallel^2 + \lambda\omega(\mathbf{x}) \tag{1}$$

Assuming the forward operator $\mathcal{F}$ is known and the measurement noise statistics are given, we can reformulate the inverse problem in a probabilistic way. In Bayesian perspective, the regularized inverse problem in Eq. 1 can be solved by Bayesian inference but aims to maximize the posterior distribution by searching a point estimator $\mathbf{x}^*$:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{y}) = \arg\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}) \tag{2}$$

where the prior distribution $p(\mathbf{x})$ (e.g., image prior (Ulyanov et al., 2018) in reconstruction problems) defines a similar regularization term and data likelihood $p(\mathbf{y}|\mathbf{x})$ corresponds to the data fidelity in Eq. 1. If we parameterize the target $\mathbf{x}$ using a generative model $\mathbf{x} = \mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}), \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ with model parameter $\boldsymbol{\phi}$, an approximate posterior distribution $\tau_{\boldsymbol{\phi}^*}(\mathbf{x})$ is obtained by minimizing the KL-divergence between the generative distribution and the target posterior distribution

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathrm{KL}(\tau_{\boldsymbol{\phi}}(\mathbf{x}) \parallel p(\mathbf{x}|\mathbf{y})) = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{\mathbf{x}\sim\tau_{\boldsymbol{\phi}}(\mathbf{x})}[-\log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{x}) + \log \tau_{\boldsymbol{\phi}}(\mathbf{x})] \tag{3}$$

Unfortunately, the probability density (likelihood) $\tau_{\boldsymbol{\phi}}(\mathbf{x})$ cannot be exactly evaluated by most of existing generative models, such as GANs or VAEs. Flow-based models offer a promising approach to compute the likelihood exactly via the change of variable theorem with invertible architectures. Therefore, we can reformulate the equation in terms of a flow-based model as

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{\mathbf{z}\sim\pi(\mathbf{z})} \left[-\log p(\mathbf{y}|\mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z})) - \log p(\mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z})) + \log \pi(\mathbf{z}) - \log \left| \det \nabla_{\mathbf{z}} \mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}) \right| \right] \tag{4}$$

Replacing data likelihood and prior terms by using data fidelity loss and regularization function in Eq. 1, we can define a new optimization problem where it can be approximated by a Monte Carlo method in practice:

$$\begin{aligned} \boldsymbol{\phi}^* &= \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{\mathbf{z}\sim\pi(\mathbf{z})} \left[\mathcal{L}_{\mathrm{D}}(\mathbf{y}, \mathcal{F}(\mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}))) + \lambda\omega(T_{\boldsymbol{\phi}}(\mathbf{z})) + \log \pi(\mathbf{z}) - \log \left| \det \nabla_{\mathbf{z}} \mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}) \right| \right] \\ &= \arg\min_{\boldsymbol{\phi}} \sum_{j=1}^{M} \left[\mathcal{L}_{\mathrm{D}}(\mathbf{y}, \mathcal{F}(\mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}_j))) + \lambda\omega(T_{\boldsymbol{\phi}}(\mathbf{z}_j)) - \log \left| \det \nabla_{\mathbf{z}} \mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}_j) \right| \right] \end{aligned} \tag{5}$$

where $\pi(\mathbf{z})$ is a constant and $\log \left| \det \nabla_{\mathbf{z}} \mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}_j) \right|$ is an entropy that is critical to encourage sample diversity and exploration to avoid generative models from collapsing to a deterministic solution.

Noted that the flow-based model is very critical and sensitive to uncertainty estimation and quantification within this variational framework. To perform accurate data uncertainty estimation, the uncertainty associated with the flow-based model must be minimized. To this end, we develop a Robust Generative Flow (RGF) model with enhanced stability, expressivity, and flexibility, while conserving efficient inference and sampling without increasing architecture depth.

4.2 ROBUST GENERATIVE FLOWS (RGF)

4.2.1 STABILITY OF INVERTIBLE ARCHITECTURES

The proposed variational framework relies on the essential assumption of the theoretical invertibility in flow-based models. However, this assumption is challenged by recent studies (Behrmann et al., 2019; 2021; Kobyzev et al., 2020) with findings that the commonly used invertible architectures $\mathcal{T}_{\boldsymbol{\phi}}$, such as additive and affine coupling blocks, suffer from exploding inverses and thus prone to becoming numerically non-invertible (Hoffman et al., 2019), which will violate the assumption underlying their main advantages, including efficient sampling and exact likelihood estimation.

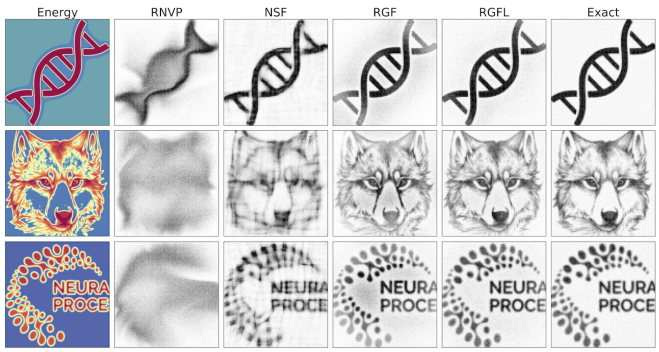

Figure 1: Sampling of 2D densities using variational inference with energy function.

Metrics	RNVP	NSF	RGF	RGFL
Precision	0.951	0.988	0.992	**0.996**
Recall	0.996	0.994	**0.997**	**0.997**
Density	0.823	0.962	0.987	**0.989**
Coverage	0.909	0.926	0.950	**0.961**
Metrics	RNVP	NSF	RGF	RGFL
Precision	0.988	0.993	0.994	**0.997**
Recall	0.997	0.998	0.998	**0.999**
Density	0.889	0.948	0.989	**1.001**
Coverage	0.912	0.951	0.946	**0.964**
Metrics	RNVP	NSF	RGF	RGFL
Precision	0.969	0.987	0.993	**0.994**
Recall	0.994	**0.996**	**0.996**	**0.996**
Density	0.829	0.942	0.987	**1.003**
Coverage	0.877	0.934	0.940	**0.965**

Table 1: Evaluation metrics.

Typically, the coupling blocks show an exploding inverse effect because the singular values of the forward mapping tend to zero as depth increases. The numerical errors introduced in both forward and inverse mapping will aggravate the imprecision and instability which renders the architecture non-invertible and results in uncontrollable model uncertainties that are intractable to characterize.

The stability of invertible neural network (INN) architectures can be analyzed by the property of bi-Lipschitz continuity if there exists a constant $L := \text{Lip}(\mathcal{T})$ and a constant $L^* := \text{Lip}(\mathcal{T}^{-1})$ such that for all $x_1, x_2; y_1, y_2 \in \mathbb{R}^d$

$$||\mathcal{T}(x_1) - \mathcal{T}(x_2)|| \leq L||x_1 - x_2||, \quad ||\mathcal{T}^{-1}(y_1) - \mathcal{T}^{-1}(y_2)|| \leq L^*||y_1 - y_2||. \tag{6}$$

Penalty terms on the Jacobian can be used to enforce INN stability locally. If $\mathcal{T}$ is Lipschitz continuous and differentiable, we have $\text{Lip}(\mathcal{T}) = \sup_{\mathbf{x}\in\mathbb{R}^d} ||J_{\mathcal{T}}(\mathbf{x})||_2$. Instead of estimating $\text{Lip}(\mathcal{T})$ using random samples, we use finite differences (FD) to approximate $\text{Lip}(\mathcal{T})$ as

$$\text{Lip}(\mathcal{T}) = \sup_{\mathbf{x}\in\mathbb{R}^d} ||J_{\mathcal{T}}(\mathbf{x})||_2 \approx \sup_{\mathbf{x}\in\mathbb{R}^d} \sup_{||\nu||_2=1} \frac{1}{\varepsilon}||\mathcal{T}(\mathbf{x}) - \mathcal{T}(\mathbf{x} + \varepsilon\nu)||_2 \tag{7}$$

where $\varepsilon > 0$ is a step size in FD. This penalty term (as a regularizer) can be added to the loss function on both directions $\mathcal{T}$ and $\mathcal{T}^{-1}$ such that we have a stable forward and inverse mapping (see more details and code implementation in Appendix 2.1.). This bi-directional FD regularization can remedy the non-invertible failures in many coupling blocks including spline function.

4.2.2 Enhanced expressivity and flexibility

Recent trends in NFs have focused on creating deeper, more complex transformations to increase the flexibility of the learned distribution. However, a deeper structure of flows often renders instability and uncertainty caused by aggregation of the numerical errors. Also, with greater model complexity comes a greater risk of overfitting while slowing down training, sampling, and inference (Giaquinto & Banerjee, 2020). To address these issues, we propose to use a gradient boosting approach for increasing the expressiveness of the neural spline flow (NSF) model (Durkan et al., 2019). Our new model is built by iteratively adding new NF components with gradient boosting, where each new NF component is fit to the residual of the previously trained components. A weight is then learned via stochastic gradient descent for each component, which results in a mixture model structure, whose flexibility increases as more components are added.

Gradient boosting flow model. A gradient boosting flow model is constructed by successively adding new components, where each new component $t_K^{(c)}$ is a K-step normalizing flow that matches the functional gradient of the loss function from the $(c-1)$ previously trained flow components $\mathcal{T}_k^{(c-1)}$. Typically, the gradient boost flow model is constructed by a convex combination of fixed and new flow components:

$$\mathcal{T}_K^{(c)}(\mathbf{z} \mid \mathbf{x}) = (1 - \beta_c)\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x}) + \beta_c t_K^{(c)}(\mathbf{z} \mid \mathbf{x}) \tag{8}$$

where $\mathbf{x}$ are the observed data, $\mathbf{z}$ are the latent variables and β_c is the weight to the new component where $\beta_c \in [0, 1]$ to sure the mixture model in Eq. 18 is a valid probability distribution.

To pursue a variational posterior that closely matches the true posterior, as shown in Eq. 14, which corresponds to the reverse KL-divergence $\mathrm{KL}(\tau_{\boldsymbol{\phi}}(\mathbf{x}) \parallel p(\mathbf{x}|\mathbf{y}))$. Thus, we seek to minimize the variational bound:

$$\mathcal{V}_{\phi}(\mathbf{x}) = \mathbb{E}_{\mathcal{T}_K^{(c)}}[\log \mathcal{T}_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}) - \log p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)]. \tag{9}$$

Boosting components updating. Given the objective function in Eq. 20, we proceed with deriving updates for new boosting components. First, at the current stage c, we let $\mathcal{T}_K^{(c-1)}$ to be fixed, and the target is learning the component $t_K^{(c)}$ and the weight β_c based on functional gradient descent (FGD) (Mason et al., 1999). We take the gradient of objective in Eq. 20 with respect to $\mathcal{T}_K^{(c)}$ at $\beta_c \to 0$:

$$\nabla_{\mathcal{T}_K^{(c)}} \mathcal{V}_{\phi}(\mathbf{x})\big|_{\beta_c \to 0} = -\log \frac{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)}{\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x})}. \tag{10}$$

Since $\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x})$ are the fixed components, then minimizing the loss $\mathcal{V}_{\phi}(\mathbf{x})$ can be achieved by selecting a new component $t_K^{(c)}$ that has the maximum inner product with the negative of the gradient (Mason et al., 1999). As a result, we can choose a $t_K^{(c)}(\mathbf{z} \mid \mathbf{x})$ based on:

$$t_K^{(c)}(\mathbf{z} \mid \mathbf{x}) = \arg\min_{t_K \in T_K} \mathop{\mathbb{E}}_{t_K(\mathbf{z}|\mathbf{x})} \left[-\log \frac{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K^{(c)})}{\mathcal{T}_K^{(c-1)}(\mathbf{z}_K^{(c)} \mid \mathbf{x})} \right] \tag{11}$$

where $\mathbf{z}_K^c$ denotes a sample transformed by component c's flow (see more details in Appendix 2.2).

Components weights updating. Once the $t_K^{(c)}(\mathbf{z}_K \mid \mathbf{x})$ is estimated, the gradient boost flow model needs to determine the corresponding $\beta_c \in [0, 1]$. However, jointly optimizing both $t_K^{(c)}$ and the weights β_c is a challenging optimization problem, we thus consider a two-step optimization strategy which means that we first train $t_K^{(c)}$ until convergence and then optimize the corresponding weight β_c by using the objective in Eq. 23. Similarly, the weights on each component can be updated by using the gradient of the loss $\mathcal{V}_{\phi}(\mathbf{x})$ with respect to β_c, as shown in Algorithm 1.

$$\frac{\partial \mathcal{V}_{\phi}(\mathbf{x})}{\partial \beta_c} = \sum_{i=1}^{n} \left(\mathop{\mathbb{E}}_{t_K^{(c)}(\mathbf{z}|\mathbf{x}_i)} \left[\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i) \right] - \mathop{\mathbb{E}}_{\mathcal{T}_K^{(c-1)}(\mathbf{z}|\mathbf{x}_i)} \left[\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i) \right] \right) \tag{12}$$

where $\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i)$ is defined as $\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i) = \log \mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x}) - \log p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)$.

Algorithm 1 Updating component weights β_c by SGD

1: Requirement and initialization: predefined C, step size λ, tolerance ϵ. Set $s = 0$, $\beta_c^{(0)} = 1/C$
2: **While** $|\beta_c^{(s)} - \beta_c^{(s-1)}| < \epsilon$ **do**
3: Generate samples $\mathbf{z}_{K,i}^{(c-1)} \sim \mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x}_i)$ and $\mathbf{z}_{K,i}^{(c)} \sim t_K^{(c)}(\mathbf{z} \mid x_i)$ for $i = 1, ..., n$
4: Estimate gradients using MC method $\nabla_{\beta_c} \mathcal{V}_{\psi}(\mathbf{x}) = 1/n \sum_{i=1}^{n} \left[\xi_{\beta_c}^{(s-1)}(\mathbf{z}_{K,i}^{(c)} \mid \mathbf{x}_i) - \xi_{\beta_c}^{(s-1)}(\mathbf{z}_{K,i}^{(c-1)} \mid \mathbf{x}_i) \right]$
5: Update weights $\beta_c^{(s)} = \beta_c^{(s-1)} - \lambda \nabla_{\beta_c}$ and clip weights $\beta_c^{(s)}$ to $[0, 1]$
6: $s = s + 1$
7: **return** $\beta_c^{(s)}$

4.3 Variance-reduced Latent Sampling

Rather than using simple random sampling (SRS) with a larger variance, Latin Hypercube Sampling (LHS) is an ideal candidate with variance reduction, which is generalized in terms of a spectrum of stratified sampling, referred to as partially stratified sampling (PSS), which shows to reduce variance associated with variable interaction but LHS reduces variance associated with additive (main) effects. A hybrid combination of LHS and PSS, named Latinized partially stratified sampling (LPSS) proposed by Shields & Zhang (2016) can reduce variance associated with variable interaction and additive effects simultaneously. However, classical LHS used in LPSS suffers from a lack of exploratory capability due to its random pair scheme. To better explore all possible inverse solutions, we propose to leverage a maximin criteria - maximizes the minimum distance between all pairs of

points, $X_n = \arg\max \min \{d(\mathbf{x}_i, \mathbf{x}_k) : i \neq k = 1, ..., M\}$ where $d(\mathbf{x}, \mathbf{x}') = \sum_{k=1}^{M} (\mathbf{x}_i - \mathbf{x}'_k)^2$ is the Euclidean distance. Instead of random pair, these criteria will greatly improve the space-filling properties of LPSS, specifically in high-dimensional space. Note that, unlike quasi-Monte Carlo (QMC) methods (Caflisch et al., 1998), e.g., Sobol sequence shown in Fig. 6 as well, which are limited in high dimensional problem (Kucherenko et al., 2015), but LPSS (not a QMC) works well with space-filling property and variance reduction in high dimensions.

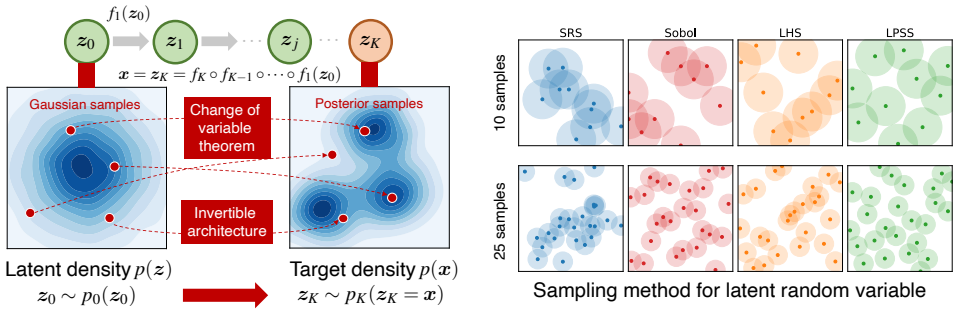

Figure 2: (Left) Illustration of mechanism for normalizing flows. The prior Gaussian samples drawn from the latent distribution are transformed to the posterior samples that match the target density by using a sequence of invertible mappings. (Right) Illustration of prior samples using four sampling algorithms for latent variables.

Algorithm 2 shows the workflow of our proposed variational framework with robust generative flows and variance-reduced latent sampling given a single measurement data.

Algorithm 2 Variational robust generative flows with latent sampling

1: **Requirements**: RGF model $\mathcal{T}_\phi$ parameterized by ϕ, number of GB component C, FD regularization coefficient λ, forward operator $\mathcal{F}$, measurement data $\mathbf{y}$, training batch size b_z, evaluation batch size e_z,
2: Draw random samples from the latent space $\mathbf{z}_j \sim p(\mathbf{z})$ using LPSS, where the sample size is $b_z \times i_z$
3: Generate image samples by $\mathbf{x}_j = \mathcal{T}_\phi(\mathbf{z}_j)$ where $\mathcal{T}_\phi(\mathbf{z}_j)$ is defined by initial gradient boosting (GB) mixture model in Eq. 18 with bi-directional FD regularized loss in Eq. 7.
4: Predict the measurement $\mathbf{y}$ by evaluating the forward operator $\mathbf{y}_j = \mathcal{F}(\mathbf{x}_j) = \mathcal{F}(\mathcal{T}_\phi(\mathbf{z}_j))$
5: Evaluate the total loss $\mathcal{L}_{\text{total}}$ defined in Eq. 5 by a summation of the data fidelity, prior and entropy loss.
6: Update new gradient boosting components based on Eq. 23 with a two-stage optimization strategy
7: Update boosting component weights according to Eq. 12 with stochastic gradient descent in Algorithm 1.
8: After training is done, generate posterior samples, and estimate the statistical quantities.

4.4 RGF Representational Capability

Next, we demonstrate that our proposed RGF can improve the representational capability of deterministic NFs at a given network size. Here we use images to define complicated two-dimensional distributions as the target distributions to be sampled. This benchmark inspired by Wu et al. (2020) aims at generating high-quality images from the exact density to compare the performance of different generative models with four blocks (more details in Appendix). Fig.1 shows the sampling of DNA, Fox and NeurIPS logo cases with different methods. As expected, RealNVP (Dinh et al., 2016) has limitations in representational capability, which results in a blurred image without detailed structures. NSF (Durkan et al., 2019) outperforms RNVP with a better structure on 2D images, but still fails to resolve details, specifically in the Fox and NeurIPS logo cases, at the selected neural architecture. Note that all the flow-based models tend to have a better representational capability as the depth increases but we fix their depth in four blocks to perform a fair comparison.

Our proposed RGF (with SRS) achieves high-quality approximation through a combination of gradient boosting and stable regularization with a shallow architecture. Although the samples in RGF are close to the ground truth, their performance can be further improved by using LPSS. Although visual differences may be slight between RGF and RGFL, we differ them with a quantitative comparison using evaluation metrics (see Table 1). RGFL shows superior performance on both fidelity and diversity aspects of the generated images. In other words, the RGF model boosted by LPSS achieves not only a more accurate approximation to the true samples, but also better captures the uncertainty (variation) in real samples. Both advantages are useful to quantify the reconstruction uncertainty.

5 EXPERIMENTS

Tasks and datasets. We evaluate our method on two image reconstruction tasks: (1) compressed sensing MRI, from fastMRI dataset (Zbontar et al., 2018) with images of size 320×320; and (2) Interferometric imaging, using blackhole images from the Event Horizon Telescope (EHT) (Akiyama et al., 2019b;c), with images of size 160×160. For both cases, we only use **one data**, which contains **one** specific measurement and the corresponding ground truth. The original images are resized to 128×128 for fastMRI and 32×32 for blackhole case.

Implementation. We use 6 flow steps and 4 residual blocks for each step so that we have a total of 24 transformations in our RGF model. The whole model is trained on a single V100 GPU by using 20000 epochs with a batch size of 32 and an initial learning rate of 1E-4 in Adam.

Evaluation metrics. After training, 1000 samples drawn from the learned posterior distribution are used to evaluate the *mean*, *standard deviation* and *absolute error*, which represents the bias between the mean image and ground truth. To evaluate the performance of the generative model, we provide a quantitative evaluation of the sample fidelity using *precision* and *density*, and the sample diversity using *recall* and *coverage*.

Baselines. We compare our proposed methods with a couple of methods including: (1) conditional variational autoencoder (cVAE) (Sohn et al., 2015), which is typically used as a baseline method; (2) Deep Probabilistic Imaging (DPI), proposed by (Sun & Bouman, 2020), mainly uses RealNVP model for image Reconstruction; (3) GlowIP, which is to use invertible generative models for inverse problems and image reconstruction (Asim et al., 2020); (4) NSF, which is neural spline flows (Durkan et al., 2019) used in our deep variational framework. RGFL is our proposed method which uses robust generative flows (RGF) with Latinized partially stratified sampling (LPSS). Although some recent methods (Wang et al., 2020; Whang et al., 2021b;a), look very promising, we can not compare them if there are no available open-source codes.

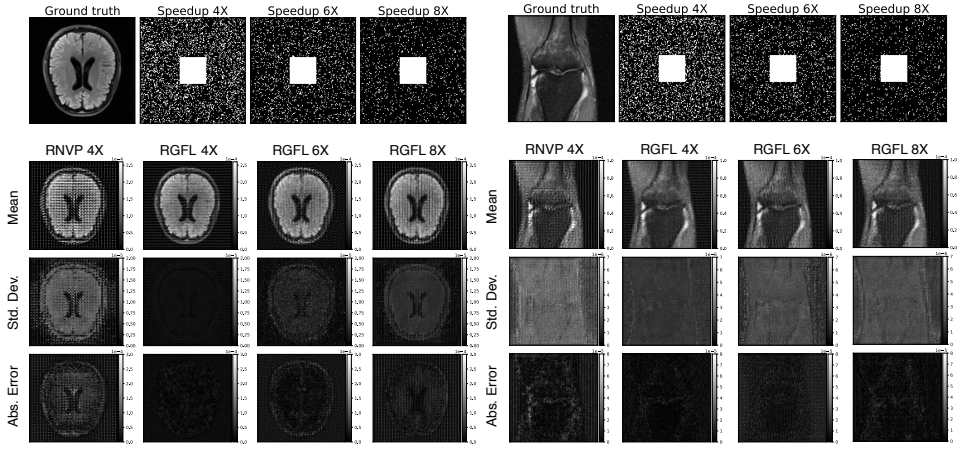

Figure 3: FastMRI reconstruction of brain case (left) and knee case (right) at three different acceleration speedup factors: 4X, 6X and 8X (each shown in a column). Row 1 shows the ground truth and sampling masks for each case. Row 2-4 shows the mean, standard deviation, absolution of error for the estimated posterior samples.

Table 2: FastMRI brain case results

Metrics	cVAE	DPI	NSF	GrowIP	RGFL
Brain case with 4× speedup					
Std. Dev. ↓	8.87E-5	1.21E-5	8.82E-6	6.90E-6	**9.73E-7**
Abs. Error ↓	8.72E-5	3.31E-5	6.91E-6	1.83E-6	**1.78E-6**
Brain case with 6× speedup					
Std. Dev. ↓	1.02E-4	1.97E-5	1.02E-5	7.33E-6	**2.35E-6**
Abs. Error ↓	1.39E-4	5.70E-5	8.04E-6	6.99E-6	**4.58E-6**
Brain case with 8× speedup					
Std. Dev. ↓	2.58E-4	3.83E-5	2.27E-5	9.14E-6	**4.10E-6**
Abs. Error ↓	4.65E-4	8.81E-5	1.33E-5	8.04E-6	**7.12E-6**

Table 3: FastMRI knee case results

Metrics	cVAE	DPI	NSF	GrowIP	RGFL
Knee case with 4× speedup					
Std. Dev. ↓	7.98E-6	5.94E-6	4.33E-6	3.84E-6	**1.24E-6**
Abs. Error ↓	7.99E-6	4.83E-6	2.55E-6	9.36E-7	**7.03E-7**
Knee case with 6× speedup					
Std. Dev. ↓	9.53E-6	9.33E-6	6.48E-6	4.71E-6	**2.05E-6**
Abs. Error ↓	1.02E-5	6.02E-6	4.87E-6	1.96E-6	**9.11E-7**
Knee case with 8× speedup					
Std. Dev. ↓	3.61E-5	1.60E-5	9.96E-6	5.18E-6	**3.71E-6**
Abs. Error ↓	3.45E-5	7.94E-6	6.07E-6	2.06E-6	**1.15E-6**

5.1 FastMRI Case Study

Partial and under-sampled noisy measurements in MRI will lead to reconstruction uncertainty. We show that our RGFL method can be successfully applied to quantify the reconstruction variation and bias on two cases (brain and knee) with acceleration factors $4\times$, $6\times$ and $8\times$. Fig. 3 shows the reconstruction results with pixel-wise statistics of the estimated posterior distribution. For both cases with a speedup $4\times$ factor, our RGFL shows a more accurate mean estimate with a smaller absolute error than the DPI baseline with larger architecture. Our advantage in terms of standard deviation is more significant due to higher model robustness and variance reduction. Although the pixel-wise variance of the reconstruction tends to be larger as the speedup factor increases (more details in Appendix), our method provides a more reliable image reconstruction compared with other baselines.

We further use the mean of the standard deviation and the absolute error to quantitatively compare the pixel-wise statistics (see Table 2 and 3). Our method outperforms the other baselines in terms of accuracy (bias) and variation of the reconstruction. Specifically, our estimation achieves significant variance reduction with 1-2 orders of magnitude. Regrading the sample fidelity and diversity metrics, our method also shows competitive performance in most cases.

5.2 Interferometric Imaging Case Study

Our approach can be also applied to study radio interferometric astronomical imaging which was used to take the first black hole images. Sparse spatial frequency measurements are used to recover the underlying astronomical image (see Fig. 4). Relatively large telescope-based gain and phase error in the measurement noise lead to a non-convex image reconstruction problem where a challenge is the potential for multi-modal posterior distribution — different solutions fit the same measurement data visually and reasonably well. In this case, a synthetic black hole is used to illustrate our capability on reconstructed uncertainty estimation and multiple modes detection (Sun & Bouman, 2020).

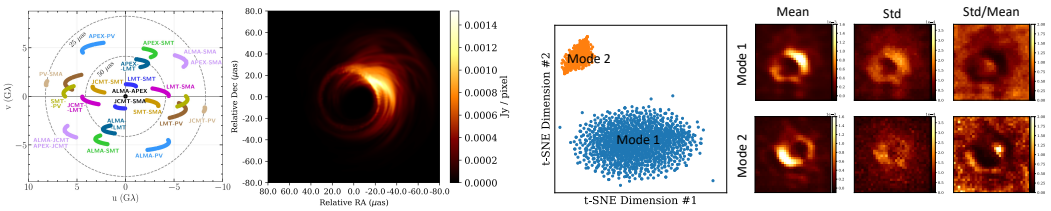

Figure 4: (Left) Measured frequency samples for EHT observing the M87* (Akiyama et al., 2019b;c) and target synthetic black hole image; (Right) t-SNE result of posterior samples and statistics of the posterior samples on each model (Sun & Bouman, 2020).

Fig. 4 shows multiple samples drawn from the learned generative model. Note that two different modes of reconstruction are captured by our RGFL model. t-SNE plots present a clear clustering of samples into two modes. Mode 1 fits the target image better than mode 2 which exhibits roughly 180-degree rotations. We perform statistical analysis for each mode to quantify the mean and standard deviation of the multi-modal posterior. Note that the uncertainty characterized by standard deviation shows a similar shape to the mean estimator. Mode 1, which is close to the correct solution, results in a smaller bias estimator than mode 2 (see Table 4). In this case, the space-filling sampling would be more important to capture the multi-modal posterior distribution. This can also be observed by the recall and coverage metrics in Table 4. Our method outperforms the other baselines with more accurate and reliable results.

Table 4: Statistical comparison of the estimated posteriors on the black hole image reconstruction (mode 1)

Metrics	Reconstructed black hole images (mode 1)				
	cVAE	DPI	NSF	GrowIP	**RGFL**
Std. Dev. ↓	1.23E-3	6.09E-4	5.43E-4	2.17E-4	**1.20E-4**
Abs. Error ↓	9.76E-4	3.85E-4	2.65E-4	**1.06E-4**	1.08E-4

6 Conclusion

We propose an uncertainty-aware framework that leverages a deep variational approach with robust generative flows and variance-reduced sampling to perform an accurate estimation of image reconstruction uncertainty. The results on multiple benchmarks and real-world tasks demonstrate our advantages in uncertainty estimation. Although the current RGFL methods show superior performance on these imaging reconstruction tasks, the total computational cost will be a major concern if we plan to scale to complex high-resolution images, e.g., 3D biomedical images.

REFERENCES

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, et al. First m87 event horizon telescope results. ii. array and instrumentation. *The Astrophysical Journal Letters*, 875(1):L2, 2019a.

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, et al. First m87 event horizon telescope results. ii. array and instrumentation. *The Astrophysical Journal Letters*, 875(1):L2, 2019b.

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, et al. First m87 event horizon telescope results. iii. data processing and calibration. *The Astrophysical Journal Letters*, 875(1): L3, 2019c.

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, et al. First m87 event horizon telescope results. iv. imaging the central supermassive black hole. *The Astrophysical Journal Letters*, 875(1):L4, 2019d.

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, et al. First m87 event horizon telescope results. v. physical origin of the asymmetric ring. *The Astrophysical Journal Letters*, 875 (1):L5, 2019e.

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, et al. First m87 event horizon telescope results. v. physical origin of the asymmetric ring. *The Astrophysical Journal Letters*, 875 (1):L5, 2019f.

Kazunori Akiyama, Juan Carlos Algaba, Antxon Alberdi, Walter Alef, Richard Anantua, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, et al. First m87 event horizon telescope results. viii. magnetic field structure near the event horizon. *The Astrophysical Journal Letters*, 910(1):L13, 2021.

Muhammad Asim, Max Daniels, Oscar Leong, Ali Ahmed, and Paul Hand. Invertible generative models for inverse problems: mitigating representation error and dataset bias. In *International Conference on Machine Learning*, pp. 399–409. PMLR, 2020.

Riccardo Barbano, Željko Kereta, Chen Zhang, Andreas Hauptmann, Simon Arridge, and Bangti Jin. Quantifying sources of uncertainty in deep learning-based image reconstruction. *arXiv preprint arXiv:2011.08413*, 2020.

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. PMLR, 2019.

Jens Behrmann, Paul Vicol, Kuan-Chieh Wang, Roger Grosse, and Jörn-Henrik Jacobsen. Understanding and mitigating exploding inverses in invertible neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 1792–1800. PMLR, 2021.

Roman Beliy, Guy Gaziv, Assaf Hoogi, Francesca Strappini, Tal Golan, and Michal Irani. From voxels to pixels and back: Self-supervision in natural-image reconstruction from fmri. *arXiv preprint arXiv:1907.02431*, 2019.

Chinmay Belthangady and Loic A Royer. Applications, promises, and pitfalls of deep learning for fluorescence image reconstruction. *Nature methods*, 16(12):1215–1225, 2019.

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

Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. In *International Conference on Machine Learning*, pp. 537–546. PMLR, 2017.

Charles Bouman and Ken Sauer. A generalized gaussian image model for edge-preserving map estimation. *IEEE Transactions on image processing*, 2(3):296–310, 1993.

Russel E Caflisch et al. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 1998:1–49, 1998.

Giannis Daras, Joseph Dean, Ajil Jalal, and Alexandros G Dimakis. Intermediate layer optimization for inverse problems using deep generative models. *arXiv preprint arXiv:2102.07364*, 2021.

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

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *arXiv preprint arXiv:1906.04032*, 2019.

T Event Horizon Telescope Collaboration, K Akiyama, A Alberdi, et al. First m87 event horizon telescope results. i. the shadow of the supermassive black hole. *The Astrophysical Journal Letters*, 875(L1):17pp, 2019.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2): 337–407, 2000.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

Yarin Gal. Uncertainty in deep learning.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Charles F Gammie, Event Horizon Telescope Collaboration, et al. First m87 event horizon telescope results. iii. data processing and calibration. *Astrophysical Journal Letters*, 875(1):L3, 2019.

Robert Giaquinto and Arindam Banerjee. Gradient boosted normalizing flows. *Advances in Neural Information Processing Systems*, 33, 2020.

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.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pp. 1861–1869. PMLR, 2015.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6629–6640, 2017.

Judy Hoffman, Daniel A Roberts, and Sho Yaida. Robust learning with jacobian regularization. *arXiv preprint arXiv:1908.02729*, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NIPS*, 2017.

Mohammad Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *International Conference on Machine Learning*, pp. 2611–2620. PMLR, 2018.

Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic gradient descent. In *ICLR: International Conference on Learning Representations*, pp. 1–15, 2015.

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

Ivan Kobyzev, Simon Prince, and Marcus Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Zhifeng Kong and Kamalika Chaudhuri. The expressive power of a class of normalizing flow models. In *International Conference on Artificial Intelligence and Statistics*, pp. 3599–3609. PMLR, 2020.

Konik Kothari, AmirEhsan Khorashadizadeh, V Maarten, and Ivan Dokmanic. Trumpets: Injective flows for inference and inverse problems. 2021.

Sergei Kucherenko, Daniel Albrecht, and Andrea Saltelli. Exploring multi-dimensional spaces: A comparison of latin hypercube and quasi monte carlo sampling techniques. *arXiv preprint arXiv:1505.02350*, 2015.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, 2017.

Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. Boosting algorithms as gradient descent. *Advances in neural information processing systems*, 12, 1999.

Michael D McKay, Richard J Beckman, and William J Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.

Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *International Conference on Machine Learning*, pp. 7176–7185. PMLR, 2020.

Frank Natterer and Frank Wübbeling. *Mathematical methods in image reconstruction*. SIAM, 2001.

Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. Survae flows: Surjections to bridge the gap between vaes and flows. *Advances in Neural Information Processing Systems*, 33, 2020.

Art B Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, pp. 439–452, 1992.

Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. Super-resolution image reconstruction: a technical overview. *IEEE signal processing magazine*, 20(3):21–36, 2003.

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

Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 5234–5243, 2018.

Michael D Shields and Jiaxin Zhang. The generalization of latin hypercube sampling. *Reliability Engineering & System Safety*, 148:96–108, 2016.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

Michael Stein. Large sample properties of simulations using latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.

David Strong and Tony Chan. Edge-preserving and scale-dependent properties of total variation regularization. *Inverse problems*, 19(6):S165, 2003.

He Sun and Katherine L Bouman. Deep probabilistic imaging: Uncertainty quantification and multi-modal solution characterization for computational imaging. *arXiv preprint arXiv:2010.14462*, 2020.

Boxin Tang. Orthogonal array-based latin hypercubes. *Journal of the American statistical association*, 88(424):1392–1397, 1993.

A Richard Thompson, James M Moran, and George W Swenson. *Interferometry and synthesis in radio astronomy*. Springer Nature, 2017.

Francesco Tonolini, Jack Radford, Alex Turpin, Daniele Faccio, and Roderick Murray-Smith. Variational inference for computational imaging inverse problems. *Journal of Machine Learning Research*, 21(179):1–46, 2020.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9446–9454, 2018.

Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, pp. 9690–9700. PMLR, 2020.

Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. Improving deterministic uncertainty estimation in deep learning for classification and regression. *arXiv preprint arXiv:2102.11409*, 2021.

Ge Wang, Jong Chul Ye, and Bruno De Man. Deep learning for tomographic image reconstruction. *Nature Machine Intelligence*, 2(12):737–748, 2020.

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8798–8807, 2018.

Jay Whang, Qi Lei, and Alex Dimakis. Solving inverse problems with a flow-based noise model. In *International Conference on Machine Learning*, pp. 11146–11157. PMLR, 2021a.

Jay Whang, Erik Lindgren, and Alex Dimakis. Composing normalizing flows for inverse problems. In *International Conference on Machine Learning*, pp. 11158–11169. PMLR, 2021b.

Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*, 2020.

Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows. *arXiv preprint arXiv:2002.06707*, 2020.

Jure Zbontar, Florian Knoll, Anuroop Sriram, Tullie Murrell, Zhengnan Huang, Matthew J. Muckley, Aaron Defazio, Ruben Stern, Patricia Johnson, Mary Bruno, Marc Parente, Krzysztof J. Geras, Joe Katsnelson, Hersh Chandarana, Zizhao Zhang, Michal Drozdzal, Adriana Romero, Michael Rabbat, Pascal Vincent, Nafissa Yakubova, James Pinkerton, Duo Wang, Erich Owens, C. Lawrence Zitnick, Michael P. Recht, Daniel K. Sodickson, and Yvonne W. Lui. fastMRI: An open dataset and benchmarks for accelerated MRI. 2018.

Zizhao Zhang, Adriana Romero, Matthew J Muckley, Pascal Vincent, Lin Yang, and Michal Drozdzal. Reducing uncertainty in undersampled mri reconstruction with active acquisition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2049–2058, 2019.

Qingping Zhou, Tengchao Yu, Xiaoqun Zhang, and Jinglai Li. Bayesian inference and uncertainty quantification for medical image reconstruction with poisson data. *SIAM Journal on Imaging Sciences*, 13(1):29–52, 2020.

Bo Zhu, Jeremiah Z Liu, Stephen F Cauley, Bruce R Rosen, and Matthew S Rosen. Image reconstruction by domain-transform manifold learning. *Nature*, 555(7697):487–492, 2018.

A VARIATIONAL FRAMEWORK

We provide an additional derivation with more details about the variational approach in Section 3.

The target reconstructed image $\mathbf{x}$ is parametrized by a generative flow-based model $\mathbf{x} = \mathcal{T}_{\boldsymbol{\phi}}(\mathbf{z}), \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ with model parameter $\boldsymbol{\phi}$. The true image posterior distribution can be derived according to the Bayes' rule:

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{\int p(\mathbf{y}|\mathbf{x})p(\mathbf{x})} \propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x}) \tag{13}$$

Our goal is to pursue an approximate posterior distribution $\tau_{\boldsymbol{\phi}^*}(\mathbf{x})$ which is achieved by minimizing the KL-divergence between the parametrized flow-based distribution $\tau_{\boldsymbol{\phi}}(\mathbf{x})$ and the target posterior distribution $p(\mathbf{x}|\mathbf{y})$

$$\begin{aligned}
\boldsymbol{\phi}^* &= \arg\min_{\boldsymbol{\phi}} \mathrm{KL}(\tau_{\boldsymbol{\phi}}(\mathbf{x}) \parallel p(\mathbf{x}|\mathbf{y})) \\
&= \arg\min_{\boldsymbol{\phi}} \int \tau_{\boldsymbol{\phi}}(\mathbf{x})[\log \tau_{\boldsymbol{\phi}}(\mathbf{x}) - \log p(\mathbf{x}|\mathbf{y})]d\mathbf{x} \\
&= \arg\min_{\boldsymbol{\phi}} \int \tau_{\boldsymbol{\phi}}(\mathbf{x})[\log \tau_{\boldsymbol{\phi}}(\mathbf{x}) - \log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{x})]d\mathbf{x} \\
&= \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{\mathbf{x}\sim\tau_{\boldsymbol{\phi}}(\mathbf{x})}[-\log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{x}) + \log \tau_{\boldsymbol{\phi}}(\mathbf{x})].
\end{aligned} \tag{14}$$

Note that this loss function can be viewed as an expectation over the maximum a posterior loss

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \{\log p(\mathbf{x}|\mathbf{y})\} = \arg\max_{\mathbf{x}} \big\{ \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}) \big\}. \tag{15}$$

with an additional term $\log \tau_{\boldsymbol{\phi}}(\mathbf{x})$. Minimizing the negative entropy term, $\mathcal{H}_\phi = \mathbb{E}_{\mathbf{x}\sim\tau_{\boldsymbol{\phi}}(\mathbf{x})}[\log \tau_{\boldsymbol{\phi}}(\mathbf{x})]$ is useful to enhance the reconstructed image distribution and avoid the generative flow-based model from collapsing to a point solution.

B ROBUST GENERATIVE FLOWS

B.1 STABILIZE THE INVERTIBLE NEURAL ARCHITECTURE

We propose to add the bi-directional FD regularization to stabilize the invertible architecture. Let's use $\mathcal{L}$ and ω_b denote the loss and regularization term respectively. The training is to minimize

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \{\mathcal{L} + \lambda_b \omega_b\} \tag{16}$$

where λ_b is the regularization coefficient. Larger coefficients yield more stable models with smaller condition numbers but a too large coefficient will harm the performance (Behrmann et al., 2021). Here we use λ_b=1e-4 for all experiments. For the bi-directional FD regularizer, we generate samples $u \sim \mathcal{N}(0, I)$ and use $\varepsilon = 0.1$ as the step size to avoid numerical errors. The core python implementation is shown in Fig. 5.

Computational cost In principle, bi-directional FD regularization is expensive. This is because the gradient computation consists of three passes: a forward pass, a backward pass, and an inverse pass. For the forward pass, the regularizer adds overhead to the computations since it needs the previous computations for noise $\mathbf{x} + \varepsilon u$. For the inverse pass, we also pass two variables $\mathbf{y} = \mathcal{F}(\mathbf{x})$ and $\hat{\mathbf{y}} = \mathcal{F}(\mathbf{x}) + \varepsilon u$ through its computations, including a forward pass to re-compute activations of the inverse mapping, an inverse pass to compute the reconstruction, and the backward pass through the inverse and forward. In practice, however, the cost can be significantly reduced by using regularization only once per every T iterations of training, and we also found $T = 10$ works similarly to $T = 1$ in terms of their stabilizing effect.

B.2 GRADIENT BOOSTING FLOW MODEL

B.2.1 GRADIENT BOOSTING

Gradient boosting (Friedman et al., 2000; Friedman, 2001) aims at minimizing a loss $F(\mathcal{T})$ where $\mathcal{T}$ is a function representing the current model. Assume an additive perturbation around $\mathcal{T}$ to $\mathcal{T} + \varepsilon t$,

```
for _, (img, label) in enumerate(train_loader):
    # fd_coeff = 1e-4, fd_inv_coeff = 1e-4
    model.train()
    projection.train()
    img, label = img.to(use_device), label.to(use_device)
    z, objective = run_flow(img)
    zs = z.view(z.size(0), -1)[:, :args.zs_dim]
    logits = projection(zs)
    classification_loss = F.cross_entropy(logits, label)
    loss += classification_loss
    # ----------------------------------------
    # For forward finite-differences regularization
    # ----------------------------------------
    if (fd_coeff > 0) and (global_iteration % args.regularize_every == 0):
        d = torch.randn(img.shape, device=img.device)
        d = (d.view(d.shape[0], -1) / torch.norm(d.view(d.shape[0], -1),
        ↪  dim=0)).view(d.shape)  # normalize to unit vector
        epsilon = 0.1
        z_noise, _ = run_flow(img + epsilon * d)
        J_forward_penalty = fd_coeff * torch.norm(z - z_noise)**2 / epsilon
        loss += J_forward_penalty
    else:
        J_forward_penalty = 0.0
    # ----------------------------------------
    # For inverse finite-differences regularization
    # ----------------------------------------
    if (fd_inv_coeff > 0) and (global_iteration % args.regularize_every == 0):
        d_inv = torch.randn(z.shape, device=z.device)
        d_inv = (d_inv.view(d_inv.shape[0], -1) /
        ↪  torch.norm(d_inv.view(d_inv.shape[0], -1), dim=0)).view(d_inv.shape)
        epsilon = 0.1
        z_noise_inv = z + epsilon * d_inv
        x_inv_noise = model.module.sample(z_noise_inv, no_grad=True)
        x_inv_noise.requires_grad = True
        J_inverse_penalty = fd_inv_coeff * torch.norm(img - x_inv_noise)**2 /
        ↪  epsilon
        loss += J_inverse_penalty
```

Figure 5: Python implementation of bi-directional FD regularizer

where t is a function representing a new component, a Taylor expansion can be written as $\varepsilon \to 0$:

$$F(\mathcal{T} + \varepsilon t) = F(\mathcal{T}) + \varepsilon \langle t, \nabla F(\mathcal{T}) \rangle + o(\varepsilon^2) \tag{17}$$

which reveals that the functional gradient $\nabla F(\mathcal{T})$ is the direction that reduces the loss at the current solution. Therefore, the goal of gradient boosting is to choose the best function t in a class of functions T (e.g., gradient boosting tree models) so that the loss $F(\mathcal{T})$ at the current model is minimized, which corresponds to solving a linear program where $\nabla F(\mathcal{T})$ defines the weights for every function in T.

B.2.2 Variational inference with gradient boosting

Gradient boosting is good at approximating posterior with generative flow-based models. A gradient boosting flow model is constructed by successively adding new components, where each new component $t_K^{(c)}$ is a K-step normalizing flow which matches the functional gradient of the loss function from the $(c-1)$ previously trained flow components $\mathcal{T}_k^{(c-1)}$.

Typically, the gradient boost flow model is constructed by a convex combination of fixed and new flow components:

$$\mathcal{T}_K^{(c)}(\mathbf{z} \mid \mathbf{x}) = (1 - \beta_c)\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x}) + \beta_c t_K^{(c)}(\mathbf{z} \mid \mathbf{x}) \tag{18}$$

where $\mathbf{x}$ are the observed data, $\mathbf{z}$ are the latent variables and β_c is the weight to the new component where $\beta_c \in [0, 1]$ to sure the mixture model in Eq. 18 is a valid probability distribution.

The benefits of gradient boost flow model over a standard mixture model with a fixed number of components is that the additional components can be sequentially added to the model with optimal weights β_c (β_c degrade to zero for non-informative components).

Our variational framework in Section A shows that our target is to approximate the posterior by minimizing the KL-divergence:

$$\mathcal{V}_\phi(\mathbf{x}) = \mathbb{E}_{\tau_{\boldsymbol{\phi}}(\mathbf{x})}[-\log p(\mathbf{y} \mid \mathbf{x}) - \log p(\mathbf{x}) + \log \tau_{\boldsymbol{\phi}}(\mathbf{x})]. \tag{19}$$

If the prior $\log p(\mathbf{x})$ is noninformative, Eq. 19 can be rewritten with defined gradient boost flow model

$$\mathcal{V}_\phi(\mathbf{x}) = \mathbb{E}_{\mathcal{T}_K^{(c)}}[\log \mathcal{T}_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}) - \log p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)]. \tag{20}$$

In our formulation of gradient boost flow model, we consider C components where C is infinite and leave C fixed by a predefined number.

B.2.3 Boosting components updating

Given the objective function in Eq. 20, we proceed with deriving updates for new boosting components. First, at the current stage c, we let $\mathcal{T}_K^{(c-1)}$ to be fixed, and the target is learning the component $t_K^{(c)}$ and the weight β_c based on functional gradient descent (FGD) (Mason et al., 1999). We take the gradient of objective in Eq. 20 with respect to $\mathcal{T}_K^{(c)}$ at $\beta_c \to 0$:

$$\nabla_{\mathcal{T}_K^{(c)}} \mathcal{V}_\phi(\mathbf{x})\big|_{\beta_c \to 0} = -\log \frac{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)}{\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x})}. \tag{21}$$

Since $\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x})$ are the fixed components, then minimizing the loss $\mathcal{V}_\phi(\mathbf{x})$ can be achieved by selecting a new component $t_K^{(c)}$ that has the maximum inner product with the negative of the gradient (Mason et al., 1999). As a result, we can choose a $t_K^{(c)}(\mathbf{z} \mid \mathbf{x})$ based on:

$$\begin{aligned} t_K^{(c)}(\mathbf{z} \mid \mathbf{x}) &= \arg\max_{t_K \in T_K} \sum_{i=1}^{n} \langle t_K(\mathbf{z} \mid \mathbf{x}_i), -\nabla_{\mathcal{T}} \mathcal{V}(\mathbf{x}_i) \rangle \\ &= \arg\min_{t_K \in T_K} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{t_K(\mathbf{z} \mid \mathbf{x}_i)} [\nabla_{\mathcal{T}} \mathcal{V}(\mathbf{x}_i)] \end{aligned} \tag{22}$$

Eq. 22 can be rearranged to show that new component $t_K^{(c)}$ minimizes:

$$t_K^{(c)}(\mathbf{z} \mid \mathbf{x}) = \arg\min_{t_K \in T_K} \mathop{\mathbb{E}}_{t_K(\mathbf{z} \mid \mathbf{x})} \left[-\log \frac{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K^{(c)})}{\mathcal{T}_K^{(c-1)}(\mathbf{z}_K^{(c)} \mid \mathbf{x})} \right] \tag{23}$$

where $\mathbf{z}_K^c$ denotes a sample transformed by component c's flow. Note that during a forward pass the model gradient boost flow model encodes data to produce $\mathbf{z}_0$. To sample from the posterior $\mathbf{z}_K \sim \mathcal{T}_K^{(c)}$, however, we transform $\mathbf{z}_0$ based on $\mathbf{z}_K$ by applying a chain of K transformations $\mathbf{z}_K = \mathcal{T}_K^{(j)} \circ \cdots \circ \mathcal{T}_1^{(j)}(\mathbf{z}_0)$, where $j \sim \text{Categorical}(\beta)$ randomly choose a component, which is similar to sampling from a mixture model. In other words, we compute a fast stochastic approximation of the likelihood $\mathcal{T}_K^{(c)}$ and inference and sampling are as fast as the non-boosted setting.

B.2.4 Two-step optimization strategy

Optimizing Eq. 23 for mixture model $\mathcal{T}_K^{(c)}$ is a nontrivial task. However, gradient boosting provides a feasible solution to simplify the optimization problem. During training, the first component is derived using a traditional objective function where no boosting is applied. In other words, no boosting during the first stage is equivalent to setting $\mathcal{T}_K^{(0)}(\mathbf{x})$ to uniform on the domain of $\mathbf{x}$. At stages $c > 1$, $\mathcal{T}_K^{(c-1)}$ consists of a convex combination of the $(c-1)$ K step flow models from the previous stages and then we train a new component $t_K^{(c)}$. Jointly optimizing both $t_K^{(c)}$ and the weights β_c is a challenging optimization problem, we thus consider a two-step optimization strategy, which means that we first train $t_K^{(c)}$ until convergence and then optimize the corresponding weight β_c by using the objective in Eq. 23.

B.2.5 COMPONENTS WEIGHTS UPDATING

Once the $t_K^{(c)}(\mathbf{z}_K \mid \mathbf{x})$ is estimated, the gradient boost flow model needs to determine the corresponding $\beta_c \in [0, 1]$. Similarly, the weights on each component can be updated by using the gradient of the loss $\mathcal{V}_\psi(\mathbf{x})$ with respect to β_c. Let's first rewrite $\mathcal{T}_K^{(c)}(\mathbf{z}_K \mid \mathbf{x})$ as the convex combination:

$$\begin{aligned}\mathcal{T}_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}) &= (1-\beta_c)\mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}) + \beta_c t_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}) \\ &= \beta_c \left(t_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}) - \mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}) \right) + \mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}).\end{aligned} \tag{24}$$

By defining $\delta_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}) \triangleq t_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}_i) - \mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}_i)$, the objective function $\mathcal{V}_\phi(\mathbf{x})$ can be written as a function of β_c:

$$\begin{aligned}\mathcal{V}_\phi(\mathbf{x}) = &\sum_{i=1}^{n} \langle \beta_c \delta_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}_i) + \mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}_i), -\log p(\mathbf{y} \mid \mathbf{x}_i, \mathbf{z}_K) \rangle \\ &+ \sum_{i=1}^{n} \langle \beta_c \delta_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}_i) + \mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}_i), \log \left(\beta_c \delta_K^{(c)}(\mathbf{z}_K \mid \mathbf{x}_i) + \mathcal{T}_K^{(c-1)}(\mathbf{z}_K \mid \mathbf{x}_i) \right) \rangle\end{aligned} \tag{25}$$

Taking the gradient of Eq. 25 with respect to β_c yields the component weight updates:

$$\frac{\partial \mathcal{V}_\phi(\mathbf{x})}{\partial \beta_c} = \sum_{i=1}^{n} \left(\mathop{\mathbb{E}}_{t_K^{(c)}(\mathbf{z}|\mathbf{x}_i)} \left[\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i) \right] - \mathop{\mathbb{E}}_{\mathcal{T}_K^{(c-1)}(\mathbf{z}|\mathbf{x}_i)} \left[\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i) \right] \right) \tag{26}$$

where $\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i)$ is defined as

$$\begin{aligned}\xi_{\beta_c}^{(s-1)}(\mathbf{z} \mid \mathbf{x}_i) &\triangleq -\log \frac{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)}{\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x})} \\ &= \log \frac{\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x})}{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)} \\ &= \log \left(\frac{(1-\beta_c^{(s-1)})\mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x}) + \beta_c^{(s-1)} t_K^{(c)}(\mathbf{z} \mid \mathbf{x})}{p(\mathbf{y} \mid \mathbf{x}, \mathbf{z}_K)} \right)\end{aligned} \tag{27}$$

The above Eq. 27 can be solved by using stochastic gradient descent (SGD). s in Eq. 27 is the iteration step. The SGD algorithm for updating weights β_c is illustrated by Algorithm 3.

Algorithm 3 Updating component weights β_c by stochastic gradient descent

1: Requirement and initialization: predefined C, step size λ, tolerance ϵ. Set $s = 0$, $\beta_c^{(0)} = 1/C$
2: **While** $|\beta_c^{(s)} - \beta_c^{(s-1)}| < \epsilon$ **do**
3: Generate samples $\mathbf{z}_{K,i}^{(c-1)} \sim \mathcal{T}_K^{(c-1)}(\mathbf{z} \mid \mathbf{x}_i)$ and $\mathbf{z}_{K,i}^{(c)} \sim t_K^{(c)}(\mathbf{z} \mid x_i)$ for $i = 1, ..., n$
4: Estimate gradients using MC method $\nabla_{\beta_c} \mathcal{V}_\psi(\mathbf{x}) = 1/n \sum_{i=1}^{n} \left[\xi_{\beta_c}^{(s-1)}(\mathbf{z}_{K,i}^{(c)} \mid \mathbf{x}_i) - \xi_{\beta_c}^{(s-1)}(\mathbf{z}_{K,i}^{(c-1)} \mid \mathbf{x}_i) \right]$
5: Update weights $\beta_c^{(s)} = \beta_c^{(s-1)} - \lambda \nabla_{\beta_c}$ and clip weights $\beta_c^{(s)}$ to $[0, 1]$
6: $s = s + 1$
7: **return** $\beta_c^{(s)}$

B.2.6 COMPUTATIONAL COST

Gradient boosting flow model compliments existing flow-based models, improving performance at the cost of additional training cycles, rather than more complex transformations. The additional training cost is caused by updating the new flow components and the corresponding component weights via stochastic gradient descent. However, sampling and inference are not slowed with gradient boosting, as each component is independent and operates in parallel.

C VARIANCE-REDUCED SAMPLING

In this section, we will provide more details including definition and derivation, for the variance-reduced sampling methods discussed in the main context and offer a comprehensive comparison of their advantages and limitations.

C.1 STRATIFIED SAMPLING

Stratified sampling (SS) divides the sample space S into a collection of M disjoint strata $\mathbf{\Omega}_k, k = 1, ..., M$ with $\cup_{k=1}^{M} \mathbf{\Omega}_k = S$ and $\mathbf{\Omega}_p \cap \mathbf{\Omega}_q = \varnothing, p \neq q$. Samples drawn from a specific stratum k, $\mathbf{x}_k = \{x_{1k}, ..., x_{Nk}\}$, are generated by randomly sampling the vector components based on

$$x_{ik} = D_{X_i}^{-1}(U_{ik}), \quad i = 1, ..., N \tag{28}$$

where N is the total number of samples, U_{ik} are iid uniformly distributed samples on $[\xi_{ik}^l, \xi_{ik}^u]$ with $\xi_{ik}^l = D_{X_i}(\zeta_{ik}^l)$ and $\xi_{ik}^u = D_{X_i}(\zeta_{ik}^u)$ and ζ_{ik}^l and ζ_{ik}^u are the lower and upper bounds of the ith vector component of stratum $\mathbf{\Omega}_k$ respectively. In many cases, the strata are defined by the bounds $[\xi_{ik}^l, \xi_{ik}^l]$.

C.2 VARIANCE REDUCTION IN SRS, SS AND LHS

For simple random sampling (SRS) (or conventional Monte Carlo sampling), the variance of the statistical estimator T_R is given $\text{Var}(T_R) = \sigma^2/N$ where $\sigma^2 = \text{Var}(g)$ where g is an arbitrary function, e.g., expectation operator. SS has been proven to unconditionally reduce the variance of statistical estimator (T_S) when compared to SRS so that the variance reduction depends on the difference between the strata means μ_k and the overall mean τ (McKay et al., 2000):

$$\text{Var}(T_S) = \text{Var}(T_R) - \frac{1}{n}\sum_{k=1}^{M} p_k(\mu_k - \tau)^2 \tag{29}$$

when the strata are sampled proportionately such that the number of samples on stratum k, $n_k = p_k n$.

LHS reduces variance by negative covariance such that the variance of a LHS estimator T_L is

$$\text{Var}(T_L) = \text{Var}(T_R) + \frac{n-1}{n}\frac{1}{n^N(n-1)^N}\sum_{R}(\mu_i - \tau)(\mu_j - \tau) \tag{30}$$

where τ is the mean value, μ_i is the mean of LHS cell i, and R is the $n^N(n-1)^N$ pairs (μ_i, μ_j) of cells having no cell coordinates in common. Note that, the variance of LHS is reduced only when the second term is negative. Stein (1987) and McKay et al. (2000) have shown that the variance is always reduced with creating negative covariance. LHS, discussed in Section 3.2, reduces variance by effectively filtering the main effects (additive effects, e.g., x+y) of the transformation but has limitations on variational interactions (e.g., xy) (Stein, 1987; McKay et al., 2000).

C.3 PARTIALLY STRATIFIED SAMPLING

SS and LHS can be seen as two extremes on a spectrum of possible stratification methods (Shields & Zhang, 2016). In true SS, the stratification occurs on all dimensions simultaneously while LHS divides each dimension individually. Partially stratified sampling (PSS) stands for the broader class of all possible stratification between these extremes where stratification occurs on low-dimensional subspaces of the N-dimensional sample space. To this end, true SS and LHS can be viewed as special cases of PSS such that stratification occurs on N-dimensional and one dimensional spaces respectively.

Let $\mathbf{\Theta}_i, i = 1, ..., N_s$ denote N_s disjoint N_i-dimensional orthogonal subspaces of the N-dimensional sample space $S(N_i \leq N, \ \forall i)$ such that $\cup_{k=1}^{N_s} \mathbf{\Theta}_i = S$ and $\mathbf{\Theta}_p \cap \mathbf{\Theta}_q = \varnothing, p \neq q$. PSS divides each subspace $\mathbf{\Theta}_i$ into a collection of M_i disjoint subsets $\mathbf{\Omega}_{ik}, k = 1, ..., M_i$. Lower N_i-dimensional random samples $\mathbf{x}_{ik} = \{x_{ik1}, ..., x_{ikN_i}\}$ are drawn within each stratum $\mathbf{\Omega}_{ik}$ of subspace $\mathbf{\Theta}_i$ based on the SS method. Following this procedure, the full N-dimensional samples $\mathbf{x}$ are assembled by randomly grouping the lower-dimensional samples generated in each subspace. Using this way, a low-dimensional sample $\mathbf{x}_{ik}$ is randomly chosen from each subspace $\mathbf{\Theta}_i$ and these terms are grouped

to create a sample. Figure 6 (left) shows how to generate four samples from a 4D sample space using PSS 2x2. First, we draw stratified samples within the 2D subspaces and then randomly pair the 2D samples to assemble 4D samples. Shields and Zhang (Shields & Zhang, 2016) derived the variance of a statistical estimator using PSS, which is given by

$$\begin{aligned}\text{Var}(T_P) = \frac{1}{n}\text{Var}(T_R) - \frac{1}{n^{N_s+1}}\sum_{i=1}^{n^{N_s}}(\mu_i - \tau)^2 + \left(\frac{1}{n^{N_s+1}} - \frac{1}{n^{2N_s}}\right)\sum_{i=1}^{n^{N_s}}\mu_i^2 + \\ \frac{n^{N_s-1}}{(n-1)^{N_s-1}}\sum\sum_{R}\mu_i\mu_j - \frac{1}{n^{2N_s}} - \frac{1}{n^{2N_s}}\sum_{i=1}^{n^{N_s}}\sum_{j=1,j\neq i}^{n^{N_s}}\mu_i\mu_j\end{aligned} \tag{31}$$

where R is the space of $n^{N_s}(n-1)^{N_s}$ cell pairs with cell mean values (μ_i, μ_j). Using $\sum \mu_i = n^{N_s}\tau$, Eq.equation 31 can be rewritten to

$$\text{Var}(T_P) = \text{Var}(T_R) + \frac{n-1}{n}\frac{1}{n^{N_s}(n-1)_s^N}\sum\sum_{R}(\mu_i - \tau)(\mu_j - \tau) \tag{32}$$

Note that Eq.equation 32 follows the similar form as Eq.equation 30 when $N_s = N$, each stratified subspace corresponds to a component of $\mathbf{x}$ resulting in a LHS. PSS has the effect of filtering out additive combinations of non-additive functions, which is important to further reduce variance on variable interactions (Shields & Zhang, 2016).

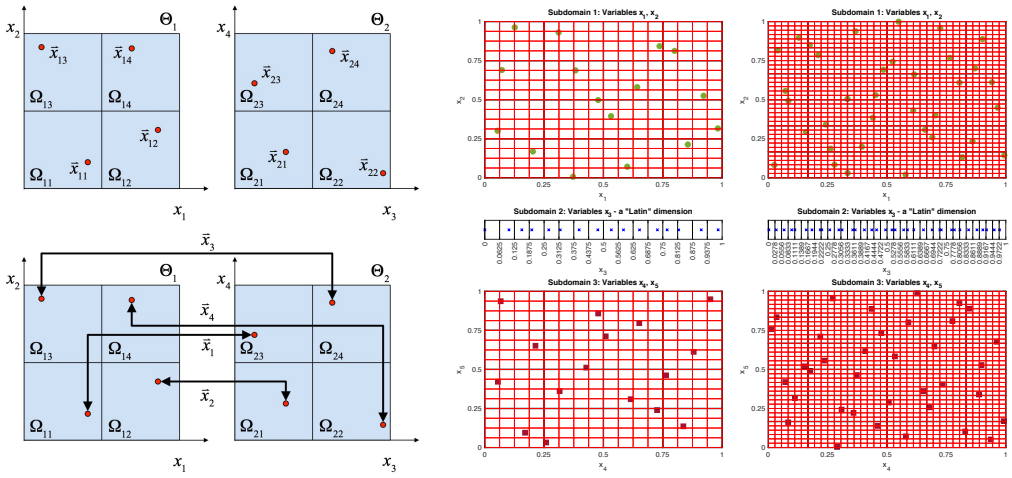

Figure 6: (Left) Four samples drawn from a four dimensional space using PSS 2x2 stratum (Shields & Zhang, 2016). (Right) Five dimensional space using LPSS 2x1x2 stratum with 16 samples and 36 samples.

C.4 LATINIZED STRATIFIED SAMPLING

To simultaneously reduce variance associated with the main effects and interaction effects, Latinized stratified sampling (LSS) is proposed by a combination of true SS and a LHS on a given N_i-dimensional subspace. The LSS typically consists of the following steps:

- Generate an LHS from the subspace without grouping the individual variables
- Stratify the domain as desired ensuring that the stratification is consistent with an LHS design
- Randomly select a point x_i from each component of the LHS without replacement such that the sample $\mathbf{x} = \{x_1, ..., x_N\}$ lies within the stratum.
- For each stratum, repeat the above steps to generate all the samples.

Note that LSS is equivalent to an Orthogonal array-based LHS (Tang, 1993; Owen, 1992) when the strata all have equal dimensions. But LSS can be applied to subspaces of different dimensions in a simple and efficient way. The benefits of LSS are to filter main effects and variable interactions but the effect will be diminished in high dimension.

C.5 LATINIZED PARTIALLY STRATIFIED SAMPLING

All the challenges discussed above can be mitigated by the use of LSS in combination with PSS, named by Latinized partially stratified sampling (LPSS), which first defines a PSS and then samples from the lower-dimensional subspace according to the LSS method. To this end, one can achieve to reduce the variance associated with both main effects and variable interactions simultaneously and is also able to scale to high dimensions. Figure 6(right) shows an example about 5D sampling space using LPSS 2x1x2 stratum with 16 and 36 samples. In our experiments, we typically divide the entire high dimensional space into 2 or 4 subspaces. For example, a 1000-dimensional problem will be divided by 500 2D or 250 4D cases. The advantages and disadvantage of these sampling methods are compared, as shown in Table 5.

Table 5: Comparison of advantages and disadvantages of sampling methods

Method	Advantages	Disadvantages
SRS	Easy to generate samples	No variance reduction (VR)
SS	VR on variable interactions	Scale poorly to high dimension
LHS	VR on main effects and scale well to high dimension	No VR on variable interactions
PSS	Improved VR on variable interaction	Limited VR on main effects and pairing penalty
LSS	VR on both main effects and variable interaction	Limited VR in high dimension
LPSS	Large VR on both main effects and variable interactions, no pairing penalty and performs well in high dimension	Optimal pairing may not be straightforward

C.6 ABLATION STUDY: THE EFFECT OF DIFFERENT SAMPLING ON VARIANCE AND BIAS

We show the effect of random sampling on variance reduction and bias estimation. Our task here is to map the latent *Gaussian* distribution $\mathbf{z} \sim N(0,1)$ to the target *Lognormal* distribution $\mathbf{x} \sim LN(3,1)$ given a fixed flow-based model but with different sampling strategies: SRS, LHS and LPSS. The task is also tested on various dimensions ranging from 1 to 1000. The estimated bias is defined by

$$\epsilon_{\mathbf{z}} = \left|\mathbb{E}[\mathbf{z}] - \hat{\mu}_z\right|, \quad \epsilon_{\mathbf{x}} = \left|\mathbb{E}[\mathbf{x}] - \hat{\mu}_x\right| \tag{33}$$

The SRS yields a significantly higher bias than the LHS and LPSS (Fig. 7(a-b))). Although increasing samples may mitigate the effect, the bias in high dimensions is still clear to see. We check whether the statistical proprieties in latent space would be preserved and propagated to the target space ($\epsilon_{\mathbf{z}}$ is correlated to $\epsilon_{\mathbf{x}}$ or not). As shown in Fig. 7 (d-e), the bias in the target space displays a similar trend as the latent space. In other words, sampling in latent space will have a direct impact on the bias estimation of the target space. Unfortunately, commonly used SRS may result in an imprecise estimation.

In this case, we fixed the normalizing flow model (NSF layers) and only change the sampling methods imposed on latent probability density. We substitute the simple affine transformation used in RealNVP (Dinh et al., 2016) by the rational-quadratic (RQ) spline (Durkan et al., 2019) implemented in `https://github.com/bayesiains/nflows`. The width, height, and slope of the RQ transformation are given by fully connected ReLU networks. We use 3 blocks with three hidden layers of dimension 64. An NSF block consists of two subsequent NSF layers with intermediate swap layers. The batch size is 256 for both 1E3 and 1E4 samples. Training was done by 2000 epochs with a learning rate $l = 0.001$. Once the model is learned, we run 20 Monte Carlo trials to measure the statistical moments for bias and variance estimators.

We denote

$$\sigma_{\mathbf{z}} = \text{Var}(\mathbb{E}[\mathbf{z}]), \quad \sigma_{\mathbf{x}} = \text{Var}(\mathbb{E}[\mathbf{x}]) \tag{34}$$

to measure the variance for the latent and target distributions using 20 repeat MC trials. Note that LPSS shows significant advantages in variance reduction, with three orders of magnitude smaller than the SRS. An increasing number of samples offers a slight improvement for SRS but is still much higher than the variance-reduced LPSS. A clear correlated trend between latent and target space further demonstrates the similar claim as the bias estimation - *sampling in latent space will dominate the statistical estimation of the target distribution*.

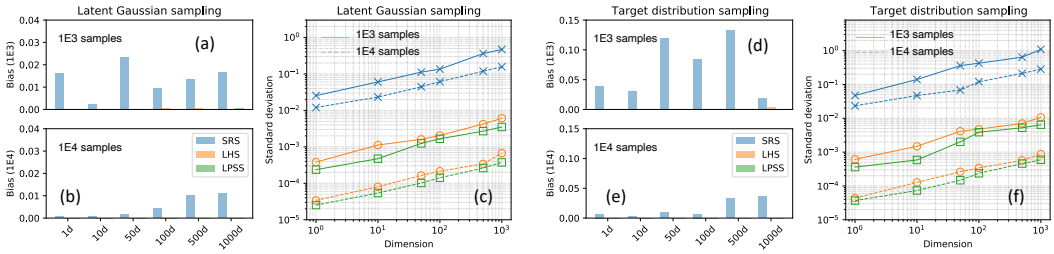

Figure 7: Variance and bias performance of different sampling algorithms on latent Gaussian distribution (left two) and target Lognormal distribution (right two). Bias estimation is shown on (a) Gaussian with 1E3 samples, (b) Gaussian with 1E4 samples, (c) Lognormal with 1E3 samples, and (d) Lognormal with 1E4 samples. Variance is shown on (a) Gaussian and (f) Lognormal distribution.

D ADDITIONAL EXPERIMENTS - DENOISING AND INPAINTING

To show the applicability of our method, we additionally provide experimental results about denoising and inpainting, which are also typical tasks in inverse problems. CelebA dataset is used here and we run the experiments on 64×64px color images with the pixel values scaled between $[0, 1]$.

D.1 DENOISING

The denoising problem can be formulated by

$$y = Ax + w, \quad A = I_n, \quad w \sim \mathcal{N}(0, \sigma I_n) \tag{35}$$

where w is additive Gaussian noise with a noise level $\sigma = 0.2$, and $n = 64 \times 64 \times 3 = 12288$ in this case. We select several samples of CelebA in distribution test set images to visualize the denoising results in Fig. 8.

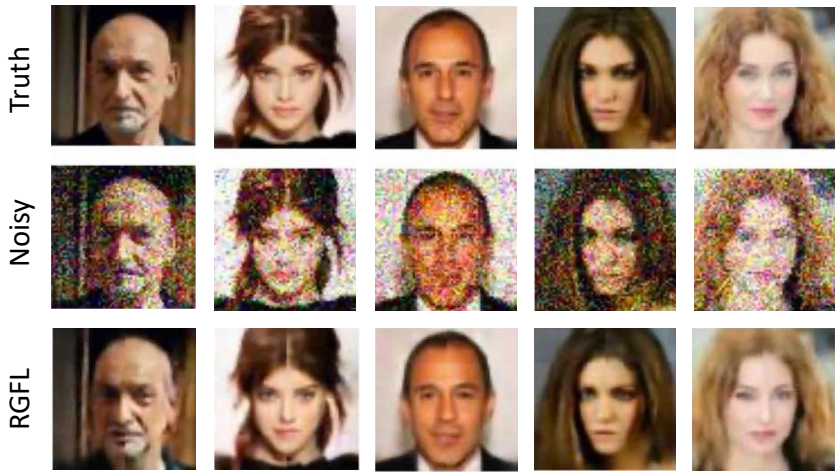

Figure 8: Image denoising - visual comparisons under the ground truth, noisy images, and denoising images with our method at noise level $\sigma = 0.2$ on CelebA in-distribution test set images.

D.2 INPAINTING

The inpainting problem can be defined as a masked image of the form

$$y = M \odot x \tag{36}$$

where M is a masking matrix with binary entries and $x \in \mathbb{R}^n$ is an n-pixel image. Similarly, we present the samples of CelebA in distribution test set images on image inpainting, as shown in Fig. 9.

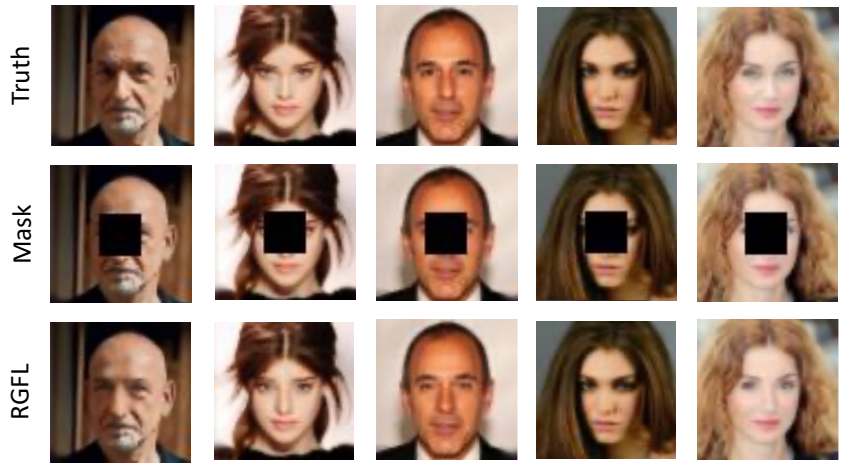

Figure 9: Image inpainting results on a sample of the CelebA test set. The first row is the ground truth, the second row is the masked images and the third row is our inpainting results.

E ADDITIONAL EXPERIMENTAL RESULTS AND DETAILS

All experiments were run using PyTorch 1.2 and on NVIDIA RTX6000 GPU. Optimization uses Adam (Kingma & Ba, 2015) with default β_1 and β_2 values.

E.1 2D IMAGE STUDY CASE

We all use 4 blocks for RealNVP, NSF and RGF flow models. All involved transformation use three hidden layers of dimension 64. We used 20 knots for both NSF layers and RGF model in the RQ-spline transformation. All training was done by 2000 epochs with batch size 256 and learning rate 0.001. After the training, 10000 random samples drawn from the latent density are used to calculate the evaluation metrics (precision, recall, density and coverage). We repeat the sampling by 20 different times and show the average score in Table 1 (main context).

E.2 COMPRESSED SENSING MRI APPLICATION

E.2.1 PROBLEM DESCRIPTION

MRI experiment is a typical compressed sensing task where under-sampled measurements result in image reconstruction uncertainty. As one of the classical inverse problems, compressed sensing MRI is often modeled as a forward model

$$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \varepsilon \tag{37}$$

where the measurements $\mathbf{y}$ are the under-sampled κ-space spatial frequency components of the image $\mathbf{x}$ with additive Gaussian noise ε with a standard deviation of 0.04% the zero-frequency amplitude. Our method is tested at three different acceleration speed-up factors: $4\times, 6\times$, and $8\times$. That means $1/4, 1/6$ and $1/8$ of all κ-space components are observed. The forward operator F often corresponds to the discrete Fourier transform (DFT). In this case, reconstructing $\mathbf{x}$ from $\mathbf{y}$ is an inverse fast Fourier transform (FFT), and that approach is typically used in many compressed sensing MRI reconstruction problems.

E.2.2 DATASET

We use the deidentified imaging dataset which is provided by NYU Langone comprises raw k-space data in several sub-dataset groups. We download the dataset from `https://fastmri.med.nyu.edu/`.

Knee MRI: Data from more than 1,500 fully sampled knee MRIs obtained on 3 and 1.5 Tesla magnets and DICOM images from 10,000 clinical knee MRIs also obtained at 3 or 1.5 Tesla. The raw dataset includes coronal proton density-weighted images with and without fat suppression. The

DICOM dataset contains coronal proton density-weighted with and without fat suppression, axial proton density-weighted with fat suppression, sagittal proton density, and sagittal T2-weighted with fat suppression. We select one scanning sample from the knee single-coil validation dataset as the ground truth.

Brain MRI: Data from 6,970 fully sampled brain MRIs obtained on 3 and 1.5 Tesla magnets. The raw dataset includes axial T1 weighted, T2 weighted, and FLAIR images. Some of the T1 weighted acquisitions included admissions of contrast agents. For the brain case, we choose one specific sample from the brain multi-coil validation dataset.

E.2.3 EXPERIMENTAL DETAILS

In this case, we use 4 blocks for RealNVP, NSF and RGF models. For affine coupling layers, we use three hidden layers and the scale-shift function of each layer is modeled as a U-Net syle fully connected networks that are composed of dense layers including the Leaky ReLU activation and the batch normalization with skip connections. For the NSF and RGF model, we use three hidden layers for width/height/slope with dimension 128 in their transformers. We use a small batch size 32 and train the model by 2000 epochs. The Adam learning rate is 1e-5 with a gradient clip coefficient 1e-3. We also use an additive MRI measurement noise with σ=5e-7 in both cases to stabilize the training.

E.2.4 ABLATION STUDY

To analyze the effect of different components on the final improvement, we provide an ablation study over the bi-directional regularization, gradient boosting, and latent sampling strategy on the brain MRI experiments with three acceleration rates. As shown in Table 7, BD-R means the bi-directional regularization, GB represents the gradient boosting and LPSS is the latent sampling strategy. In terms of the absolute error quantity, GB plays a more important role, and combining GB and BD-R shows a pretty close performance to the best performance (our RGFL). On the other side, LPSS is more powerful than GB and BD-R on variance reduction improvement, which is consistent with our expectation and the results displayed by the toy example in Section 3.6.

Table 6: An ablation study to study the effect of bi-direction regularization, gradient boosting and latent sampling strategy on brain MRI study.

	Brain with 4X		Brain with 6x		Brain with 8x	
	Abs. Error	Std. Dev.	Abs. Error	Std. Dev.	Abs. Error	Std. Dev.
BD-R only	6.56E-06	8.30E-06	7.67E-06	9.28E-06	1.15E-05	1.97E-05
GB only	3.42E-06	4.91E-06	5.03E-06	6.84E-06	8.48E-06	8.10E-06
LPSS only	6.18E-06	1.05E-06	7.20E-06	3.11E-06	9.69E-06	5.26E-06
BD-R & GB	2.03E-06	3.82E-06	4.91E-06	5.79E-06	7.94E-06	6.93E-06
BD-R, GB & LPSS (**RGFL**)	**1.78E-06**	**9.73E-07**	**4.58E-06**	**2.35E-06**	**7.12E-06**	**4.10E-06**

E.3 INTERFEROMETIC IMAGING APPLICATION

E.3.1 PROBLEM DESCRIPTION

In interferometric imaging, radio telescopes are linked to receive sparse spatial frequency measurements of an astronomical target (Thompson et al., 2017). The Fourier measurements are then used to recover the underlying astronomical image. Due to sparse and noisy measurements, multiple image modes can fit the same observed data well. The Event Horizon Telescope (EHT) applied this technique to capture the first black hole picture, by linking telescopes across the globe (Akiyama et al., 2019b;c;d;e;f; 2021).

The Fourier measurement is obtained by correlating the data from a pair of telescopes and the measurement equation is given by:

$$M_{a,b}^t = g_a^t g_b^t \exp[-i(\phi_a^t - \phi_b^t)] F_{a,b}^t \mathbf{x} + n_{a,b}^t, \tag{38}$$

where a and b index the telescopes, $F_{a,b}^t \mathbf{x}$ extracts the Fourier component from image $\mathbf{x}$ corresponding to the baseline between telescope a and b at time t which represents time. Typically, the measurement

noise consists of three sources: (1) Gaussian thermal noise $n^t_{a,b} \sim \mathcal{N}(0, \nu^2_{a,b})$, (2) time-dependent phase error ϕ^t_a and ϕ^t_b, and (3) time-dependent gain error g^t_a and g^t_b.

If the gain errors and the phase errors are large, the measurements can be combined into robust data products that are invariant to telescope-based errors, named closure phase, $C^{ph.t}_{a,b,c}$ and closure amplitude, $C^{amp.,t}_{a,b,c,d}$:

$$C^{ph.,t}_{a,b,c} = \angle(M^t_{a,b}, M^t_{b,c}, M^t_{c,a}) \approx f^{ph.,t}_{a,b,c}(\mathbf{x}), \quad C^{amp.,t}_{a,b,c,d} = \frac{|M^t_{a,b}||M^t_{c,d}|}{|M^t_{a,c}||M^t_{b,d}|} \approx f^{amp.,t}_{a,b,c,d}(\mathbf{x}) \tag{39}$$

The above nonlinear closure quantities can be used to constrain the non-convex image reconstruction problem, which is given by

$$\mathcal{L}(\mathbf{y}, f(\mathbf{x})) = \sum_{a,b,c} |M^{ph.,t}_{a,b,c} - f^{ph.,t}_{a,b,c}(\mathbf{x})|^2/\sigma^2_{a,b,c} + \sum_{a,b,c,d} |\log M^{amp.,t}_{a,c,b,d} - \log f^{amp.,t(\mathbf{x})}_{a,b,c,d}|^2/\sigma^2_{a,b,c,d} \tag{40}$$

where $\sigma_{a,b,c}$ and $\sigma_{a,b,c,d}$ are the standard deviations of the corresponding closure term computed based on each telescope's system equivalent flux density. Note that Eq. 40 only approximates the true data log-likelihood. In this case, we demonstrate our method on non-convex reconstruction with closure quantities using real datasets from `https://github.com/achael/eht-imaging`.

E.3.2 Dataset

The black hole case is one of the important radio interferometric astronomical imaging problems. Typically, radio telescopes are linked to obtaining sparse spatial frequency measurements of an astronomical target. Then the Fourier measurements are employed to recover the underlying astronomical image. The Event Horizon Telescope (EHT) applied this technique to capture the first picture of a black hole, by linking telescopes across the globe. More background and technical details can be found via (Event Horizon Telescope Collaboration et al., 2019; Akiyama et al., 2019a; Gammie et al., 2019; Akiyama et al., 2019e;f).

E.3.3 Experimental details

In this case, we blurred the original data to a lower resolution by 32x32. The field of view of the image in micro-arcsecond is set to 160. The fwhm of the image prior in micro-arcsecond is set to 50. As similar as the FastMRI case, we use 4 blocks for all flow-based models. For affine coupling layers, we use three hidden layers and the scale-shift function of each layer is modeled as a U-Net syle fully connected networks that are composed of dense layers including the Leaky ReLU activation and the batch normalization with skip connections. For the NSF and RGF model, we use three hidden layers for width/height/slope with dimension 128 in their transformers. We use a smaller batch size of 16 and train the model by 10000 epochs. The Adam learning rate is 1e-4 with a gradient clip coefficient of 1e-3. Once the learned model is done, we generate 1024 random samples from the latent density and estimate the mean and variance. These samples are also used to analyze the different modes and t-SNE visualization.

E.3.4 Ablation study

We provide an ablation study for the blackhole image case to analyze the effect of bi-directional regularization, gradient boosting, and latent sampling strategy on our RGFL performance. We found a similar trend that means GB shows a bigger impact on the absolute error metric, and combining GB and BD-R shows a pretty close performance to the best performance (RGFL). However, compared with BG and BD-R, LPSS has a more sensitive impact on variance reduction.

F Discussion of computational cost

The major limitation of this work is the training cost which mainly includes three parts, bi-directional FD regularization, gradient boosting, and latent space sampling. The gradient boosting scheme is the most expensive one since it needs additional training cycles for updating the new flow components and the optimal weights. However, introducing gradient boosting does not affect the prediction and

Table 7: An ablation study to study the effect of bi-direction regularization, gradient boosting, and latent sampling strategy on blackhole image study.

	blackhole images	
	Abs. Error	Std. Dev.
BD-R only	2.50E-04	5.12E-04
GB only	1.28E-04	3.22E-04
LPSS only	2.59E-04	1.35E-04
BD-R & GB	1.17E-04	2.64E-04
BD-R, GB & LPSS (**RGFL**)	**1.08E-04**	**1.20E-04**

sampling efficiency. The bi-directional FD regularization is imposed on both forward mapping and backward mapping, so it has an impact on training and prediction but it is not significant as the gradient boosting. Latent space sampling with LPSS is the minor one in terms of total computational cost although it is slower than simple random sampling.

Our future work will mainly address these computational concerns to improve the scalability on large-scale problems such that the proposed methods will make a bigger impact and scope to solving inverse problems, not only limited by underdetermined imaging systems but also more general image reconstruction tasks in machine learning and computer vision community.

G Additional experimental results

G.1 FastMRI samples

We provide additional generated samples from the learned model. Figure 10 - Figure 13 show the additional 40 samples for the brain case with speedup 4x, 6x, 8x and 10x. The knee cases are shown in Figure 14 - Figure 17 for speedup 4x, 6x, 8x and 10x respectively.

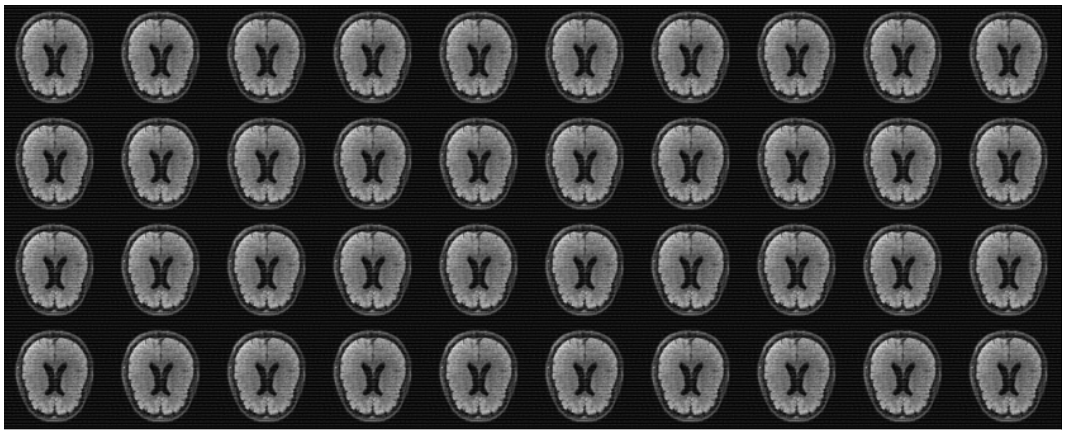

Figure 10: Brain with speedup 4x: additional 40 generated samples from the learned model.

G.2 Black hole samples

For the black hole case study, we also offer more generated samples, as shown18 and it is noted that the two modes are clear to be observed from these samples as we discussed in the main context.

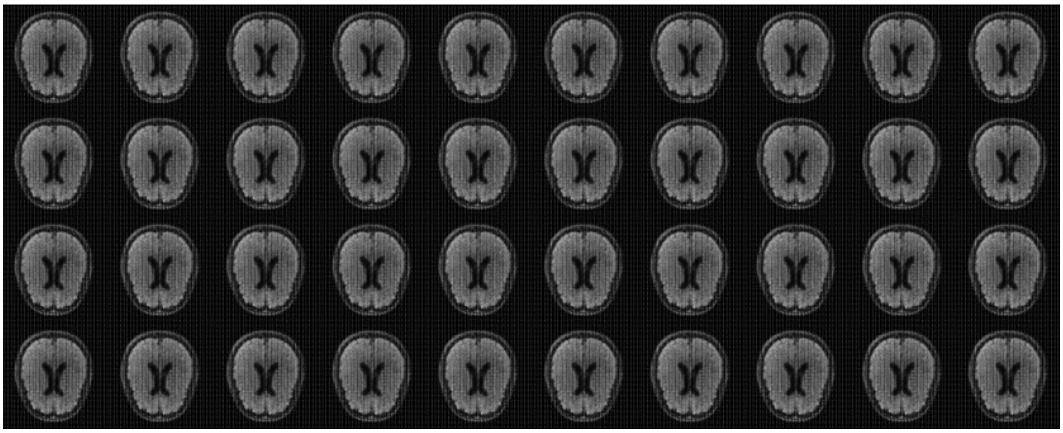

Figure 11: Brain with speedup 6x: additional 40 generated samples from the learned model.

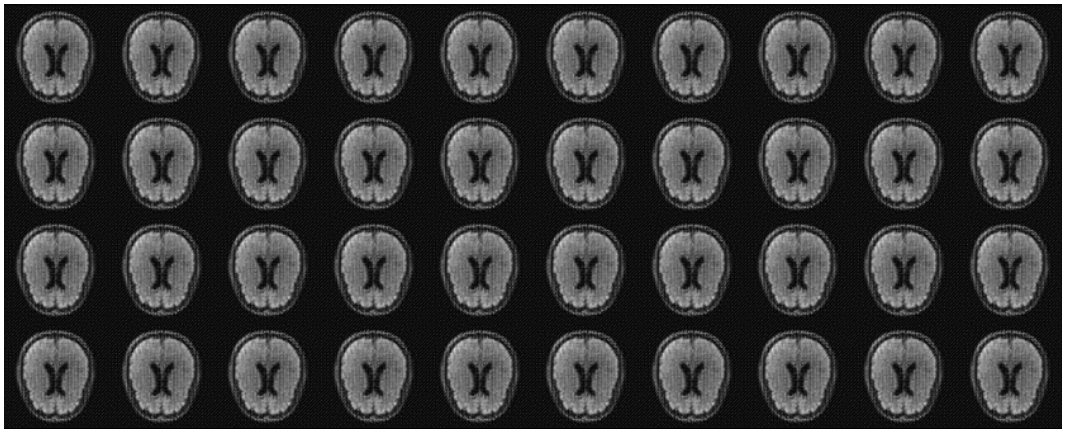

Figure 12: Brain with speedup 8x: additional 40 generated samples from the learned model.

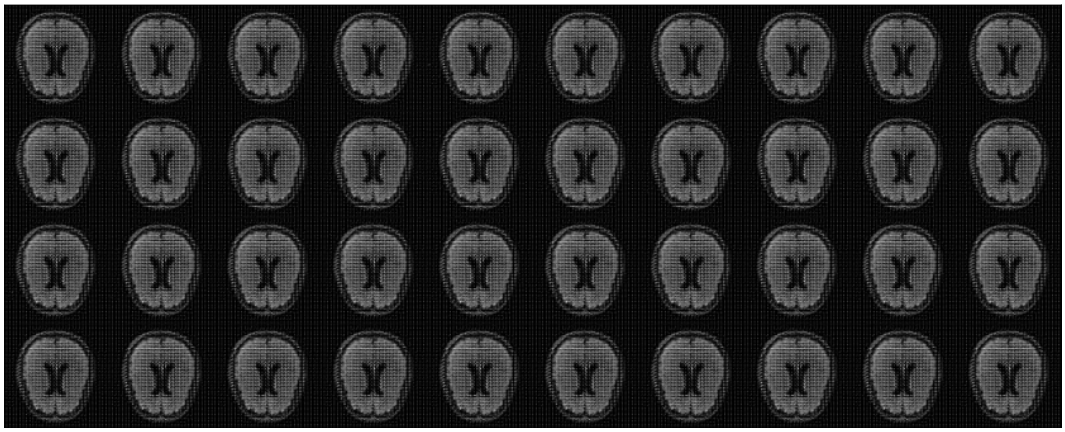

Figure 13: Brain with speedup 10x: additional 40 generated samples from the learned model.

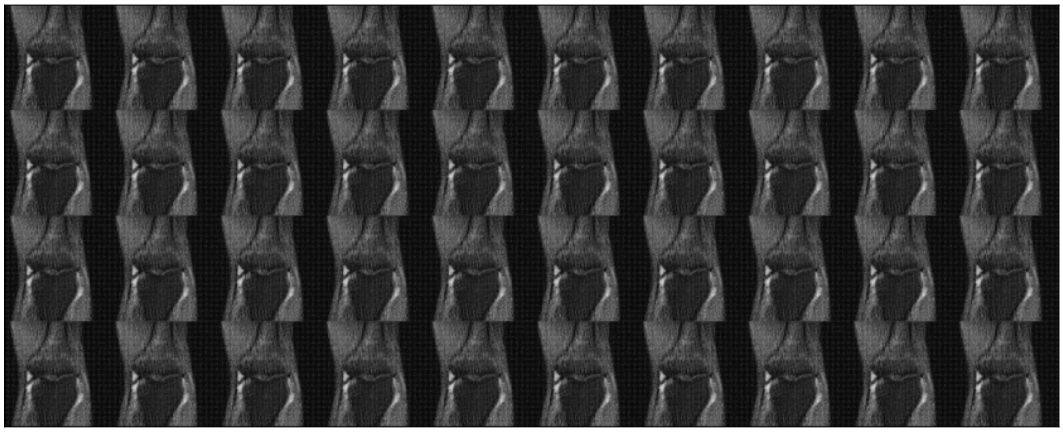

Figure 14: Knee with speedup 4x: additional 40 generated samples from the learned model.

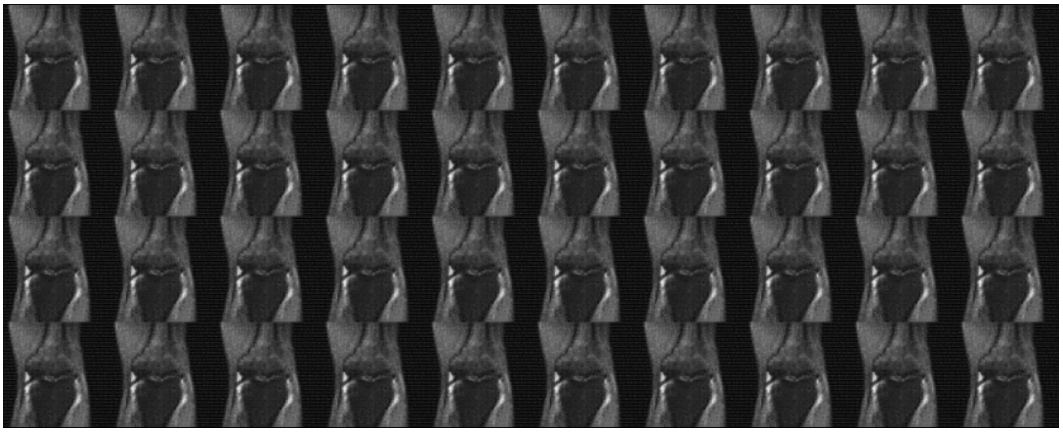

Figure 15: Knee with speedup 6x: additional 40 generated samples from the learned model.

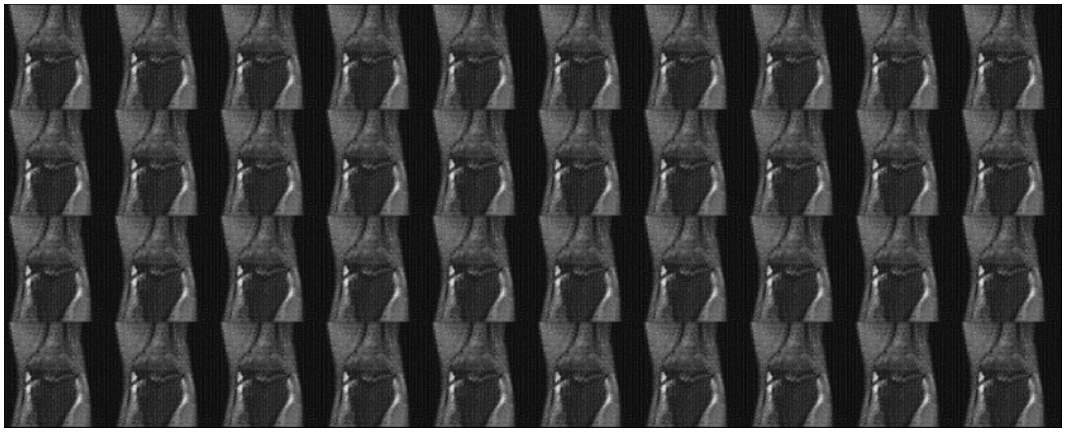

Figure 16: Knee with speedup 8x: additional 40 generated samples from the learned model.

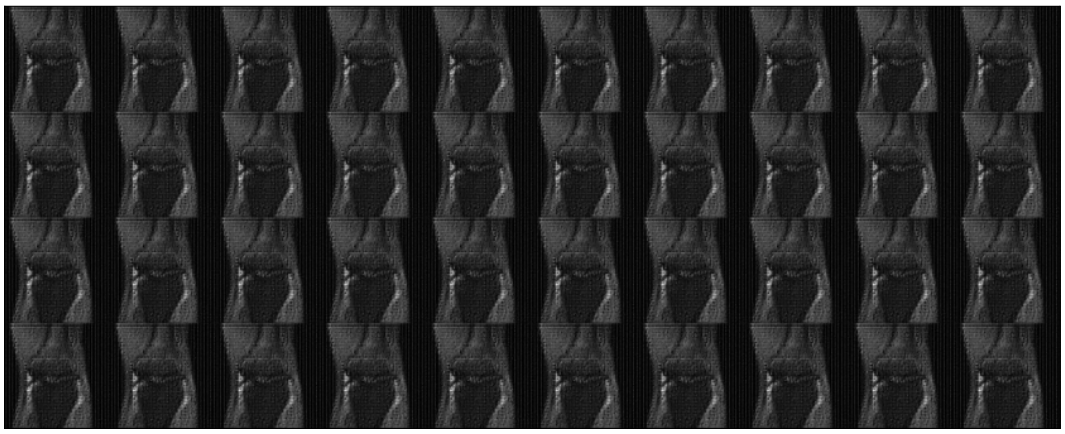

Figure 17: Knee with speedup 10x: additional 40 generated samples from the learned model.

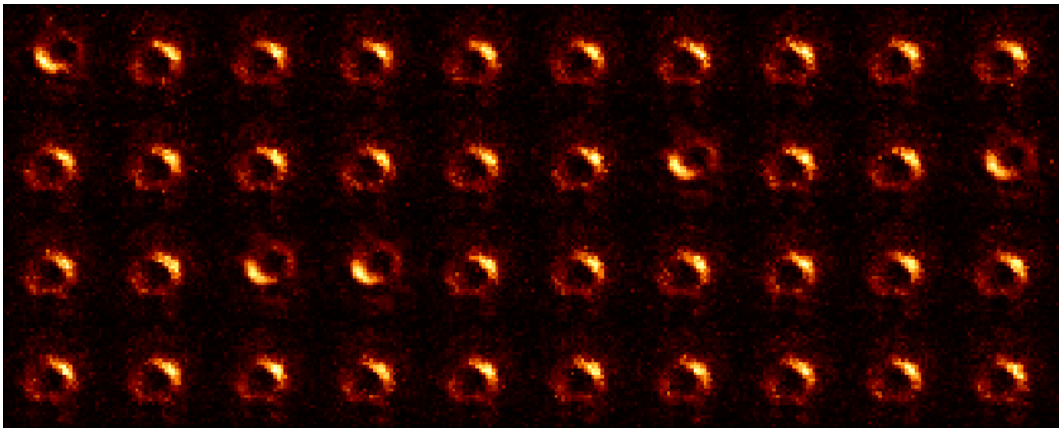

Figure 18: Black hole: additional 40 generated samples from the learned model.Note that two different modes are clear to see in more samples.