# Fast Sampling of Diffusion Models with Exponential Integrator

**Anonymous Author(s)**
Affiliation
Address
`email`

## Abstract

Our goal is to develop a fast sampling method for Diffusion models (DMs) with a small number of steps while retaining high sample quality. To achieve this, we systematically analyze the sampling procedure in DMs and identify key factors that affect the sample quality, among which the method of discretization is most crucial. By carefully examining the learned diffusion process, we propose Diffusion Exponential Integrator Sampler (DEIS). It is based on the Exponential Integrator designed for discretizing ordinary differential equations (ODEs) and leverages a semilinear structure of the learned diffusion process to reduce the discretization error. The proposed method can be applied to any DMs and can generate high-fidelity samples in as few as 10 steps. By directly using pre-trained DMs, we achieve superior sampling performance when the number of score function evaluation (NFE) is limited, e.g., 4.17 FID with 10 NFEs, 2.86 FID with only 20 NFEs on CIFAR10.

## 1 Introduction

The Diffusion model (DM) [10] is a generative modeling method developed recently that relies on the basic idea of reversing a given simple diffusion process. A time-dependent score function is learned for this purpose and DMs are thus also known as score-based models [27]. Compared with other generative models such as generative adversarial networks (GANs), in addition to great scalability, the DM has the advantage of stable training; it avoids mode-collapsing and is not hyperparameter sensitive [6, 18]. DMs have recently achieved impressive performances on a variety of tasks, including unconditional image generation [10, 27, 24, 8], text conditioned image generation [22, 23], text generation [12, 2], 3D point cloud generation [21], inverse problem [16, 29], etc.

However, the remarkable performance of DMs comes at the cost of extremely slow sampling; it takes much longer time to produce high-quality samples compared with GANs. See Appendix A for discussions on existing methods. The objective of this work is to establish a principled discretization scheme for the learned backward diffusion processes in DMs so as to achieve fast sampling. Since the most expensive part in sampling a DM is the evaluation of the neural network that parameterizes the backward diffusion, we seek a discretization method that requires a small number of network function evaluation (NFE). We start with a family of marginal equivalent SDEs/ODEs associated with DMs and investigate numerical error sources, which include fitting error and discretization error. We observe that even with the same trained model, different discretization schemes can have dramatically different performances in terms of discretization error. We find out that the *Exponential Integrator (EI)* [11] that utilizes the semilinear structure of the backward diffusion has minimum error. To further reduce the discretization error, we propose to either use high order polynomials to approximate the nonlinear term in the ODE or employ Runge Kutta methods on a transformed ODE.

**Figure 1:** Generated images with various DMs. Latent diffusion [24] (Left), $256 \times 256$ image with text *A shirt with inscription "World peace"* (15 NFE). VE diffusion [27] (Mid), FFHQ $256 \times 256$ (12 NFE). VP diffusion [10] (Right), CIFAR10 (7 NFE) and CELEBA (5 NFE).

The resulting algorithms, termed *Diffusion Exponential Integrator Sampler (DEIS)*, achieve the best sampling quality with limited NFEs.

Our contributions are summarized as follows: 1) We investigate a family of marginal equivalent SDEs/ODEs for fast sampling and conduct a systematic error analysis for their numerical solvers. 2) We propose DEIS, an efficient sampler that can be applied to any DMs to achieve superior sampling quality with a limited number of NFEs. 3) We conduct comprehensive experiments to validate the efficacy of DEIS. For instance, with a pre-trained model [27], DEIS is able to reach 4.17 FID with 10 NFEs, and 2.86 FID with 20 NFEs on CIFAR10.

## 2 Background on Diffusion Models

A DM consists of a fixed forward diffusion (noising) process that adds noise to the data, and a learned backward diffusion (denoising) process that gradually removes the added noise.

**Forward noising diffusion**: The forward diffusion of a DM for $D$-dimensional data is a linear diffusion described by the stochastic differential equation (SDE) [25]

$$d\boldsymbol{x} = \boldsymbol{F}_t \boldsymbol{x} dt + \boldsymbol{G}_t d\boldsymbol{w}, \tag{1}$$

where $\boldsymbol{F}_t \in \mathbb{R}^{D\times D}$ denotes the linear drift coefficient, $\boldsymbol{G}_t \in \mathbb{R}^{D\times D}$ denotes the diffusion coefficient, and $\boldsymbol{w}$ is a standard Wiener process. The diffusion Eq (1) is initiated at the training data and simulated over a fixed time window $[0, T]$. Denote by $p_t(\boldsymbol{x}_t)$ the marginal distribution of $\boldsymbol{x}_t$ and by $p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ the conditional distribution from $\boldsymbol{x}_0$ to $\boldsymbol{x}_t$, then $p_0(\boldsymbol{x}_0)$ represents the underlying distribution of the training data. The simulated trajectories are represented by $\{\boldsymbol{x}_t\}_{0\leq t\leq T}$. The parameters $\boldsymbol{F}_t$ and $\boldsymbol{G}_t$ are chosen such that the conditional marginal distribution $p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ is a simple Gaussian distribution, denoted as $\mathcal{N}(\mu_t \boldsymbol{x}_0, \Sigma_t)$, and the distribution $\pi(\boldsymbol{x}_T) := p_T(\boldsymbol{x}_T)$ is easy to sample from.

**Backward denoising diffusion**: Under mild assumptions [1, 27], the forward diffusion Eq (1) is associated with a reverse-time diffusion process

$$d\boldsymbol{x} = [\boldsymbol{F}_t \boldsymbol{x} dt - \boldsymbol{G}_t \boldsymbol{G}_t^T \nabla \log p_t(\boldsymbol{x})]dt + \boldsymbol{G}_t d\boldsymbol{w}, \tag{2}$$

where $\boldsymbol{w}$ denotes a standard Wiener process in the reverse-time direction. The distribution of the trajectories of Eq (2) with terminal distribution $\boldsymbol{x}_T \sim \pi$ coincides with that of Eq (1) with initial distribution $\boldsymbol{x}_0 \sim p_0$, that is, Eq (2) matches Eq (1) in probability law.

**Training**: The basic idea of DMs is to use a time-dependent network $\boldsymbol{s}_\theta(\boldsymbol{x}, t)$, known as a score network, to approximate the score $\nabla \log p_t(\boldsymbol{x})$. This is achieved by score matching techniques [13, 31] where the score network $\boldsymbol{s}_\theta$ is trained by minimizing the denoising score matching loss

$$\mathcal{L}(\theta) = \mathbb{E}_{t\sim \mathrm{Unif}[0,T]} \mathbb{E}_{p(\boldsymbol{x}_0)p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)}[\|\nabla \log p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \boldsymbol{s}_\theta(\boldsymbol{x}_t, t)\|^2_{\Lambda_t}]. \tag{3}$$

Here $\nabla \log p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ has a closed form expression as $p_{0t}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ is a simple Gaussian distribution, and $\Lambda_t$ denotes a time-dependent weight. We refer the reader to [10, 27] for more details on choices of $\Lambda_t$ and training techniques.

## 3 Fast Sampling with learned score models

Once the score network $\boldsymbol{s}_\theta(\boldsymbol{x}, t) \approx \nabla \log p_t(\boldsymbol{x})$ is trained, it can be used to generate new samples by solving the backward SDE Eq (2) with $\nabla \log p_t(\boldsymbol{x})$ replaced by $\boldsymbol{s}_\theta(\boldsymbol{x}, t)$. It turns out there are infinitely many diffusion processes one can use. In this work, we consider a family of SDEs

$$d\hat{\boldsymbol{x}} = [\boldsymbol{F}_t \hat{\boldsymbol{x}} - \frac{1+\lambda^2}{2} \boldsymbol{G}_t \boldsymbol{G}_t^T \boldsymbol{s}_\theta(\hat{\boldsymbol{x}}, t)]dt + \lambda \boldsymbol{G}_t d\boldsymbol{w}, \tag{4}$$

parameterized by $\lambda \geq 0$. Here we use $\hat{\boldsymbol{x}}$ to distinguish the solution to the SDE associated with the learned score from the ground truth $\boldsymbol{x}$ in Eq (1) and (2). When $\lambda = 0$, Eq (4) reduces to an ODE known as the *probability flow ODE* [27]. The reverse-time diffusion Eq (2) with an approximated score is a special case of Eq (4) with $\lambda = 1$. The following Proposition (Proof in Appendix B) justifies the usage of Eq (4) for generating samples.

**Proposition 1.** *When $\boldsymbol{s}_\theta(\boldsymbol{x}, t) = \nabla \log p_t(\boldsymbol{x})$ for all $\boldsymbol{x}, t$, and $\hat{p}_T^* = \pi$, the marginal distribution $\hat{p}_t^*$ of Eq (4) matches $p_t$ of the forward diffusion Eq (1) for all $0 \leq t \leq T$.*

To generate a new sample, one can sample $\hat{\boldsymbol{x}}_T^*$ from $\pi$ and solve Eq (4) to obtain a sample $\hat{\boldsymbol{x}}_0^*$. The objective of this work is to develop an efficient sampling scheme from Eq (4) with less discretization errors. In this section, we focus on the ODE approach with $\lambda = 0$.

We investigate the discretization error of solving the probability flow ODE ($\lambda = 0$)

$$\frac{d\hat{\boldsymbol{x}}}{dt} = \boldsymbol{F}_t \hat{\boldsymbol{x}} - \frac{1}{2} \boldsymbol{G}_t \boldsymbol{G}_t^T \boldsymbol{s}_\theta(\hat{\boldsymbol{x}}, t). \tag{5}$$

Eq (5) is a semilinear stiff ODE [11]. The exact solution to this ODE is

$$\hat{\boldsymbol{x}}_t = \Psi(t, s)\hat{\boldsymbol{x}}_s + \int_s^t \Psi(t, \tau)[-\frac{1}{2} \boldsymbol{G}_\tau \boldsymbol{G}_\tau^T \boldsymbol{s}_\theta(\hat{\boldsymbol{x}}_\tau, \tau)]d\tau, \tag{6}$$

where $\Psi(t, s)$ satisfying $\frac{\partial}{\partial t}\Psi(t, s) = \boldsymbol{F}_t \Psi(t, s), \Psi(s, s) = \boldsymbol{I}$ is known as the transition matrix associated with $\boldsymbol{F}_\tau$. We study the discretization error in solving Eq (5) to reduce gap between numerical results within a small number of NFEs and exact result Eq (6).

**Exponential Integrator over Euler method.** The Euler method is an elementary explicit numerical method for ODEs and is widely used in numerical softwares [32]. When applied to Eq (5), it reads

$$\hat{\boldsymbol{x}}_{t-\Delta t} = \hat{\boldsymbol{x}}_t - [\boldsymbol{F}_t \hat{\boldsymbol{x}}_t - \frac{1}{2} \boldsymbol{G}_t \boldsymbol{G}_t^T \boldsymbol{s}_\theta(\hat{\boldsymbol{x}}_t, t)]\Delta t. \tag{7}$$

This is used in many existing works in DMs [27, 9]. This approach however has low accuracy and is sometimes unstable when the stepsize is not sufficiently small. To improve the accuracy, we propose to use the *Exponential Integrator (EI)*, a method that leverages the semilinear structure of Eq (5). When applied to Eq (5), the EI reads

$$\hat{\boldsymbol{x}}_{t-\Delta t} = \Psi(t - \Delta t, t)\hat{\boldsymbol{x}}_t + [\int_t^{t-\Delta t} -\frac{1}{2} \Psi(t - \Delta t, \tau) \boldsymbol{G}_\tau \boldsymbol{G}_\tau^T d\tau] \boldsymbol{s}_\theta(\hat{\boldsymbol{x}}_t, t). \tag{8}$$

It is effective if the nonlinear term $\boldsymbol{s}_\theta(\hat{\boldsymbol{x}}_t, t)$ does not change much along the solution. In fact, for any given $\Delta t$, Eq (8) solves Eq (5) exactly if $\boldsymbol{s}_\theta(\hat{\boldsymbol{x}}_t, t)$ is constant over the time interval $[t - \Delta t, t]$.

**Parametrization: $\epsilon_\theta(\boldsymbol{x}, t)$ over $\boldsymbol{s}_\theta(\boldsymbol{x}, t)$.** We find another source of large discretizaiton error is caused by rapidly changing score $\nabla \log p_t(\boldsymbol{x})$. It is found that the parameterization [10] $\nabla \log p_t(\boldsymbol{x}) \approx -\boldsymbol{L}_t^{-T} \epsilon_\theta(\boldsymbol{x}, t)$, where $\boldsymbol{L}_t$ can be any matrix satisfying $\boldsymbol{L}_t \boldsymbol{L}_t^T = \Sigma_t$, leads to significant improvements of accuracy. The network $\epsilon_\theta$ tries to follow $\epsilon$ which is sampled from a standard Gaussian and thus has a small magnitude. In comparison, the parameterization $\boldsymbol{s}_\theta = -\boldsymbol{L}_t^{-T} \epsilon_\theta$ can take large value as $\boldsymbol{L}_t \to 0$ as $t$ approaches 0. It is thus better to approximate $\epsilon_\theta$ than $\boldsymbol{s}_\theta$ with a neural network. We adopt this parameterization and rewrite Eq (5) as

$$\frac{d\hat{\boldsymbol{x}}}{dt} = \boldsymbol{F}_t \hat{\boldsymbol{x}} + \frac{1}{2} \boldsymbol{G}_t \boldsymbol{G}_t^T \boldsymbol{L}_t^{-T} \epsilon_\theta(\hat{\boldsymbol{x}}, t). \tag{9}$$

Applying the EI to Eq (9) yields

$$\hat{\boldsymbol{x}}_{t-\Delta t} = \Psi(t - \Delta t, t)\hat{\boldsymbol{x}}_t + [\int_t^{t-\Delta t} \frac{1}{2} \Psi(t - \Delta t, \tau) \boldsymbol{G}_\tau \boldsymbol{G}_\tau^T \boldsymbol{L}_\tau^{-T} d\tau] \epsilon_\theta(\hat{\boldsymbol{x}}_t, t). \tag{10}$$

| | FID for various DEIS | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 26.91 | $108^{+1}$ | $185^{+1}$ | $193^{+3}$ | 22.28 | 21.53 | 21.43 | 19.72 | 16.31 | **15.37** |
| 10 | 11.14 | 14.72 | $13.19^{+2}$ | $28.65^{+2}$ | 7.56 | 6.72 | 6.50 | 6.09 | 4.57 | **4.17** |
| 20 | 5.47 | 3.50 | $2.97^{+1}$ | 3.92 | 3.70 | 3.32 | 3.17 | 3.54 | 3.05 | **2.86** |
| 50 | 3.27 | 2.60 | $\mathbf{2.55}^{+1}$ | $2.57^{+2}$ | 2.70 | 2.62 | 2.59 | 2.67 | 2.59 | 2.57 |

**Table 1:** DEIS for VPSDE on CIFAR10 with limited NFE. For $\rho$RK-DEIS, the upper right number indicates extra NFEs used. Bold numbers denote the best performance achieved with similar NFE.

Then we develop several fast sampling algorithms, all coined as the *Diffusion Exponential Integrator Sampler (DEIS)*, based on Eq (10), for DMs. Interestingly, the discretization Eq (10) based on EI coincides with the popular deterministic DDIM when the forward diffusion Eq (1) is VPSDE [26]. The update Eq (10) can be further improved by using a polynomial of time, rather than a constant, to approximation $\epsilon_\theta$ over the interval $[t - \Delta t, t]$. The resulting approach resembles the classical Adams–Bashforth [11] method, thus we term it $t$**AB-DEIS**. Another major factor that affects the performance of sampling is the choice of time discretization.

## 4 Exponential Integrator: simplify probability Flow ODE

Next we present a different perspective to DEIS based on ODE transformations. The probability ODE Eq (9) can be transformed into a simple non-stiff ODE, and then off-the-shelf ODE solvers can be applied to solve the ODE effectively. To this end, we introduce variable $\hat{\boldsymbol{y}}_t := \Psi(t,0)\hat{\boldsymbol{x}}_t$ and rewrite Eq (9) into

$$\frac{d\hat{\boldsymbol{y}}}{dt} = \frac{1}{2}\Psi(t,0)\boldsymbol{G}_t\boldsymbol{G}_t^T\boldsymbol{L}_t^{-T}\epsilon_\theta(\Psi(0,t)\hat{\boldsymbol{y}}, t). \tag{11}$$

Note that, departing from Eq (9), Eq (11) does not possess semi-linear structure. Thus, we can apply off-the-shelf ODE solvers to Eq (11) without worrying about the semi-linear structure. This transformation Eq (11) can be further improved by taking into account the analytical form of $\Psi, \boldsymbol{G}_t, \boldsymbol{L}_t$. Here we present treatment for VPSDE; the results can be extended to other (scalar) DMs such as VESDE. See Appendix C for the proof.

**Proposition 2.** *For the VPSDE, with $\hat{\boldsymbol{y}}_t = \sqrt{\frac{\alpha_0}{\alpha_t}}\hat{\boldsymbol{x}}_t$ and the time-scaling $\beta(t) = \sqrt{\alpha_0}(\sqrt{\frac{1-\alpha_t}{\alpha_t}} - \sqrt{\frac{1-\alpha_0}{\alpha_0}})$, Eq (9) can be transformed into*

$$\frac{d\hat{\boldsymbol{y}}}{d\rho} = \epsilon_\theta(\sqrt{\frac{\alpha_{\beta^{-1}(\rho)}}{\alpha_0}}\hat{\boldsymbol{y}}, \beta^{-1}(\rho)), \quad \rho \in [\beta(0), \beta(T)]. \tag{12}$$

Based on the transformed ODE Eq (12), we propose two variants of the DEIS algorithm: $\rho$**RK-DEIS** when applying classical RK methods, and $\rho$**AB-DEIS** when applying Adams-Bashforth methods. We remark that the difference between $t$AB-DEIS and $\rho$AB-DEIS lies in the fact that $t$AB-DEIS fits polynomials in $t$ which may not be polynomials in $\rho$.

## 5 Experiments and Conclusion

We conduct experiment on CIFAR10 with pre-trained model from [27]. As in Fig 2, compared with other samplers, ODE sampler based on RK45 [27], SDE samplers based on Euler-Maruyama (EM) [27] and SDE adaptive step size solver [14], DEIS can converge much faster. We also include comaprison among various DEIS algorithms in Tab 1 (See Appendix D for more experiment results). In summary, by utilizing structural information in Eq (5), DEIS can significantly accelerate DM sampling.

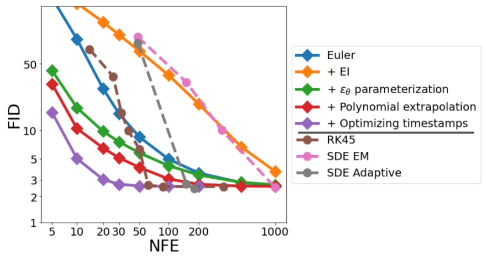

**Figure 2:** The effects of EI, parametrization, polynomial approximation, and time discretization.

## References

[1] Anderson, B. D. Reverse-time diffusion equation models. Stochastic Process. Appl., 12 (3):313–326, May 1982. ISSN 0304-4149. doi: 10.1016/0304-4149(82)90051-5. URL https://doi.org/10.1016/0304-4149(82)90051-5.

[2] Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. Advances in Neural Information Processing Systems, 34:17981–17993, 2021.

[3] Bao, F., Li, C., Zhu, J., and Zhang, B. Analytic-DPM: An Analytic Estimate of the Optimal Reverse Variance in Diffusion Probabilistic Models. 2022. URL http://arxiv.org/abs/2201.06503.

[4] Chen, T., Liu, G.-H., and Theodorou, E. A. Likelihood training of schrödinger bridge using forward-backward sdes theory. arXiv preprint arXiv:2110.11291, 2021.

[5] Chen, Y., Georgiou, T. T., and Pavon, M. Stochastic control liaisons: Richard sinkhorn meets gaspard monge on a schrodinger bridge. SIAM Review, 63(2):249–313, 2021.

[6] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. Generative adversarial networks: An overview. IEEE Signal Processing Magazine, 35(1):53–65, 2018.

[7] De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. Diffusion schrödinger bridge with applications to score-based generative modeling. Advances in Neural Information Processing Systems, 34, 2021.

[8] Dhariwal, P. and Nichol, A. Diffusion Models Beat GANs on Image Synthesis. 2021. URL http://arxiv.org/abs/2105.05233.

[9] Dockhorn, T., Vahdat, A., and Kreis, K. Score-Based Generative Modeling with Critically-Damped Langevin Diffusion. pp. 1–13, 2021. URL http://arxiv.org/abs/2112.07068.

[10] Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In Advances in Neural Information Processing Systems, volume 2020-Decem, 2020. ISBN 2006.11239v2. URL https://github.com/hojonathanho/diffusion.

[11] Hochbruck, M. and Ostermann, A. Exponential integrators. Acta Numerica, 19:209–286, 2010.

[12] Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. Argmax flows and multinomial diffusion: Learning categorical distributions. Advances in Neural Information Processing Systems, 34, 2021.

[13] Hyvärinen, A. Estimation of non-normalized statistical models by score matching. J. Mach. Learn. Res., 6:695–708, 2005. ISSN 15337928.

[14] Jolicoeur-Martineau, A., Li, K., Piché-Taillefer, R., Kachman, T., and Mitliagkas, I. Gotta go fast when generating data with score-based models. arXiv preprint arXiv:2105.14080, 2021.

[15] Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. arXiv preprint arXiv:2206.00364, 2022.

[16] Kawar, B., Vaksman, G., and Elad, M. Snips: Solving noisy inverse problems stochastically. Advances in Neural Information Processing Systems, 34, 2021.

[17] Kim, D., Shin, S., Song, K., Kang, W., and Moon, I.-C. Score matching model for unbounded data score. arXiv preprint arXiv:2106.05527, 2021.

[18] Kingma, D. P. and Welling, M. An introduction to variational autoencoders. arXiv preprint arXiv:1906.02691, 2019.

[19] Kong, Z. and Ping, W. On fast sampling of diffusion probabilistic models. arXiv preprint arXiv:2106.00132, 2021.

[20] Liu, L., Ren, Y., Lin, Z., and Zhao, Z. Pseudo Numerical Methods for Diffusion Models on Manifolds. (2021):1–23, 2022. URL http://arxiv.org/abs/2202.09778.

[21] Lyu, Z., Kong, Z., Xu, X., Pan, L., and Lin, D. A conditional point diffusion-refinement paradigm for 3d point cloud completion. arXiv preprint arXiv:2112.03530, 2021.

[22] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. 2021. URL http://arxiv.org/abs/2112.10741.

[23] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents, 2022.

[24] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-Resolution Image Synthesis with Latent Diffusion Models. 2021. URL http://arxiv.org/abs/2112.10752.

[25] Särkkä, S. and Solin, A. Applied stochastic differential equations, volume 10. Cambridge University Press, 2019.

[26] Song, J., Meng, C., and Ermon, S. Denoising Diffusion Implicit Models. 2020. URL http://arxiv.org/abs/2010.02502.

[27] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-Based Generative Modeling through Stochastic Differential Equations. ArXiv preprint, abs/2011.13456, 2020. URL https://arxiv.org/abs/2011.13456.

[28] Song, Y., Durkan, C., Murray, I., and Ermon, S. Maximum likelihood training of score-based diffusion models. Advances in Neural Information Processing Systems, 34, 2021.

[29] Song, Y., Shen, L., Xing, L., and Ermon, S. Solving inverse problems in medical imaging with score-based generative models. arXiv preprint arXiv:2111.08005, 2021.

[30] Vargas, F., Thodoroff, P., Lamacraft, A., and Lawrence, N. Solving Schrödinger bridges via maximum likelihood. Entropy, 23(9):1134, 2021.

[31] Vincent, P. A connection between score matching and denoising autoencoders. Neural Comput., 23(7):1661–1674, July 2011. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco_a_00142. URL https://doi.org/10.1162/neco_a_00142.

[32] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[33] Wang, G., Jiao, Y., Xu, Q., Wang, Y., and Yang, C. Deep generative learning via schrödinger bridge. In International Conference on Machine Learning, pp. 10794–10804. PMLR, 2021.

[34] Watson, D., Ho, J., Norouzi, M., and Chan, W. Learning to efficiently sample from diffusion probabilistic models. arXiv preprint arXiv:2106.03802, 2021.

[35] Whalen, P., Brio, M., and Moloney, J. Exponential time-differencing with embedded Runge–Kutta adaptive step control. J. Comput. Phys., 280:579–601, January 2015. ISSN 0021-9991. doi: 10.1016/j.jcp.2014.09.038. URL https://doi.org/10.1016/j.jcp.2014.09.038.

[36] Zhang, Q. and Chen, Y. Diffusion normalizing flow. Advances in Neural Information Processing Systems, 34, 2021.

## A Related works

A lot of research has been conducted to speed up the sampling of DMs. In [19, 34] the authors optimize denosing process by modifying the underlying stochastic process. However, such acceleration can not generate high quality samples with a small number of discretization steps. In [26] the authors use a non-Markovian forward noising. The resulted algorihtm, DDIM, achieves significant acceleration than DDPMs. More recently, the authors of [3] optimize the backward Markovian process to approximate the non-Markovian forward process and get an analytic expression of optimal variance in denoising process. Another strategy to make the forward diffusion nonlinear and trainable [36, 30, 7, 33, 4] in the spirit of Schrödinger bridge [5]. This however comes with a heavy training overhead.

More closely related to our method is [20], which interprets update step in stochastic DDIM as a combination of gradient estimation step and transfer step. It modifies high order ODE methods to provide an estimation of the gradient and uses DDIM for transfer step. However, the decomposition of DDIM into two separate components is not theoretically justified. Based on our analysis on Exponential Integrator, Liu et al. [20] uses Exponential Integral but with a Euler discretization-based approximation of the nonlinear term. This approximation is inaccurate and may suffer large discretization error if the step size is large.

The semilinear structure presented in probability flow ODE has been widely investigated in physics and numerical simulation [11, 35], from which we get inspirations. The stiff property of the ODEs requires more efficient ODE solvers instead of black-box solvers that are designed for general ODE problems. In this work, we investigate sovlers for differential equations in diffusion model and take advantage of the semilinear structure.

## B Proof of Proposition 1

The proof is inspired by [36]. We show that the marginal distribution induced by Eq (4) does not depend on the choice of $\lambda$ and equals the marginal distribution induced by Eq (2) when the score model is perfect.

Consider the distribution $q$ induced by the SDE

$$d\boldsymbol{x} = [\boldsymbol{F}_t\boldsymbol{x} - \frac{1+\lambda^2}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla\log q_t(\boldsymbol{x})]dt + \lambda\boldsymbol{G}_t d\boldsymbol{w}. \tag{13}$$

Eq (13) is simulated from $t = T$ to $t = 0$. According to the Fokker-Planck-Kolmogorov (FPK) Equation, $q$ solves the partial differential equation

$$\begin{aligned}\frac{\partial q_t(\boldsymbol{x})}{\partial t} &= -\nabla\cdot\{[\boldsymbol{F}_t\boldsymbol{x} - \frac{1+\lambda^2}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla\log q_t(\boldsymbol{x})]q_t(\boldsymbol{x})\} - \frac{\lambda^2}{2}\langle\boldsymbol{G}_t\boldsymbol{G}_t^T, \frac{\partial^2}{\partial x_i\partial x_j}q_t(\boldsymbol{x})\rangle \\ &= -\nabla\cdot\{[\boldsymbol{F}_t\boldsymbol{x} - \frac{1}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla\log q_t(\boldsymbol{x})]q_t(\boldsymbol{x})\} + \nabla\cdot\{[\frac{\lambda^2}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla\log q_t(\boldsymbol{x})]q_t(\boldsymbol{x})\} - \\ &\quad \frac{\lambda^2}{2}\langle\boldsymbol{G}_t\boldsymbol{G}_t^T, \frac{\partial^2}{\partial x_i\partial x_j}q_t(\boldsymbol{x})\rangle,\end{aligned}$$

where $\nabla\cdot$ denotes the divergence operator. Since

$$\nabla\cdot\{[\frac{\lambda^2}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla\log q_t(\boldsymbol{x})]q_t(\boldsymbol{x})\} = \nabla\cdot[\frac{\lambda^2}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla q_t(\boldsymbol{x})] = \langle\frac{\lambda^2}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T, \frac{\partial^2}{\partial x_i\partial x_j}q_t(\boldsymbol{x})\rangle, \tag{14}$$

we obtain

$$\frac{\partial q_t(\boldsymbol{x})}{\partial t} = -\nabla\cdot\{[\boldsymbol{F}_t\boldsymbol{x} - \frac{1}{2}\boldsymbol{G}_t\boldsymbol{G}_t^T\nabla\log q_t(\boldsymbol{x})]q_t(\boldsymbol{x})\}. \tag{15}$$

Eq (15) shows that the above partial differential equation does not depend on $\lambda$. Thus, the marginal distribution of Eq (13) is independent of the value of $\lambda$.

## C Proof of Proposition 2

We start our proof with Eq (11). In VPSDE, Eq (11) reduce to

$$\frac{d\hat{\boldsymbol{y}}}{dt} = -\frac{1}{2}\sqrt{\frac{\alpha_0}{\alpha_t}}\frac{d\log\alpha_t}{dt}\frac{1}{\sqrt{1-\alpha_t}}\epsilon_\theta(\Psi(0,t)\hat{\boldsymbol{y}}, t). \tag{16}$$

Now we consider a rescaled time $\rho$, which satisfies the following equation

$$\frac{d\rho}{dt} = -\frac{1}{2}\sqrt{\frac{\alpha_0}{\alpha_t}}\frac{d\log\alpha_t}{dt}\frac{1}{\sqrt{1-\alpha_t}}. \tag{17}$$

Plugging Eq (17) into Eq (16), we reach

$$\frac{d\hat{\boldsymbol{y}}}{d\rho} = \epsilon_\theta(\Psi(0,t)\hat{\boldsymbol{y}}, t). \tag{18}$$

In VPSDE, we $\alpha_t$ is a monotonically decreasing function with respect to $t$. Therefore, there exists a bijective mapping between $\rho$ and $t$ based on Eq (17), which we define as $\beta$ and $\rho = \beta(t)$. Furthermore, we can solve Eq (17) for $\beta$

$$\beta(t) = \sqrt{\alpha_0}\left(\sqrt{\frac{1-\alpha_t}{\alpha_t}} - \sqrt{\frac{1-\alpha_0}{\alpha_0}}\right). \tag{19}$$

## D More experiment details

### D.1 Important technical details and modifications

- It is found that correcting steps and an extra denoising step can improve image quality at additional NFE costs [27, 14]. For a fair comparison, we disable the correcting steps, extra denoising step, or other heuristic clipping tricks for all methods and experiments in this work unless stated otherwise.
- Due to numerical issues, we set ending time $t_0$ in DMs during sampling a non-zero number. Song et al. [27] suggests $t_0 = 10^{-3}$ for VPSDE and $t_0 = 10^{-5}$ for VESDE. In practice, we find the value of $t_0$ and time scheduling have huge impacts on FIDs. This finding is not new and has been pointed out by existing works [14, 17, 26]. Interestingly, we found different algorithms have different preferences for $t_0$ and time scheduling. We report the best FIDs for each method among different choices of $t_0$ and time scheduling in Tab 1. We use $t_0$ suggested by the original paper and codebase for different checkpoints and quadratic time scheduling suggested by Song et al. [26] unless stated otherwise. We include a comprehensive study about $t_0$ and time scheduling in Appendix D.3
- Because PNDM needs 12 NFE for the first 3 steps, we compare PNDM only when NFE is great than 12. However, our proposed iPNDM can work when NFE is less than 12.
- We include the comparison against A-DDIM [3] with its official checkpoints and implementation in Appendix D.5.
- We only provide qualitative results for text-to-image experiment with pre-trained model [23].
- We include proposed $r$-th order iPNDM in Appendix D.2. We use $r = 3$ by default unless stated otherwise.

### D.2 Improved PNDM

By Eq (10), PNDM can be viewed as a combination of Exponential Integrator and linear multistep method based on the Euler method. More specifically, it uses a linear combination of multiple score evaluations instead of using only the latest score evaluation. PNDM follows the steps

$$\hat{\epsilon}_t^{(3)} = \frac{1}{24}(55\epsilon_t - 59\epsilon_{t+\Delta t} + 37\epsilon_{t+2\Delta t} - 9\epsilon_{t+3\Delta t}), \tag{20}$$

$$\hat{\boldsymbol{x}}_{t-\Delta t} = \sqrt{\frac{\alpha_{t-\Delta t}}{\alpha_t}}\hat{\boldsymbol{x}}_t + [\sqrt{1-\alpha_{t-\Delta t}} - \sqrt{\frac{\alpha_{t-\Delta t}}{\alpha_t}}\sqrt{1-\alpha_t}]\hat{\epsilon}_t^{(4)}, \tag{21}$$

where $\epsilon_t = \epsilon_\theta(\hat{\boldsymbol{x}}_t, t), \epsilon_{t+\Delta t} = \epsilon_\theta(\hat{\boldsymbol{x}}_{t+\Delta t}, t+\Delta t)$. The coefficients in Eq (20) are derived based on black-box ODE Euler discretization with fixed step size. Similarly, there exist lower order approximations

$$\hat{\epsilon}_t^{(0)} = \epsilon_t \tag{22}$$

$$\hat{\epsilon}_t^{(1)} = \frac{3}{2}\epsilon_t - \frac{1}{2}\epsilon_{t+\Delta t} \tag{23}$$

$$\hat{\epsilon}_t^{(2)} = \frac{1}{12}(23\epsilon_t - 16\epsilon_{t+\Delta t} + 5\epsilon_{t+2\Delta t}). \tag{24}$$

**Table 2:** Comparison between PNDM and $t$AB-DEIS. The bold results denote the best result with a fixed NFE on each dataset. The advantage of DEIS is obvious when NFE is small.

| FID \ NFE / Method | 5 | 10 | 20 | 50 |
|---|---|---|---|---|
| PNDM | - | - | 6.42 | 3.03 |
| iPNDM | 70.07 | 9.36 | 4.21 | 3.00 |
| DDIM | 30.64 | 11.71 | 6.12 | 4.25 |
| $t$AB1 | 20.01 | 6.09 | 3.81 | 3.32 |
| $t$AB2 | 16.53 | 4.57 | 3.41 | 3.09 |
| $t$AB3 | **16.10** | **4.17** | **3.33** | **2.99** |

**(a)** CIFAR10

| FID \ NFE / Method | 5 | 10 | 20 | 50 |
|---|---|---|---|---|
| PNDM | - | - | 7.60 | 3.51 |
| iPNDM | 59.87 | 7.78 | 5.58 | 3.34 |
| 0-DEIS | 30.42 | 13.53 | 6.89 | 4.17 |
| 1-DEIS | 26.65 | 8.81 | 4.33 | 3.19 |
| 2-DEIS | 25.13 | 7.20 | 3.61 | 3.04 |
| 3-DEIS | **25.07** | **6.95** | **3.41** | **2.95** |

**(b)** CELEBA

Originally, PNDM uses Runge-Kutta for warming start and costs 4 score network evaluation for each of the first 3 steps. To reduce the NFE in sampling, the improved PNDM (iPNDM) uses lower order multistep for warming start. We summarize iPNDM in Alg 1. We include a comparison with $t$AB-DEIS in Tab 2, we adapt uniform step size for $t$AB-DEIS when NFE=50 in CIFAR10 as we find its performance is slightly better than the quadratic one.

**Algorithm 1** Improved PNDM (iPNDM)

```
Input: $\{t_i\}_{i=0}^N, t_i = i\Delta t$, order $r$
Instantiate: $\boldsymbol{x}_{t_N}$, Empty $\epsilon$-buffer
for $i$ in $N, N-1, \cdots, 1$ do
    $j = \min(N - i + 1, r)$
    $\epsilon$-buffer.append($\epsilon_\theta(\hat{\boldsymbol{x}}_{t_i}, t_i)$)
    Simulate $\hat{\epsilon}_{t_i}^{(j)}$ based on $j$ and $\epsilon$-buffer
    $\hat{\boldsymbol{x}}_{t_{i-1}} \leftarrow$ Simulate Eq (21) with $\hat{\boldsymbol{x}}_{t_i}$ and $\hat{\epsilon}_{t_i}^{(j)}$
end for
```

## D.3 Impact of $t_0$ and time scheduling on FIDs

We present a study about sampling with difference $t_0$ and time scheduling based VPSDE. We consider two choices of $t_0$ $(10^{-3}, 10^{-4})$ and three choices for time scheduling. The first time scheduling follows the power function in $t$

$$t_i = (\frac{N-i}{N} t_0^{\frac{1}{\kappa}} + \frac{i}{N} t_N^{\frac{1}{\kappa}})^\kappa, \tag{25}$$

the second time scheduling follows power function in $\rho$

$$\rho_i = (\frac{N-i}{N} \rho_0^{\frac{1}{\kappa}} + \frac{i}{N} \rho_N^{\frac{1}{\kappa}})^\kappa, \tag{26}$$

and the last time scheduling follows a uniform step in $\log \rho$ space

$$\log \rho_i = \frac{N-i}{N} \log \rho_0 + \frac{i}{N} \log \rho_N. \tag{27}$$

We include the comparison between different $t_0$ and time scheduling in Tab 3 to 5. We notice $t_0$ has a huge influence on image FIDs, which is also noticed and investigated across different studies [17, 9]. Among various scheduling, we observe $t$AB-DEIS has obvious advantages when NFE is small and $\rho$RK-DEIS is competitive when we NFE is relatively large.

## D.4 More abalation study

We include more quantitative comparisons of the introduced ingredients in Tab 6 for Fig 2. Since ingredients $\epsilon_\theta$-based parameterization and polynomial extrapolation are only compatible with the

|  | FID for various DEIS with $\kappa = 1$ in Eq (25) |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 47.59 | $207^{+1}$ | $238^{+1}$ | $212^{+3}$ | 35.14 | 32.51 | 32.02 | 25.99 | **25.06** | 44.29 |
| 10 | 16.60 | 84.55 | $66.81^{+2}$ | $78.57^{+2}$ | 10.47 | 8.85 | 8.18 | 9.51 | 7.71 | **7.18** |
| 15 | 10.39 | $46.36^{+1}$ | 47.45 | $41.27^{+1}$ | 6.69 | 5.70 | 5.24 | 6.47 | 5.51 | **5.01** |
| 20 | 7.93 | 34.87 | $28.35^{+1}$ | 27.21 | 5.27 | 4.56 | 4.24 | 5.20 | 4.50 | **4.14** |
| 50 | 4.36 | 11.58 | $7.00^{+1}$ | $7.48^{+2}$ | 3.32 | 3.08 | **2.99** | 3.32 | 3.09 | **2.99** |

|  | FID for various DEIS with $\kappa = 2$ in Eq (25) |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 30.64 | $256^{+1}$ | $357^{+1}$ | $342^{+3}$ | 24.58 | 23.60 | 23.48 | 20.01 | 16.52 | **16.10** |
| 10 | 11.71 | 56.62 | $56.51^{+2}$ | $103^{+2}$ | 7.56 | 6.72 | 6.50 | 6.09 | 4.57 | **4.17** |
| 15 | 7.67 | $10.62^{+1}$ | 14.96 | $36.15^{+1}$ | 4.93 | 4.40 | 4.26 | 4.29 | 3.57 | **3.37** |
| 20 | 6.11 | 6.33 | $4.74^{+1}$ | 12.81 | 4.16 | 3.84 | 3.77 | 3.81 | 3.41 | **3.33** |
| 50 | 4.24 | 3.88 | $3.75^{+1}$ | $3.78^{+2}$ | 3.70 | 3.68 | 3.69 | 3.62 | **3.61** | 3.36 |

|  | FID for various DEIS with $\kappa = 3$ in Eq (25) |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 34.07 | $356^{+1}$ | $388^{+1}$ | $377^{+3}$ | 29.80 | 29.35 | 29.38 | 24.87 | 22.57 | **22.06** |
| 10 | 14.59 | 115 | $171^{+2}$ | $267^{+2}$ | 10.73 | 10.16 | 10.11 | 8.11 | 6.36 | **5.97** |
| 15 | 9.22 | $32.94^{+1}$ | 77.44 | $103^{+1}$ | 6.45 | 6.03 | 5.98 | 5.21 | 4.26 | **4.05** |
| 20 | 7.27 | 13.06 | $11.55^{+1}$ | 50.56 | 5.17 | 4.83 | 4.78 | 4.45 | 3.88 | **3.75** |
| 50 | 4.64 | 3.76 | **3.68**$^{+1}$ | $3.74^{+2}$ | 3.92 | 3.82 | 3.79 | 3.81 | 3.72 | 3.71 |

|  | FID for various DEIS with Eq (27) |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 54.58 | $216^{+1}$ | $335^{+1}$ | $313^{+3}$ | 49.25 | 48.56 | 48.47 | 37.99 | 28.45 | **26.11** |
| 10 | 20.03 | 14.72 | $13.19^{+2}$ | $28.65^{+2}$ | 14.05 | 12.63 | 12.18 | 10.36 | 7.03 | **5.71** |
| 15 | 11.99 | $5.03^{+1}$ | 5.88 | $6.88^{+1}$ | 7.72 | 6.67 | 6.29 | 6.22 | 4.69 | **4.13** |
| 20 | 8.92 | 4.12 | $3.97^{+1}$ | 4.14 | 5.79 | 5.05 | 4.78 | 4.97 | 4.10 | **3.80** |
| 50 | 5.05 | **3.67** | $3.75^{+1}$ | $3.73^{+2}$ | 4.01 | 3.84 | 3.79 | 3.89 | 3.74 | 3.72 |

**Table 3:** DEIS for VPSDE on CIFAR10 with $t_0 = 10^{-3}$.

exponential integrator, we cannot combine them with the Euler method. We also provide performance when applying quadratic timestamp scheduling to Euler Tab 7 directly. We find sampling with small NFE and large NFE have different preferences for time schedules.

We also report the performance of the RK45 ODE solver for VPSDE on CIFAR10 in Tab 8 [1]. As a popular and well-developed ODE solver, RK45 has decent sampling performance when NFE $\geq 50$. However, the sampling quality with limited NFE is not satisfying. Such results are within expectation as RK45 does not take advantage of the structure information of diffusion models. The overall performance of RK45 solver is worse than iPNDM and DEIS when NFE is small.

### D.5 Comparison with Analytic-DDIM (A-DDIM) [3]

We also compare our algorithm with Analytic-DDIM (A-DDIM) in terms of fast sampling performance. We failed to reproduce the significant improvements claimed in [3] in our default CIFAR10 checkpoint. There could be two factors that contribute to this. First, we use a score network trained with continuous time loss objective and different weights [27]. However, Analytic-DDIM is proposed for DDPM with discrete times and finite timestamps. Second, some tricks have huge impacts on the sampling quality in A-DDIM. For instance, A-DDIM heavily depends on clipping value in the last few steps [3]. A-DDIM does not provide high-quality samples without proper clipping when NFE is low.

[1] We use $scipy.integrate.solve\_ivp$ and tune tolerance to get different performances on different NFE. We find different combinations of absolute tolerance and relative tolerance may result in the same NFE but different FID. We report the best FID in that case.

| NFE | | | | | FID for various DEIS with $\kappa = 1$ in Eq (25) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 42.38 | $239^{+1}$ | $232^{+1}$ | $199^{+3}$ | 33.09 | 31.06 | 30.67 | 26.01 | **20.57** | 42.35 |
| 10 | 17.23 | 143 | $95.13^{+2}$ | $130^{+2}$ | 12.44 | 11.04 | 10.41 | 12.01 | 10.57 | **8.04** |
| 15 | 12.06 | $99.76^{+1}$ | 77.37 | $88.56^{+1}$ | 9.12 | 8.25 | 7.79 | 9.08 | 8.20 | **7.50** |
| 20 | 9.71 | 82.89 | $57.54^{+1}$ | 66.61 | 7.57 | 6.89 | 6.50 | 7.60 | 6.90 | **6.45** |
| 50 | 5.76 | 31.56 | $13.10^{+1}$ | $15.73^{+2}$ | 4.64 | 4.25 | **4.06** | 4.67 | 4.28 | 4.10 |
| | | | | | FID for various DEIS with $\kappa = 2$ in Eq (25) | | | | | |
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 26.91 | $271^{+1}$ | $362^{+1}$ | $348^{+3}$ | 22.28 | 21.53 | 21.43 | 19.72 | 16.31 | **15.37** |
| 10 | 11.14 | 66.25 | $63.53^{+2}$ | $111^{+2}$ | 7.65 | 6.89 | 6.67 | 6.74 | 5.49 | **5.02** |
| 15 | 7.06 | $13.48^{+1}$ | 17.15 | $44.83^{+1}$ | 4.69 | 4.16 | 3.99 | 4.38 | 3.78 | **3.50** |
| 20 | 5.47 | 6.62 | $4.15^{+1}$ | 15.14 | 3.70 | 3.32 | 3.17 | 3.57 | 3.19 | **3.03** |
| 50 | 3.27 | 2.65 | **2.55**$^{+1}$ | $2.57^{+2}$ | 2.70 | 2.62 | 2.59 | 2.70 | 2.61 | 2.59 |
| | | | | | FID for various DEIS with $\kappa = 3$ in Eq (25) | | | | | |
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 32.11 | $364^{+1}$ | $393^{+1}$ | $383^{+3}$ | 28.87 | 28.58 | 28.62 | 25.78 | 23.66 | **23.38** |
| 10 | 13.18 | 135 | $199^{+2}$ | $298^{+2}$ | 9.89 | 9.38 | 9.33 | 7.74 | 6.20 | **5.77** |
| 15 | 7.92 | $42.04^{+1}$ | 99.64 | $122^{+1}$ | 5.41 | 4.99 | 4.91 | 4.48 | 3.65 | **3.37** |
| 20 | 5.92 | 17.05 | $16.66^{+1}$ | 64.40 | 4.04 | 3.69 | 3.60 | 3.54 | 3.05 | **2.86** |
| 50 | 3.36 | 2.77 | $2.57^{+1}$ | $2.71^{+2}$ | 2.73 | 2.63 | 2.60 | 2.67 | 2.59 | **2.57** |
| | | | | | FID for various DEIS with Eq (27) | | | | | |
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 54.85 | $230^{+1}$ | $382^{+1}$ | $370^{+3}$ | 51.94 | 51.62 | 51.58 | 43.84 | 39.91 | **38.76** |
| 10 | 19.80 | 23.35 | $25.08^{+2}$ | $82.17^{+2}$ | 14.63 | 13.43 | 13.07 | 11.14 | 7.78 | **6.02** |
| 15 | 11.29 | $5.63^{+1}$ | 7.46 | $8.90^{+1}$ | 7.31 | 6.28 | 5.90 | 5.89 | 4.35 | **3.71** |
| 20 | 7.91 | 3.84 | $3.05^{+1}$ | 4.14 | 4.91 | 4.19 | 3.91 | 4.23 | 3.35 | **3.00** |
| 50 | 3.82 | 2.60 | **2.56**$^{+1}$ | $2.59^{+2}$ | 2.86 | 2.70 | 2.64 | 2.79 | 2.63 | 2.58 |

**Table 4:** DEIS for VPSDE on CIFAR10 with $t_0 = 10^{-4}$ in Eq (25)

| | | | | | FID for various DEIS with $\kappa = 7$ in Eq (26) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| NFE | DDIM | $\rho$2Heun | $\rho$3Kutta | $\rho$4RK | $\rho$AB1 | $\rho$AB2 | $\rho$AB3 | $t$AB1 | $t$AB2 | $t$AB3 |
| 5 | 53.20 | $108^{+1}$ | $185^{+1}$ | $193^{+3}$ | 47.56 | 46.36 | 46.13 | 36.98 | 28.76 | **25.76** |
| 10 | 18.99 | 18.75 | $20.27^{+2}$ | $54.92^{+2}$ | 13.38 | 11.84 | 11.21 | 10.92 | 8.26 | **6.87** |
| 15 | 10.91 | $4.89^{+1}$ | 6.31 | $9.79^{+1}$ | 6.90 | 5.86 | 5.42 | 6.12 | 4.86 | **4.33** |
| 20 | 7.81 | 3.50 | **2.97**$^{+1}$ | 3.92 | 4.84 | 4.10 | 3.80 | 4.48 | 3.69 | 3.38 |
| 50 | 3.84 | 2.60 | **2.58**$^{+1}$ | $2.59^{+2}$ | 2.86 | 2.69 | 2.64 | 2.82 | 2.66 | 2.61 |

**Table 5:** DEIS for VPSDE on CIFAR10 with $t_0$ and time scheduling suggested by Karras et al. [15]

To compare with A-DDIM, we conduct another experiment with checkpoints provided by [3] and integrate iPNDM and DEIS into the provided codebase; the results are shown in Tab 9. We use piecewise linear function to fit discrete SDE coefficients in [3] for DEIS. Without any ad-hoc tricks, the plugin-and-play iPNDM is comparable or even slightly better than A-DDIM when the NFE budget is small, and DEIS is better than both of them.

## D.6 Sampling quality on ImageNet $32 \times 32$

We conduct experiments on ImageNet $32 \times 32$ with pre-trained VPSDE model provided in [28]. Again, we observe significant improvement over DDIM and iPNDM methods when the NFE budget

| | FID with various NFE | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | 5 | 10 | 20 | 30 | 50 | 100 | 200 | 500 | 1000 |
| Euler | 246.16 | 90.52 | 27.38 | 14.99 | 8.46 | 4.96 | 3.54 | 2.81 | 2.62 |
| + EI | 283.67 | 216.47 | 137.20 | 100.74 | 68.03 | 37.93 | 18.81 | 6.66 | 3.69 |
| $+\epsilon_\theta$ | 42.38 | 17.23 | 9.71 | 7.56 | 5.76 | 4.24 | 3.37 | 2.83 | 2.67 |
| + Poly | 30.67 | 10.41 | 6.50 | 5.13 | 4.06 | 3.07 | 2.69 | 2.58 | 2.57 |
| + Opt $\{t_i\}$ | 15.37 | 5.02 | 3.03 | 2.70 | 2.59 | 2.57 | 2.56 | 2.56 | 2.56 |

**Table 6:** Quantitative comparison in Fig 2 for introduced ingredients, Exponential Integrator (EI), $\epsilon_\theta$-based score parameterization, polynomial extrapolation, and optimizing time discretization $\{t_i\}$, where we change uniform stepsize to quadratic one $t_0 = 10^{-4}$. We include Tab 3 to 5 for more ablation studies regarding time discretization.

| | FID with various NFE | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | 5 | 10 | 20 | 30 | 50 | 100 | 200 | 500 | 1000 |
| Uniform | 246.16 | 90.52 | 27.38 | 14.99 | 8.46 | 4.96 | 3.54 | 2.81 | 2.62 |
| Quadratic | 294.01 | 138.73 | 39.82 | 19.26 | 8.49 | 3.96 | 2.88 | 2.61 | 2.57 |

**Table 7:** Effects of different timesteps on the Euler method. We use $t_0 = 10^{-4}$ which has lower FID score compared with the default $t_0 = 10^{-3}$ [27] in the experiments.

is low. Even with 50 NFE, DEIS is able to outperform blackbox ODE solver in terms of sampling quality.

## D.7 More results on VPSDE

We include mean and standard deviation for CELEBA in Tab 11.

## D.8 More reuslts on VESDE

Though VESDE does not achieve the same accelerations as VPSDE, our method can significantly accelerate VESDE sampling compared with previous method for VESDE. We show the accelerated FID for VESDE on CIFAR10 in Tab 12 and sampled images in Fig 3.

## D.9 Checkpoint used and code licenses

Our code will be released in the future. We implemented our approach in Jax and PyTorch. We have also used code from a number of sources in Tab 13.

We list the used checkpoints and the corresponding experiments in Tab 14.

| NFE | 14 | 26 | 32 | 38 | 50 | 62 | 88.2 | 344 |
|---|---|---|---|---|---|---|---|---|
| FID | 61.11 | 36.64 | 15.18 | 9.88 | 6.32 | 2.63 | 2.56 | 2.55 |

**Table 8:** Quantitative performance of RK45 ODE solver with $t_0 = 10^{-4}$ in Fig 2.

| FID \ NFE / Method | 5 | 10 | 20 | 50 |
|---|---|---|---|---|
| A-DDIM | 51.47 | 14.06 | 6.74 | 4.04 |
| 1-iPNDM | 30.13 | 13.01 | 8.25 | 5.65 |
| 2-iPNDM | 84.00 | 10.45 | 6.79 | 4.73 |
| 3-iPNDM | 105.38 | 14.03 | 5.79 | 4.24 |
| $t$AB1-DEIS | 20.45 | 8.11 | 4.91 | 3.88 |
| $t$AB2-DEIS | 18.87 | 7.47 | 4.66 | 3.79 |
| $t$AB3-DEIS | **18.43** | **7.12** | **4.53** | **3.78** |

**Table 9:** Comparison with A-DDIM on the checkpoint and time scheduling provided by [3] on CIFAR10

| FID \ NFE / Method | 5 | 10 | 20 | 50 |
|---|---|---|---|---|
| iPNDM | 54.62 | 15.32 | 9.26 | 8.26 |
| DDIM | 49.08 | 23.52 | 13.69 | 9.44 |
| $t$AB1-DEIS | 34.69 | 13.94 | 9.55 | 8.41 |
| $t$AB2-DEIS | 29.50 | 11.36 | 8.79 | 8.29 |
| $t$AB3-DEIS | **28.09** | **10.55** | **8.58** | **8.25** |

**Table 10:** Sampling quality on VPSDE ImageNet32 $\times$ 32 with the checkpoint provided by Song et al. [28]. Blackbox ODE solver reports FID 8.34 with ODE tolerance $1 \times 10^{-5}$ (NFE around 130).

| Dataset | FID \ NFE / Method | 5 | 10 | 20 | 50 |
|---|---|---|---|---|---|
| CELEBA | PNDM | - | - | 7.60±0.12 | 3.51±0.03 |
| | iPNDM | 59.87±1.01 | 7.78±0.18 | 5.58±0.11 | 3.34±0.04 |
| | DDIM | 30.42±0.87 | 13.53±0.48 | 6.89±0.11 | 4.17±0.04 |
| | $t$AB1-DEIS | 26.65±0.63 | 8.81±0.23 | 4.33±0.07 | 3.19±0.03 |
| | $t$AB2-DEIS | 25.13±0.56 | 7.20±0.21 | 3.61±0.05 | 3.04±0.02 |
| | $t$AB3-DEIS | **25.07±0.49** | **6.95**±0.09 | **3.41±0.04** | **2.95±0.03** |

**Table 11:** Mean and standard deviation of multiple runs with 4 different random seeds on the checkpoint and time scheduling provided by Liu et al. [20] on CELEBA.

| SDE | FID \ NFE / Method | 5 | 10 | 20 | 50 |
|---|---|---|---|---|---|
| VESDE | $t$AB0-DEIS | 103.52±2.09 | 46.90±0.38 | 27.64±0.05 | 19.86±0.03 |
| | $t$AB1-DEIS | **56.33±0.87** | 26.16±0.12 | 18.52±0.03 | 16.64±0.01 |
| | $t$AB2-DEIS | 58.65±0.25 | **20.89±0.09** | 16.94±0.03 | 16.33±0.02 |
| | $t$AB3-DEIS | 96.70±0.90 | 25.01±0.03 | **16.59±0.03** | **16.31±0.02** |

**Table 12:** FID results of DEIS on VESDE CIFAR10. We note the Predictor-Corrector algorithm proposed in [27] have $\geq$ 100 FID if sampling with limited NFE budget ($\leq$ 50).

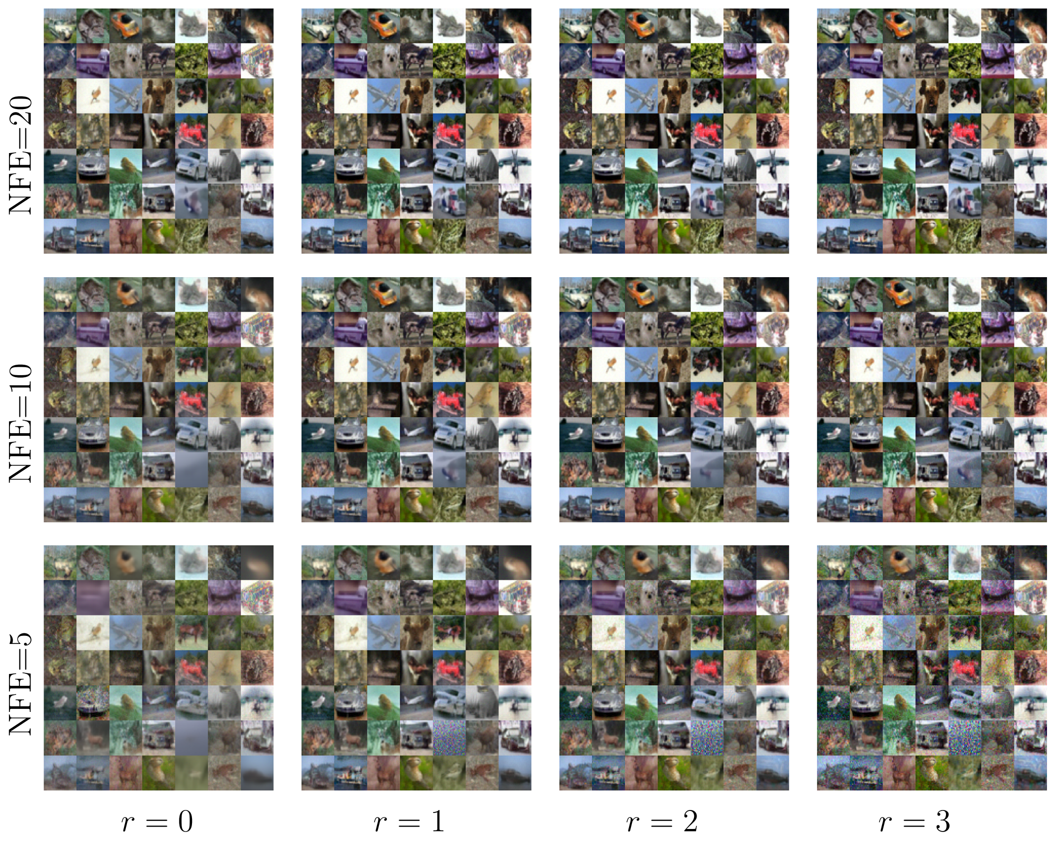

**Figure 3:** Generated images with $t$AB$r$-DEIS on VESDE CIFAR10.

| URL | Citation | License |
|---|---|---|
| `https://github.com/yang-song/score_sde` | [27] | Apache License 2.0 |
| `https://github.com/luping-liu/PNDM` | [20] | Apache License 2.0 |
| `https://github.com/CompVis/latent-diffusion` | [24] | MIT |
| `https://github.com/baofff/Analytic-DPM` | [3] | Unknown |

**Table 13:** Code Used and License

| Experiment | Citation | License |
|---|---|---|
| CIFAR10 Tab 1, 3 to 5, 8, 11 and 12, FFHQ Fig 1 | [27] | Apache License 2.0 |
| CIFAR10 Tab 9 | [3] | Unknown |
| CELEBA Tab 11 | [20] | Apache License 2.0 |
| ImageNet 32 $\times$ 32 Tab 10 | [28] | Unknown |
| Text-to-image | [24] | MIT |

**Table 14:** Checkpoints for experiments