OPTIMIZING FEW-STEP DIFFUSION SAMPLERS BY GRADIENT DESCENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Denoising Diffusion Probabilistic Models (DDPMs) have emerged as a flexible family of generative models rivaling GANs and autoregressive models in sample quality and likelihoods. DDPMs however typically require hundreds of inference steps to generate a high-fidelity image, despite recent progress on speeding up diffusion model sampling. We introduce Differentiable Diffusion Sampler Search (DDSS): a method that learns few-step samplers for any pre-trained DDPM by using gradient descent. We propose Generalized Gaussian Diffusion Models (GGDM), a family of non-Markovian samplers for diffusion models, and we show how to improve the generated samples of pre-trained DDPMs by optimizing the degrees of freedom of the GGDM sampler family with respect to a perceptual loss. Our optimization procedure backpropagates through the sampling process using the reparametrization trick and gradient rematerialization. Searching our novel GGDM family with DDSS, we achieve strong results on unconditional image generation on both CIFAR and ImageNet 64x64 (*e.g.,* FID scores on CIFAR10 of 7.59 with only 10 inference steps, and 4.67 with 25 steps, compared to 13.62 and 6.56 with the strongest respective DDIM($\eta = 0$) baselines). Our method is compatible with any pre-trained DDPM without re-training, only needs to be applied once, and does not finetune the parameters of the pre-trained DDPM.

1 INTRODUCTION

Denoising Diffusion Probabilistic Models (DDPM) (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) have emerged as a powerful family of generative models, capable of synthesizing high-quality samples across various modalities, *e.g.,* images, audio, and 3D shapes (Ho et al., 2020; 2021; Cai et al., 2020; Chen et al., 2021a;b; Luo & Hu, 2021). Recent work (Dhariwal & Nichol, 2021; Ho et al., 2021) shows that DDPMs can outperform Generative Adversarial Networks (GAN) (Goodfellow et al., 2014; Brock et al., 2018) in generation quality, but unlike GANs, DDPMs admit likelihood computation and much more stable training dynamics (Arjovsky et al., 2017; Gulrajani et al., 2017). That being said, GANs are much more efficient at generation time, typically

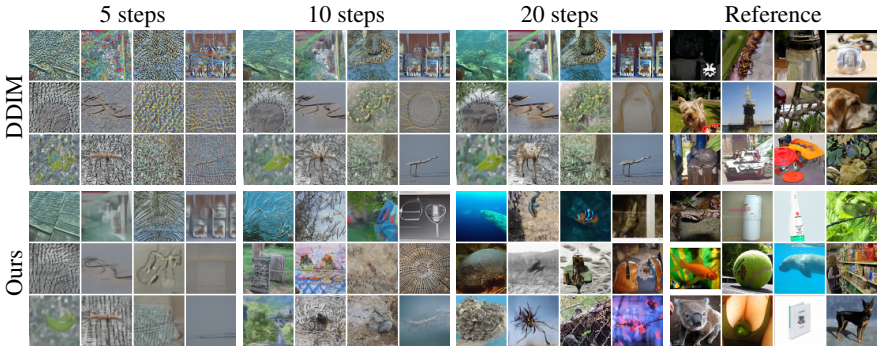

Figure 1: Non-cherrypicked samples for a DDPM trained on ImageNet 64x64, comparing the strongest DDIM($\eta = 0$) baseline and our approach. All samples were generated with the same random seeds. For reference, we include DDPM samples using all 4,000 stpes (top right) and real images (bottom right).

requiring a single forward pass through the generator network, whereas DDPMs require hundreds of inference steps. Instead of learning a generator directly, DDPMs learn to model the score function of the marginals of a diffusion process that interpolates between the data distribution and a simple prior, leading to a wide variety of feasible strategies for sampling (Song et al., 2021b). In particular, at inference time, DDPMs allow controlling the number of forward passes to the modeled score function (a.k.a. *inference steps*) (Song et al., 2020; Nichol & Dhariwal, 2021).

It has been shown both empirically and mathematically that, for any sufficiently good DDPM, more inference steps leads to better log-likelihood and sample quality (Nichol & Dhariwal, 2021; Kingma et al., 2021). In practice, the minimum number of inference steps to achieve competitive sample quality is highly problem-dependent, *e.g.,* depends on the complexity of the dataset, and the strength of the conditioning signal if the task is conditional. Given the importance of generation speed, recent work (Song et al., 2020; Chen et al., 2021a; Watson et al., 2021) has explored reducing the number of steps required for high quality sampling with pretrained diffusion models.

This paper treats the selection of efficient samplers for diffusion models as a differentiable optimization problem, and proposes *Differentiable Diffusion Sampler Search* (DDSS). Our key observation is that one can unroll the sampling chain of a diffusion model and use reparametrization trick (Kingma & Welling, 2013) and gradient rematerialization (Kumar et al., 2019a) to optimize over a class of parametric few-step samplers with respect to a global objective function. Our class of parameteric samplers, which we call Generalized Gaussian Diffusion Model (GGDM), includes Denoising Diffusion Implicit Models (DDIM) (Song et al., 2020) as a special case and is motivated by the success of DDIM on few-step diffusion sampling.

An important challenge for efficient DDPM sampling is the mismatch between the training objective (*e.g.,* ELBO or weighted ELBO) and sample quality. Prior work (Watson et al., 2021; Song et al., 2021a) finds that samplers that are optimal with respect to ELBO often lead to worse sample quality and Fréchet Inception Distance (FID) scores (Heusel et al., 2017), especially with few inference steps. We propose the use of a *perceptual* loss within the DDSS framework to find high-fidelity diffusion samplers, motivated by prior work showing that their optimization leads to solutions that correlate better with human perception of quality. We empirically find that using DDSS with the Kernel Inception Distance (KID) (Bińkowski et al., 2018) as the perceptual loss indeed leads to few-step samplers with significantly better image quality than prior work (see Figure 1). Moreover, our method is robust to different choices of kernels.

Our main contributions are as follows:

1. We propose Differentiable Diffusion Sampler Search (DDSS), using the reparametrization trick and gradient rematerialization to optimize any parametric space of few-step diffusion samplers.
2. We identify a parametric family of Generalized Gaussian Diffusion Model (GGDM) that admits high-fidelity few-step samplers for diffusion models.
3. We show that using DDSS to optimize samplers by minimizing the Kernel Inception Distance leads to few-step diffusion samplers with state-of-the-art sample quality scores.

2 Background on Denoising Diffusion Implicit Models

We start with a brief review on DDPM (Ho et al., 2020) and DDIM (Song et al., 2020). DDPMs fix a Markovian forward process

$$q(\boldsymbol{x}_0, ..., \boldsymbol{x}_T) = q(\boldsymbol{x}_0) \prod_{t=1}^{T} q(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}) \tag{1}$$

where $q(\boldsymbol{x}_0)$ is the data distribution, and, following the notation of Watson et al. (2021),

$$q(\boldsymbol{x}_t | \boldsymbol{x}_s) = \mathcal{N}(\boldsymbol{x}_t | f_{ts} \boldsymbol{x}_s, g_{ts}^2 \boldsymbol{I}_d). \tag{2}$$

DDPMs are trained to model $\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t)$ by maximizing a (possibly reweighted) evidence lower bound (ELBO), which is equivalent to minimizing a weighted sum of denoising score matching objectives (DSM) (Vincent, 2011):

$$\mathbb{E}_q \left[D_{\mathrm{KL}}[q(\boldsymbol{x}_T|\boldsymbol{x}_0) \| p(\boldsymbol{x}_T)] + \sum_{t=2}^{T} D_{\mathrm{KL}}[q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \| p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)] - \log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1) \right] \tag{3}$$

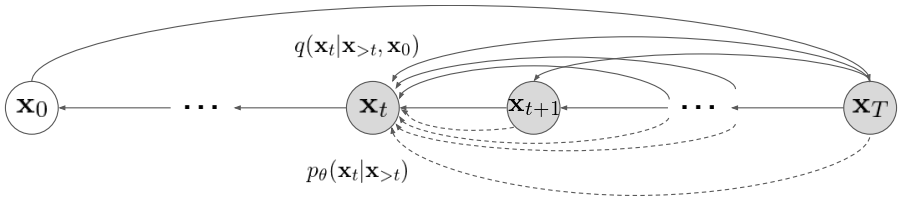

Figure 2: Probabilistic graphical model for GGDM.

and the model is parametrized as

$$p_\theta(\boldsymbol{x}_s|\boldsymbol{x}_t) = q\left(\boldsymbol{x}_s \middle| \boldsymbol{x}_t, \hat{\boldsymbol{x}}_0 = \frac{1}{f_{t0}}(\boldsymbol{x}_t - g_{t0}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t))\right) \tag{4}$$

$$= \mathcal{N}\left(\boldsymbol{x}_s \middle| \frac{1}{g_{t0}^2}(f_{s0}g_{ts}^2\boldsymbol{x}_0 + f_{ts}g_{s0}^2\boldsymbol{x}_t), \frac{g_{s0}^2 g_{ts}^2}{g_{t0}^2}\boldsymbol{I}_d\right) \tag{5}$$

for each contiguous $s < t$.

The seminal work of Song et al. (2020) presents Denoising Diffusion Implicit Models (DDIM): a family of evidence lower bounds (ELBOs) with corresponding forward diffusion processes and samplers. All of these ELBOs share the same marginals as DDPM, but allow arbitrary choices of posterior variances. Specifically, Song et al. (2020) note that is it possible to construct alternative ELBOs with only a subsequence of the original timesteps $S \subset \{1, ..., T\}$ that shares the same marginals as the construction above (i.e., $q_S(\boldsymbol{x}_t|\boldsymbol{x}_0) = q(\boldsymbol{x}_t|\boldsymbol{x}_0)$ for every $t \in S$, so q_S defines a faster sampler compatible with the pre-trained model) by simply using the new contiguous timesteps in the equations above. They also show it is also possible to construct an *infinite* family of non-Markovian processes $\{q_\sigma : \sigma \in [0, 1]^{T-1}\}$ where each q_σ also shares the same marginals as the original forward process with:

$$q_\sigma(\boldsymbol{x}_0, ..., \boldsymbol{x}_t) = q(\boldsymbol{x}_0)q(\boldsymbol{x}_T|\boldsymbol{x}_0)\prod_{t=1}^{T-1} q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_{t+1}, \boldsymbol{x}_0) \tag{6}$$

and where the posteriors are defined as

$$q(\boldsymbol{x}_s|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| f_{s0}\boldsymbol{x}_0 + \frac{\sqrt{g_{s0}^2 - \sigma_t^2}}{g_{t0}}(\boldsymbol{x}_t - f_{t0}\boldsymbol{x}_0), \sigma_t^2\boldsymbol{I}_d\right) \tag{7}$$

Song et al. (2020) empirically find that the extreme case of using all-zero variances (*a.k.a.* DDIM($\eta = 0$)) consistently helps with sample quality in the few-step regime. Combined with a good selection of timesteps to evaluate the modeled score function (*a.k.a. strides*), DDIM($\eta = 0$) establishes the current state-of-the-art for few-step diffusion model sampling with the smallest inference step budgets.

3 Generalized Gaussian Diffusion Models

We now present Generalized Gaussian Diffusion Model (), a family of Gaussian diffusion processes that includes DDIM as a special case. We define a joint distribution with no independence assumptions

$$q_{\mu,\sigma}(\boldsymbol{x}_0, ..., \boldsymbol{x}_T) = q(\boldsymbol{x}_0)q(\boldsymbol{x}_T|\boldsymbol{x}_0)\prod_{t=1}^{T-1} q_{\mu,\sigma}(\boldsymbol{x}_t|\boldsymbol{x}_{>t}, \boldsymbol{x}_0) \tag{8}$$

where the new factors are defined as

$$q_{\mu,\sigma}(\boldsymbol{x}_t|\boldsymbol{x}_{>t}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \sum_{u \in S_t} \mu_{tu}\boldsymbol{x}_u, \sigma_t^2\boldsymbol{I}_d\right) \tag{9}$$

(letting $S_t = \{0, ..., T\} \setminus \{1, ..., t\}$ for notation compactness), with σ_t and μ_{tu} free parameters $\forall t \in \{1, ..., T\}, u \in S_t$. In other words, when predicting the next, less noisy image, the sampler can

take into account *all* the previous, noisier images in the sampling chain, and similarly to DDIM, we can also control the sampler's variances. As we prove in the appendix (A.2), this construction admits Gaussian marginals, and we can differentiably compute the marginal coefficients from arbitrary choices of μ and σ:

Theorem 1. Given some $t \in \{1, ..., T\}$, let $a_{tu}^{(1)} = \mu_{tu}\ \forall u \in S_t$ and $v_t^{(1)} = \sigma_t^2$. For each $i \in \{1, ..., T-t\}$, recursively define

$$a_{tu}^{(i+1)} = a_{t,t+i}^{(i)} \mu_{t+i,u} + a_{tu}^{(i)}\ \forall u \in S_{t+i} \quad \text{and} \quad v_t^{(i+1)} = v_t^{(i)} + \left(a_{t,t+i}^{(i)} \sigma_{t+i}\right)^2.$$

Then, it follows that

$$q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{>t+i}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \sum_{u \in S_{t+i}} a_{tu}^{(i+1)} \boldsymbol{x}_u, v_t^{(i+1)} \boldsymbol{I}_d\right). \tag{10}$$

In other words, instead of letting f_{t0} and g_{t0} (or equivalently, each of the $f_{t,t-1}$ and $g_{t,t-1}$, i.e., the β_t coefficients in Ho et al. (2020)) *define* the forward process as done by a DDPM, the GGDM family lets the μ_{tu} and σ_t define the process, and consequently, $f_{t0} = \sum_{u \in S_{t+i}} a_{tu}^{(i+1)} \boldsymbol{x}_u$, and $g_{t0}^2 = v_t^{(i+1)}$. The reverse process is thus defined as $p(\boldsymbol{x}_T) \prod_{t=1}^{T} p(\boldsymbol{x}_{t-1} | \boldsymbol{x}_t)$ with $p(\boldsymbol{x}_T) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d))$ and

$$p_\theta(\boldsymbol{x}_t | \boldsymbol{x}_{>t}) = q_{\mu,\sigma}\left(\boldsymbol{x}_t | \boldsymbol{x}_{>t}, \hat{\boldsymbol{x}}_0 = \frac{1}{f_{t0}}(\boldsymbol{x}_t - g_{t0} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t))\right). \tag{11}$$

The corresponding ELBO is

$$\mathbb{E}_q\left[\log p_\theta(\boldsymbol{x}_0 | \boldsymbol{x}_{>0}) - \sum_{t=1}^{T-1} D_{\mathrm{KL}}[q(\boldsymbol{x}_t | \boldsymbol{x}_{>t}, \boldsymbol{x}_0) \| p_\theta(\boldsymbol{x}_t | \boldsymbol{x}_{>t})] - D_{\mathrm{KL}}[q(\boldsymbol{x}_T | \boldsymbol{x}_0) \| p(\boldsymbol{x}_T)]\right] \tag{12}$$

where we recover the usual connection to DSM:

$$D_{\mathrm{KL}}[q(\boldsymbol{x}_t | \boldsymbol{x}_{>t}, \boldsymbol{x}_0) \| p_\theta(\boldsymbol{x}_t | \boldsymbol{x}_{>t})] = \frac{\mu_{t0}^2 g_{t0}^2}{2 f_{t0}^2 \sigma_t^2} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \right\|^2. \tag{13}$$

3.1 Ignoring the matching marginals condition

Unlike DDIM, the GGDM family does not guarantee that the marginals of the new forward process match that of the original DDPM. We empirically find, however, that this condition can often be too restrictive and better samplers exist where the marginals $q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| a_{t0}^{(T-t+1)} \boldsymbol{x}_0, v_t^{(T-t+1)} \boldsymbol{I}_d\right)$ of the new forward process differ from the original DDPM's marginals. We verify this empirically by applying DDSS to both the family of DDIM sigmas and DDPM variances: both sampler families have the same number of parameters (the reverse process variances), but the latter does not adjust the mean coefficients like DDIM to ensure matching marginals and still achieves similar or better scores than the former across sample quality metrics (and even outperforms the DDIM($\eta = 0$) baseline); see Section 5.2.

4 Differentiable Diffusion Sampler Search

We now describe DDSS, our approach to search for high-fidelity samplers with a limited budget of $K < T$ steps. We observe we can backpropagate through the diffusion sampling process via the reparamterization trick. Equipped with this, we can now use gradient-descent to optimize any given differentiable loss function over model samples.

We begin with a pre-trained DDPM and a family of K-step samplers that we wish to search for this DDPM. We parametrize this family's degrees of freedom as simple transformations of trainable variables. We experiment with the following families in this paper, but emphasize that DDSS is applicable to any other family where model samples are differentiable with respect to the trainable variables:

- **DDIM**: we parametrize the posterior variances with $\sigma_t = g_{s0}\ \mathrm{sigmoid}(v_t)$, where $v_1, ..., v_K$ are trainable variables (the g_{s0} constant ensures real-valued mean coefficients; see the square root in Equation 7).
- **DDPM-G**: we parametrize the marginal variances with $g_{t0} = \mathrm{cumsum}(\mathrm{softmax}([\boldsymbol{v}; 1]))_t$, ensuring they are monotonically increasing with respect to t (concatenating a one to ensure K degrees of freedom).
- **GGDM**: we parametrize all the μ_{tu} and σ_t as sigmoids of trainable variables.
- **GGDM +PRED**: we parametrize all the μ_{tu} and σ_t identically to GGDM, but also learn the marginal coefficients with a $\mathrm{cumsum} \circ \mathrm{softmax}$ parametrization (identical to DDPM-G) instead of computing them via Theorem 1, as well as the coefficients that predict $\boldsymbol{x}_0$ from $a_t \boldsymbol{x}_t - b_t \boldsymbol{\epsilon}$ with $1 + \mathrm{softplus}$ and $\mathrm{softplus}$ parametrizations, instead of using $\frac{1}{f_{t0}}$ and $\frac{g_{t0}}{f_{t0}}$.
- **[family]+TIME**: for any sampler family, we additionally parametrize the timesteps used to call the pre-trained score model with a $\mathrm{cumsum} \circ \mathrm{softmax}$ parametrization (identical to DDPM-G).

As we will show in our experiments, despite the fact that out pre-trained DDPMs were trained with discrete timesteps, we find that learning the timesteps is still helpful. In principle, this should only be possible for DDPMs trained with continuous time sampling (Chen et al., 2021a; Song et al., 2021b; Kingma et al., 2021), but in practice we find that DDPMs trained with continuously embedded discrete timesteps are still well-behaved when applied at a timestep not present during training. We hypothesize that this is due to the regularity of the sinusoidal positional encodings (following Vaswani et al. (2017)) used by these models when trained with a sufficiently large number of timesteps T.

4.1 Perceptual Loss

We can find our sampler in this family by optimizing the desired search family's trainable variables; however, the question of which objective to use remains. Prior work has shown that optimizing log-likelihood can actually worsen sample quality and FID scores in the few-step regime (Watson et al., 2021; Song et al., 2021a). Thus, we instead design a *perceptual* loss which simply compares mean statistics between the DDPM's samples against real samples in neural network feature space. These types of objectives have been shown in prior work to prefer minima that better correlate with human perception of sample quality (Johnson et al., 2016; Heusel et al., 2017), which we also observe in our experiments. Specifically, we rely on the representations of the penultimate layer of a pre-trained InceptionV3 classifier (Szegedy et al., 2016), and we optimize the Kernel Inception Distance (KID) (Bińkowski et al., 2018). Letting the inception features of an image be $\phi(\boldsymbol{x})$ and ψ the collection of all the introduced trainable variables, our exact objective is the unbiased estimator of the squared maximum mean discrepancy (MMD) used by Bińkowski et al. (2018):

$$\mathcal{L}_{\text{KID}} = \left\| \mathop{\mathbb{E}}_{\boldsymbol{x}_p \sim p_\psi(\boldsymbol{x}_p)} f^*(\boldsymbol{x}_p) - \mathop{\mathbb{E}}_{\boldsymbol{x}_q \sim q(\boldsymbol{x}_0)} f^*(\boldsymbol{x}_q) \right\|_2^2 \tag{14}$$

$$\approx \frac{1}{n(n-1)} \sum_{i \neq j}^{n} \left(k_\phi(\boldsymbol{x}_p^{(i)}, \boldsymbol{x}_p^{(j)}) + k_\phi(\boldsymbol{x}_q^{(i)}, \boldsymbol{x}_q^{(j)}) \right) - \frac{2}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k_\phi(\boldsymbol{x}_p^{(i)}, \boldsymbol{x}_q^{(j)}) \tag{15}$$

where $p_\psi(\boldsymbol{x}_p)$ is the marginal distribution of the final generated sample of the sampler family parametrized by ψ, $q(\boldsymbol{x}_0)$ is the distribution of real data, k is any differentiable, positive definite kernel, $k_\phi(\boldsymbol{x}, \boldsymbol{y}) = k(\phi(\boldsymbol{x}), \phi(\boldsymbol{y}))$ and $f^*(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{x}'_p \sim p_\psi(\boldsymbol{x}_p)} k_\phi(\boldsymbol{x}, \boldsymbol{x}'_p) - \mathbb{E}_{\boldsymbol{x}'_q \sim q(\boldsymbol{x}_0)} k_\phi(\boldsymbol{x}, \boldsymbol{x}'_q)$ is the witness function attaining the supremum in the MMD.

Since the sampling chain of any Gaussian diffusion process admits using the reparametrization trick (Kingma & Welling, 2013), our loss function is fully differentiable with respect to the trainable variables ψ. We empirically find that using the perceptual features is crucial; i.e., by trying $\phi(\boldsymbol{x}) = \boldsymbol{x}$ to compare images directly on pixel space rather than neural network feature space (as above), we observe that our method makes samples consistently worsen in apparent quality as training progresses.

Table 1: FID and IS scores for DDSS against baseline methods for a DDPM trained on CIFAR10 with the L_{simple} objective proposed by (Ho et al., 2020). FID scores (lower is better) are the numbers at the left of each entry, and IS scores (higher is better) are at the right.

Sampler \ K	5	10	15	20	25
DDPM (linear stride)	84.27 / 5.396	43.39 / 7.034	31.40 / 7.609	25.94 / 7.879	22.60 / 8.043
DDPM (quadratic stride)	76.25 / 5.435	42.03 / 6.965	27.78 / 7.714	20.21 / 8.128	16.17 / 8.350
DDIM (linear stride)	44.41 / 6.750	19.11 / 7.965	14.06 / 8.190	11.82 / 8.420	10.52 / 8.512
DDIM (quadratic stride)	32.66 / 7.090	13.62 / 8.190	9.318 / 8.495	7.500 / 8.641	6.560 / 8.759
GGDM +PRED+TIME	**13.77 / 8.520**	**8.227 / 8.903**	**6.115 / 9.050**	**4.722 / 9.261**	**4.250 / 9.186**

Table 2: FID / IS scores for DDSS against baseline methods for a DDPM trained on ImageNet 64x64 with the L_{hybrid} objective proposed by Nichol & Dhariwal (2021).

Sampler \ K	5	10	15	20	25
DDPM (linear stride)	122.0 / 5.878	58.78 / 10.67	39.30 / 13.22	31.36 / 14.72	26.36 / 15.71
DDPM (quadratic stride)	394.8 / 1.351	129.5 / 5.997	80.10 / 9.595	61.34 / 11.60	49.60 / 13.01
DDIM (linear stride)	135.4 / 5.898	40.70 / 12.23	28.54 / 13.99	24.23 / 14.75	22.13 / 15.16
DDIM (quadratic stride)	409.1 / 1.380	148.6 / 5.533	67.65 / 9.842	45.60 / 11.99	36.11 / 13.23
GGDM +PRED+TIME	**55.14 / 12.90**	**37.32 / 14.76**	**24.69 / 17.23**	**20.69 /17.92**	**18.40 / 18.12**

4.2 Gradient Rematerialization

In order for backpropagation to be feasible under reasonable memory constraints, one final problem must be addressed: since we are taking gradients with respect to model samples, the cost in memory to maintain the state of the forward pass scales linearly with the number of inference steps, which can quickly become unfeasible considering the large size of typical DDPM architectures. To address this issue, we use gradient rematerialization (Kumar et al., 2019b). Instead of storing a particular computation's output from the forward pass required by the backward pass computation, we recompute it on the fly. To trade $\mathcal{O}(K)$ memory cost for $\mathcal{O}(K)$ computation time, we simply rematerialize calls to the pre-trained DDPM (i.e., the estimated score function), but keep in memory all the progressively denoised images from the sampling chain. In JAX (Bradbury et al., 2018), this is trivial to implement by simply wrapping the score function calls with `jax.remat`.

5 Experiments

In order to emphasize that our method is compatible with any pre-trained DDPM, we apply our method on pre-trained DDPM checkpoints from prior work. Specifically, we experiment with the DDPM trained by Ho et al. (2020) with L_{simple} on CIFAR10, as well as a DDPM following the exact configuration of Nichol & Dhariwal (2021) trained on ImageNet 64x64 (Deng et al., 2009) with their L_{hybrid} objective (with the only difference being that we trained the latter ourselves for 3M rather than 1.5M steps). Both of these models utilize adaptations of the UNet architecture (Ronneberger et al., 2015) that incorporate self-attention layers (Vaswani et al., 2017).

We evaluate all of our models on both FID and Inception Score (IS) (Salimans et al., 2016), comparing the samplers discovered by DDSS against DDPM and DDIM baselines with linear and quadratic strides. As previously mentioned, more recent methods for fast sampling are outperformed by DDIM when the budget of inference steps is as small as those we utilize in this work (5, 10, 15, 20, 25). All reported results on both of these approximate sample quality metrics were computed by comparing 50K model and training data samples, as is standard in the literature. Also as is standard, IS scores are computed 10 times, each time on 5K samples, and then averaged.

In all of our experiments, we optimize the DDSS objective presented in Section 4.1 with the following hyperparameters:

1. For every family of models we search over, we initialize the degrees of freedom such that training begins with a sampler matching DDPM with K substeps following Song et al. (2020); Nichol & Dhariwal (2021).

Table 3: FID / IS scores for the KID kernel ablation on CIFAR10. When not learning the timesteps, we fix them to a quadratic stride, as Table 1 shows this performs best on CIFAR10.

Sampler $\setminus$ K	5	10	15	20	25
DDSS (linear kernel)					
GGDM +PRED+TIME	13.77 / 8.520	8.227 / 8.903	6.115 / 9.050	**4.722** / **9.261**	**4.250** / **9.186**
GGDM +PRED	14.26 / 8.406	8.617 / 8.842	5.939 / 9.035	4.893 / 9.153	4.574 / 9.145
GGDM +TIME	12.85 / 8.383	**7.858** / 8.895	6.265 / **9.075**	5.367 / 9.136	4.887 / 9.229
GGDM)	14.45 / 8.281	8.154 / 8.892	7.045 / 8.939	5.477 / 9.183	4.815 / 9.189
DDSS (cubic kernel)					
GGDM +PRED+TIME	14.41 / **8.527**	8.217 / **9.007**	**5.895** / 9.036	4.932 / 9.092	4.278/ **9.286**
GGDM +PRED	14.39 / 8.401	8.977 / 8.870	6.517 / 8.970	4.915 / 9.132	4.471 / 9.247
GGDM +TIME	**12.35** / 8.406	7.879 / 8.852	6.682 / 8.999	5.639 / 9.058	4.631 / 9.189
GGDM	14.57 / 8.297	8.232 / 8.836	6.727 / 8.904	5.569 / 9.177	4.668 / 9.192

2. We apply gradient updates using the Adam optimizer (Kingma & Ba, 2015). We sweeped over the learning rate and used $\lambda = 0.0005$. We did not sweep over other Adam hyperparameters and kept $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1 \times 10^{-8}$.
3. We tried batch sizes of 128 and 512 and opted for the latter, finding that it leads to better sample quality upon inspection. Since the loss depends on averages over examples as our experiments are on unconditional generation, this choice was expected.
4. We run all of our experiments for 50K training steps and evaluate the discovered samplers at this exact number of training steps. We did not sweep over this value.

We include our main results in Table 1 for CIFAR10 and Table 2 for ImageNet 64x64, comparing DDSS applied to GGDM +PRED+TIME against DDPM and DDIM baselines with linear and quadratic strides. All models use a linear kernel, i.e., $k_\phi(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{x})^\top \phi(\boldsymbol{y})$, which we found to perform slightly better than the cubic kernel used by Bińkowski et al. (2018) (we ablate this in section 5.1). We omit the use of the learned variances of the ImageNet 64x64 model (i.e., following Nichol & Dhariwal (2021)), as we search for the variances ourselves via DDSS. We include samples for 5, 10 and 25 steps comparing the strongest DDIM baselines to DDSS + GGDM with a learned stride; see Figures 1 and 3. We include additional ImageNet 64x64 samples in the appendix (A.1).

5.1 Ablations for KID kernel and GGDM variants

As our approach is compatible with any choice of KID kernel, we experiment with different choices of kernels. Namely, we try the simplest possible linear kernel, $k_\phi(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{x})^\top \phi(\boldsymbol{y})$, as well as the cubic kernel $k_\phi(\boldsymbol{x}, \boldsymbol{y}) = \left(\frac{1}{d}\phi(\boldsymbol{x})^\top \phi(\boldsymbol{y}) + 1\right)^3$ used by Bińkowski et al. (2018). We compare the performance of these two kernels, as well as different variations of GGDM (i.e., with and without TIME and PRED as defined in Section 4). Results are included for CIFAR10 across all budgets in Table 3. We also include a smaller version of this ablation for ImageNet 64x64 in the appendix (A.3).

The results in this ablation show that the contributions of the linear kernel, timestep learning, and the empirical choice of learning the coefficients that predict $\boldsymbol{x}_0$ all slightly contribute to better FID and IS scores. Importantly, however, removing any of these additions still allows us to comfortably outperform the strongest baselines.

5.2 Search space ablation

Now, in order to further demonstrate the key importance of optimizing our GGDM family to find high-fidelity samplers, we also apply DDSS to the less general DDIM and DDPM-G families. We show that, while we still attain better scores than a regular DDPM, searching these less flexible families of samplers does not yield improvements as significant as with out novel GGDM family. In particular, optimizing the DDIM sigma coefficients does not outperform the corresponding DDIM($\eta = 0$) baseline on CIFAR10, which is not a surprising result as Song et al. (2020) show empirically that most choices of the σ_t degrees of freedom lead to worse FID scores than setting them all to 0. These results also show that optimizing the DDPM-G can outperform DDSS applied

Table 4: FID / IS scores for the DDSS search space ablation on CIFAR10. All runs fix the timesteps to a quadratic stride and use a linear kernel except for the last row (we only include the GGDM results for ease of comparison).

Sampler $\setminus K$	5	10	15	20	25
DDIM($\eta = 0$)	32.66 / 7.090	13.62 / 8.190	9.318 / 8.495	7.500 / 8.641	6.560 / 8.759
DDSS					
DDPM-G	33.08 / 7.096	15.33 / 8.559	9.693 / 8.845	7.297 / 8.924	6.172 / 9.057
DDIM	32.61 / 7.084	16.29 / 7.966	11.31 / 8.372	9.120 / 8.563	7.853 / 8.644
GGDM	14.45 / 8.281	8.154 / 8.892	7.045 / 8.939	5.477 / 9.183	4.815 / 9.189
GGDM +PRED+TIME	13.77 / 8.520	8.227 / 8.903	6.115 / 9.050	4.722 / 9.261	4.250 / 9.186

to the DDIM family, and even the strongest DDIM($\eta = 0$) baselines for certain budgets, justifying our choice of not enforcing the marginals to match (as discussed in Section 3.1).

6 Discussion

When applied to a sufficiently flexible family (such as the GGDM family proposed in this work), DDSS consistently finds samplers that achieve better image generation quality than the strongest baselines in the literature for very few steps. This is qualitatively apparent in non-cherrypicked samples (*e.g.,* DDIM($\eta = 0$) tends to generate blurrier images and with less background details as the budget decreases), and multiple quantitative sample quality metrics (FID and IS) also reflect these results. Still, we observe limitations to our method. Finding samplers with inference step budgets as small as $K < 10$ that have little apparent loss in quality remains challenging with our proposed search family. And, while on CIFAR10 the metrics indicate significant relative improvement over sample quality metrics, the relative improvement on ImageNet 64x64 is less pronounced. We hypothesize that this is an inherent difficulty of ImageNet due to its high diversity of samples, and that in order to retain sample quality and diversity, it might be impossible to escape some minimum number of inference steps with score-based models as they might be crucial to mode-breaking.

Beyond the empirical gains of applying our procedure, our findings shed further light into properties of pre-trained score-based generative models. First, we show that without fine-tuning a DDPM's parameters, these models are already capable of producing high-quality samples with very few inference steps, though the default DDPM sampler in this regime is usually suboptimal when using a few-step sampler. We further show that better sampling paths exist, and interestingly, these are determined by alternative variational lower bounds to the data distribution that make use of the score-based model but do not necessarily share the same marginals as the original DDPM forward process. Our findings thus suggest that enforcing this marginal-sharing constraint is unnecessary and can be too restrictive in practice.

7 Other Related Work

Besides DDIM (Song et al., 2020), there have been more recent attempts at reducing the number of inference steps for DDPMs. Jolicoeur-Martineau et al. (2021) proposed a dynamic step size SDE solver that can reduce the number of calls to the modeled score function to ~ 150 on CIFAR10 (Krizhevsky et al., 2009) with minimal cost in FID scores, but quickly falls behind DDIM($\eta = 0$) with as many as 50 steps. Watson et al. (2021) proposed a dynamic programming algorithm that chooses log-likelihood optimal strides, but find that log-likelihood reduction has a mismatch with FID scores, particularly with in the very few step regime, also falling behind DDIM($\eta = 0$) in this front. Other methods that have been shown to help sample quality in the few-step regime include non-Gaussian variants of diffusion models (Nachmani et al., 2021) and adaptively adjusting the sampling path by introducing a noise level estimating network (San-Roman et al., 2021), but more thorough evaluation of sample quality achieved by these approaches is needed with budgets as small as those considered in this work.

Other approaches to sampling DDPMs have also been recently proposed, though not for the explicit purpose of efficient sampling. Song et al. (2021b) derive a reverse SDE that, when discretized, uses different coefficients than the ancestral samplers considered in this work. The same authors also

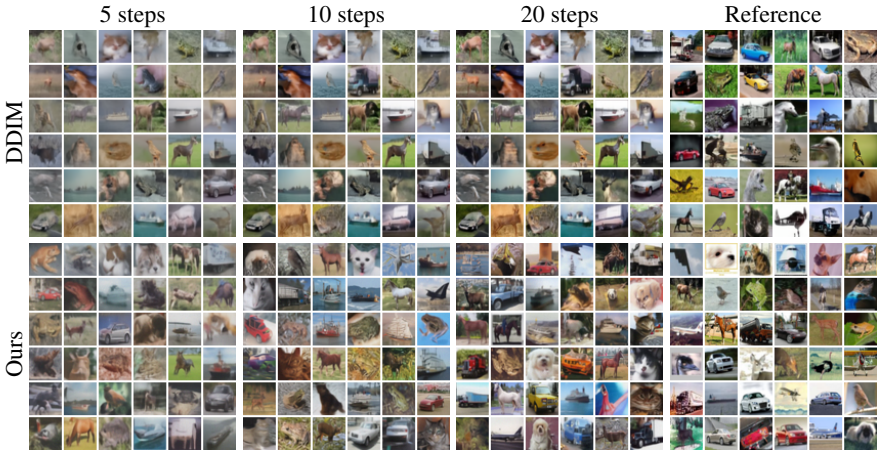

Figure 3: Non-cherrypicked samples for a DDPM trained on CIFAR10, comparing the strongest DDIM($\eta = 0$) baseline and our approach. All samples were generated with the same random seeds. For reference, we include DDPM samples using all 1,000 stpes (top right) and real images (bottom right).

derive "corrector" steps, which introduce additional calls to the pre-trained DDPM as a form of gradient ascent (Langevin dynamics) that help with quality but introduce computation cost, as well as an alternative sampling procedure using a probability flow ODE that shares the same marginals as the DDPM's original forward process. Huang et al. (2021) generalize this family of samplers to a "plug-in reverse SDE" that interpolates between a probability flow ODE and the reverse SDE, similarly to how the DDIM η interpolates between an implicit probabilistic model and a stochastic reverse process. Our proposed search family includes discretizations of most of these cases for Gaussian processes, notably missing corrector steps, where reusing a single timestep is considered.

8 Conclusion and Future Work

We propose Differentiable Diffusion Sampler Search (DDSS), a method for finding few-step samplers for Denoising Diffusion Probabilistic Models. We show how to optimize a perceptual loss over a space of diffusion processes that makes use of a pre-trained DDPM's samples by leveraging the reparametrization trick and gradient rematerialization. Our results qualitatively and quantitatively show that DDSS is able to significantly improve sample quality for unconditional image generation over prior methods on efficient DDPM sampling. The success of our method hinges on searching a novel, wider family of Generalized Gaussian Diffusion Model (GGDM) than those identified in prior work (Song et al., 2020). DDSS does not fine-tune the pre-trained DDPM, only needs to be applied once, has few hyperparameters, and does not require re-training the DDPM.

Our findings suggest future directions to further reduce the number of inference steps while retaining high fidelity in generated samples. For instance, it is plausible to use different representations for the perceptual loss instead of those of a classifier, *e.g.,* use representations from an unsupervised model such as SimCLR (Chen et al., 2020), to using internal representations learned by the pre-trained DDPM itself, which would eliminate the burden of additional computation. Moreover, considering the demonstrated benefits of applying DDSS to our proposed GGDM family of samplers (as opposed to narrower families like DDIM), we motivate future work on identifying more general families of samplers and investigating whether they help uncover even better samplers or lead to overfitting. Finally, identifying other variants of perceptual losses (*e.g.,* that do not sample from the model), or alternative optimization strategies (*e.g.,* gradient-free methods) that lead to similar results is important future work. This would make DDSS itself a more efficient procedure, as gradient-based optimization of our proposed loss requires extensive memory or computation requirements to back-propagate through the whole sampling chain.

REPRODUCIBILITY STATEMENT

We plan to open source our work for the camera-ready version of the paper. Still, we provide sufficient details in the paper to completely reproduce our work, and the datasets and model checkpoints we experiment with are already publicly available. We include a theorem that shows how to compute the coefficients of the marginals of GGDM (Equation 10), which is a necessary and the most non-trivial step to reproduce our results. We also include a proof in the paper, immediately after the theorem statement. Additionally, we indicate the reader how to implement key ideas in the paper, *e.g.,* in Section 4.1 we provide pseudocode showing exactly how we parametrize learnable, monotonically increasing timesteps, and in Section 4.2 we explain how to implement gradient rematerialization with JAX. Finally, our experiments section is thorough on all the hyperparameters we use, and any exceptions such as early stopping are always documented where appropriate.

ETHICS STATEMENT

We acknowledge that generative models have potentially dangerous applications, and that making them more efficient (as we do in our work) allows more people to utilize them. Still, methods outside the scope of this work like GANs are already much more efficient and can still achieve high-fidelity samples. Bias is also an important concern when working on such image generative models, more work needs to be done to investigate and mitigate bias of our samplers. We do not believe that our research will be directly used for harmful purposes, nor that its release or open sourcing will pose any risk to the community.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. In *arXiv*, 2017.

Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning Gradient Fields for Shape Generation. In *ECCV*, 2020.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, and William Chan. WaveGrad: Estimating Gradients for Waveform Generation. In *ICLR*, 2021a.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, Najim Dehak, and William Chan. WaveGrad 2: Iterative Refinement for Text-to-Speech Synthesis . In *INTERSPEECH*, 2021b.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big Self-Supervised Models are Strong Semi-Supervised Learners. In *NeurIPS*, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *arXiv preprint arXiv:2105.05233*, 2021.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

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. *arXiv preprint arXiv:1706.08500*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *NeurIPS*, 2020.

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *arXiv preprint arXiv:2106.15282*, 2021.

Chin-Wei Huang, Jae Hyun Lim, and Aaron Courville. A variational perspective on diffusion-based generative models and score matching. *arXiv preprint arXiv:2106.02808*, 2021.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pp. 694–711. Springer, 2016.

Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080*, 2021.

Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *ICLR*, 2013.

Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, and Aaron Courville. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis. In *NeurIPS*, 2019a.

Ravi Kumar, Manish Purohit, Zoya Svitkina, Erik Vee, and Joshua R Wang. Efficient rematerialization for deep networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 15172–15181, 2019b.

Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2837–2845, 2021.

Eliya Nachmani, Robin San Roman, and Lior Wolf. Non gaussian denoising diffusion models. *arXiv preprint arXiv:2106.07582*, 2021.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29: 2234–2242, 2016.

Robin San-Roman, Eliya Nachmani, and Lior Wolf. Noise estimation for generative diffusion models. *arXiv preprint arXiv:2104.02600*, 2021.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. *NeurIPS*, 2019.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *arXiv e-prints*, pp. arXiv–2101, 2021a.

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*, 2021b.

Markus Svensén and Christopher M Bishop. Pattern recognition and machine learning, 2007.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *CVPR*, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NIPS*, 2017.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to efficiently sample from diffusion probabilistic models. *arXiv preprint arXiv:2106.03802*, 2021.

A APPENDIX

A.1 ADDITIONAL IMAGENET 64X64 SAMPLES

We provide additional samples for our results on ImageNet 64x64. The DDIM($\eta = 0$) samples (left) use a linear stride, while our DDSS + GGDM samples (right) use a learned stride. For reference, we include DDPM samples with all 4,000 steps (bottom left) and real samples (bottom middle).

A.2 PROOF FOR THEOREM 1

Theorem 1. Given some $t \in \{1, ..., T\}$, let $a_{tu}^{(1)} = \mu_{tu}\ \forall u \in S_t$ and $v_t^{(1)} = \sigma_t^2$. For each $i \in \{1, ..., T-t\}$, recursively define

- $a_{tu}^{(i+1)} = a_{t,t+i}^{(i)} \mu_{t+i,u} + a_{tu}^{(i)}\ \forall u \in S_{t+i}$
- $v_t^{(i+1)} = v_t^{(i)} + \left(a_{t,t+i}^{(i)} \sigma_{t+i}\right)^2$

Then, it follows that

$$q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{>t+i}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \sum_{u \in S_{t+i}} a_{tu}^{(i+1)} \boldsymbol{x}_u, v_t^{(i+1)} \boldsymbol{I}_d\right)$$

Proof. Let us prove this result with mathematical induction. Note that, for each such t, we have by definition that

$$q_{\mu,\sigma}(\boldsymbol{x}_{t+1} | \boldsymbol{x}_{>t+1}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+1}} \mu_{t+1,u} \boldsymbol{x}_u, \sigma_{t+1}^2 \boldsymbol{I}_d\right)$$

and

$$q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{t+1}, \boldsymbol{x}_{>t+1}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_t} \mu_{tu} \boldsymbol{x}_u, \sigma_t^2 \boldsymbol{I}_d\right) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_t} a_{tu}^{(1)} \boldsymbol{x}_u, v_t^{(1)} \boldsymbol{I}_d\right)$$

Therefore, following Svensén & Bishop (2007) (2.115), by prior conjugacy it follows that

$$\begin{aligned}
q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{>t+1}, \boldsymbol{x}_0) &= \mathcal{N}\left(\boldsymbol{x}_t \middle| a_{t,t+1}^{(1)} \textstyle\sum_{u \in S_{t+1}} \mu_{t+1,u} \boldsymbol{x}_u + \sum_{u \in S_{t+1}} a_{tu}^{(1)}, \left(v_t^{(1)} + a_{t,t+1}^{(1)} \sigma_{t+1}^2 a_{t,t+1}^{(1)}\right) \boldsymbol{I}_d\right) \\
&= \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+1}} \left(a_{t,t+1}^{(1)} \mu_{t+1,u} + a_{tu}^{(1)}\right) \boldsymbol{x}_u, v_t^{(2)} \boldsymbol{I}_d\right) \\
&= \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+1}} a_{tu}^{(2)} \boldsymbol{x}_u, v_t^{(2)} \boldsymbol{I}_d\right).
\end{aligned}$$

This proves the base case for our induction argument. Now, let us prove the inductive step. Suppose there exists some integer $j \in \{1, ..., T-t+1\}$ such that

$$q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{>t+j}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+j}} a_{tu}^{(j+1)} \boldsymbol{x}_u, v_t^{(j+1)} \boldsymbol{I}_d\right).$$

By definition, we already know $q(\boldsymbol{x}_{t+j+1} | \boldsymbol{x}_{>t+j+1}, \boldsymbol{x}_0)$, so we have

$$q_{\mu,\sigma}(\boldsymbol{x}_{t+j+1} | \boldsymbol{x}_{>t+j+1}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_{t+j+1} \middle| \textstyle\sum_{u \in S_{t+j+1}} \mu_{t+j+1,u} \boldsymbol{x}_u, \sigma_{t+j+1}^2 \boldsymbol{I}_d\right)$$

and (rewriting the inductive hypothesis)

$$q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{t+j+1}, \boldsymbol{x}_{>t+j+1}, \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+j}} a_{tu}^{(j+1)} \boldsymbol{x}_u, v_t^{(j+1)} \boldsymbol{I}_d\right).$$

Therefore, by prior conjugacy again, it follows that

$$\begin{aligned}
&q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_{>t+j+1}, \boldsymbol{x}_0) \\
=&\mathcal{N}\left(\boldsymbol{x}_t \middle| a_{t,t+j+1}^{(j+1)} \textstyle\sum_{u \in S_{t+j}} \mu_{t+j+1,u} \boldsymbol{x}_u + \sum_{u \in S_{t+j}} a_{tu}^{(j+1)}, \left(v_t^{(j+1)} + a_{t,t+j+1}^{(j+1)} \sigma_{t+j+1}^2 a_{t,t+j+1}^{(j+1)}\right) \boldsymbol{I}_d\right) \\
=&\mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+j}} \left(a_{t,t+j+1}^{(j+1)} \mu_{t+j+1,u} + a_{tu}^{(j+1)}\right) \boldsymbol{x}_u, v_t^{(j+2)} \boldsymbol{I}_d\right) \\
=&\mathcal{N}\left(\boldsymbol{x}_t \middle| \textstyle\sum_{u \in S_{t+j+1}} a_{tu}^{(j+2)} \boldsymbol{x}_u, v_t^{(j+2)} \boldsymbol{I}_d\right).
\end{aligned}$$

This concludes the proof of the inductive step. Hence, we have proven the result for any $i \in \{1, ..., T-t\}$. In particular,

$$q_{\mu,\sigma}(\boldsymbol{x}_t | \boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \middle| a_{t0}^{(T-t+1)} \boldsymbol{x}_0, v_t^{(T-t+1)} \boldsymbol{I}_d\right). \quad \square$$

A.3 ADDITIONAL ABLATION OF KID KERNEL AND GGDM VARIANTS FOR IMAGENET 64X64

We also ran a smaller version of the ablation results presented in Section 5.1, but for ImageNet 64x64 instead of CIFAR10, as these are more computationally intensive to do a full grid search. Results for a step budget $K = 15$ are included below. When not learning the timesteps, we fix them to a linear stride, as Table 2 shows this performs best on ImageNet 64x64.

Sampler $\setminus$ K	15
DDSS (linear kernel)	
GGDM +PRED+TIME	**24.69** / 17.23
GGDM +PRED	27.08 / 16.44
GGDM +TIME	25.73 / **17.27**
GGDM	28.34 / 16.63
DDSS (cubic kernel)	
GGDM +PRED+TIME	26.52 / 16.29
GGDM +PRED	27.82 / 16.3
GGDM +TIME	26.87 / 16.99
GGDM	28.83 / 16.32