DC-Solver: Improving Predictor-Corrector Diffusion Sampler via Dynamic Compensation

Wenliang Zhao, Haolin Wang, Jie Zhou, and Jiwen Lu*

[1] Department of Automation, Tsinghua University, China
[2] Beijing National Research Center for Information Science and Technology, China
zhaowl20@mails.tsinghua.edu.cn, wanghowlin@gmail.com,
{jzhou,lujiwen}@tsinghua.edu.cn

Abstract. Diffusion probabilistic models (DPMs) have shown remarkable performance in visual synthesis but are computationally expensive due to the need for multiple evaluations during the sampling. Recent predictor-corrector diffusion samplers have significantly reduced the required number of function evaluations (NFE), but inherently suffer from a misalignment issue caused by the extra corrector step, especially with a large classifier-free guidance scale (CFG). In this paper, we introduce a new fast DPM sampler called DC-Solver, which leverages dynamic compensation (DC) to mitigate the misalignment of the predictor-corrector samplers. The dynamic compensation is controlled by compensation ratios that are adaptive to the sampling steps and can be optimized on only 10 datapoints by pushing the sampling trajectory toward a ground truth trajectory. We further propose a cascade polynomial regression (CPR) which can instantly predict the compensation ratios on unseen sampling configurations. Additionally, we find that the proposed dynamic compensation can also serve as a plug-and-play module to boost the performance of predictor-only samplers. Extensive experiments on both unconditional sampling and conditional sampling demonstrate that our DC-Solver can consistently improve the sampling quality over previous methods on different DPMs with a wide range of resolutions up to 1024×1024. Notably, we achieve 10.38 FID (NFE=5) on unconditional FFHQ and 0.394 MSE (NFE=5, CFG=7.5) on Stable-Diffusion-2.1. Code is available at https://github.com/wl-zhao/DC-Solver.

Keywords: Diffusion Model · Fast Sampling · Visual Generation

1 Introduction

Diffusion probabilistic models (DPMs) [8, 28, 33, 36] have emerged as the new state-of-the-art generative models, demonstrating remarkable quality in various visual synthesis tasks [3–7, 10, 16, 21–24, 26–29, 32, 38, 42]. Recent advances in large-scale pre-training of DPMs on image-text pairs also allow the generation of

* Corresponding author

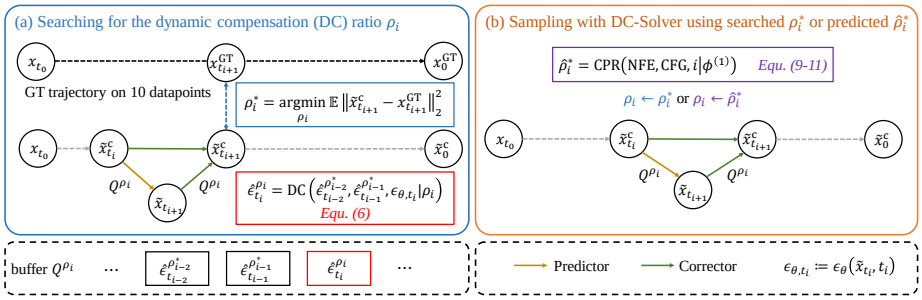
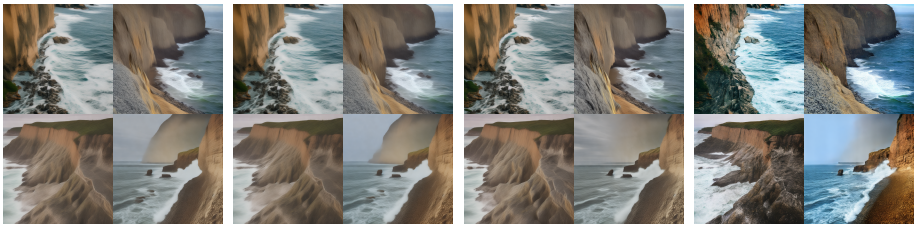


Fig. 1: The main idea of *DC-Solver*. (a) Searching. We propose dynamic compensation (DC) to mitigate the misalignment issue in the predictor-corrector diffusion sampler. The compensation is controlled by the ratios $\{\rho_i\}$ which are adaptive to the sampling step and can be optimized by pushing the sampling trajectory toward the ground truth trajectory on only 10 datapoints. **(b) Sampling.** The compensation ratios can be either efficiently searched as in (a) or instantly predicted by the cascade polynomial regression (CPR) given the desired NFE and CFG.

high-fidelity images given the text prompts [28]. However, sampling from DPMs requires gradually performing denoising from Gaussian noises, leading to multiple evaluations of the denoising network $\boldsymbol{\epsilon}_\theta$, which is computationally expensive and time-consuming. Therefore, it is of great interest to design fast samplers of DPMs [19, 20, 43, 45] to improve the sampling quality with few numbers of function evaluations (NFE).

Recent efforts on accelerating the sampling of DPMs can be roughly divided into training-based methods [17, 25, 30, 35, 39] and training-free methods [15,19,20,34,43–45]. The latter families of approaches are generally preferred in applications because they can be applied to any pre-trained DPMs without the need for fine-tuning or distilling the denoising network. Modern training-free DPM samplers [19, 20, 43, 45] mainly focus on solving the diffusion ODE instead of SDE [1, 8, 36, 44], since the stochasticity would deteriorate the sampling quality with few NFE. Specifically, [20, 43] adopt the exponential integrator [12] to significantly reduce the approximation error of the sampling process. More recently, Zhao *et al.* [45] proposed a predictor-corrector framework called UniPC, which can enhance the sampling quality without extra model evaluations. However, the extra corrector step will cause a misalignment between the intermediate corrected result $\tilde{\boldsymbol{x}}^c_{t_i}$ and the reused model output $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$. The influence of the misalignment has been witnessed in an analysis of UniPC [45], and it has been proven that re-computing the $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}^c_{t_i}, t_i)$ to ensure the alignment is indeed beneficial. However, naively re-computing $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}^c_{t_i}, t_i)$ would bring extra evaluations of the $\boldsymbol{\epsilon}_\theta$ and double the total computational costs.

In this paper, we propose a new fast sampler for DPMs called DC-Solver, which leverages dynamic compensation (DC) to mitigate the misalignment issue in the predictor-corrector framework. Specifically, we adopt the Lagrange interpolation of previous model outputs at a new timestep, which is controlled by a



(a) DPM-Solver++ [20] (MSE 0.443) **(b)** DEIS [43] (MSE 0.436) **(c)** UniPC [45] (MSE 0.434) **(d)** DC-Solver (**Ours**) (MSE 0.394)

Fig. 2: Qualitative comparisons on Stable-Diffusion-2.1. Images above are sampled from SD2.1 (768×768) using the text prompt "*A photo of a serene coastal cliff with waves crashing against the rocks below*" with a classifier-free guidance scale of 7.5 and only **5** number of function evaluations (NFE). We provide the generated images from 4 random initial noises for each method. We show that DC-Solver is able to generate high-resolution and photo-realistic images with more details. Best viewed in color.

learned compensation ratio ρ_i^*. The compensation ratios are optimized by minimizing the ℓ_2-distance between the intermediate sampling results and a ground truth trajectory, which can be achieved in less than 5min on only 10 datapoints. By examining the learned compensation ratios on different numbers of function evaluations (NFE) and classifier-free guidance scale (CFG), we further propose a cascade polynomial regression (CPR) that can instantly predict the desired compensation ratios on unseen NFE/CFG. Equipped with CPR, our DC-Solver allows users to freely adjust the configurations of CFG/NFE and substantially accelerates the sampling process. We also illustrate our method in Figure 1.

We perform extensive experiments on both unconditional sampling and conditional sampling tasks, where we show that DC-Solver consistently outperforms previous methods by large margins in 5∼10 NFE. In the experiments on the state-of-the-art Stable-Diffusion [28] (SD), we find DC-Solver can obtain the best sampling quality on different CFG (1.5∼7.5), NFE (5∼10) and pre-trained models (SD1.4, SD1.5, SD2.1, SDXL). Notably, DC-Solver achieves 0.394 MSE on SD2.1 with a guidance scale of 7.5 and only 5 NFE. By performing the cascade polynomial regression to the compensation ratios searched on only a few configurations, our DC-Solver can generalize to unseen NFE/CFG and surpass previous methods. Besides, we find the proposed dynamic compensation can also serve as a plug-and-play component to boost the performance of predictor-only solvers like [20, 34]. We provide some qualitative comparisons between our DC-Solver and previous methods in Figure 2, where it can be clearly observed that DC-Solver can generate high-resolution and photo-realistic images with more details in only 5 NFE.

2 Related Work

Diffusion probabilistic models. Diffusion probabilistic models (DPMs), originally proposed in [8,33,36], have demonstrated impressive ability in high-fidelity

visual synthesis. The basic idea of DPMs is to train a denoising network $\boldsymbol{\epsilon}_\theta$ to learn the reverse of a Markovian diffusion process [8] through score-matching [36]. To reduce the computational costs in high-resolution image generation and add more controllability, Rombach *et al.* [28] propose to learn a DPM on latent space and adopt the cross-attention [37] to inject conditioning inputs. Based on the latent diffusion models [28], a series of more powerful DPMs called Stable-Diffusion [28] are released, which are trained on a large-scale text-image dataset LAION-5B [31] and soon become famous for the high-resolution text-to-image generation. In practical usage, classifier-free guidance [9] (CFG) is usually adopted to encourage the adherence between the text prompt and the generated image. Despite the impressive synthesis quality of DPMs, they suffer from heavy computational costs during the inference due to the need for multiple evaluations of the denoising network. In this paper, we focus on designing a fast sampler that can accelerate the sampling process of a wide range of DPMs and is suitable to different CFG, thus promoting the application of DPMs.

Fast DPM samplers. Developing fast samplers for DPMs has gained increasing attraction since the prevailing of Stable Diffusion [28]. Modern fast samplers of DPMs usually work by discretizing the diffusion ODE or SDE. Among those, ODE-based methods [19, 20, 34, 45] are shown to be more effective in few-step sampling due to the absence of stochasticity. The widely used DDIM [34] can be viewed as a 1-order approximation of the diffusion ODE. DPM-Solver [19] and DEIS [43] adopt exponential integrator to develop high-order solvers and significantly reduce the sampling error. DPM-Solver++ [20] investigates the data-prediction parameterization and multistep high-order solver which are proven to be useful in practice, especially for conditional sampling. UniPC [45] borrows the merits of the predictor-corrector paradigm [11] in numeral analysis and finds the corrector can substantially improve the sampling quality in the few-step sampling. However, UniPC [45] suffers from a misalignment issue caused by the extra corrector step, which is observed also and mentioned in their original paper. In this work, we aim to mitigate the misalignment through a newly proposed approach called dynamic compensation.

3 Method

3.1 Preliminaries: Fast Sampling of DPMs

We start by briefly reviewing the basic ideas of diffusion probabilistic models (DPMs) and how to efficiently sample from them. DPMs aim to model the data distribution $q_0(\boldsymbol{x}_0)$ by learning the reverse of a forward diffusion process. Given the noise schedule $\{\alpha_t, \sigma_t\}_{t=0}^T$, the diffusion process gradually adds noise to a clean data point $\boldsymbol{x}_0$ and the equivalent transition can be computed by $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \in \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and the resulting distribution $q_T(\boldsymbol{x}_T)$ is approximately Gaussian. During training, a network $\boldsymbol{\epsilon}_\theta$ is learned to perform score matching [2] by estimating the $\boldsymbol{\epsilon}$ given the current $\boldsymbol{x}_t$, timestep t and the condition c. Specifically, the training objective is to minimize:

$$\mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}, t} \left[w(t) \| \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, c) - \boldsymbol{\epsilon} \|_2^2 \right]. \tag{1}$$

The above simple objective makes it more stable to train DPMs on large-scale image-text pairs and enables the generation of high-fidelity visual content. However, sampling from DPMs is computationally expensive due to the need for multiple evaluations of the denoising network $\boldsymbol{\epsilon}_\theta$ (*e.g.*, 200 steps for DDIM [34]).

Modern fast samplers for DPMs [19, 20, 43] significantly reduce the required number of function evaluations (NFE) by solving the diffusion ODE with a multistep paradigm, which leverages the model outputs of previous points to improve convergence. Recently, UniPC [45] proposes to use a corrector to refine the result at each sampling step, which can further improve the sampling quality. Denote the sampling timesteps as $\{t_i\}_{i=0}^M$ and let Q be the buffer to store previous model outputs of the denoising network, the update logic of modern samplers of DPMs from t_{i-1} to t_i can be summarized as follows:

$$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \text{Predictor}(\tilde{\boldsymbol{x}}^{\text{c}}_{t_{i-1}}, Q), \tag{2}$$

$$\tilde{\boldsymbol{x}}^{\text{c}}_{t_i} \leftarrow \text{Corrector}(\tilde{\boldsymbol{x}}_{t_i}, \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i), Q) \quad \text{(optional)} \tag{3}$$

$$Q \overset{\text{buffer}}{\leftarrow} \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i), \tag{4}$$

where $\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}$ denote the refined result after the corrector and $\tilde{\boldsymbol{x}}^{\text{c}}_{t_i} = \tilde{\boldsymbol{x}}_{t_i}$ if no corrector is used as in [20, 43].

3.2 Better Alignment via Dynamic Compensation

Although the extra corrector step (3) can improve the theoretical convergence order, there exists a misalignment between $\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}$ and $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$, *i.e.*, the $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$ pushed into the buffer Q is not computed from the corrected intermediate result $\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}$. It is also witnessed in [45] that replacing the $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$ with $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}, t_i)$ (which would bring an extra forward of $\boldsymbol{\epsilon}_\theta$) can further improve the sampling quality. The effects of the misalignment will be further amplified by the large guidance scale in the widely used classifier-free guidance [9] (CFG) for conditional sampling:

$$\bar{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t, c) = s \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, c) + (1 - s) \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t,), \tag{5}$$

where $s > 1$ is the guidance scale and $s = 7.5$ is usually adopted in text-to-image synthesis on Stable-Diffusion [28].

Dynamic compensation. The aforementioned misalignment issue motivates us to seek for a better method to approximate $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}, t_i)$ after (3) with no extra NFE. To achieve this, we propose a new method called dynamic compensation (DC) that leverages the previous model outputs stored in the buffer Q to approach the target $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}, t_i)$. Given a ratio ρ_i, let $t'_i = \rho_i t_i + (1 - \rho_i) t_{i-1}$, we adopt the following estimation based on Lagrange interpolation:

$$\hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}^{\text{c}}_{t_i}, t_i) = \sum_{k=0}^{K} \prod_{\substack{0 \le l \le K \\ l \neq k}} \frac{t'_i - t_{i-l}}{t_{i-k} - t_{i-l}} \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-k}}, t_{i-k}), \tag{6}$$

Algorithm 1 Searching.

Require: current timestep t_i, a ground truth trajectory $\boldsymbol{x}_t^{\mathrm{GT},N}$, the (corrected) intermediate results $\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c},N}$, a buffer Q, learning rate α, number of iterations L.
$\rho_i \leftarrow 1.0$, $Q^{\mathrm{copy}} \leftarrow Q$
for $l = 1$ **to** L **do**
  compute $\hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c},N}, t_i)$ via (6)
  $Q^{\rho_i} \leftarrow [Q_{[:-1]}^{\mathrm{copy}}, \hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c},N}, t_i)]$
  $\tilde{\boldsymbol{x}}_{t_{i+1}}^{N} \leftarrow \mathrm{Pred}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c},N}, Q^{\rho_i})$
  $\boldsymbol{x}_{t_{i+1}}^{\mathrm{c},N} \leftarrow \mathrm{Corr}(\tilde{\boldsymbol{x}}_{t_{i+1}}^{N}, \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}^{N}, t_i), Q^{\rho_i})$
  $\rho_i \leftarrow \rho_i - \alpha \nabla_{\rho_i} \|\boldsymbol{x}_{t_{i+1}}^{\mathrm{c},N} - \boldsymbol{x}_t^{\mathrm{GT},N}\|_2^2$
end for
return: ρ_i, Q^{ρ_i}

Algorithm 2 Sampling.

Require: sampling timesteps $\{t_i\}_{i=0}^{M}$, initial noise $\tilde{\boldsymbol{x}}_{t_0}^{\mathrm{c}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, compensation ratios $\{\rho_i^*\}_{i=0}^{M-1}$ either searched by (8) or directly predicted by (11).
for $i = 0$ **to** $M - 1$ **do**
  if $i \geq K$ **then**
    compute $\hat{\boldsymbol{\epsilon}}^{\rho_i^*}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c}}, t_i)$ via (6)
    $Q \leftarrow [Q_{[:-1]}, \hat{\boldsymbol{\epsilon}}^{\rho_i^*}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c}}, t_i)]$
  end if
  $\tilde{\boldsymbol{x}}_{t_{i+1}} \leftarrow \mathrm{Pred}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c}}, Q)$
  $\boldsymbol{x}_{t_{i+1}}^{\mathrm{c}} \leftarrow \mathrm{Corr}(\tilde{\boldsymbol{x}}_{t_{i+1}}, \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i), Q)$
end for
return: $\boldsymbol{x}_{t_M}^{\mathrm{c}}$

where K represents the order of the Lagrange interpolation and $\{\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-k}}, t_{i-k})\}_{k=0}^{K}$ are previous model outputs retrieved from buffer Q. The above estimation is then used to replace the last item in Q to obtain a new buffer:

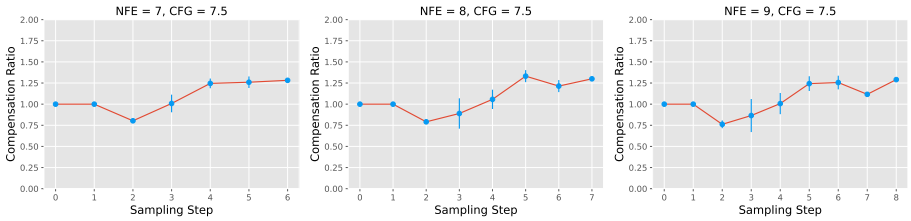
$$Q^{\rho_i} \leftarrow [Q_{[:-1]}, \hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}_{t_i}^{c}, t_i)], \tag{7}$$

where $Q_{[:-1]}$ denotes the elements in Q except the last one. Note that when $\rho_i = 1.0$ we have $\hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}_{t_i}^{c}, t_i) = \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$, which implies that the buffer Q is not updated. By varying the ρ_i, we can obtain a trajectory of $\hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}_{t_i}^{c}, t_i)$ and our goal is to find an optimal ρ_i^* which can minimize the local error to push the sampling trajectory toward the ground truth trajectory. Since the optimal compensation ratio ρ_i^* is different across the sampling timesteps, we name our method dynamic compensation.

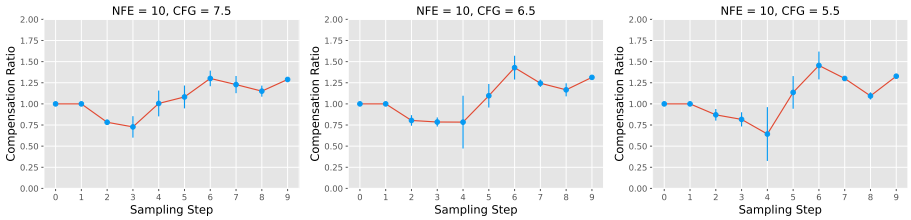
Searching for the optimal ρ_i^*. The optimal compensation ratios $\{\rho_i^*\}$ can be viewed as learnable parameters and optimized through backpropagation. Given a DPM, we first obtain ground truth trajectories $\{\boldsymbol{x}_t^{\mathrm{GT}}\}$ of N initial noises. During each sampling step, we minimize the following objective:

$$\rho_i^* = \arg\min_{\rho_i} \mathbb{E}\|\tilde{\boldsymbol{x}}_{t_{i+1}}^{\mathrm{c}}(\tilde{\boldsymbol{x}}_{t_i}^{\mathrm{c}}, Q^{\rho_i}) - \boldsymbol{x}_{t_{i+1}}^{\mathrm{GT}}\|_2^2, \tag{8}$$

where $\tilde{\boldsymbol{x}}_{t_{i+1}}^{\mathrm{c}}$ is computed similar to (2) and (3), and the expectation is approximated over the N datapoints. The above objective ensures that the local approximation error on the selected N datapoints is reduced with an optimal compensation ratio ρ_i^*. We find in our experiments that $N = 10$ is sufficient in order to learn the optimal $\{\rho_i^*\}_{i=1}^{M}$ which also works well on any other initial noises. Besides, we show that both the local and global convergence of DC-Solver are guaranteed under mild conditions (see Supplementary). When an optimal ρ_i^* is searched, we replace the buffer Q with $Q^{\rho_i^*}$ and move to the next sampling step. We also list the detailed searching procedure in Algorithm 1.



(a) CFG $= 7.5$, NFE $\in [7, 8, 9]$



(b) NFE $= 10$, CFG $\in [7.5, 6.5, 5.5]$

Fig. 3: Relationship between compensation ratios and CFG/NFE. We adopt the widely used Stable-Diffusion-1.5 [28] and search for the optimal compensation ratios for different CFG and NFE and find that the compensation ratios evolve continuously with the variations in CFG/NFE.

Sampling with DC-Solver. After obtaining the optimal compensation ratios $\{\rho_i^*\}$, we can directly apply them in our DC-Solver to sample from the pre-trained DPM. Similar to the searching stage, we update the buffer with $Q^{\rho_i^*}$ after each sampling step to improve the alignment between the intermediate result and the model output (see Algorithm 2 for details). Note that the dynamic compensation (6) does not introduce any extra NFE, thus the overall computational costs are almost unchanged.

3.3 Generalization to Unseen NFE & CFG

Although the compensation ratio ρ_i^* can be obtained via (8), the optimization still requires extra time costs (about 1min for NFE=5). Since the ρ_i^* is specifically optimized for a diffusion ODE, the optimal choice for ρ_i^* is different when NFE or CFG varies. This issue would limit the application of conditional sampling (5), where the users may try different combinations of NFEs and CFGs. Therefore, it is vital to design a method to estimate the optimal compensation ratios without extra time costs of searching. To this end, we propose a technique called cascade polynomial regression that can instantly compute the desired compensation ratios given the CFG and NFE.

Cascade polynomial regression. To investigate how to efficiently estimate the compensation ratios, we start by searching for the optimal compensation ratios on the widely used Stable-Diffusion-1.5 [28] for different configurations of CFG and NFE and plot the relationship between the compensation ratios and CFG/NFE in Figure 3. For each configuration, we perform the search for

10 runs and report the averaged results as well as the corresponding standard deviation. Our key observation is that the learned optimal compensation ratios evolve almost continuously when CFG/NFE changes. Inspired by the shapes of the curves in Figure 3, we propose a cascade polynomial regression to directly predict the compensation ratios. Formally, define the p-order polynomial with the coefficients $\boldsymbol{\phi} \in \mathbb{R}^{p+1}$ as $f^{(p)}(a|\boldsymbol{\phi}) = \sum_{j=0}^{p} \boldsymbol{\phi}_j a^j$, we predict the compensation ratios as follows:

$$\boldsymbol{\phi}_{j,k}^{(2)} = f_1^{(p_1)}(\text{NFE}|\boldsymbol{\phi}_{j,k}^{(1)}), 0 \leq j \leq p_3, 0 \leq k \leq p_2 \tag{9}$$

$$\boldsymbol{\phi}_{j}^{(3)} = f_2^{(p_2)}(\text{CFG}|\boldsymbol{\phi}_{j}^{(2)}), 0 \leq j \leq p_3 \tag{10}$$

$$\hat{\rho}_i^* = f_3^{(p_3)}(i|\boldsymbol{\phi}^{(3)}), 2 \leq i \leq \text{NFE} - 1 \tag{11}$$

The above formulation indicates that we model the change of compensation ratios w.r.t. sampling steps via a polynomial, whose coefficients are determined by the CFG, NFE, and the $\boldsymbol{\phi}^{(1)} \in \mathbb{R}^{(p_3+1)\times(p_2+1)\times(p_1+1)}$. As we will show in Section 4.4, $\boldsymbol{\phi}^{(1)}$ can be obtained by applying the off-the-shelf regression toolbox (such as `curve_fit` in `scipy`) on the pre-computed optimal compensation ratios of few configurations of NFE/CFG. With cascade polynomial regression, we can efficiently compute the compensation ratios with neglectable extra costs, making our DC-Solver more practical in real applications.

3.4 Discussion

Recently, a concurrent work DPM-Solver-v3 [46] proposes to learn several coefficients called empirical model statistics (EMS) of the pre-trained model to obtain a better parameterization during sampling. Our DC-Solver has several distinctive advantages: 1) DPM-Solver-v3 requires extensive computational resources to optimize and save the EMS parameters (*e.g.*, 1024 datapoints, 11h on 8 GPUs, 125MB disk space), while our DC-Solver only needs a *scalar* compensation ratio ρ_i for each step and can be searched more efficiently in both time and memory (10 datapoints, <5min on a single GPU). 2) The EMS is specific to different CFG, and adjusting CFG requires another training of EMS to obtain good results. Our DC-Sovler adopts cascade polynomial regression to predict the desired compensation ratios on unseen CFG/NFE *instantly*. 3) Our proposed dynamic compensation is a more general technique that can boost the performance of both predictor-only and predictor-corrector samplers.

4 Experiments

4.1 Implementation Details

Our DC-Solver follows the predictor-corrector paradigm by applying the dynamic compensation to UniPC [45]. We set $K = 2$ in (6) and skip the compensation when $i < K$, which is equivalent to $\rho_0 = \rho_1 = 1.0$. During the searching stage, we set the number of datapoints $N = 10$. We use a 999-step

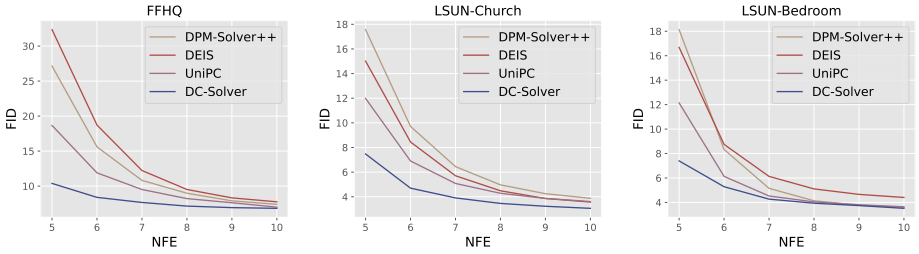


Fig. 4: Unconditional sampling results. We compare our DC-Solver with previous methods on FFHQ [13], LSUN-Church [41], and LSUN-Bedroom [41]. The FID↓ on different numbers of function evaluations (NFE) is used to measure the sampling quality. We show that DC-Solver significantly outperforms other methods, especially with few NFE.

DDIM [34] to generate the ground truth trajectory $\boldsymbol{x}_t^{\mathrm{GT}}$ in the conditional sampling while we found a 200-step DDIM is enough for unconditional sampling. We use AdamW [18] to optimize the compensation ratios for only $L = 40$ iterations, which can be finished in 5min on a single NVIDIA RTX 3090 GPU. We use $p_1 = p_2 = 2$ and $p_3 = 4$ for the cascade polynomial regression.

4.2 Main Results

We perform extensive experiments on both unconditional and conditional sampling on different datasets to evaluate our DC-Solver. Following common practice [20, 45], we use FID↓ of the generated images in unconditional sampling and MSE↓ between the generated latents and the ground truth latents on 10K prompts in conditional sampling. Our experiments demonstrate that our DC-Solver achieves better sampling quality than previous methods including DPM-Solver++ [20], DEIS [43] and UniPC [45] both qualitatively and quantitatively.

Unconditional sampling. We start by comparing the unconditional sampling quality of different methods. We adopt the widely used latent-diffusion models [28] pre-trained on FFHQ [13], LSUN-Bedroom [41], and LSUN-Church [41]. We use the 3-order version for all the methods and report the FID↓ on 5∼10 NFE, as shown in Figure 4. We find our DC-Solver consistently outperforms previous methods on different datasets. With the dynamic compensation, DC-Solver improves over UniPC significantly, especially with fewer NFE. Compared with UniPC, DC-Solver reduces the FID by 8.28, 4,51, 4.75 on FFHQ, LSUN-Church, and LSUN-Bedroom respectively when NFE=5.

Conditional sampling. We conduct experiments on Stable-Diffusion-1.5 [28] to compare the conditional sampling performance of different methods. Following common practice [20, 45], we report the mean squared error (MSE) between the generated latents and the ground truth latents (obtained by a 999-step DDIM [34]) on 10K samples. The input prompts for the diffusion models are

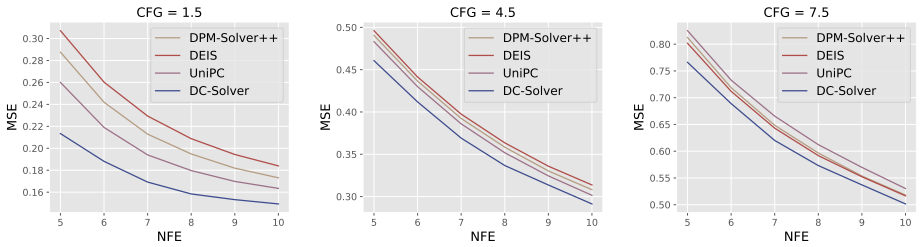


Fig. 5: Conditional sampling results. We compare the sampling quality of different methods using the Stable-Diffusion-1.5 with classifier-free guidance (CFG) varying from 1.5 to 7.5. The sampling quality is measured by the mean squared error (MSE↓) between the generated latents and the ground truth latents obtained by a 999-step DDIM. We randomly select 10K captions from MS-COCO2014 as the text prompts. We observe that DC-Solver consistently achieves better sampling quality on different NFE/CFG.

randomly sampled from MS-COCO2014 validation dataset [14]. Apart from the default guidance scale CFG for Stable-Diffusion-1.5, we also conducted experiments with CFG=1.5/4.5. The results in Figure 5 demonstrate that our DC-Solver achieves the lowest MSE on all of the three guidance scales. Notably, we find that the performance enhancement over UniPC achieved by DC-Solver surpasses the differences observed among those three previous methods.

4.3 Ablation study

We conduct ablation studies on the design of our method and the hyper-parameters on FFHQ [13]. The comparisons of the sampling quality measured by FID↓ of different configurations are summarized in Table 1.

Compensation methods. Firstly, we evaluate the effectiveness of the proposed dynamic compensation in Table 1a. We start from the baseline method UniPC [45] and apply different compensation methods. As discussed in Section 3.2, the baseline with no compensation is equivalent to $\rho_i \equiv 1.0, \forall i$. We then conduct experiments by setting ρ_i to other constants, *i.e.*, $\rho_i \equiv 0.9$ or $\rho_i \equiv 1.1$, which also corresponds to performing interpolation or extrapolation in (6). Since the compensation ratio is constant across the sampling steps, we call these "static compensation". We find that adjusting the ρ_i can indeed influence the performance significantly, and the static compensation with $\rho_i \equiv 1.1$ outperforms the baseline method. As shown in the last row, our proposed dynamic compensation further improves the sampling quality by large margins.

Number of datapoints. We investigate how the number of datapoints would affect the performance of our DC-Solver. We compare the sampling quality when using 5,10,20,30 datapoints and list the results in Table 1b. We also provide the memory costs during the searching stage. We demonstrate that $N = 10$ is enough to obtain satisfactory results while further increasing the number of datapoints will not bring significant improvement.

Table 1: Ablation studies. We perform ablation studies on the design of our method and the hyper-parameters. Sampling quality is measured by FID↓ on FFHQ [13]. The configurations with the best trade-offs are selected and highlighted in gray.

(a) Compensation method.

Compensation Method	NFE			
	5	6	8	10
Baseline [45]	18.66	11.89	8.21	6.99
Static ($\rho_i \equiv 0.9$)	26.43	16.50	9.84	7.84
Static ($\rho_i \equiv 1.1$)	13.99	10.21	7.86	6.90
Dynamic ($\rho_i = \rho_i^*$)	10.38	8.39	7.14	6.82

(b) Number of datapoints.

#Datapoints	Memory (GB)	NFE			
		5	6	8	10
5	9.15	12.39	9.79	7.05	6.84
10	12.10	10.38	8.39	7.14	6.82
20	18.61	10.37	8.31	7.01	6.63
30	22.44	10.93	8.40	6.95	6.70

(c) Order of dynamic compensation.

DC Order K	NFE			
	5	6	8	10
1	12.70	9.44	7.07	6.55
2	10.38	8.39	7.14	6.82
3	11.63	8.89	6.98	6.72

(d) Number of optimization iterations.

#Iterations	Time (s)	NFE			
		5	6	8	10
20	11.4	11.34	8.69	6.96	6.55
40	22.2	10.38	8.39	7.14	6.82
60	33.4	10.63	8.38	7.00	6.65

Order of dynamic compensation. According to (6), the order K controls how the $\hat{\boldsymbol{\epsilon}}^{\rho_i}(\tilde{\boldsymbol{x}}^c_{t_i}, t_i)$ varies with ρ_i. The results in Table 1c indicate that $K = 2$ can produce the best sampling quality, indicating that performing Lagrange interpolation on a parabola-like trajectory is the optimal choice.

Number of optimization iterations. We now examine how many iterations are required to learn the dynamic compensation ratios. In Table 1d, we report the FID of different optimization iterations as well as the time costs for each sampling step. We find the optimization converges after about 40 iterations. In this case, the actual time cost for each NFE is around $(\text{NFE} - 2) \times 22.2$s since we do not need to learn for the first two steps ($\rho_0 = \rho_1 = 1.0$). Note that the time costs in the searching stage will not affect the inference speed since we can directly predict the compensation ratios using the CPR described in Section 3.3.

4.4 More Analyses

In this section, we will provide in-depth analyses of DC-Solver, including some favorable properties and more quantitative/qualitative results.

Comparisons with different pre-trained DPMs. In our main results Section 4.2, we have evaluated the effectiveness of DC-Solver on conditional sampling using Stable-Diffusion-1.5. We now provide comparisons on more different pre-trained DPMs in Table 2, where we report the MSE between the generated latents to the ground truth similar to Figure 5. Specifically, we consider three versions of Stable-Diffusion (SD): 1) SD1.4 is the previous version of SD1.5, which is widely used in [20, 45] to evaluate the conditional sampling quality; 2) SD2.1 is trained using another parameterization called v-prediction [30] and can generate 768×768 images; 3) SDXL is the latest Stable-Diffusion model that can

Table 2: Comparisons with different DPMs. We compare the sampling quality between DC-Solver and previous methods using different pre-trained Stable-Diffusion (SD) models including SD1.4, SD2.1, and SDXL, which can generate images of various resolutions from 512×512 to 1024×1024. We compare the MSE↓ with 5∼10 NFE with the default classifier-free guidance scale of each model. We show that our DC-Solver consistently outperforms previous methods by large margins.

Method	NFE					
	5	6	7	8	9	10
SD1.4, ϵ-prediction, CFG=7.5, 512×512						
DPM-Solver++ [20]	0.803	0.711	0.642	0.590	0.547	0.510
DEIS [43]	0.795	0.706	0.636	0.586	0.544	0.508
UniPC [45]	0.813	0.724	0.658	0.607	0.563	0.525
DC-Solver (Ours)	**0.760**	**0.684**	**0.615**	**0.565**	**0.527**	**0.496**
SD2.1, v-prediction, CFG=7.5, 768×768						
DPM-Solver++ [20]	0.443	0.421	0.404	0.390	0.379	0.370
DEIS [43]	0.436	0.416	0.400	0.387	0.376	0.368
UniPC [45]	0.434	0.415	0.400	0.390	0.381	0.373
DC-Solver (Ours)	**0.394**	**0.364**	**0.336**	**0.309**	**0.315**	**0.294**
SDXL, ϵ-prediction, CFG=5.0, 1024×1024						
DPM-Solver++ [20]	0.745	0.659	0.601	0.558	0.527	0.502
DEIS [43]	0.778	0.683	0.619	0.571	0.538	0.511
UniPC [45]	0.718	0.645	0.593	0.553	0.524	0.500
DC-Solver (Ours)	**0.689**	**0.626**	**0.574**	**0.529**	**0.510**	**0.487**

generate realistic images of 1024×1024. Note that we use the default CFG for all the models (CFG=7.5 for SD1.4 and SD2.1, CFG=5.0 for SDXL). We demonstrate that DC-Solver consistently outperforms previous methods with 5∼10 NFE, indicating that our method has a wide application and can be applied to any pre-trained DPMs to accelerate the sampling.

Generalization to unseen NFE & CFG. Based on the observation of the optimal compensation ratios and the proposed cascade polynomial regression (CPR) in Section 3.3, our DC-Solver can be applied to unseen NFE and CFG without extra time costs for the searching stage. This is important because the users might frequently adjust the NFE and CFG to generate the desired images. To evaluate the effectiveness of the CPR, we first search the optimal compensation ratios for CFG $\in [1.5, 4.5, 7.5, 10.5]$ and NFE $\in [10, 15, 20]$ (which covers most of the use cases in real applications). We then use the `curve_fit` in the `scipy` library to obtain the $\boldsymbol{\phi}^{(1)}$ in (9) and predict the compensation ratios $\hat{\rho}_i^*$ on unseen configurations where CFG $\in [3.0, 6.0, 9.0]$ and NFE $\in [12, 14, 16, 18]$. The results of DC-Solver with the predicted compensation ratios on unseen NFE and CFG on SD2.1 can be found in Table 3, where we also provide the results of previous methods [20, 43, 45] for comparisons. We observe that DC-Solver with the compensation ratios predicted by CPR can still achieve lower MSE on all the unseen configurations. These results indicate that in order to use DC-Solver in

Table 3: Generalization to unseen NFE & CFG. By performing the cascade polynomial regression to the compensation ratios searched on CFG $\in [1.5, 4.5, 7.5, 10.5]$ and NFE $\in [10, 15, 20]$, our DC-Solver can generalize to unseen NFE and CFG and outperform previous methods by large margins. The sampling quality is measured by the MSE↓ between the generated latents and the ground truth on SD2.1 [28].

CFG	Method	NFE			
		12	14	16	18
3.0	DPM-Solver++ [20]	0.212	0.209	0.198	0.196
	DEIS [43]	0.215	0.210	0.199	0.198
	UniPC [45]	0.211	0.208	0.206	0.205
	DC-Solver (Ours)	**0.103**	**0.093**	**0.087**	**0.083**
6.0	DPM-Solver++ [20]	0.312	0.304	0.293	0.289
	DEIS [43]	0.312	0.305	0.293	0.290
	UniPC [45]	0.311	0.304	0.298	0.296
	DC-Solver (Ours)	**0.215**	**0.196**	**0.182**	**0.169**
9.0	DPM-Solver++ [20]	0.404	0.393	0.385	0.377
	DEIS [43]	0.402	0.391	0.380	0.374
	UniPC [45]	0.406	0.394	0.386	0.377
	DC-Solver (Ours)	**0.338**	**0.314**	**0.293**	**0.275**

real scenarios, we only need to perform CPR on sparsely selected configurations of CFG and NFE.

Enhance any solver with dynamic compensation. Although our DC-Solver was originally designed to mitigate the misalignment issue in the predictor-corrector frameworks, we will show that the dynamic compensation (DC) can also boost the performance of predictor-only DPM samplers. Similar to (8), we can also search for an optimal ρ_i^* to minimize $\|\tilde{\boldsymbol{x}}_{t_{i+1}}(\tilde{\boldsymbol{x}}_{t_i}, Q^{\rho_i}) - \boldsymbol{x}_{t_{i+1}}^{\mathrm{GT}}\|_2^2$. To verify this, we conduct experiments on DDIM [34] and DPM-Solver++ [20] by applying the DC to them and the results are shown in Table 4. The FID↓ on FFHQ [13] is reported as the evaluation metric. We show that DC can significantly improve the sampling quality of the two baseline predictor-only solvers. These results indicate that our dynamic compensation can serve as a plug-and-play module to enhance any existing solvers of DPMs.

Visualizations. We now provide some qualitative comparisons between our DC-Solver and previous methods on SD2.1 with CFG=7.5 and NFE=5, as shown in Figure 2. The images sampled from 4 random initial noises are displayed. We find that while other methods tend to produce blurred images with few NFE, our DC-Solver can generate photo-realistic images with more details.

Inference speed and memory. We compare the inference speed and memory of DC-Solver with previous methods, as shown in Table 5. For all the methods, we sample from the Stable-Diffusion-2.1 [28] using a single NVIDIA RTX 3090 GPU with a batch size of 1 and NFE=5/10/15. Our results show that DC-Solver achieves similar speed and memory to previous methods, indicating that

Table 4: Applying DC to predictor-only solvers. We compare the FID↓ on FFHQ [13] using two methods DDIM [34] and DPM-Solver++ [20] as the baselines. We show that dynamic compensation (DC) can also significantly boost the performance of predictor-only solvers.

Method	NFE					
	5	6	7	8	9	10
DDIM [34]	57.92	42.67	32.82	26.96	23.25	19.09
+ DC (Ours)	16.56	15.50	12.51	11.33	9.62	9.21
DPM-Solver++ [20]	27.80	16.01	11.16	9.17	8.04	7.40
+ DC (Ours)	11.97	8.64	7.70	7.32	7.10	6.94

Table 5: Comparisons of inference speed and memory. We compare the inference speed and memory cost of different sampling methods with batch size 1 on SD2.1 [28] using a single NVIDIA RTX 3090 GPU. For inference time, we report the mean and std of 10 runs for each method and NFE. Our DC-Solver achieves similar speed to previous methods with the same NFE.

Method	Memory	Inference Time (s)		
	(GB)	NFE = 5	NFE = 10	NFE = 15
DPM-Solver++ [20]	14.21	1.515(±0.003)	2.833(±0.007)	4.168(±0.005)
UniPC [45]	14.37	1.533(±0.004)	2.865(±0.004)	4.203(±0.003)
DC-Solver (Ours)	14.37	1.532(±0.003)	2.867(±0.005)	4.203(±0.004)

DC-Solver can improve the sample quality without introducing noticeable extra computational costs during the inference.

Limitations. Despite the effectiveness of DC-Solver, it cannot be used with SDE-based samplers [40] because of the stochasticity. How to apply DC-Solver to SDE samplers requires future investigation of a stochasticity-aware metric instead of the ℓ_2-distance in (8).

5 Conclusions

In this paper, we have proposed a new fast sampler of DPMs called DC-Solver, which leverages the dynamic compensation to effectively mitigate the misalignment issue in previous predictor-corrector samplers. We have shown that the optimal compensation ratios can be either searched efficiently using only 10 datapoints on a single GPU in 5min, or instantly predicted by the proposed cascade polynomial regression on unseen CFG/NFE. Extensive experiments have demonstrated that DC-Solver significantly outperforms previous methods in 5$\sim$10 NFE, and can be applied to different pre-trained DPMs including SDXL. We have also found that the proposed dynamic compensation can also serve as a plug-and-play module to boost the performance of predictor-only methods. We hope our investigation on dynamic compensation can inspire more effective approaches in the few-step sampling of DPMs.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2022ZD0160102, and in part by the National Natural Science Foundation of China under Grant 62125603, Grant 62321005, Grant 62336004.

References

1. Bao, F., Li, C., Zhu, J., Zhang, B.: Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. ICLR (2022)
2. Batzolis, G., Stanczuk, J., Schönlieb, C.B., Etmann, C.: Conditional image generation with score-based diffusion models. arXiv preprint arXiv:2111.13606 (2021)
3. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18392–18402 (2023)
4. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. NeurIPS **34**, 8780–8794 (2021)
5. Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A.H., Chechik, G., Cohen-Or, D.: An image is worth one word: Personalizing text-to-image generation using textual inversion. arXiv preprint arXiv:2208.01618 (2022)
6. Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., Guo, B.: Vector quantized diffusion model for text-to-image synthesis. In: CVPR. pp. 10696–10706 (2022)
7. Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., Cohen-Or, D.: Prompt-to-prompt image editing with cross attention control. arXiv preprint arXiv:2208.01626 (2022)
8. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. NeurIPS **33**, 6840–6851 (2020)
9. Ho, J., Salimans, T.: Classifier-free diffusion guidance. NeurIPS (2021)
10. Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., Fleet, D.J.: Video diffusion models. arXiv preprint arXiv:2204.03458 (2022)
11. Hochbruck, M., Ostermann, A.: Explicit exponential runge–kutta methods for semilinear parabolic problems. SIAM Journal on Numerical Analysis **43**(3), 1069–1090 (2005)
12. Hochbruck, M., Ostermann, A.: Exponential integrators. Acta Numerica **19**, 209–286 (2010). https://doi.org/10.1017/S0962492910000048
13. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: CVPR. pp. 4401–4410 (2019)
14. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755. Springer (2014)
15. Liu, L., Ren, Y., Lin, Z., Zhao, Z.: Pseudo numerical methods for diffusion models on manifolds. ICLR (2022)
16. Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: ICCV. pp. 9298–9309 (2023)
17. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. arXiv preprint arXiv:2209.03003 (2022)

18. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
19. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. NeurIPS (2022)
20. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. arXiv preprint arXiv:2211.01095 (2022)
21. Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.Y., Ermon, S.: Sdedit: Guided image synthesis and editing with stochastic differential equations. arXiv preprint arXiv:2108.01073 (2021)
22. Mokady, R., Hertz, A., Aberman, K., Pritch, Y., Cohen-Or, D.: Null-text inversion for editing real images using guided diffusion models. In: CVPR. pp. 6038–6047 (2023)
23. Mou, C., Wang, X., Xie, L., Zhang, J., Qi, Z., Shan, Y., Qie, X.: T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. arXiv preprint arXiv:2302.08453 (2023)
24. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., Chen, M.: Glide: Towards photorealistic image generation and editing with text-guided diffusion models. ICML (2022)
25. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: ICML. pp. 8162–8171. PMLR (2021)
26. Parmar, G., Kumar Singh, K., Zhang, R., Li, Y., Lu, J., Zhu, J.Y.: Zero-shot image-to-image translation. In: ACM SIGGRAPH 2023 Conference Proceedings. pp. 1–11 (2023)
27. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988 (2022)
28. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR. pp. 10684–10695 (2022)
29. Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., Aberman, K.: Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In: CVPR. pp. 22500–22510 (2023)
30. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. ICLR (2022)
31. Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., Komatsuzaki, A.: Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. arXiv preprint arXiv:2111.02114 (2021)
32. Shi, Y., Xue, C., Pan, J., Zhang, W., Tan, V.Y., Bai, S.: Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. arXiv preprint arXiv:2306.14435 (2023)
33. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML. pp. 2256–2265. PMLR (2015)
34. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. ICLR (2021)
35. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models (2023)
36. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: ICLR (2021)
37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

38. Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., Zhu, J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. arXiv preprint arXiv:2305.16213 (2023)
39. Watson, D., Chan, W., Ho, J., Norouzi, M.: Learning fast samplers for diffusion models by differentiating through sample quality. In: ICLR (2021)
40. Xue, S., Yi, M., Luo, W., Zhang, S., Sun, J., Li, Z., Ma, Z.M.: Sa-solver: Stochastic adams solver for fast sampling of diffusion models. arXiv preprint arXiv:2309.05019 (2023)
41. Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)
42. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: ICCV. pp. 3836–3847 (2023)
43. Zhang, Q., Chen, Y.: Fast sampling of diffusion models with exponential integrator. arXiv preprint arXiv:2204.13902 (2022)
44. Zhang, Q., Tao, M., Chen, Y.: gddim: Generalized denoising diffusion implicit models. arXiv preprint arXiv:2206.05564 (2022)
45. Zhao, W., Bai, L., Rao, Y., Zhou, J., Lu, J.: Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. NeurIPS (2023)
46. Zheng, K., Lu, C., Chen, J., Zhu, J.: Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. arXiv preprint arXiv:2310.13268 (2023)