Analyzing and Improving Training-Free Fast Sampling of Text-to-Image Diffusion Models

Zhenyu Zhou[1,2], Defang Chen[3,4†], Siwei Lyu[4], Chun Chen[1,2], and Can Wang[1,2]

[1]State Key Laboratory of Blockchain and Data Security, Zhejiang University
[2]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
[3]University of California, Berkeley [4]State University of New York at Buffalo

Abstract. Text-to-image diffusion models have achieved unprecedented success but still struggle to produce high-quality results under limited sampling budgets. Existing training-free sampling acceleration methods are typically developed independently, leaving the overall performance and compatibility among these methods unexplored. In this paper, we bridge this gap by systematically elucidating the design space, and our comprehensive experiments identify the sampling time schedule as the most pivotal factor. Inspired by the geometric properties of diffusion models revealed through the Frenet-Serret formulas, we propose *cons**t**ant **t**o**t**al **r**otation **s**chedule* (TORS), a scheduling strategy that ensures uniform geometric variation along the sampling trajectory. TORS outperforms previous training-free acceleration methods and produces high-quality images with 10 sampling steps on Flux.1-Dev and Stable Diffusion 3.5. Extensive experiments underscore the adaptability of our method to unseen models, hyperparameters, and downstream applications. Code is available at `https://github.com/zju-pi/TORS`.

1 Introduction

Diffusion-based generative models [10,40,43] synthesize data by progressively denoising noisy samples, which can be modeled using ordinary differential equations (ODEs) [44]. Despite their impressive generative capabilities, diffusion models typically require hundreds of sampling steps, making sampling acceleration a long-standing research topic [10, 34, 44]. Existing acceleration methods can be categorized into training-based and training-free methods. Training-based methods fine-tune the original model to distill multi-step sampling into fewer steps [28, 37,42,54,58]. However, such methods become increasingly resource-intensive and costly as the model sizes grow exponentially, with state-of-the-art text-to-image diffusion models reaching billions of parameters (*e.g.*, 8B Stable Diffusion 3.5 [6], 12B FLUX [1], 20B Qwen-Image [49], 24B Playground v3 [20]). To improve deployment efficiency without additional training, recent work has shifted toward training-free approaches, such as fast ODE solvers [13, 24, 41, 55–57], optimized time schedules [4, 29, 35, 52], and feature caching [17, 21, 27, 48].

† Corresponding author

Fig. 1: Text-to-image generation using Flux.1-Dev [1]. Our proposed TORS attains image quality comparable to the 50-step baseline using only 10 steps.

However, existing training-free methods have largely been developed in isolation, and no systematic study, particularly for state-of-the-art text-to-image diffusion models, has examined their design components or compared their relative contributions to overall performance. In this paper, we bridge this gap by elucidating the design space of ODE solvers, time schedules, and feature caching methods. Through comprehensive experiments on five key components of training-free acceleration methods, we identify the time schedule as the most pivotal factor affecting model performance, while the default uniform time schedule is suboptimal, leading to slow convergence in image structure. Based on this observation, we introduce an improved time scheduling strategy that leverages the geometric properties of sampling trajectories, specifically curvature and torsion under the Frenet-Serret formulas. Our method enforces a constant total geometric variation along the sampling trajectory, yielding high-quality images with consistent structure that closely match converged multi-step results with just a few steps. Moreover, we verify the adaptability of our method by transferring it to unseen LoRA-fine-tuned variants, architectures, hyperparameters, and downstream applications such as image editing. Finally, we conduct a comprehensive compatibility evaluation across various training-free acceleration methods, demonstrating performance improvements and strong robustness of our method. In summary, our contributions are as follows:

- We elucidate the design space of mainstream training-free acceleration methods, quantify the impact of each component on sampling acceleration, and pinpoint the outer time schedule as the dominant factor.
- We propose an advanced scheduling strategy, TORS, based on the geometric regularity of flow-based text-to-image sampling trajectories, which maintains constant total variation for stable and efficient generation.
- Extensive evaluation demonstrating that TORS consistently yields significant performance gains while exhibiting architecture-agnostic adaptability, hyperparameter robustness, and seamless task transferability.

2 Background

Given a data sample $\mathbf{x}_0 \in \mathbb{R}^d$ from the target data distribution p_0, the forward process in diffusion models gradually adds Gaussian noise to the sample, following a stochastic differential equation (SDE) [44]:

$$\mathrm{d}\mathbf{x}_t = f(t)\mathbf{x}_t\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \quad t \in [0, T]\,, \tag{1}$$

where $f(t) : \mathbb{R} \to \mathbb{R}$, and $g(t) : \mathbb{R} \to \mathbb{R}$ are drift and diffusion coefficients, and $\mathbf{w}_t \in \mathbb{R}^d$ is the Wiener process [30]. The backward process reconstructs the data through a reverse-time SDE, $\mathrm{d}\mathbf{x}_t = [f(t)\mathbf{x}_t - g^2(t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}_t$, which shares the same marginal distributions $\{p_t\}_{t=0}^T$ with the forward process [44]. This reverse-time SDE has a *probability flow* ordinary differential equation (PF-ODE) counterpart [4, 44], $\mathrm{d}\mathbf{x}_t = [f(t)\mathbf{x}_t - \frac{1}{2}g^2(t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)]\mathrm{d}t$. The analytically intractable $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ is known as the *score function* [11], which is the target to be predicted by diffusion models. Depending on the choice of $f(t)$ and $g(t)$, two specific forms of linear SDEs, namely, the variance-preserving (VP) SDE [10] and the variance-exploding (VE) SDE [43] are widely used.

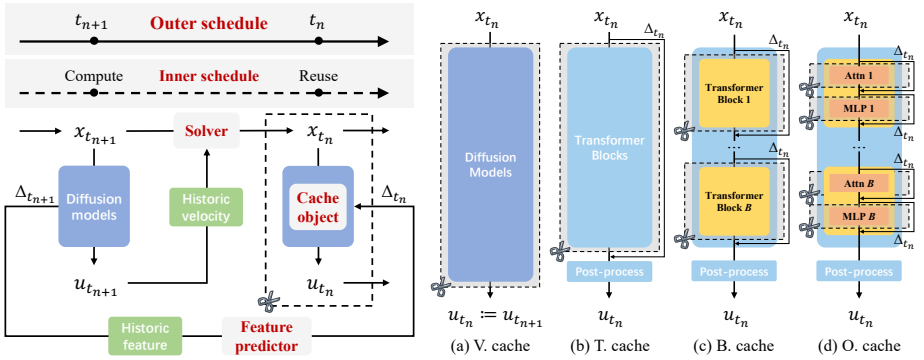
In contrast, flow-based models [19, 23] aim to use a time-dependent *velocity filed* $\mathbf{u}_t(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ to generate a probability path between a data distribution p_0 and a noise distribution $p_T = \mathcal{N}(\mathbf{0}, \mathbf{I})$ by

$$\mathrm{d}\mathbf{x}_t = \mathbf{u}_t(\mathbf{x}_t)\mathrm{d}t, \quad t \in [0, T]\,. \tag{2}$$

Without loss of generality, we set $T = 1$ thereafter. To eliminate the intractable velocity field, *conditional velocity field* $\mathbf{u}_t(\cdot|\mathbf{x}_0)$ is used to define the generative forward process. We have $\mathbf{u}_t(\mathbf{x}_t|\mathbf{x}_0) = \frac{\dot{\sigma_t}}{\sigma_t}\mathbf{x}_t + \alpha_t\left(\frac{\dot{\alpha_t}}{\alpha_t} - \frac{\dot{\sigma_t}}{\sigma_t}\right)\mathbf{x}_0$ given $\mathbf{x}_t = \alpha_t\mathbf{x}_0 + \sigma_t\mathbf{x}_1$. Instead of regressing the intractable velocity field with the objective $\mathbb{E}_{t,p_t(\mathbf{x}_t)}\|\mathbf{u}_\theta(\mathbf{x}_t) - \mathbf{u}_t(\mathbf{x}_t)\|_2^2$, the model is trained by conditional flow matching objective $\mathbb{E}_{t,p_t(\mathbf{x}_t|\mathbf{x}_0),p_0(\mathbf{x}_0)}\|\mathbf{u}_\theta(\mathbf{x}_t) - \mathbf{u}_t(\mathbf{x}_t|\mathbf{x}_0)\|_2^2$. The backward process is simply reversing Equation (2) in time and replacing the velocity by a well-trained model $\mathbf{u}_\theta$. For text-to-image generation, an additional text prompt $\mathbf{c}$ is input into the model, initiating a sampling process governed by $\mathrm{d}\mathbf{x}_t = \mathbf{u}_\theta(\mathbf{x}_t, \mathbf{c})\mathrm{d}t$. Diffusion models and flow-based models are equivalent by $\alpha_t = \exp(\int_0^t f(u)\mathrm{d}u)$, and $\sigma_t = \sqrt{\int_0^t g_u^2 \exp(-2\int_0^u f(s)\mathrm{d}s)\mathrm{d}u}$. Practically, flow-based models are increasingly favored due to their conceptual simplicity, cultivating a series of cutting-edge text-to-image models such as Stable Diffusion v3.5 [6] and Flux [1]. Given that the primary focus of this paper is on state-of-the-art text-to-image models, we employ a flow-based formulation while still referring to them as diffusion models.

3 Elucidating the design Space of Sampling Acceleration

Diffusion models generate samples by simulating $\mathrm{d}\mathbf{x}_t = \mathbf{u}_\theta(\mathbf{x}_t, \mathbf{c})\mathrm{d}t$ starting from Gaussian noise $\mathbf{x}_1$. A common discretization scheme is the Euler method, which



(a) Unified framework of diffusion sampling. **(b)** Four types of cache objects.

Fig. 2: (a) Diffusion sampling follows a pre-defined outer schedule $\{t_n\}_{n=0}^{N}$, while an inner schedule determines whether each step performs computation or feature reuse. A sample $\mathbf{x}_{t_{n+1}}$ transitions from t_{n+1} to t_n with the solver's update rule, utilizing stored historical velocities. Various types of cache objects can be selected, and a feature predictor is employed to estimate the current feature Δ_{t_n}. This perspective also applies to U-Net architectures. (b) Four types of cache objects: velocity cache (V. cache), transformer cache (T. cache), block cache (B. cache), and operation cache (O. cache)

reads $\mathbf{x}_{t_n} = \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1})\mathbf{u}_\theta(\mathbf{x}_{t_{n+1}}, \mathbf{c})$. Due to the multi-step backward process, diffusion sampling is computationally expensive, and the growing parameter scales of modern text-to-image models further intensify the computational and memory demands [1,6,31]. While various training-free acceleration methods have been proposed to improve sampling efficiency, they are generally developed in isolation. In Figure 2a, we provide a unified perspective on these methods, including fast ODE solvers [13,24,55–57], efficient time schedules [4,29,35,52], and feature caching [17, 21, 27, 48]. Below, we elucidate five key components within this unified design space.

Solver. Truncation errors in the Euler method accumulate rapidly when the number of sampling steps is small. To address this, multi-step higher-order solvers [24, 55, 56] utilize historical velocities for more accurate sampling:

$$\mathbf{x}_{t_n} = \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1}) \sum_{i=1}^{O_S} \omega_{n+1,i} \mathbf{u}_\theta(\mathbf{x}_{t_{n+i}}, \mathbf{c}), \tag{3}$$

where $\{\omega_{n+1,i}\}_{i=1}^{O_S}$ are theoretically deduced coefficients to achieve an O_S-th order global truncation error.

Outer schedule. The sampling process of diffusion models starts from establishing the semantic structure and then refines the image details. An appropriate outer schedule ($0 = t_0 < t_1 < \cdots < t_N = 1$) is essential for high image quality, especially under limited sampling budgets.

Orthogonal to solvers and outer schedules, feature caching divides the sampling process into a series of "compute-reuse" cycles, where the stored features

at previous compute steps are utilized in each reuse step to save computations. Feature caching contains three components:

Inner schedule determines whether to compute or reuse features at each t_n of the given outer schedule. It is typically governed by a hyperparameter called *cycle length*. A cycle length of C indicates that each "compute-reuse" cycle consists of one compute step followed by $C-1$ reuse steps.

Cache object determines the content to be cached. Different cache objects impose different demand on both latency and VRAM. As shown in Figure 2b, they can be categorized into four types: (a) *Velocity cache* bypasses all computations during reuse steps and outputs the velocity stored from the last compute step, effectively equivalent to removing all reuse steps from the outer schedule. (b) *Transformer cache* retains the residual outputs after processing through all the transformer blocks. (c) *Block cache* preserves the residual from each individual transformer block. (d) *Operation cache* delves into each block, storing the residuals of the attention and MLP modules which account for the majority of computations within each transformer block.

Feature predictor operates similarly to solvers. It forecasts features required in the current "reuse" step by utilizing historical features from previous "compute" steps. An example of feature prediction with $C=2$ at the reuse step t_n is given by

$$\Delta_{t_n} = \sum_{i=1}^{O_F} \gamma_{n,i} \Delta_{t_{n+1+C(i-1)}}, \tag{4}$$

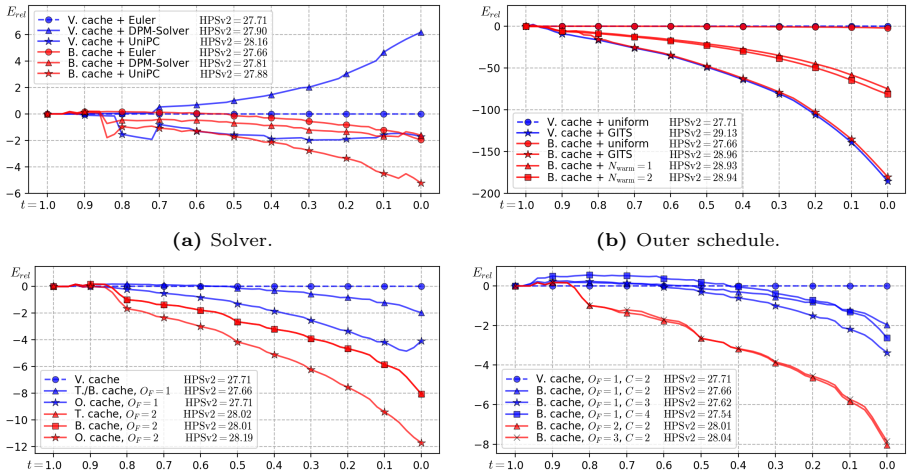
where O_F is the maximum caching order and $\{\gamma_{n,i}\}_{i=1}^{O_F}$ are designed caching coefficients. A direct feature reuse is given by setting $O_F = 1$ and $\gamma_{n,1} = 1$, *i.e.*, $\Delta_{t_n} = \Delta_{t_{n+1}}$.

4 Analyzing Training-Free Sampling Acceleration

In this section, we conduct comprehensive experiments to analyze the contribution of each component in training-free acceleration methods, examining their effects on both the *sampling trajectory* $\{\mathbf{x}_{t_n}\}_{n=1}^{N}$ and overall performance.

Settings. As our primary focus is sampling acceleration, we evaluate models under a 10-step compute setting, where state-of-the-art text-to-image models struggle to maintain high visual quality. To integrate feature caching methods into a unified perspective, we employ $N=20$ sampling steps with a cycle length of $C=2$, resulting in 10 total compute steps. Unless otherwise specified, we employ Euler discretization and a uniform outer schedule, both of which are default settings in recent text-to-image models [1, 6, 31, 34]. For cache configuration, we implement a block cache with a maximum caching order of $O_F = 1$ by default. When $O_F > 1$, we employ the feature predictor from TaylorSeers [21], which calculates adaptive caching coefficients via Taylor expansion.

Metrics. Given a baseline sampling trajectory $\{\hat{\mathbf{x}}_{t_n}\}_{n=1}^{N_b}$ with N_b sampling steps and a reference sampling trajectory $\{\bar{\mathbf{x}}_{t_n}\}_{n=1}^{N_r}$ with N_r sampling steps, the



(a) Solver. **(b)** Outer schedule. **(c)** Cache object. **(d)** Inner schedule and feature predictor.

Fig. 3: Comprehensive experiments examining the impact of each acceleration method on sampling acceleration. The experiments are conducted on Flux.1-Dev model [1]. The outer schedule is identified as the most influential factor (see the vertical scale range).

relative trajectory error of a target sampling trajectory $\{\mathbf{x}_{t_n}\}_{n=1}^{N}$ is defined as

$$E_{rel}(t) = \|\bar{\mathbf{x}}_t - \mathbf{x}_t\|_2 - \|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2. \tag{5}$$

The relative formulation facilitates clearer visualization. The baseline and reference trajectories are generated using velocity cache with $N_b = 10$ and $N_r = 50$ steps ($C = 1$), respectively, corresponding to standard sampling without acceleration. A negative E_{rel} indicates that the target trajectory is closer to the reference trajectory than the baseline trajectory. As the L_2 distance does not adequately correlate with visual quality, we also report HPSv2 (Human Preference Score v2 [50]), which better predicts human perceptual preferences, to evaluate the visual quality of decoded samples on the target trajectories. All the metrics are averaged over 200 prompts. Results on Flux.1-Dev [1] model are shown in Figure 3. Results for Stable Diffusion v3.5 are included in Appendix C.3.

Solver. In addition to the Euler discretization, we further incorporate two fast solvers, namely, DPM-Solver [24] and UniPC [56], both set with a maximum solver order of $O_S = 2$. The results in Figure 3a demonstrate that although these solvers exhibit distinct behaviors in terms of relative trajectory error, they all enhance image quality.

Outer schedule. Beyond the default uniform outer schedule, we re-implement GITS [4], an effective schedule obtained utilizing dynamic programming by approximating global truncation error with accumulated local errors. As illustrated in Figure 3b, GITS significantly improves both relative trajectory error and image quality. Inspecting the resulting GITS schedule, we observe a dense allocation of compute steps in the early sampling stage. To better understand this

phenomenon, we adopt a simple reallocating strategy for block cache, where the "compute-reuse" cycles initiate after N_{warm} consecutive compute steps while omitting the last few compute steps to maintain a total of 10 compute steps for fair comparison. This warm-up strategy reallocates the computational effort toward earlier sampling stages, yielding considerable performance gains. In contrast, shifting compute steps toward the final stage degrades performance. Similar results are observed on Stable Diffusion 3.5, as detailed in Appendix C.3. Therefore, we conclude that the early sampling stage is crucial for state-of-the-art text-to-image models and merits prioritized resource allocation.

Cache object. We compare four representative cache objects that progressively extend into deeper network modules. With a maximum caching order of $O_F = 1$, the transformer cache and block cache are equivalent. Results in Figure 3c demonstrate that naive caching ($O_F = 1$) offers limited benefit, while higher caching orders yield modest improvements, particularly for deeper levels. However, deeper caching also increase latency and VRAM demands. For instance, to generate one single image using Flux.1-Dev, velocity cache requires 9.85 seconds and 36.32 GB of VRAM. In contrast, the deepest operation cache incurs a 6% to 14% latency increase and requires 38.48 GB to 42.78 GB of VRAM, even surpassing the memory needed for a doubled batch size with velocity cache (38.88 GB).

Inner schedule and feature predictor. We conduct ablation studies on the inner schedule and feature predictor by adjusting the cycle length C and maximum caching order O_F, respectively. As shown in Figure 3d, adjusting inner schedule does not improve performance, whereas a higher caching order slightly enhances performance, albeit at the cost of increased latency and VRAM requirements.

All experiments above are replicated on Stable Diffusion v3.5 [6], with results detailed in Appendix C.3. The trends for solver and outer schedule remain consistent. However, while feature caching methods may enhance image quality on Flux [1] models, no performance gains are observed on Stable Diffusion v3.5 [6].

Summary of analyses. To summarize, our key findings on training-free sampling acceleration are as follows:

- *Outer schedule* impacts the performance most: allocating more compute steps to the early sampling stages substantially enhances performance.
- Higher-order *solvers* consistently yield slight improvements in image quality.
- The effectiveness of the *feature predictor* varies across text-to-image models.
- Deeper *cache object* without higher-order feature predictor offers no additional benefit.
- Increasing the cycle length for the *inner schedule* offers no performance gain.

Given the significant performance gains achieved by optimizing the outer schedule, we focus on further enhancements to this aspect in the subsequent sections.

5 Improving Training-Free Sampling Acceleration

5.1 Uniform Outer Schedule is Suboptimal

A widely recognized principle in the sampling dynamics of diffusion models is that early steps define semantic content, while later steps refine visual details [2]. However, we observe slow structural convergence for text-to-image models with the default uniform outer schedule. As shown in Figure 4, image structure fluctuates notably before stabilizing at around 30 steps.

To understand this phenomenon, we first extend the findings on the geometric regularity in diffusion models, previously explored for VE SDEs [2, 4], to the flow-based framework utilized by state-of-the-art text-to-image models. We conduct Principal Component Analysis (PCA) on 100 sampling trajectories, each with 100 sampling steps. The results presented in Figure 5a reveal strong trajectory regularity, with an average explained variance exceeding 99% using the top *three* principal components. Therefore, we can closely characterize each trajectory in the subspace spanned by these principal components, which is referred to as the *projected sampling trajectory*. We further observe that these projected sampling trajectories begin with a high-curvature phase (Figures 5a, 12), which requires smaller step sizes to reduce truncation errors. This is consistent with the empirically successful approaches such as GITS [4] and the warm-up strategy. In contrast, a coarse uniform outer schedule that underrepresents this early phase tends to induce prolonged structural instability in the generated images. Motivated by these geometric insights and experimental evidence, we introduce an improved scheduling strategy tailored for modern text-to-image diffusion models.

5.2 Constant Total Rotation Schedule

The core mechanism of the GITS algorithm lies in approximating the global truncation error by accumulating local errors. In contrast, our approach advances this idea by explicitly incorporating the underlying geometric properties of sampling trajectories, in particular, curvature and torsion in the three-dimensional subspace, to construct a more computationally efficient outer scheduling strategy.

Curvature and torsion. A spatial curve that is at least twice differentiable in three-dimensional Euclidean space $\mathbb{R}^3$ can be characterized by the *Frenet–Serret formulas* [3, 32], describing the evolution of an orthonormal frame formed by the tangent $\mathbf{T}$, normal $\mathbf{N}$, and binormal $\mathbf{B}$ vectors along the curve. These formulas quantify the geometric properties of the curve through two scalar functions: the curvature κ, measuring the change rate of the tangent direction, and the torsion τ, capturing the rate at which the curve deviates from its osculating plane. Consider $\mathbf{r}(t)$ as a projected sampling trajectory in $\mathbb{R}^3$. Since different trajectories may trace the same geometrical shape at different speeds, we adopt an arc-length parameterization to ensure a speed-independent representation. Specifically, assuming that $\dot{\mathbf{r}} \neq 0$, we define $\mathbf{r}(s) = \mathbf{r}(t(s))$ where $s(t) = \int_0^t \|\dot{\mathbf{r}}(u)\| \mathrm{d}u$ is the arc length. With a non-degenerate curve $\mathbf{r}(s)$, the

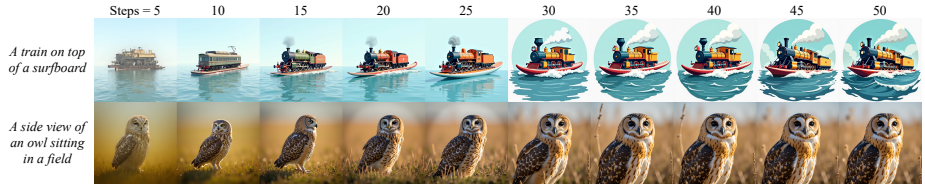
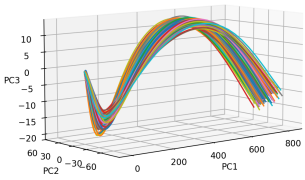
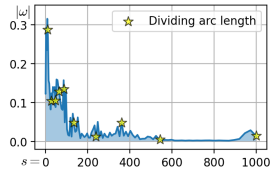


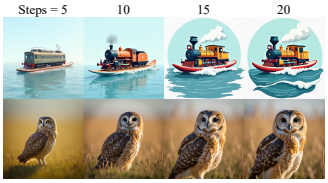
Fig. 4: Text-to-image generation with Flux.1-Dev [1] using the default uniform outer schedule. As the number of sampling steps increases, the image structure continues to change and only stabilizes after around 30 steps.



(a) Trajectory regularity observed on Flux.1-Dev.



(b) Selected dividing arc lengths for TORS.



(c) TORS yields faster structural convergence.

Fig. 5: We extend the insights of sampling regularity from diffusion models to flow-based models and propose TORS by utilizing the geometric properties of sampling trajectories. (a) Visualization of 100 sampling trajectories generated by Flux.1-Dev reveals a strong trajectory regularity. All sampling trajectories starting from Gaussian noises are uniformly shifted to the origin. (b) A 10-step example of our proposed TORS, which ensures a constant total rotation change $\int_0^S |\omega(s)|\mathrm{d}s$ along the sampling trajectory. (c) TORS yields considerably faster structural convergence compared with the uniform outer schedule (Figure 4).

Frenet-Serret formulas are:

$$\frac{\mathrm{d}\mathbf{T}}{\mathrm{d}s} = \kappa\mathbf{N}, \quad \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}s} = -\kappa\mathbf{T} + \tau\mathbf{B}, \quad \frac{\mathrm{d}\mathbf{B}}{\mathrm{d}s} = -\tau\mathbf{N}. \tag{6}$$

Curvature $\kappa = \|\frac{\mathrm{d}\mathbf{T}}{\mathrm{d}s}\|$ measures how sharply a curve bends at a given point, while torsion $\tau = -\mathbf{N} \cdot \frac{\mathrm{d}\mathbf{B}}{\mathrm{d}s}$ measures the rate at which the curve deviates from being planar, capturing its twist into the third dimension [32]. The fundamental theorem of space curves states that curvature and torsion uniquely determine the shape of a regular curve in three-dimensional space (up to rigid motion), a principle that underpins many practical applications in geometry, physics, and computer vision.

Total rotation. The uniform outer schedule overlooks these geometric properties, adopting excessively large step sizes during the initial stage of sampling, where both curvature and torsion are high. Consequently, this leads to a prolonged structural instability in the images before convergence. To address this, we utilize these geometric properties to derive an improved efficient outer schedule. Within the Frenet-Serret framework, the angular velocity vector of a space curve, known as the *Darboux vector*, is defined by $\boldsymbol{\omega} = \tau\mathbf{T} + \kappa\mathbf{B}$ [7]. Its mag-

nitude, $|\boldsymbol{\omega}| = \sqrt{\kappa^2 + \tau^2}$, represents the instantaneous rotation rate, which is usually called the *total curvature* of the curve. Therefore, we define the *total rotation* of a segment as

$$\Theta_{[s_1,s_2]} = \int_{s_1}^{s_2} \sqrt{\kappa^2(s) + \tau^2(s)}\mathrm{d}s. \tag{7}$$

TORS schedule. Leveraging the geometric insights of projected sampling trajectory, we allocate more compute steps to regions where the curvature and torsion are high and fewer compute steps otherwise. To achieve this, we propose adjusting step sizes to maintain a constant change in total rotation. Specifically, we compute the average curvature $\kappa(s)$ and torsion $\tau(s)$ over a fine-grained arc-length grid across 100 projected sampling trajectories. These statistics, defined over $s \in [0, S]$, sever as indicators of the model's geometric behavior. Utilizing these statistics, we partition the trajectory into N segments, each representing an equal share of the total rotation, *i.e.*, $\frac{\Theta_{[0,S]}}{N}$. Based on the selected dividing arc lengths (see Figure 5b), our proposed efficient time schedule is obtained by mapping each arc-length s back to its corresponding timestamp t. We name this scheduling approach the *const**a**nt t**o**tal **r**otation **s**chedule* (TORS), reflecting its strategic focus on maintaining a constant change in total rotation throughout sampling. The full algorithm is presented in Algorithm 1. As shown in Figure 5c, TORS yields considerably faster convergence on image quality compared with the uniform outer schedule.

Justification for total rotation. Define the Frenet-Serret basis $\mathbf{F}(s) = [\mathbf{T}(s), \mathbf{N}(s), \mathbf{B}(s)]$. By the Frenet-Serret formulas, $\|\mathbf{F}'(s)\|_F = \sqrt{2\kappa^2 + 2\tau^2} = \sqrt{2}\,|\boldsymbol{\omega}(s)|$, where $|\boldsymbol{\omega}|$ characterizes the instantaneous rotation rate of the orthonormal frame. To assess its role in discretization accuracy, we examine the local truncation error of Euler steps along $\mathbf{r}(s)$. A Taylor expansion in the Frenet-Serret frame yields

$$\boldsymbol{\epsilon} = \frac{h^2}{2}\kappa\,\mathbf{N} + \frac{h^3}{6}\kappa\tau\,\mathbf{B} + O(h^4), \tag{8}$$

neglecting subdominant terms as κ is dominated by τ along diffusion trajectories (Figure 12). This decomposition reveals two geometric error sources: normal bending (κ, $O(h^2)$) and binormal twisting ($\kappa\tau$, $O(h^3)$). While classical convergence analysis ($h \to 0$) ignores higher-order terms, in the few-step regime with large h, the $O(h^3)$ binormal error becomes non-negligible—potentially comparable to the $O(h^2)$ normal error as their ratio scales with $h\tau/3$. The total rotation rate $|\boldsymbol{\omega}| = \sqrt{\kappa^2 + \tau^2}$ naturally couples these two dominant error sources into a single scalar, serving as a principled indicator for adaptive discretization. Equidistributing total rotation Θ thus ensures that the regions of high bending or twisting receive proportionally denser sampling steps, providing unified control over the leading geometric errors.

6 Experiments

In this section, we conduct comprehensive experiments to demonstrate the superiority of our proposed TORS and examine its compatibility with the aforemen-

Table 1: Results on text-to-image generation. Our proposed TORS achieve state-of-the-art results among sampling acceleration methods.

(a) Results on Flux.1-Dev.

Methods	IR	CS	AS	HPSv2
Baselines				
50 steps	0.96	30.61	5.73	30.15
20 steps	0.93	30.76	5.68	29.30
10 steps	0.71	30.08	5.53	27.70
Sampling acceleration methods (10 steps)				
FORA [39]	0.71	30.32	5.53	27.71
TaylorSeers [21]	0.77	30.38	5.58	28.19
DPM-Solver [24]	0.73	30.27	5.58	27.98
UniPC [56]	0.78	30.41	5.58	28.16
TPDM [53]	0.76	30.07	5.69	28.82
GITS [4]	0.90	30.53	5.65	29.13
TORS (ours)	**0.97**	**30.97**	**5.71**	**29.30**

(b) Results on Stable Diffusion 3.5 medium.

Methods	IR	CS	AS	HPSv2
Baselines				
50 steps	0.97	33.26	5.37	28.64
20 steps	0.94	33.32	5.36	27.95
10 steps	0.55	32.77	5.23	25.31
Sampling acceleration methods (10 steps)				
FORA [39]	0.56	32.81	5.27	25.40
TaylorSeers [21]	0.54	32.73	5.22	25.25
DPM-Solver [24]	0.61	32.88	5.26	25.73
UniPC [56]	0.69	32.88	5.29	26.43
TPDM [53]	0.60	33.10	5.26	25.47
GITS [4]	0.75	32.72	5.25	25.89
TORS (ours)	**0.86**	**33.13**	**5.33**	**26.90**

Table 2: GenEval evaluation on Flux.1-dev.

Model	**Steps**	**Single Object**	**Two Object**	**Counting**	**Colors**	**Attribute Binding**	**Position**	**Overall↑**
Baseline	50	0.99	0.81	**0.69**	**0.81**	0.41	0.20	0.65
Baseline	10	0.96	0.71	0.59	0.69	0.28	0.20	0.57
TORS (ours)	10	**1.00**	**0.82**	0.67	0.80	**0.49**	**0.22**	**0.67**

tioned sampling acceleration methods. Sample quality is measured by Image Reward (IR) [51], CLIP Score (CS) [33], Aesthetic Score (AS) [38] and HPSv2 [50] on the DrawBench dataset [36]. Two recent text-to-image diffusion/flow-based models, namely, Flux.1-Dev [1] and Stable Diffusion 3.5 medium [6] are employed with the guidance scale of 7, unless otherwise specified. All experiments are conducted using up to 4 NVIDIA RTX A6000 GPUs.

6.1 Performance Comparison

We compare our proposed TORS with cutting-edge training-free sampling acceleration methods. For feature caching, we report results using FORA [39] and TaylorSeers [21] which are representative methods tailored for transformer-based diffusion models. In terms of solvers, we include two advanced fast solvers, namely DPM-Solver [24] and UniPC [56]. For the outer schedule, we compare our scheduling strategy with GITS [4], an efficient training-free time schedule developed through dynamic programming. We further compare TORS with representative training-based scheduling methods [45,53]. While approaches like LD3 [45] demonstrate promising results in low-resolution regimes, they are computationally prohibitive due to the requirement of backpropagating gradients through the entire sampling chain. Therefore, we re-implement TPDM [53] that sidesteps the

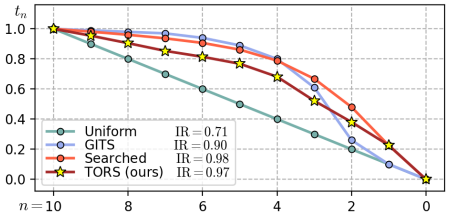


Fig. 6: Visualizing different outer schedules. TORS achieves comparable performance with the searched optimal schedule.

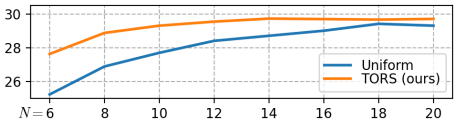


Fig. 7: Ablation on NFE evaluated by HPSv2. TORS largely outperforms the default schedule in few-step generation.

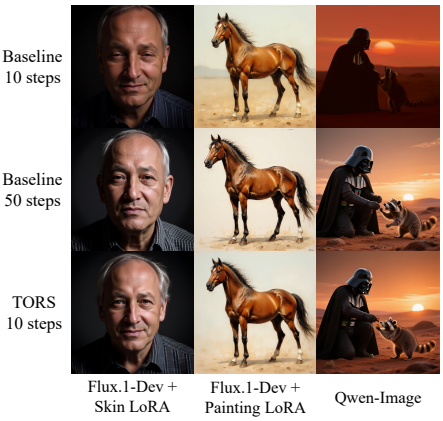


Fig. 8: TORS schedules generated for Flux generalizes well on unseen LoRA-fine-tuned variants as well as the 20B Qwen-Image model.

unrolled computational graph via reinforcement learning. Implementation details are included in Appendix B.3. The results of performance comparison are shown in Table 1, where our proposed TORS consistently outperforms existing sampling acceleration methods under a 10-step compute budget. Notably, TORS achieves nearly 5× acceleration on the Flux model, closely approaching the performance of the 50-step baseline. Our method delivers visually superior results, as shown in Figure 1 and Appendix C.5. Furthermore, evaluated on GenEval [8] (Table 2), TORS with 10 steps significantly outperforms the 10-step baseline and even surpasses the 50-step baseline in overall score.

To further illustrate the superiority of TORS, we design a complex outer schedule parameterized by α, β and p, aiming to encompass a variety of shapes:

$$t_n = \left(1 - I_{\frac{N-n}{N}}(\alpha, \beta)\right)\left(\frac{n}{N}\right)^p, \tag{9}$$

where $I_t(\alpha, \beta) = \frac{B_t(\alpha,\beta)}{B_1(\alpha,\beta)}$ and $B_t(\alpha, \beta) = \int_0^t x^{\alpha-1}(1-x)^{\beta-1}\mathrm{d}x$ is called the *incomplete beta function*. We employ Bayesian optimization to search for an optimal outer schedule within the parameter space, which minimizes the distance between pairs of 10-step and 50-step samples. We conduct 1000 iterations of parameter searching, obtaining an optimal set of parameters as $\alpha = 6.23, \beta = 1.34$ and $p = 0.18$. In Figure 6, we visualize various 10-step outer schedules. Our proposed TORS achieves performance comparable to the searched optimal outer schedule in terms of Image Reward, demonstrating the effectiveness of our scheduling strategy. Moreover, as shown in Figure 7, TORS schedules largely outperforms the default uniform schedule in few-step generation.

Table 3: Quantitative results for adaptability.

Model	N	IR	CS	AS	HPSv2
Flux + Skin LoRA	50	0.70	29.86	5.50	28.91
Flux + Skin LoRA	10	0.42	29.35	5.41	26.59
+ **TORS (ours)**	10	**0.67**	**30.15**	**5.57**	**28.24**
Flux + Painting LoRA	50	0.99	30.42	5.79	29.26
Flux + Painting LoRA	10	0.81	30.15	5.79	27.56
+ **TORS (ours)**	10	**0.98**	**31.03**	**5.81**	**28.52**
Qwen-Image	50	1.18	33.78	5.57	30.80
Qwen-Image	10	0.91	33.18	5.46	27.10
+ **TORS (ours)**	10	**1.12**	**33.76**	**5.55**	**29.28**

Table 4: Ablation study on guidance scales. TORS schedules generated with different guidance scales remain effective across different inference guidance scales.

Inference CFG	Base.	TORS CFG			
		3	5	7	9
3	0.88	0.93	0.93	0.95	0.97
5	0.72	0.92	0.92	0.95	0.92
7	0.71	0.96	0.93	0.97	0.94
9	0.78	0.91	0.87	0.91	0.92

Table 5: Ablation studies evaluating the robustness of TORS schedules on prompt distribution, sample sizes and step counts.

(a) Prompt distribution.

Prompt set	IR	CS	AS	HPSv2
Anime	0.94	31.00	5.71	29.57
Concept	0.94	31.04	5.69	29.35
Paintings	0.95	30.96	5.70	29.37
Photo	0.96	31.01	5.69	29.31

(b) Sample sizes.

#Samples	IR	HPSv2
25	0.96	29.41
50	0.96	29.43
75	0.95	29.37
100	0.97	29.30

(c) Steps counts.

#Steps	IR	HPSv2
25	0.81	28.23
50	0.91	28.61
75	0.94	29.03
100	0.97	29.30

6.2 Ablation studies

To compute the geometric statistics required for TORS scheduling, we generate 100 sampling trajectories, each with 100 sampling steps, using text prompts randomly sampled from the MS-COCO dataset [18]. The computational overhead is approximately three and one A6000 hours for Flux.1-Dev and Stable Diffusion 3.5 medium, respectively. Notably, these pre-computed statistics enable the instantaneous generation of TORS schedules for an arbitrary number of sampling steps with negligible overhead. To further demonstrate the efficiency and effectiveness, we provide additional experiments that underscore the adaptability, robustness, and transferability of our TORS schedules.

Adaptability. We verify the adaptability of our method in Figure 8 and Table 3 by **directly applying** the TORS schedules, pre-computed for Flux, to different LoRA-fine-tuned variants and distinct architectures such as Qwen-Image [49]. Notably, our method exhibits strong generalization ability to unseen models, maintaining high image quality under constrained sampling budgets.

Robustness. In Table 4, we conduct an ablation study on guidance scales by generating TORS schedules under various configurations (referred to as TORS CFG) and evaluating their performance with different inference CFGs. The results demonstrate that our method maintains its efficacy regardless of the guidance scale employed. Furthermore, Table 5 provides comprehensive ablations on the prompt distributions, sample sizes, and step counts used for schedule generation. Our method is largely insensitive to the specific prompt set and the number

Table 6: Performance of image editing evaluated on PIE-Bench using Flux.1-Kontext.

Method	Structure	Background Preservation				CLIP Similarity	
	Dist.$_{\times 10^3}$↓	PSNR↑	LPIPS$_{\times 10^3}$↓	MES$_{\times 10^4}$↓	SSIM$_{\times 10^2}$↑	Whole↑	Edit↑
Baseline $N = 50$	47.52	28.33	59.32	90.17	91.67	25.71	22.65
Baseline $N = 10$	65.29	25.05	75.99	112.10	87.69	25.64	22.59
TORS $N = 10$	**52.46**	**27.59**	**64.87**	**90.90**	**90.37**	**25.76**	**22.70**

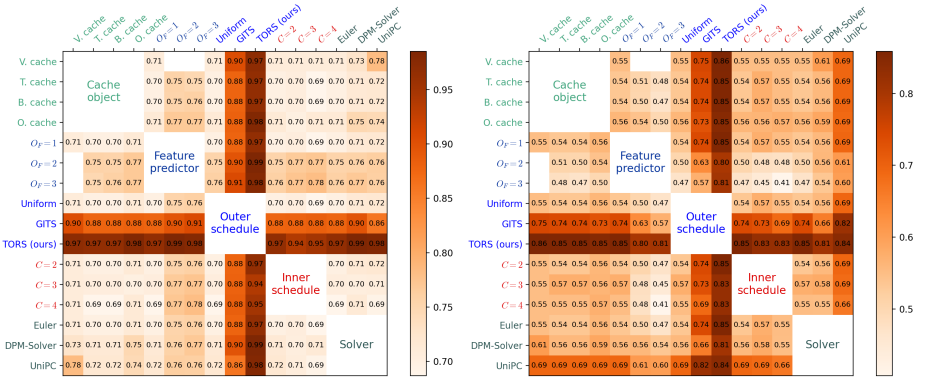
Fig. 9: Qualitative results of image editing using Flux.1-Kontext.

of samples. However, maintaining a sufficient number of sampling steps during profiling is essential for capturing accurate geometric statistics of the trajectory.

Transferability. We further demonstrate the transferability of our method by deploying TORS schedules to downstream applications like image editing. Specifically, we **seamlessly reuse** the TORS schedules computed for Flux on Flux.1-Kontext [14], one of the state-of-the-art image editing models. Quantitative evaluations on PIE-Bench [12] (Table 6) reveal that our method remains highly effective for image manipulation, significantly outperforming the default time schedule. As illustrated in Figure 9, TORS preserves superior layout consistency relative to 50-step reference generations.

6.3 Compatibility Evaluation

Beyond the univariate experiments presented in Section 4, we conduct a thorough pairwise compatibility evaluation among the aforementioned sampling acceleration methods. The heatmap in Figure 10 presents results under a sampling budget of 10 compute steps, covering all possible combinations of sampling acceleration groups. Several key observations emerge: (a) Our proposed TORS demonstrates strong robustness, consistently achieving enhanced performance when combined with diverse acceleration strategies. (b) Among all the acceleration strategies, the outer schedule predominantly drives the performance improvements. (c) In most cases, fast solvers are compatible with other strategies and provide moderate performance gains. (d) Neither the cache object nor the cache cycle significantly affect performance. (e) The compatibility of the feature



(a) Compatibility evaluation on Flux.1-Dev. **(b)** Compatibility evaluation on SD 3.5 medium.

Fig. 10: Compatibility evaluation on acceleration methods using Image Reward. The performance can be enhanced by integrating appropriate acceleration strategies.

predictor with other strategies remains consistent within a single text-to-image model, though it may consistently enhance or degrade the performance.

7 Conclusion

Training-free sampling acceleration methods have improved the efficiency of text-to-image generation but have largely been developed in isolation. In this paper, we bridge this gap by elucidating and analyzing the design space of these methods from a unified perspective, through which we identify that the outer time schedule plays a dominant role in performance gains. We reveal the defects of the commonly used uniform schedule and introduce TORS, an improved efficient outer schedule that leverages the geometric properties of diffusion sampling trajectories. Extensive results demonstrate the superiority of TORS. Furthermore, we examine the compatibility among the discussed acceleration methods, emphasizing the strong robustness of TORS when integrated with various acceleration strategies. Our work opens new avenues for sampling acceleration in state-of-the-art text-to-image diffusion models, and we believe the pursuit of ultimate training-free sampling acceleration through incorporation and refinement of multiple components is an important direction for future exploration.

Acknowledgements

Zhenyu Zhou and Can Wang are supported by the National Natural Science Foundation of China (No. 62476244), the Starry Night Science Fund of Zhejiang University Shanghai Institute for Advanced Study, China (Grant No: SN-ZJU-SIAS-001) and the advanced computing resources provided by the Supercomputing Center of Hangzhou City University.

References

1. BlackForestLabs: Flux.1-dev. https://huggingface.co/black-forest-labs/FLUX.1-dev (2024)
2. Chen, D., Zhou, Z., Mei, J.P., Shen, C., Chen, C., Wang, C.: A geometric perspective on diffusion models. arXiv preprint arXiv:2305.19947 (2023)
3. Chen, D., Zhou, Z., Wang, C., Lyu, S.: Geometric regularity in deterministic sampling dynamics of diffusion-based generative models. Journal of Statistical Mechanics: Theory and Experiment **2025**(12), 124002 (2025)
4. Chen, D., Zhou, Z., Wang, C., Shen, C., Lyu, S.: On the trajectory regularity of ODE-based diffusion sampling. In: International Conference on Machine Learning. pp. 7905–7934. PMLR (2024)
5. Chen, P., Shen, M., Ye, P., Cao, J., Tu, C., Bouganis, C.S., Zhao, Y., Chen, T.: Delta-dit: A training-free acceleration method tailored for diffusion transformers. arXiv preprint arXiv:2406.01125 (2024)
6. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: Forty-first International Conference on Machine Learning (2024)
7. Farouki, R.T.: Pythagorean-hodograph curves: algebra and geometry inseparable. Springer (2008)
8. Ghosh, D., Hajishirzi, H., Schmidt, L.: Geneval: An object-focused framework for evaluating text-to-image alignment. Advances in Neural Information Processing Systems **36**, 52132–52152 (2023)
9. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In: Advances in Neural Information Processing Systems. pp. 6626–6637 (2017)
10. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems (2020)
11. Hyvärinen, A.: Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research **6**, 695–709 (2005)
12. Ju, X., Zeng, A., Bian, Y., Liu, S., Xu, Q.: Pnp inversion: Boosting diffusion-based editing with 3 lines of code. International Conference on Learning Representations (ICLR) (2024)
13. Karras, T., Aittala, M., Aila, T., Laine, S.: Elucidating the design space of diffusion-based generative models. In: Advances in Neural Information Processing Systems (2022)
14. Labs, B.F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., Dockhorn, T., English, J., English, Z., Esser, P., et al.: Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. arXiv preprint arXiv:2506.15742 (2025)
15. Lewiner, T., Gomes, J.D., Lopes, H., Craizer, M.: Curvature and torsion estimators based on parametric curve fitting. Computers & Graphics **29**(5), 641–655 (2005). https://doi.org/https://doi.org/10.1016/j.cag.2005.08.004, https://www.sciencedirect.com/science/article/pii/S0097849305001275
16. Li, L., Li, H., Zheng, X., Wu, J., Xiao, X., Wang, R., Zheng, M., Pan, X., Chao, F., Ji, R.: Autodiffusion: Training-free optimization of time steps and architectures for automated diffusion model acceleration. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7105–7114 (2023)

17. Li, S., Hu, T., Khan, F.S., Li, L., Yang, S., Wang, Y., Cheng, M.M., Yang, J.: Faster diffusion: Rethinking the role of unet encoder in diffusion models. CoRR (2023)
18. 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: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
19. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)
20. Liu, B., Akhgari, E., Visheratin, A., Kamko, A., Xu, L., Shrirao, S., Lambert, C., Souza, J., Doshi, S., Li, D.: Playground v3: Improving text-to-image alignment with deep-fusion large language models. arXiv preprint arXiv:2409.10695 (2024)
21. Liu, J., Zou, C., Lyu, Y., Chen, J., Zhang, L.: From reusing to forecasting: Accelerating diffusion models with taylorseers. arXiv preprint arXiv:2503.06923 (2025)
22. Liu, L., Ren, Y., Lin, Z., Zhao, Z.: Pseudo numerical methods for diffusion models on manifolds. In: International Conference on Learning Representations (2022)
23. 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)
24. 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. In: Advances in Neural Information Processing Systems (2022)
25. 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)
26. Ma, X., Fang, G., Bi Mi, M., Wang, X.: Learning-to-cache: Accelerating diffusion transformer via layer caching. Advances in Neural Information Processing Systems **37**, 133282–133304 (2024)
27. Ma, X., Fang, G., Wang, X.: Deepcache: Accelerating diffusion models for free. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15762–15772 (2024)
28. Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., Salimans, T.: On distillation of guided diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14297–14306 (2023)
29. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: International Conference on Machine Learning. pp. 8162–8171. PMLR (2021)
30. Oksendal, B.: Stochastic differential equations: an introduction with applications. Springer Science & Business Media (2013)
31. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. In: International Conference on Learning Representations (2024)
32. Pressley, A.N.: Elementary differential geometry. Springer Science & Business Media (2010)
33. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
34. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10684–10695 (2022)
35. Sabour, A., Fidler, S., Kreis, K.: Align your steps: Optimizing sampling schedules in diffusion models. arXiv preprint arXiv:2404.14507 (2024)

36. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. In: Advances in Neural Information Processing Systems. pp. 36479–36494 (2022)
37. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. In: International Conference on Learning Representations (2022)
38. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in Neural Information Processing Systems **35**, 25278–25294 (2022)
39. Selvaraju, P., Ding, T., Chen, T., Zharkov, I., Liang, L.: Fora: Fast-forward caching in diffusion transformer acceleration. arXiv preprint arXiv:2407.01425 (2024)
40. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning. pp. 2256–2265. PMLR (2015)
41. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: International Conference on Learning Representations (2021)
42. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models. In: International Conference on Machine learning (2023)
43. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. In: Advances in Neural Information Processing Systems (2019)
44. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations (2021)
45. Tong, V., Trung-Dung, H., Liu, A., Broeck, G.V.d., Niepert, M.: Learning to discretize denoising diffusion odes. arXiv preprint arXiv:2405.15506 (2024)
46. Wang, Y., Wang, X., Dinh, A.D., Du, B., Xu, C.: Learning to schedule in diffusion probabilistic models. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 2478–2488 (2023)
47. Watson, D., Chan, W., Ho, J., Norouzi, M.: Learning fast samplers for diffusion models by differentiating through sample quality. In: International Conference on Learning Representations (2021)
48. Wimbauer, F., Wu, B., Schoenfeld, E., Dai, X., Hou, J., He, Z., Sanakoyeu, A., Zhang, P., Tsai, S., Kohler, J., et al.: Cache me if you can: Accelerating diffusion models through block caching. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6211–6220 (2024)
49. Wu, C., Li, J., Zhou, J., Lin, J., Gao, K., Yan, K., ming Yin, S., Bai, S., Xu, X., Chen, Y., Chen, Y., Tang, Z., Zhang, Z., Wang, Z., Yang, A., Yu, B., Cheng, C., Liu, D., Li, D., Zhang, H., Meng, H., Wei, H., Ni, J., Chen, K., Cao, K., Peng, L., Qu, L., Wu, M., Wang, P., Yu, S., Wen, T., Feng, W., Xu, X., Wang, Y., Zhang, Y., Zhu, Y., Wu, Y., Cai, Y., Liu, Z.: Qwen-image technical report (2025), `https://arxiv.org/abs/2508.02324`
50. Wu, X., Hao, Y., Sun, K., Chen, Y., Zhu, F., Zhao, R., Li, H.: Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. arXiv preprint arXiv:2306.09341 (2023)
51. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: learning and evaluating human preferences for text-to-image generation. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. pp. 15903–15935 (2023)

52. Xue, S., Liu, Z., Chen, F., Zhang, S., Hu, T., Xie, E., Li, Z.: Accelerating diffusion sampling with optimized time steps. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8292–8301 (2024)
53. Ye, Z., Chen, Z., Li, T., Huang, Z., Luo, W., Qi, G.J.: Schedule on the fly: Diffusion time prediction for faster and better image generation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 23412–23422 (2025)
54. Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W.T., Park, T.: One-step diffusion with distribution matching distillation. arXiv preprint arXiv:2311.18828 (2023)
55. Zhang, Q., Chen, Y.: Fast sampling of diffusion models with exponential integrator. In: International Conference on Learning Representations (2023)
56. Zhao, W., Bai, L., Rao, Y., Zhou, J., Lu, J.: Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. arXiv preprint arXiv:2302.04867 (2023)
57. Zhou, Z., Chen, D., Wang, C., Chen, C.: Fast ode-based sampling for diffusion models in around 5 steps. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4401–4410 (2024)
58. Zhou, Z., Chen, D., Wang, C., Chen, C., Lyu, S.: Simple and fast distillation of diffusion models. In: Advances in Neural Information Processing Systems. vol. 37, pp. 40831–40860 (2024)