Score-Based Diffusion Models as Principled Priors for Inverse Imaging: Supplemental

Contents

A Implementation Details **1**

A.1 Score-Based Priors 1

A.1.1 Log-probability computation 1

A.1.2 Gradient estimation 1

A.2 Posterior Sampling Experiments 2

A.3 2D Experiments 2

B Image-Restoration Metrics **3**

C Score-Based Priors vs. Discrete-Flow Priors **3**

A. Implementation Details

In this section, we discuss practical considerations for numerically computing log-probabilities and gradients under a score-based prior. We also discuss the experimental setups of our presented results.

A.1. Score-Based Priors

A.1.1 Log-probability computation

Recall that for a pretrained score model $\mathbf{s}_\theta(\mathbf{x}, t)$, the log-probability formula is given by

$$\log p_0(\mathbf{x}_0) = \log p_T(\mathbf{x}_T) + \int_0^T \nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t; \theta) dt, \quad (1)$$

where $\tilde{\mathbf{f}}(\mathbf{x}_t, t; \theta)$ comes from the probability flow ODE:

$$d\mathbf{x}_t = \left[\mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2} g(t)^2 \mathbf{s}_\theta(\mathbf{x}_t, t) \right] dt =: \tilde{\mathbf{f}}(\mathbf{x}_t, t) dt. \quad (2)$$

Given an image $\mathbf{x}$, to compute $\log p_\theta(\mathbf{x})$ under the score-based prior parameterized by θ, we have to solve an initial-value problem, where $\mathbf{x}_0 = \mathbf{x}$ and $\frac{d\mathbf{x}_t}{dt} = \tilde{\mathbf{f}}(\mathbf{x}_t, t; \theta)$.

Log-probability estimation. The two implementation decisions that most affect log-probability accuracy are: (1) which ODE solver to use and (2) how to estimate the divergence in Eq. 1. To deal with (1), our code uses Diffrax [5], a JAX library for differential equations, to easily swap out solvers and adaptively select time steps. As for (2), we use Hutchinson-Skilling estimation with multiple trace estimators to reduce the variance of log-probability and gradient calculations.

ODE solver. Tab. A.1.1 shows how different solvers affect time-efficiency and KL divergence to a ground-truth distribution. The ground-truth is the Gaussian distribution used in Main Sec. 3.2. This suggests that Bogacki–Shampine's 3/2 method and Dormand-Prince's 5/4 method offer a good balance between efficiency and accuracy. Note, however, that score-based priors trained on different datasets may show different trends. It is always a good idea to evaluate the runtime of different solvers for a given score-based prior to find the most efficient solver.

Solver	$D_{\mathrm{KL}}(q\|p)$ (↓)	NFE low. bound (↓)
Euler* (1st order)	0.848	4092
Heun (2nd order)	0.478	312
Bosh3 (3rd order)	0.453	81
Tsit5 (5th order)	0.521	255
Dopri5 (5th order)	0.284	65
Dopri8 (8th order)	0.422	1440

Table 1. KL divergence depending on the solver used for log-probability computation. "Euler" used a fixed step-size of $1/4092$. All other solvers used adaptive step-sizing, with the number of function evaluations ("NFE") calculated as the number of solver steps times the order of the solver. The KL divergence was estimated from 512 samples from the ODE sampler.

Trace estimation. For high-dimensional data, trace estimation is necessary to estimate the divergence in Eq. 1. This causes variance in the estimated log-probabilities and gradients. Song et al. [8] use Hutchinson-Skilling with one trace estimator, but we use multiple trace estimators to reduce variance. In our implementation, the same trace estimators are applied to each image in a batch. Figs. 1 and 2 show the variance of densities and gradients, respectively, depending on the number of trace estimators used.

A.1.2 Gradient estimation

Adjoint ODE. To compute the exact gradient $\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$, we would need to backpropagate through the ODE solve.

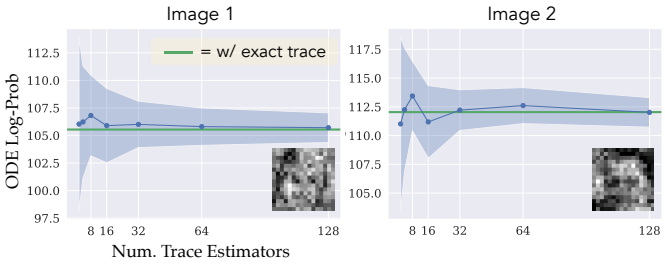

Figure 1. Mean and variance of log-probability values vs. number of trace estimators. The score-based prior was fit to the ground-truth Gaussian distribution used in Main Sec. 3.2. For each number of trace estimators (1, 2, 4, 8, 16, 32, 64, or 128), 50 trials of log-probability estimation were done with different random seeds (using the Dopri5 solver with adaptive step-sizing). The solid blue line indicates the mean of these trials, and the shaded region indicates one std. dev. above and below the mean. The solid green line shows the value resulting from exact trace calculation. The evaluated image is inset. As more trace estimators are used, the variance of the log-probability decreases.

This is too memory-intensive, so we opt for the continuous adjoint method [1, 5], which solves a secondary ODE that gives the gradient of the idealized continuous-time primary ODE. This adjoint method best balances our memory, speed, and accuracy requirements. Direct backpropagation through the probability flow ODE could be possible with improved gradient-checkpointing.

A.2. Posterior Sampling Experiments

Gaussian ground-truth distribution. The Gaussian distribution is defined for 16×16 grayscale images. The mean and covariance were fit by expectation-maximization to images from the CelebA training set (each image was first center-cropped to 140×140 and then rescaled to 16×16). The covariance was preconditined by adding 0.01 along the diagonal. To generate a batch of training data for a score model, samples are randomly drawn from the resulting Gaussian distribution.

Score model. All score models that were trained on 32×32 images had an NCSN++ architecture [8] with 64 filters in the initial layer. The score model trained on the Gaussian ground-truth distribution in Main Sec. 3.2 had 128 filters in the initial layer.

DPI implementation. We adapted the PyTorch implementation of DPI [9] [1] for JAX/Flax. For all presented results on image posterior sampling, we used a RealNVP architecture with 64 affine-coupling layers. The RealNVP was optimized with stochastic gradient descent (SGD) with a batch size of 64. We used Adam optimizer with a learning rate of 0.0002 and clipped gradients to have norm 1.

DPI sampling. Once optimized, the RealNVP can be sampled to obtain samples from the approximate posterior. Occasionally the RealNVP produces a clearly out-of-distribution sample, so we remove such outliers by discarding any sample with a pixel value whose magnitude is greater than 2. Although not needed in most cases, we applied this postprocessing step before computing statistics of DPI-estimated posteriors.

DPI optimization time. The main computational bottleneck is computing log-probabilities for each batch. Since we use adaptive step-size controllers, the time required for each SGD step is variable. In our experiments, we found it ranged from 30 seconds/step to 200 seconds/step. The time required for each ODE solve could also depend on the complexity of the distribution underlying the score-based prior. For example, we found CelebA priors to be faster (about 50 seconds/step for interferometric imaging experiments) and CIFAR-10 priors to be slower (about 200 seconds/step for deblurring a CIFAR-10 image, which was the slowest case). The RealNVP generally converges within 5000-10000 SGD steps, although we ran the optimization for 20000-50000 steps to be sure of convergence. We used v4-8 TPUs to perform the optimization.

Although DPI with a score-based prior takes a long time to optimize, it is extremely efficient for *sampling*. Sampling 128 samples (32×32 RGB images) takes about 2.76 seconds. In contrast, the diffusion-based baselines that we include in the main text are much slower. To get 128 samples, SDE+Proj [7] takes 20.8 seconds; Score-ALD [4] (with 5 Langevin-dynamics steps at each annealing level) takes 51.8 seconds; and DPS [2] takes 34.1 seconds.

Furthermore, our framework gives a reliable and rich posterior automatically. This saves human time and effort that would have been spent on carefully handcrafting and validating regularizers/priors.

A.3. 2D Experiments

Supp. Fig. 5 and Main Fig. 4 compare our posterior-sampling approach to baselines (SDE+Proj, Score-ALD, DPS) on a toy 2D posterior. Our samples were generated from a RealNVP with 32 affine-coupling layers. All methods used the same true score model. Since baseline methods do not provide posterior probabilities (only samples), we used kernel density estimation (KDE) to approximate a probability density function (PDF) from 10000 samples. In Fig. 5, PDFs were estimated with `scipy.stats.gaussian_kde`, which includes automatic KDE bandwidth selection. In Main Fig. 4, `sklearn.neighbors.KernelDensity` was used with a bandwidth-0.03 Gaussian kernel, since `scipy.stats.gaussian_kde` does not do well on multimodal distributions.

[1] https://github.com/HeSunPU/DPI

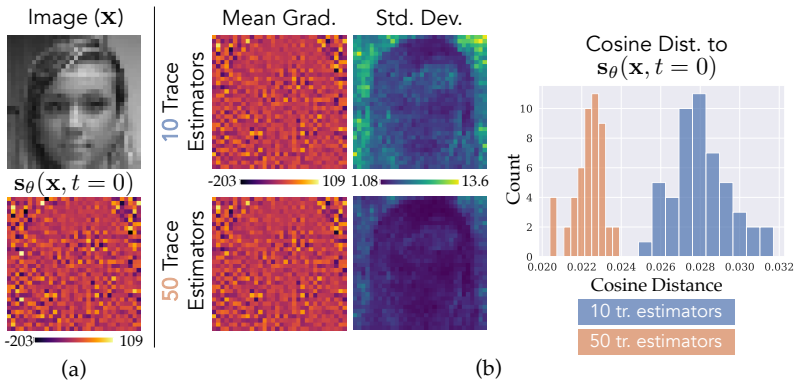

Figure 2. Mean and variance of $\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ with 10 vs. 50 trace estimators. The score-based prior was trained on 32×32 grayscale CelebA images. **(a)** Test image $\mathbf{x}$ and gradient according to the learned score model, $\mathbf{s}_\theta(\mathbf{x}, t)$, evaluated at $t = 0$. (In reality, we set $t = 10^{-3}$ for numerical stability and perturbed $\mathbf{x}$ with noise accordingly.) Since the test image was drawn as $\mathbf{x} \sim p_0$, the score-model output should equal the true $\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$. **(b)** Results of estimating the gradient $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ with the probability flow ODE including trace estimation. For both "10 trace estimators" and "50 trace estimators", 50 trials of gradient estimation were done with the continuous adjoint method. "Mean Grad." and "Std. Dev." are the average gradient and std. dev. of the gradient of all these runs. "Cosine Dist. to $\mathbf{s}_\theta(\mathbf{x}, t=0)$" shows the histogram of the cosine distance between each gradient estimate and the score-model output, which we consider to be ground-truth. The results in (b) are evidence that trace estimation gives a good approximation of the gradient in expectation, but using fewer trace estimates causes higher variance. With 10 trace estimates, the median relative std. dev. of the gradient is 16%. With 50 trace estimates, it is 8.6%. (Relative std. dev. is computed as $|\sigma|/|\mu|$.) Also note that regions of highest variance are in the image background.

B. Image-Restoration Metrics

For our results on deblurring, we evaluated our chosen posterior-sampling approach against baselines (SDE+Proj, Score-ALD, DPS) using standard image-restoration metrics (MSE, SSIM, PSNR). Fig. 3 shows the evaluated metrics. We emphasize that such metrics do not reflect the correctness of the posterior. But for applications that call for high-quality posterior samples, Fig. 3 suggests that our framework is still preferable to baselines.

CelebA Image

	MSE (↓)	SSIM (↑)	PSNR (↑)	MSE (↓)	SSIM (↑)	PSNR (↑)
Sc.-ALD	0.0093	0.781	20.37	0.0203	0.539	16.98
DPS	0.0070	0.823	21.66	0.0186	0.585	17.47
SDE+Proj	0.0120	0.745	19.38	0.0216	0.525	16.74
Ours	**0.0034**	**0.880**	**24.75**	**0.0068**	**0.757**	**21.65**
	(a) CelebA prior			(b) CIFAR-10 prior		

CIFAR-10 Image

	MSE (↓)	SSIM (↑)	PSNR (↑)	MSE (↓)	SSIM (↑)	PSNR (↑)
Sc.-ALD	0.0194	0.635	17.15	0.0383	0.469	14.20
DPS	0.0183	0.651	17.42	0.0312	0.519	15.15
SDE+Proj	0.0158	0.669	18.03	0.0291	0.521	15.44
Ours	**0.0109**	**0.764**	**19.62**	**0.0157**	**0.684**	**18.05**
	(c) CIFAR-10 prior			(d) CelebA prior		

Figure 3. Image-restoration metrics (deblurring example). For Main Figs. 7 and 8, average MSE, SSIM, and PSNR of 128 estimated samples from each method compared to the true source image were evaluated. Our posterior samples outperform baseline samples for every combination of a source image and prior (e.g., CIFAR-10 prior applied to a CelebA source image).

C. Score-Based Priors vs. Discrete-Flow Priors

Although discrete normalizing flows (e.g., RealNVP [3], Glow [6]) are generative networks that provide image probabilities, they suffer from two limitations: (1) they are restricted to invertible network architectures, which limits the ability to express a diverse and sophisticated image distribution; and (2) their probability function does not generalize well outside of training data. We note that a score-based diffusion model (following the probability flow ODE) is actually a *continuous*-time normalizing flow [8].

Main Fig. 6 and Main Tab. 1 show the results of using a discrete normalizing-flow (NF) image prior for denoising. Fig. 4 shows results for deblurring. In both experiments, the NF used was a RealNVP with 64 affine-coupling layers, and DPI optimization was done with a learning-rate of 10^{-5} and gradients clipped to a norm of 1. Compared to the score-based prior trained on the same dataset, the NF prior resulted in less visually-convincing samples and caused unstable optimization of the DPI posterior. The inferior quality of samples might be due to the limited expressiveness of a discrete NF. The instability might be due to the NF's inability to generalize to non-training images. This is relevant for inference algorithms that are randomly initialized (like DPI), as randomly-initialized images are likely far away

from the prior. We note that, although an NF prior consistently performed poorly in our experiments, clever initialization might make DPI optimization more stable with an NF prior, and other NF architectures might be more expressive than a RealNVP architecture.

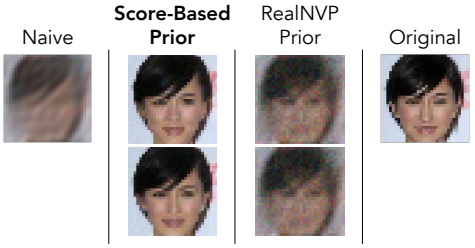

Figure 4. Score-based prior vs. RealNVP prior. A score-based diffusion model and a RealNVP were each trained on the same CelebA training set. We then applied each of their probability functions as the prior in DPI for the task of deblurring (the same task as in Main Fig. 7). Two samples are shown from each estimated posterior.

While a RealNVP is not as expressive as a diffusion model, our experiments with DPI suggest that a RealNVP can model a *posterior* that is sufficiently constrained by measurements. If the inverse problem is extremely ill-posed — meaning the posterior is almost indistinguishable from the prior — then a RealNVP would probably not sufficiently capture the distribution. DPI is not restricted to discrete normalizing flows, though. As long as the generative model used to approximate the posterior is invertible, it can be optimized via the variational objective.

References

[1] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *NeurIPS*, 31, 2018. 2

[2] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023. 2

[3] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 3

[4] Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alexandros G Dimakis, and Jonathan I Tamir. Robust compressed sensing mri with deep generative priors. *NeurIPS*, 2021. 2

[5] Patrick Kidger. On neural differential equations. *arXiv preprint arXiv:2202.02435*, 2022. 1, 2

[6] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018. 3

[7] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. In *ICLR*, 2022. 2

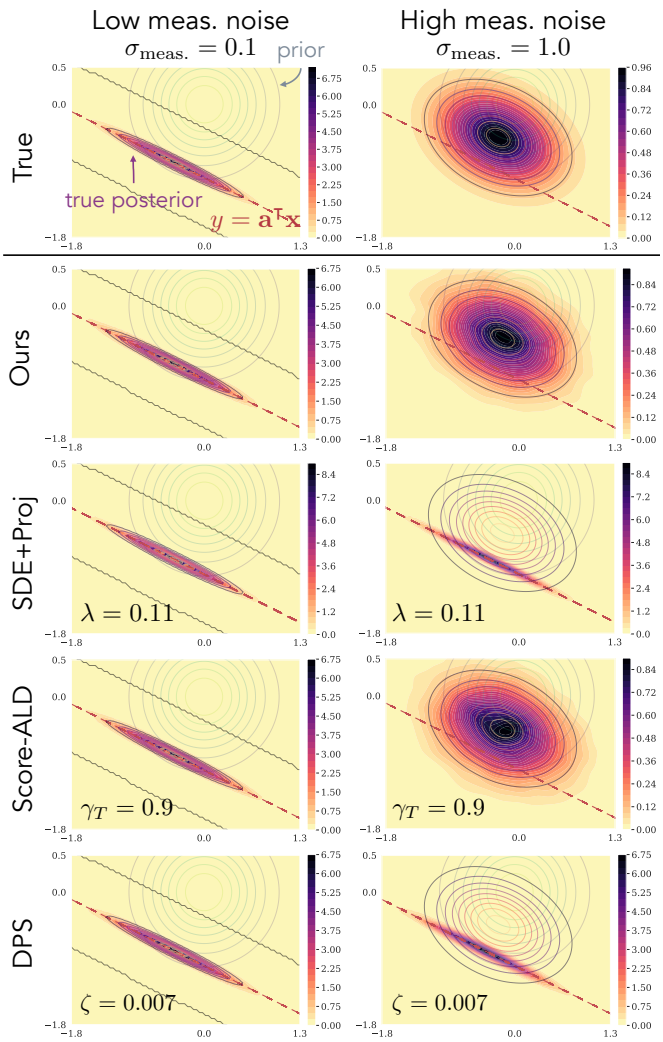

Figure 5. In this toy example with a Gaussian prior and linear measurements, we see that our method samples from the true posterior regardless of the measurement noise. For SDE+Proj, Score-ALD, and DPS, the optimal hyperparameter value was found (according to sample-approximated KL divergence to the true posterior) for the "Low meas. noise" case. The same value was then applied for the "High meas. noise" case. SDE+Proj and DPS severely underestimate the spread of the posterior. Score-ALD works well here since the approximation of the measurement-likelihood converges to the true likelihood distribution as ALD continues. In general, though, Score-ALD can become unstable when the measurement annealing rate (i.e., the sequence of γ_t's) is not well-tuned.

[8] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021. 1, 2, 3

[9] He Sun and Katherine L Bouman. Deep probabilistic imaging: Uncertainty quantification and multi-modal solution characterization for computational imaging. In *AAAI*, pages 2628–2637, 2021. 2