CAB: Accelerating Flow and Diffusion Sampling via Rectification and Corrected Adams-Bashforth

Anuska Roy **Pravin Nair**
Department of Electrical Engineering
Indian Institute of Technology Madras, Chennai, India
ee25s070@smail.iitm.ac.in, pravinnair@ee.iitm.ac.in

Abstract

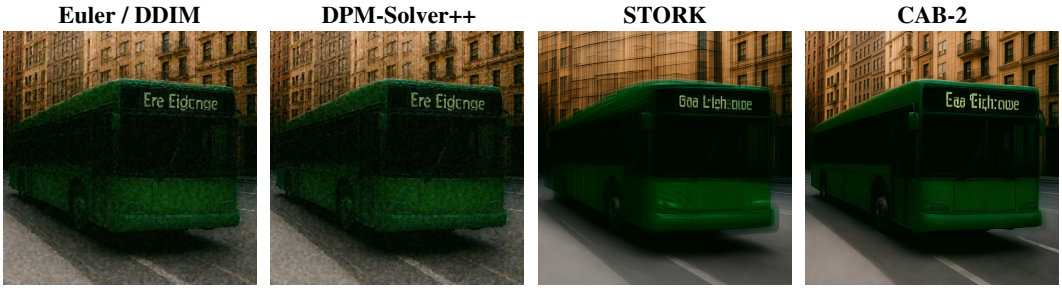
Flow and diffusion models achieve high-fidelity, high-resolution image synthesis, but often require many function evaluations (NFEs) at sampling time. Existing acceleration methods either require additional training through distillation or rely on training-free high-order solvers, and both can degrade sample quality at low NFE budgets. We propose CAB (Corrected Adams-Bashforth), a training-free sampler that accelerates both flow and diffusion models. CAB first transforms the sampling dynamics to a common rectified coordinate system, and then applies a multistep Adams–Bashforth predictor augmented with a simple correction term based on past velocity evaluations and therefore incurs no additional NFEs. The resulting method is simple, has the same algorithmic form across model classes, and has at least third-order local truncation error and second-order global error. Experiments on pretrained flow and diffusion models, including class-conditional and large-scale text-to-image benchmarks, show that CAB improves quality–NFE trade-offs in the low-step regime of 6–20 NFEs. It also remains competitive with strong training-free samplers at higher step counts across most tested models. The official implementation is available at `https://github.com/Anuska-Roy/CAB`.

1 Introduction

Flow and diffusion models have become the dominant choice for high-fidelity image and video generation [1–3], and they increasingly serve as backbones for editing and restoration pipelines [4–8]. Despite their impressive quality, sampling from these models remains computationally expensive because generation often involves numerically integrating a learned reverse-time ordinary differential equation over many neural network evaluations [2, 9, 10]. This cost is particularly limiting in practical deployment, where latency and throughput matter, motivating continued interest in training-free few-step samplers that improve the quality-compute trade-off without any additional training [10–12]. The most important regime is often the low-NFE setting, where only 6 to 20 network evaluations are affordable, but it is also the most challenging, since discretization errors accumulate rapidly and visibly degrade sample fidelity [10, 12].

Training-free ODE solvers have greatly accelerated flow and diffusion sampling, but each family comes with its own limitations. First-order methods such as DDIM are simple and robust, yet typically require many function evaluations for high-quality generation [13]. Multistep methods such as PNDM improve over first-order baselines, but the low-NFE regime remains challenging [12, 14]. Dedicated high-order diffusion solvers such as DEIS and DPM-Solver exploit the semi-linear structure of diffusion ODEs, though prior fast high-order samplers can become unstable under large guidance scales, motivating stabilized variants such as DPM-Solver++ [15, 10, 13]. UniPC improves few-step accuracy over DPM-Solver++, but satisfactory sampling below 10 steps remains difficult in general [12]. DPM-Solver-v3 improves low-NFE quality by introducing empirical model statistics computed on the pretrained model, which yields a more model-adaptive solver requiring

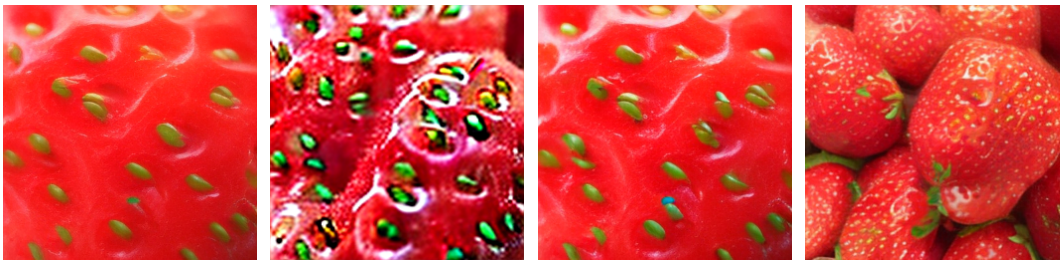
40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Prompt 1: *"A green bus with signage on a city street."* CAB-2 produces a sharper and detail-enhanced sample at 6 NFEs, while existing samplers either retain strong noise artifacts or oversmooth the sample.



Prompt 2: *"The tennis player looks solemn holding his racket."* CAB-2 preserves facial structure, pose, and racket details better at 6 NFEs, while competing solvers show grain and distortion.



Class-conditional generation: CAB-2 produces a more realistic sample with clearer global structure, while competing samplers show oversmoothing, artifacts, or distorted texture at 8 NFEs.

Figure 1: Qualitative comparison of training-free samplers on QWEN-Image 1024×1024 flow generation [18] and DiT 256×256 VP diffusion generation [19].

additional model-dependent estimation [16]. STORK reduces the cost of stabilized Runge–Kutta sampling through virtual NFEs based on Taylor approximation, improving efficiency at the expense of added approximation and solver simplicity [17]. Consequently, existing samplers still face a practical trade-off among accuracy, stability, generality, and simplicity in the low-NFE regime [12, 13, 17]. Figure 1 illustrates this low-NFE trade-off, where existing training-free samplers can exhibit noise, oversmoothing, or structural distortions at low NFE budget.

Contribution: In this work, we propose **Corrected Adams–Bashforth (CAB)**, a simple framework for constructing explicit multistep samplers for flow and diffusion models via an appropriate rectification of the reverse-time ODE. CAB augments Adams–Bashforth (AB) updates with a lightweight correction term that improves low-NFE stability while retaining the key efficiency advantage of a single network evaluation per iteration. Unlike UniPC, which is formulated in log-SNR diffusion coordinates, CAB applies a noise-to-signal rectification to obtain a common explicit learned-field ODE for both diffusion and flow models. We also prove that the corrected schemes preserve the convergence order of the underlying AB solvers. Empirically, CAB improves quality–NFE trade-offs across academic benchmarks and large-scale text-to-image/video models. For example, CAB produces sharper and more coherent samples in the low-NFE settings illustrated in Figure 1.

2 Background and Related Work

Let $\boldsymbol{x}_0 \in \mathbb{R}^d$ denote a data sample drawn from an unknown distribution p_0. Flow and diffusion models define a family of intermediate marginals $\{p_t\}_{t\in[0,T]}$ that transport p_0 to a tractable reference distribution p_T, typically chosen as $\mathcal{N}(\mathbf{0}, \mathbf{I})$ [1, 9, 20, 21]. We use the forward direction $t : 0 \to T$ to

describe the corruption process, while sampling integrates the corresponding reverse-time dynamics from $t = T$ to $t = 0$ starting from the simple prior p_T. Since p_0 is unknown, the intermediate marginals are specified through a forward transition kernel $p_{0t}(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$ [1, 2], which induces

$$p_t(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{x}_0 \sim p_0}[p_{0t}(\boldsymbol{x} \mid \boldsymbol{x}_0)] .$$

A useful unifying viewpoint is that many widely used constructions, including VP/DDPM-type diffusion models [1, 9], VE/EDM-type parameterizations [9, 22], and rectified-flow paths [23], admit the affine-Gaussian forward form

$$\boldsymbol{x}_t = s_t \boldsymbol{x}_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{1}$$

or equivalently,

$$p_{0t}(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}(s_t \boldsymbol{x}_0, \sigma_t^2 \mathbf{I}).$$

Different model families correspond to different choices of nonnegative scalar schedules $s_t, \sigma_t \geq 0$ [1, 9, 22, 23]. On intervals where $s_t > 0$, we define the noise-to-signal ratio $\rho_t := \sigma_t / s_t$. For all the commonly used schedules in [1, 9, 22, 23, 2], ρ_t is differentiable and strictly increasing. Under this regularity condition, the affine path (1) can be equivalently realized by a linear forward stochastic differential equation (SDE) initialized at $\boldsymbol{x}_0 \sim p_0$, [24, 10],

$$d\boldsymbol{x}_t = f(t)\boldsymbol{x}_t \, dt + g(t) \, d\boldsymbol{w}_t, \qquad f(t) = \frac{\dot{s}_t}{s_t}, \qquad g(t)^2 = 2\,\sigma_t \left(\dot{\sigma}_t - \frac{\dot{s}_t}{s_t} \sigma_t \right), \tag{2}$$

where $\boldsymbol{w}_t$ denotes a standard Wiener process. In particular, the monotonicity condition $\dot{\rho}_t \geq 0$ ensures that the diffusion coefficient $g(t)$ is real-valued and nonnegative. This monotonicity condition is also required for the noise-to-signal rectification used to derive the proposed solver in Section 3.

Given the forward SDE in (2), there exists a corresponding reverse-time generative process [9]. Starting from the prior $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, samples from the data distribution can be generated by integrating the associated reverse-time ODE from $t = T$ to $t = 0$. In the noise-prediction parameterization commonly used in diffusion models, the reverse-time ODE is,

$$\frac{d\boldsymbol{x}_t}{dt} = f(t)\boldsymbol{x}_t + \frac{g(t)^2}{2\sigma_t} \epsilon_\theta(\boldsymbol{x}_t, t), \qquad t \in [0, T], \tag{3}$$

with initialization $\boldsymbol{x}_T \sim p_T$ [9, 13, 12, 17]. Here, $\epsilon_\theta(\boldsymbol{x}_t, t)$ denotes a neural network trained to predict the noise component ϵ from $\boldsymbol{x}_t$ in the forward path (1). Equivalent parameterizations are also common. In particular, the data prediction model $\hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t)$ can directly predict $\boldsymbol{x}_0$ from $\boldsymbol{x}_t$, where its relation with $\epsilon_\theta(\boldsymbol{x}_t, t)$ is given as,

$$\hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t) = \frac{\boldsymbol{x}_t - \sigma_t \epsilon_\theta(\boldsymbol{x}_t, t)}{s_t}. \tag{4}$$

while the reverse-time velocity field in (3) may be learned directly as

$$\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) = \dot{s}_t \, \hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t) + \dot{\sigma}_t \, \epsilon_\theta(\boldsymbol{x}_t, t)$$

as in rectified flow [2, 23]. Thus, noise prediction, clean-sample prediction, and direct velocity prediction are equivalent parameterizations of the same reverse-time dynamics.

Fast solvers: One class of existing fast solvers accelerates generation through additional training, including progressive distillation, consistency-style models, and one-step diffusion models [25–31]. In contrast, training-free samplers [11, 14, 15, 10, 13, 12, 32, 33, 16, 34, 35, 17] retain the pretrained model and accelerate generation through improved inference-time solver design. Among these, the most widely used methods solve the reverse-time ODE in (3). Early training-free samplers such as DDIM discretize (3) using a first-order update, while PNDM combines Runge–Kutta startup with linear multistep integration [11, 14]. DPM-Solver reparameterizes (3) using the log-SNR variable $\lambda_t = \log \frac{s_t}{\sigma_t}$ and, under noise prediction, rewrites the ODE (3) in a semi-linear form that admits high-order exponential-integrator updates [10]. DPM-Solver++ [13] retains the same log-SNR reparameterization but replaces noise prediction with data-prediction, yielding improved guided sampling and stabilized multistep variants. UniPC develops a unified predictor–corrector framework for the same reverse dynamics, yielding arbitrary-order samplers with zero-extra-NFE correction and improved low-NFE performance [12]. DPM-Solver-v3 further improves low-NFE quality by incorporating empirical model statistics computed on the pretrained model [16]. When the reverse-time velocity field is learned directly in (3), as in flow matching and rectified flow, generation is performed by standard ODE solvers, with Euler, Heun, and related Runge–Kutta schemes being the most common choices [2, 23]. More recently, STORK introduced stabilized Taylor–Runge–Kutta solvers that address both stiffness and dependence on semi-linear diffusion structure, extending naturally beyond classical diffusion ODEs to flow-matching models [17].

3 Corrected Adams-Bashforth Solver

In this section, we develop a training-free sampler by reparameterizing the ODE in (3) to noise-signal coordinates, and designing a corrected Adams-Bashforth method for flow and diffusion sampling.

3.1 Rectification of the reverse-time ODE in noise-to-signal coordinates

Our key observation is that the reverse-time ODE in (3) admits a particularly simple form after an appropriate change of state and time variables. Specifically, by rescaling the state as $\boldsymbol{y}_t = \boldsymbol{x}_t/s_t$ and reparameterizing time using the noise-to-signal ratio $\rho_t = \sigma_t/s_t$, the linear drift induced by the schedules (s_t, σ_t) is removed. In the resulting coordinates, the reverse dynamics reduce to a rectified ODE whose right-hand side is given directly by the learned noise field as shown in Lemma 3.1.

Lemma 3.1 (Rectified form in noise-to-signal coordinates)**.** *Assume $s : [0,T] \to \mathbb{R}_{>0}$ and $\sigma : [0,T] \to \mathbb{R}_{>0}$ are C^1. Define the noise-to-signal ratio $\rho_t := \sigma_t/s_t$, and assume that $\rho : [0,T] \to \mathbb{R}_{>0}$ is strictly increasing, so that the inverse map $t = t(\rho)$ exists. Consider the reverse-time ODE in* (3)*. Define the rescaled state and its parametrization in ρ space as*

$$\boldsymbol{y}_t = \frac{\boldsymbol{x}_t}{s_t}, \qquad \hat{\boldsymbol{y}}_\rho = \boldsymbol{y}_{t(\rho)}.$$

Then the ρ-reparameterized trajectory satisfies

$$\frac{d\hat{\boldsymbol{y}}_\rho}{d\rho} = \epsilon_\theta\big(s_{t(\rho)}\,\hat{\boldsymbol{y}}_\rho,\ t(\rho)\big). \tag{5}$$

The proof is given in Appendix A.1. As in (3), the noise-prediction model in (5) can be replaced by equivalent neural parameterizations. In particular,

$$\epsilon_\theta\big(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)\big) = \frac{\hat{\boldsymbol{y}}_\rho - \hat{\boldsymbol{x}}_\theta\big(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)\big)}{\rho} = \frac{\boldsymbol{v}_\theta\big(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)\big) - \dot{s}_{t(\rho)}\,\hat{\boldsymbol{y}}_\rho}{s_{t(\rho)}\,\dot{\rho}_{t(\rho)}}, \tag{6}$$

where recall that $\hat{\boldsymbol{x}}_\theta$ denotes the data-prediction model and $\boldsymbol{v}_\theta$ the reverse-time velocity field. The assumption required in Lemma 3.1 that $\rho_t = \sigma_t/s_t$ is strictly increasing, i.e., $\dot{\rho}_t > 0$, is precisely the condition under which the affine-Gaussian path (1) is generated by the linear SDE in Section 2.

3.2 Proposed solver

We develop our sampler by numerically integrating the rectified reverse-time ODE in (5). Although the underlying time variable t is typically discretized uniformly, the induced noise-to-signal variable $\rho_t = \sigma_t/s_t$ is in general non-uniformly spaced; accordingly, our solver is formulated directly on the ρ-grid. Since (5) has the learned nonlinear field as the right-hand side of the ODE, we can directly apply explicit linear multistep predictors on the ρ-grid [36, 37]. However, linear-multistep methods such as Adams–Bashforth (AB) may become inaccurate at large step sizes required for low-NFE sampling [36, 37]. Classical predictor–corrector and defect-correction ideas address this issue by using an additional correction stage [36, 38–40]. Inspired by this perspective, we correct the Adams-Bashforth prediction using an extrapolation defect, the discrepancy between the current learned velocity and its linear Lagrange extrapolation from previous steps.

We now derive the resulting update on the nonuniform ρ-grid induced by a reverse-time discretization. Specifically, sampling integrates the rectified ODE in (5) from $t = T$ to $t = 0$ using grid points

$$T = t_0 > t_1 > \cdots > t_N = 0.$$

For any $i \geq 2$, we next derive the update from $\boldsymbol{y}_{t_i}$ to $\boldsymbol{y}_{t_{i+1}}$. Let

$$\epsilon_i := \epsilon_\theta(s_{t_i}\boldsymbol{y}_{t_i}, t_i), \qquad \epsilon_{i-1} := \epsilon_\theta(s_{t_{i-1}}\boldsymbol{y}_{t_{i-1}}, t_{i-1}), \qquad \epsilon_{i-2} := \epsilon_\theta(s_{t_{i-2}}\boldsymbol{y}_{t_{i-2}}, t_{i-2})$$

denote the learned noise predictions at timesteps t_i, t_{i-1} and t_{i-2}. Further, let step sizes be

$$h_i := \rho_{t_{i+1}} - \rho_{t_i}, \qquad h_{i-1} := \rho_{t_i} - \rho_{t_{i-1}}, \qquad h_{i-2} := \rho_{t_{i-1}} - \rho_{t_{i-2}},$$

and the step ratios be defined as

$$r_i := \frac{h_i}{h_{i-1}}, \qquad r_{i-1} := \frac{h_{i-1}}{h_{i-2}}.$$

Algorithm 1 CAB-p sampling ($p = 2$ or 3)

Require: Prior sample $\boldsymbol{x}_{t_0} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ with $t_0 = T$; reverse grid $\{t_i\}_{i=0}^{N}$; schedules s_t, σ_t; corrector weights $\{\gamma_i\}_{i=2}^{N}$; noise prediction model $\epsilon_\theta(\boldsymbol{x}_t, t)$.
Ensure: Generated sample $\boldsymbol{x}_{t_N}$.

$\epsilon_0 \leftarrow \epsilon_\theta(\boldsymbol{x}_{t_0}, t_0), \quad \boldsymbol{y}_{t_0} \leftarrow \boldsymbol{x}_{t_0}/s_{t_0}$
$\rho_0 \leftarrow \sigma_{t_0}/s_{t_0}, \quad \rho_1 \leftarrow \sigma_{t_1}/s_{t_1}, \quad h_0 \leftarrow \rho_1 - \rho_0$
$\boldsymbol{y}_{t_1} \leftarrow \boldsymbol{y}_{t_0} + h_0\epsilon_0$
for $i = 1$ to $N - 1$ **do**
  $\epsilon_i \leftarrow \epsilon_\theta(\boldsymbol{x}_{t_i}, t_i), \quad \boldsymbol{y}_{t_i} \leftarrow \boldsymbol{x}_{t_i}/s_{t_i}$
  $\rho_i \leftarrow \sigma_{t_i}/s_{t_i}, \quad \rho_{i+1} \leftarrow \sigma_{t_{i+1}}/s_{t_{i+1}}$
  $h_i \leftarrow \rho_{i+1} - \rho_i, \quad r_i \leftarrow h_i/h_{i-1}$
  if $i = 1$ **then**
    $r_i \leftarrow h_i/h_{i-1}$
    $\boldsymbol{y}_{t_2} \leftarrow \boldsymbol{y}_{t_1} + h_i\big[\big(1 + \frac{r_i}{2}\big)\,\epsilon_i - \frac{r_i}{2}\epsilon_{i-1}\big]$
  else
    $\boldsymbol{y}^{\mathrm{pred}}_{t_{i+1}} \leftarrow \mathrm{ABPredictor}(p, i)$
    $\epsilon^{\mathrm{ext}}_i \leftarrow (1 + r_{i-1})\epsilon_{i-1} - r_{i-1}\epsilon_{i-2}$
    $\boldsymbol{y}_{t_{i+1}} \leftarrow \boldsymbol{y}^{\mathrm{pred}}_{t_{i+1}} + \gamma_i h_i(\epsilon_i - \epsilon^{\mathrm{ext}}_i)$
  end if
  $\boldsymbol{x}_{t_{i+1}} \leftarrow s_{t_{i+1}}\boldsymbol{y}_{t_{i+1}}$
end for
return $\boldsymbol{x}_{t_N}$

AB predictors used in Algorithm 1.

Order $p = 2$:

$$\mathrm{ABPredictor}(2, i) = \boldsymbol{y}_{t_i} + h_i\Big[\Big(1 + \frac{r_i}{2}\Big)\,\epsilon_i - \frac{r_i}{2}\epsilon_{i-1}\Big].$$

Order $p = 3$:

$$\mathrm{ABPredictor}(3, i) = \boldsymbol{y}_{t_i} + h_i\left(\beta_0\epsilon_i + \beta_1\epsilon_{i-1} + \beta_2\epsilon_{i-2}\right),$$

where

$$\beta_0 = 1 + \frac{r_i(2r_{i-1} + 1)}{2(r_{i-1} + 1)} + \frac{r_{i-1}r_i^2}{3(r_{i-1} + 1)},$$

$$\beta_1 = -\frac{r_i}{6}\left(2r_{i-1}r_i + 3r_{i-1} + 3\right),$$

$$\beta_2 = \frac{r_{i-1}^2 r_i(2r_i + 3)}{6(r_{i-1} + 1)}.$$

Then the CAB-2 update from $\boldsymbol{y}_{t_i}$ to $\boldsymbol{y}_{t_{i+1}}$ is given by the AB2 predictor

$$\boldsymbol{y}^{\mathrm{pred}}_{t_{i+1}} = \boldsymbol{y}_{t_i} + h_i\Big[\Big(1 + \frac{r_i}{2}\Big)\,\epsilon_i - \frac{r_i}{2}\epsilon_{i-1}\Big], \tag{7}$$

followed by the correction

$$\boldsymbol{y}_{t_{i+1}} = \boldsymbol{y}^{\mathrm{pred}}_{t_{i+1}} + \gamma_i h_i\big(\epsilon_i - \epsilon^{\mathrm{ext}}_i\big), \tag{8}$$

where

$$\epsilon^{\mathrm{ext}}_i = (1 + r_{i-1})\epsilon_{i-1} - r_{i-1}\epsilon_{i-2}. \tag{9}$$

is the linear Lagrange extrapolation of the velocity at ρ_{t_i} from the two previous steps $\rho_{t_{i-1}}$ and $\rho_{t_{i-2}}$, and $\gamma_i \geq 0$ is the corrector weight. The CAB correction is an extrapolation-defect term, which measures how far the current learned velocity deviates from its linear extrapolation using previous velocities, and feeds this error back into the update with weight γ_i. Thus, the correction vanishes when the learned field varies linearly across steps, and becomes active when the learned field has high nonlinear variation, which is exactly when Adams–Bashforth becomes error-prone (refer Appendix A.6). In particular, the correction is designed to refine the Adams–Bashforth prediction while keeping the solver explicit, lightweight, and single-evaluation per step. The complete CAB procedures with two-step AB2 and three-step AB3 predictors, denoted CAB-2 and CAB-3 respectively, are given in Algorithm 1. We next show that the proposed CAB schemes retain the formal order of accuracy of the underlying Adams–Bashforth methods [36, 37] despite the added correction.

Theorem 3.2 (Accuracy of CAB-2 and CAB-3)**.** *Consider the rectified ODE* (5)*, and let the numerical approximations $\{\boldsymbol{y}_{t_i}\}_{i=0}^{N}$ be generated by Algorithm 1 on the induced grid $\rho_{t_0} > \rho_{t_1} > \cdots > \rho_{t_N}$. Assume that the rectified velocity field ϵ_θ in* (5) *is Lipschitz in $\boldsymbol{y}$ uniformly in ρ, the exact solution $\hat{\boldsymbol{y}}(\rho)$ is sufficiently smooth, the variable-step ratios are uniformly bounded away from 0 and ∞, the corrector weights are uniformly bounded, and the starting values are accurate to the target global order. Then the corrected variable-step CAB schemes satisfy*

$$\max_{0 \leq i \leq N} \|\boldsymbol{y}_{t_i} - \hat{\boldsymbol{y}}(\rho_{t_i})\| \leq \begin{cases} C_2\, h_{\max}^2, & \textit{for CAB-2}, \\ C_3\, h_{\max}^3, & \textit{for CAB-3 if } \gamma_i = \mathcal{O}(h_i), \end{cases} \tag{10}$$

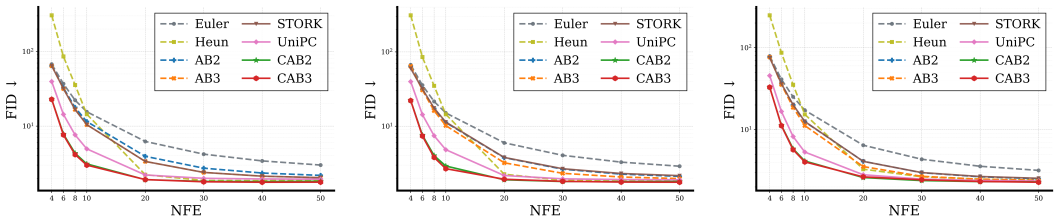
where $h_{\max} := \max_i |\rho_{t_{i+1}} - \rho_{t_i}|$ and the constants C_2 and C_3 are independent of the step sizes $\{h_i\}$. If used with a constant corrector weight $\gamma_i = \mathcal{O}(1)$, CAB-3 is second-order globally. Moreover, the local truncation errors are $\mathcal{O}(h_i^3)$ for CAB-2 and $\mathcal{O}(h_i^4)$ for CAB-3 when $\gamma_i = \mathcal{O}(h_i)$.

Table 1: Comparison of fast flow and diffusion samplers with CAB.

Method (NFE regime)	Integration space; update mechanism	Distinction from CAB
PNDM / PLMS (20–50)	DDIM path; AB/PLMS history extrapolation	History extrapolation only; no field-defect correction
DEIS (10–20)	t or ρ; polynomial noise extrapolation with exponential integration	Related ρ-space prediction, but no defect-based correction
DPM-Solver (10–20)	$-log(\rho)$; high-order exponential / multistep integration	Uses high-order integration rather than history-based defect correction
DPM-Solver++ (10–20)	$-log(\rho)$; data-prediction exponential multistep solver	No adaptive defect-controlled correction weight
UniPC (5–20)	$-log(\rho)$; multistep predictor + zero-NFE UniC corrector	Uses an order-dependent weighted combination of field differences; CAB uses one extrapolation residual scaled by a single scalar
DPM-Solver-v3 (5–10)	$-log(\rho)$; multistep residual approximation	Uses model-specific offline statistics
STORK (4–50)	Original ODE; Taylor-predicted RK stages	Predicts stages/derivatives rather than correcting a field defect
Conjugate Integrators (≥ 30)	Transformed state; time-varying linear transformation	Requires a chosen transformation of the dynamics
Bespoke Solvers (8–20)	Learned coordinates; model- and NFE-specific optimized solver	Requires solver optimization for each setting
CAB (6–20)	$\mathbf{y} = \mathbf{x}/s,\ \rho = \sigma/s$; history-based field-defect correction	Uses one defect scaled by a fixed or adaptive scalar; no evaluation at the predicted state or offline model statistics

We defer the proof to Appendix A.4, where we present a unified analysis in which CAB-2 and CAB-3 arise as special cases. Thus, CAB retains the standard convergence behavior of explicit multistep integration while improving low-NFE generation quality in practice, as shown in Section 4. We next provide a detailed comparison of our proposed solver with existing solvers in Table 1, with respect to the evolution of the updates and NFEs required.

Implementation details. Since CAB is a multistep method, the initial updates are handled separately as in Algorithm 1, where we use an Euler step for $i = 0$ and an AB2 step for $i = 1$. Though Algorithm 1 is written in terms of a noise-prediction field, when the pretrained network uses data prediction, as in DPM-Solver++ [13], or direct velocity prediction, as in rectified flow [23], we use the equivalent noise-field expressions in (6). Because sampling proceeds from $t = T$ to $t = 0$, the induced noise-to-signal coordinate $\rho_t = \sigma_t/s_t$ can produce a numerically negligible terminal step. We therefore merge the last two ρ-steps in our implementation. This affects only the grid construction; all reported NFEs are counted by the number of network evaluations, not by the number of grid intervals. Finally, Theorem 3.2 shows that CAB-3 preserves third-order global accuracy when the corrector weight is $\gamma_i = \mathcal{O}(h_i)$. In practice, however, a constant weight $\gamma_i = \gamma$ gives better low-NFE sample quality. We therefore select a single model-specific $\gamma \in (0, 1.5)$ and keep it fixed across all sampling steps and generated samples for both CAB-2 and CAB-3.



(a) VP schedule on CIFAR-10. (b) VE schedule on CIFAR-10. (c) VP schedule on ImageNet.

Figure 2: FID versus NFE for CIFAR-10 with VP/VE schedules and ImageNet with the VP schedule. CAB-2 and CAB-3 consistently improve over the corresponding AB baselines in the low-NFE regime. They remain competitive with, and often outperform, strong training-free fast samplers.

4 Experiments

In this section, we compare CAB with fast training-free flow and diffusion samplers such as DPM-Solver++, UniPC, and STORK, across multiple generation settings. We also provide ablations on the roles of rectification and correction, the effect of the corrector weight γ on distributional and



(a) DDIM (b) DPM++ (c) STORK (d) CAB-2

DiT at 8 NFEs. CAB-2 retains the cauliflower structure with sharper boundaries and no texture artifacts. Other samplers show oversmoothing, high-frequency artifacts, or distorted local texture.



(e) DDIM (f) DPM++ (g) STORK (h) CAB-2

DiT at 10 NFEs. CAB-2 preserves sharper facial detail and more coherent structure, DDIM and STORK are noticeably blurred, while DPM-Solver++ introduces stronger texture artifacts.

Figure 3: Comparison of training-free samplers on 256×256 class-conditional ImageNet generation.

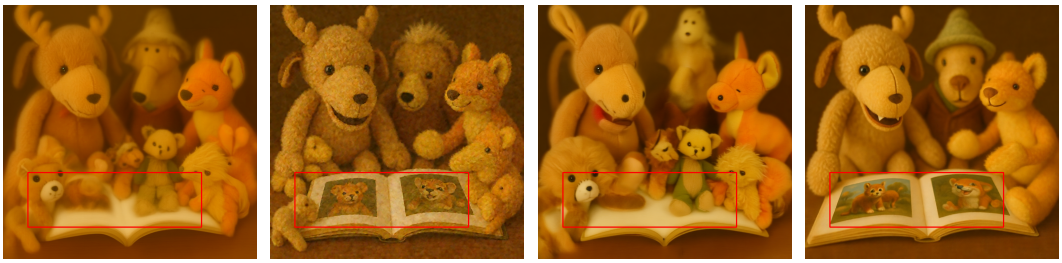
Table 2: FID comparison on DiT/ImageNet 256×256, where lower is better. Sampler-O3 denotes the corresponding third-order variant using three past velocity evaluations, analogous to CAB-3.

NFE	Euler	DPM++-O3	DPM++	STORK-O3	STORK	AB2	AB3	CAB-2	CAB-3
4	120.83	106.64	**75.26**	127.99	117.12	116.99	116.77	88.04	87.82
6	73.20	96.38	26.64	100.23	52.27	51.52	46.44	15.13	**14.81**
8	39.71	144.08	24.90	78.00	21.18	20.70	16.75	**9.21**	9.72
10	24.22	141.58	22.76	57.62	12.65	12.39	10.35	**7.99**	8.79
20	9.94	40.38	12.19	17.98	7.48	7.44	7.21	**6.86**	7.08
30	8.21	15.06	9.34	11.49	7.10	7.09	7.05	**6.86**	6.93
40	7.63	9.59	8.42	9.44	7.02	7.01	6.98	**6.91**	6.97
50	7.40	8.02	7.99	8.53	6.99	6.98	6.97	**6.95**	7.01

perceptual quality, runtime, and memory cost. Additional results, such as empirical verification of Theorem 3.2, additional comparisons, and limitations, are deferred to the Appendix.

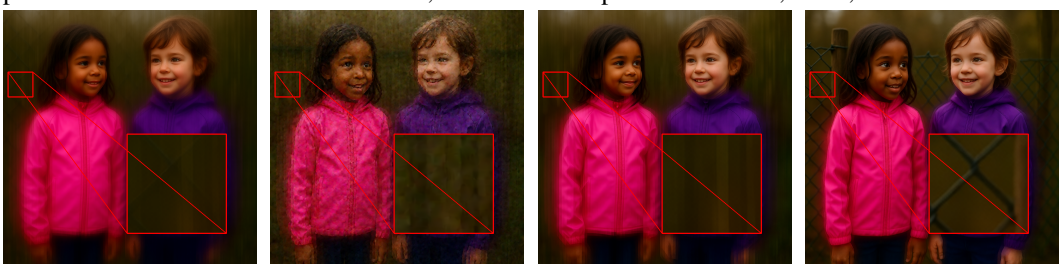
Unconditional image generation on CIFAR-10 (32×32) **and ImageNet** (64×64) We evaluate CAB on three pixel-space EDM models from [22]: CIFAR-10 with VP and VE schedules, and ImageNet with VP schedule. For each sampler, we generate 50K images and report Fréchet Inception Distance (FID) as a function of NFE in Figure 2. In the low-NFE regime, CAB-2 and CAB-3 consistently improve over their uncorrected Adams-Bashforth counterparts with $\gamma = 0.9$, demonstrating that the proposed correction improves finite-step sampling accuracy. CAB remains effective across both VP and VE parameterizations and often outperforms strong training-free baselines such as UniPC and STORK at small NFE budgets. As the number of NFEs increases, CAB remains competitive while retaining a simple one-network-evaluation-per-step update.

Conditional ImageNet 256×256 **generation with DiT.** We next evaluate CAB on DiT for class-conditional ImageNet 256×256 generation. Table 2 reports FID using 10K generated samples across NFE budgets with classifier guidance scale 1.25. For UniPC, DPM-Solver++, and STORK, we report the best-performing settings together with higher-order variants; in this setting, aggressive multi-evaluation corrections can overcorrect and degrade FID (also shown visually in Figure. 10 in



(a) Euler (b) DPM++ (c) STORK (d) CAB-2

Prompt: *"Stuffed animals looking a pictures of other animals in a book."* At 8 NFEs, CAB-2 preserves the context ' animals in book', while other outputs contain blur, noise, or semantic loss.



(e) Euler (f) DPM++ (g) STORK (h) CAB-2

Prompt: *"Two kids in pink and purple jackets standing by a fence."* At 8 NFEs, CAB-2 generates sharper details and preserves 'fence', while competing samplers show blur, noise, or missing structure.

Figure 4: Comparison of training-free samplers on QWEN-Image 1024×1024 flow generation [18].

Table 3: ImageReward Comparison on Qwen-Image in the low-NFE regime. CAB-3 (with $\gamma = 0.2$) is strongest at $4-10$ NFEs, while CAB-2 remains highly competitive and achieves the best score at $N = 20$; at larger NFEs, the gap to the strongest baseline becomes small.

NFE	Euler	Heun	DPM-Solver++	UniPC	STORK	AB2	AB3	CAB-2	CAB-3
4	−0.7949	−1.1742	−1.3069	−0.6505	−0.5147	−0.6547	−0.6414	−0.3544	**−0.3474**
6	−0.1279	−0.2009	0.0455	0.1377	0.1306	0.0637	0.0975	0.3393	**0.3798**
8	0.3439	0.1499	0.5866	0.5471	0.5512	0.4996	0.5183	0.6561	**0.6760**
10	0.6499	0.4081	0.8069	0.7671	0.7734	0.7482	0.7662	0.8689	**0.8856**
20	1.1545	1.0057	1.1912	1.1682	1.1756	1.1686	1.1768	**1.2026**	1.1921
30	1.2644	1.1344	1.2729	1.2651	**1.2855**	1.2809	1.2843	1.2655	1.2584
40	1.2806	1.2543	1.2669	1.2756	1.2654	1.2838	**1.2852**	1.2707	1.2746
50	1.2824	1.2738	1.2840	1.2839	**1.2934**	1.2818	1.2844	1.2866	1.2920

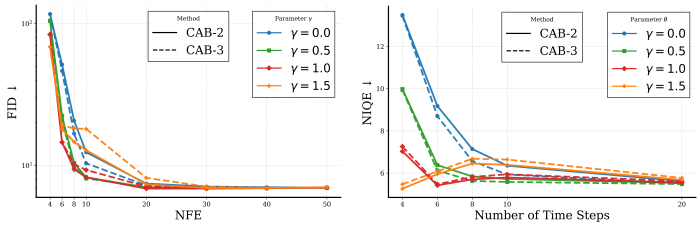
Appendix). CAB uses a more controlled correction and achieves the strongest low-NFE performance: CAB-3 is best at 6 NFEs, while setting $\gamma = 0.9$ in CAB-2, it obtains the best FID from 8 to 50 NFEs. Figure 1 and Figure 3 provide qualitative comparisons using classifier guidance scale 4. CAB-2 with $\gamma = 0.5$ produces sharper and more coherent samples, and baselines often exhibit oversmoothing or high-frequency texture artifacts, consistent with the quantitative trends in Figure 2 and Table 2.

High-Resolution (1024×1024) **Text-to-Image Generation (QWEN-Image)** We evaluate CAB for high-resolution text-to-image synthesis using the recent QWEN-Image model [18]. Table 3 reports ImageReward [41], a learned human-preference score, averaged over 200 prompts. CAB-3 achieves the best ImageReward score at $N = 4, 6, 8$, and 10, while CAB-2 is best at $N = 20$. Both CAB variants with $\gamma = 0.2$ substantially improve over the uncorrected AB2 baseline, highlighting the contribution of the correction term. As NFE increases, the gap narrows, and strong baselines such as STORK, DPM-Solver++, and UniPC marginally outperform CAB. The qualitative results in Figures 1 and 4 are consistent with the ImageReward trends, where, CAB-2 produces sharper samples with fewer noise artifacts, less oversmoothing, and better preservation of prompt-relevant objects such as the bus, racket, book, and fence at 6–8 NFEs. In contrast, competing samplers often exhibit grainy textures, blurred structure, or partial semantic loss.

Table 4: HunyuanVideo evaluation. CAB-2 gives the best overall final score across most step budgets.

Metric	Method	NFE					Metric	Method	NFE				
		4	6	8	10	20			4	6	8	10	20
Visual	DPM++	45.07	44.88	45.78	47.35	51.74	**Align.**	DPM++	30.87	36.88	40.13	41.04	42.50
	STORK	45.18	47.44	**49.35**	50.47	**53.09**		STORK	35.30	37.61	40.00	40.53	42.06
	CAB-2	**46.08**	**47.70**	49.03	**50.54**	52.71		CAB-2	**37.92**	**39.80**	40.97	42.12	**43.17**
	CAB-3	46.03	47.05	47.88	49.33	51.53		CAB-3	37.72	39.67	**41.41**	**42.34**	42.73
Temp.	DPM++	62.22	62.88	63.04	63.13	**63.13**	**Final**	DPM++	138	146	149	152	157
	STORK	**63.71**	**63.47**	**63.22**	**63.30**	63.03		STORK	144	149	152	154	158
	CAB-2	62.84	62.72	62.78	63.04	62.86		CAB-2	**147**	**150**	**153**	**156**	**159**
	CAB-3	62.73	62.42	62.45	62.73	62.72		CAB-3	146	149	151	154	157

Text-to-video generation with HunyuanVideo. We evaluate CAB on HunyuanVideo, a text-to-video generator trained with flow matching [42]. Table 4 reports low-NFE results averaged over 200 EvalCrafter prompts [43], where 20-frame videos at 512×320 resolution are generated. We evaluate visual quality, temporal consistency, and text–video alignment, where alignment combines CLIP similarity [44] with BLIP-BLEU [45]. CAB-2 ($\gamma = 0.9$) achieves the best final score for most NFE budgets, despite variation in individual metrics. It is particularly strong in text–video alignment at 4 and 6 NFEs, while matching strong baselines in visual quality and temporal consistency. These results suggest that CAB improves low-NFE sampling for video generation, beyond image settings.



(a) DiT on ImageNet 256×256. (b) FID–NIQE trade-off.

Figure 6: Ablation of the CAB correction weight γ on DiT/ImageNet 256×256. Stronger correction improves low-NFE FID, while moderate correction yields better NIQE.

Table 5: Runtime (in sec) and maximum GPU memory allocated (in GB) across samplers for QWEN-Image.

Sampler	Time	GPU Mem.
STORK	9.9620	57.949
DPM++	9.9679	57.940
AB2	9.9539	57.941
AB3	9.9710	57.942
CAB-2	9.9556	57.942
CAB-3	9.9731	57.942

Table 6: Transition-wise variation of the learned ODE field before and after noise-to-signal rectification. Entries report normalized change per grid interval, with consecutive field change in parentheses.

NFE	Grid	$2 \to 3$	$3 \to 4$	$4 \to 5$	$5 \to 6$
6	t-space	541.03 (90.17)	470.22 (78.37)	482.09 (80.35)	646.12 (107.69)
	ρ-space	**12.79** (**38.37**)	**39.61** (**39.61**)	**103.56** (**51.78**)	**280.98** (**84.29**)

NFE	Grid	$2 \to 3$	$3 \to 4$	$4 \to 5$	$5 \to 6$	$6 \to 7$	$7 \to 8$
8	t-space	981.57 (122.70)	929.34 (116.17)	687.91 (85.99)	588.78 (73.60)	587.45 (73.43)	838.06 (104.76)
	ρ-space	**8.83** (**35.33**)	**32.98** (**43.98**)	**62.86** (**41.90**)	**119.28** (**47.71**)	**212.67** (**56.71**)	**486.83** (**92.73**)

Ablation studies Figure 6 studies the effect of the CAB correction weight γ on DiT/ImageNet model. Nonzero correction substantially improves low-NFE performance over the uncorrected case $\gamma = 0$, especially in the aggressive 4-10 NFE regime. This supports the interpretation of the correction as an extrapolation-defect adjustment. The best value of γ is however metric-dependent. Stronger correction, e.g., $\gamma = 1.0$, can improve FID suggesting better agreement with dataset-level distributional statistics under this metric. In our NIQE evaluations, moderate correction $\gamma = 0.5$ is frequently preferred, indicating a trade-off between distributional metrics and perceptual metrics. Table 6 quantifies the effect of noise-to-signal rectification on field variation along QWEN-Image sampling trajectories, averaged over 10 prompts. In the original coordinates, ODE uses the learned velocity field, whereas after rectification the right-hand side becomes the learned noise field in Lemma 3.1. The rectified ρ-space dynamics exhibit substantially smaller, more structured transition-wise changes at 6 and 10 NFEs, indicating a smoother field for multistep updates and supporting the

Table 7: Stochastic sampling on DiT/ImageNet-256 ($\eta = 0.1$). FID $\downarrow$.

NFE	Sampler		
	DDIM	CAB-SDE	CAB-ODE
8	39.83	19.45	**9.21**
10	24.40	12.00	**7.99**
15	12.90	8.09	**7.09**

Table 8: Generalizability of the correction on CIFAR-10 (VP). FID $\downarrow$.

NFE	Euler		Heun		STORK		AB	
	Base	+Corr.	Base	+Corr.	Base	+Corr.	AB2	CAB-2
6	36.94	12.90	85.83	85.76	32.66	23.21	23.78	**7.81**
8	22.21	5.92	35.57	35.27	18.03	6.02	14.42	**4.32**
10	15.54	3.91	14.44	14.34	11.65	3.25	9.99	**3.11**

role of rectification on improved low-NFE behavior of CAB (refer Appendix A.6 for more details). We compare runtime and memory on an A100 GPU for QWEN-Image 1024×1024 generation at 10 NFEs, averaged over 10 prompts with batch size 1 and *torch.bfloat16*. CAB has the same cost range as other samplers, with negligible memory overhead from storing one additional past evaluation.

Extension to Stochastic Sampling. We extend CAB to stochastic sampling by considering the variance-controlled family in noise-to-signal coordinates,

$$d\mathbf{y}_\rho = (1+\eta^2)\boldsymbol{\epsilon}_\theta(s_{t(\rho)}\mathbf{y}_\rho, t(\rho))\, d\rho + \eta\sqrt{2\rho}\, d\bar{\mathbf{W}}_\rho, \tag{11}$$

where $\eta = 0$ recovers the deterministic probability-flow ODE, $\eta = 1$ corresponds to the full reverse SDE, and $0 < \eta < 1$ interpolates between them. We discretize the drift using CAB-2 and sample the Gaussian increments analytically, retaining one network evaluation per step. Table 7 shows that CAB-2-SDE substantially improves over DDIM-SDE [11] on DiT/ImageNet-256 at $\eta = 0.1$, reducing FID from 39.83 to 19.45 at 8 NFEs and from 24.40 to 12.00 at 10 NFEs. Deterministic CAB-2 remains stronger, indicating that stochastic perturbations can make few-step multistep correction more challenging.

Generalizability of the Correction. We evaluate the extrapolation-defect correction with Euler, Heun, and STORK predictors on CIFAR-10 (VP). Table 8 shows substantial improvements for Euler and STORK, reducing FID from 15.54 to 3.91 and 11.65 to 3.25, respectively, at 10 NFEs. Heun shows negligible gains, while CAB-2 consistently achieves the lowest FID, highlighting the effectiveness of the correction with history-based Adams–Bashforth prediction.

5 Conclusion

We introduced CAB, a simple training-free framework for accelerating flow and diffusion sampling by rectifying the reverse-time ODE associated with affine-Gaussian paths. In the resulting noise-to-signal coordinates, CAB applies corrected Adams–Bashforth updates with one network evaluation per step while preserving the second- or third-order accuracy of the underlying predictors. Across our evaluated image and video settings, CAB provides a favorable low-NFE trade-off, often improving perceptual quality or distributional metrics over strong training-free baselines.

Acknowledgments and Disclosure of Funding

This work was supported by the Anusandhan National Research Foundation (ANRF), Government of India, under the Prime Minister Early Career Research Grant (PM-ECRG), Grant No. `ANRF/ECRG/2024/001835/ENS`.

Broader impact. CAB reduces the inference cost of flow and diffusion generation by improving sample quality at low NFE budgets. This can lower latency, energy use, and hardware requirements, making generative models more accessible in resource-constrained settings. However, faster sampling can also increase the scale at which synthetic or misleading visual content is generated. Deployment should therefore be accompanied by watermarking or misuse-detection mechanisms.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Proc. Advances in neural information processing systems*, 33:6840–6851, 2020.

[2] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. *Proc. International Conference on Learning Representations*, 2023.

[3] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. *Proc. Advances in neural information processing systems*, 35:8633–8646, 2022.

[4] Yi Huang, Jiancheng Huang, Yifan Liu, Mingfu Yan, Jiaxi Lv, Jianzhuang Liu, Wei Xiong, He Zhang, Shifeng Chen, and Liangliang Cao. Diffusion model-based image editing: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):4409–4437, 2025.

[5] Chitwan Saharia, William Chan, Huiwen Chang, Chris A. Lee, Jonathan Ho, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[6] Andreas Lugmayr, Martin Danelljan, Andrés Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[7] Xin Li, Yulin Ren, Xin Jin, Cuiling Lan, Xingrui Wang, Wenjun Zeng, Xinchao Wang, and Zhibo Chen. Diffusion models for image restoration and enhancement: A comprehensive survey. *International Journal of Computer Vision*, 133(11):8078–8108, 2025.

[8] Yimu Wang, Xuye Liu, Wei Pang, Li Ma, Shuai Yuan, Paul Debevec, and Ning Yu. Survey of video diffusion models: Foundations, implementations, and applications. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

[9] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *Proc. International Conference on Learning Representations*, 2021.

[10] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Proc. Advances in neural information processing systems*, 35:5775–5787, 2022.

[11] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *Proc. International Conference on Learning Representations*, 2021.

[12] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Proc. Advances in Neural Information Processing Systems*, 36:49842–49869, 2023.

[13] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, pages 1–22, 2025.

[14] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *Proc. International Conference on Learning Representations*, 2022.

[15] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *Proc. International Conference on Learning Representations*, 2023.

[16] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. *Proc. Advances in Neural Information Processing Systems*, 36:55502–55542, 2023.

[17] Zheng Tan, Weizhen Wang, Andrea L. Bertozzi, and Ernest K. Ryu. STORK: Faster diffusion and flow matching sampling by resolving both stiffness and structure-dependence. *Proc. International Conference on Learning Representations*, 2026.

[18] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

[19] William Peebles and Saining Xie. Scalable diffusion models with transformers. *Proc. IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[20] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Proc. Advances in Neural Information Processing Systems*, 31, 2018.

[21] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *Proc. International Conference on Learning Representations*, 2019.

[22] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Proc. Advances in neural information processing systems*, 35:26565–26577, 2022.

[23] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *Proc. International Conference on Learning Representations*, 2023.

[24] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Proc. Advances in neural information processing systems*, 34:21696–21707, 2021.

[25] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *Proc. International Conference on Learning Representations*, 2022.

[26] Nikita Starodubcev, Ilya Drobyshevskiy, Denis Kuznedelev, Artem Babenko, and Dmitry Baranchuk. Scale-wise distillation of diffusion models. *Proc. International Conference on Learning Representations*, 2026.

[27] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *Proc. International Conference on Machine Learning*, pages 32211–32252, 2023.

[28] Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. *Proc. International Conference on Learning Representations*, 2025.

[29] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. *Proc. International Conference on Learning Representations*, 2024.

[30] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6613–6623, 2024.

[31] Junsong Chen, Shuchen Xue, Yuyang Zhao, Jincheng Yu, Sayak Paul, Junyu Chen, Han Cai, Song Han, and Enze Xie. Sana-sprint: One-step diffusion with continuous-time consistency distillation. *Proc. IEEE/CVF International Conference on Computer Vision*, pages 16185–16195, 2025.

[32] Shuchen Xue, Mingyang Yi, Weijian Luo, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhi-Ming Ma. Sa-solver: Stochastic adams solver for fast sampling of diffusion models. *Proc. Advances in Neural Information Processing Systems*, 36:77632–77674, 2023.

[33] Yilun Xu, Mingyang Deng, Xiang Cheng, Yonglong Tian, Ziming Liu, and Tommi Jaakkola. Restart sampling for improving generative processes. *Proc. Advances in Neural Information Processing Systems*, 36:76806–76838, 2023.

[34] Neta Shaul, Juan Perez, Ricky T. Q. Chen, Ali Thabet, Albert Pumarola, and Yaron Lipman. Bespoke solvers for generative flow models. *Proc. International Conference on Learning Representations*, 2024.

[35] Eric Frankel, Sitan Chen, Jerry Li, Pang Wei Koh, Lillian J. Ratliff, and Sewoong Oh. S4s: Solving for a fast diffusion model solver. *Proc. International Conference on Machine Learning*, 2025.

[36] Ernst Hairer, Syvert P. Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer, 2 edition, 1993.

[37] J. C. Butcher. Numerical methods for ordinary differential equations in the 20th century. *Journal of Computational and Applied Mathematics*, 125(1–2):1–29, 2000.

[38] Uri M. Ascher and Linda R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, 1998.

[39] Klaus Böhmer, P. W. Hemker, and H. J. Stetter. The defect correction approach. In Klaus Böhmer and H. J. Stetter, editors, *Defect Correction Methods: Theory and Applications*, volume 5 of *Computing Supplementum*, pages 1–32. Springer, Vienna, 1984. doi: 10.1007/978-3-7091-7023-6_1.

[40] Benjamin W. Ong and Raymond J. Spiteri. Deferred correction methods for ordinary differential equations. *Journal of Scientific Computing*, 83(3):60, 2020. doi: 10.1007/s10915-020-01235-8.

[41] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Proc. Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

[42] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

[43] Yaofang Liu, Xiaodong Cun, Xuebo Liu, Xintao Wang, Yong Zhang, Haoxin Chen, Yang Liu, Tieyong Zeng, Raymond Chan, and Ying Shan. Evalcrafter: Benchmarking and evaluating large video generation models. *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22139–22149, 2024.

[44] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *Proc. International Conference on Machine Learning*, pages 8748–8763, 2021.

[45] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. *Proc. International Conference on Machine Learning*, pages 12888–12900, 2022.

[46] Jan Verschelde. Variable step methods, 2022. Lecture notes for MCS 471, University of Illinois Chicago.

[47] Xiangyu Fan, Zesong Qiu, Fanzhou Wang, Peng Gao, and Lei Yang. Qwen-image-lightning. `https://github.com/ModelTC/LightX2V-Qwen-Image-Lightning`, 2025.

[48] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15853–15863, 2025.

A Theoretical results and additional experiments

A.1 Proof of Lemma 3.1

Starting from the reverse-time ODE in (3),

$$\frac{d\boldsymbol{x}_t}{dt} = \frac{\dot{s}_t}{s_t}\boldsymbol{x}_t + \left(\dot{\sigma}_t - \frac{\dot{s}_t}{s_t}\sigma_t\right)\epsilon_\theta(\boldsymbol{x}_t, t). \tag{12}$$

Consider the rescaled state.

$$\boldsymbol{y}_t := \frac{\boldsymbol{x}_t}{s_t}, \qquad \text{so that} \qquad \boldsymbol{x}_t = s_t\boldsymbol{y}_t.$$

Differentiating $\boldsymbol{x}_t = s_t\boldsymbol{y}_t$ with respect to t gives

$$\frac{d\boldsymbol{x}_t}{dt} = \dot{s}_t\,\boldsymbol{y}_t + s_t\frac{d\boldsymbol{y}_t}{dt}.$$

Substituting this into (12), we obtain

$$\dot{s}_t\, \boldsymbol{y}_t + s_t \frac{d\boldsymbol{y}_t}{dt} = \frac{\dot{s}_t}{s_t}(s_t \boldsymbol{y}_t) + \left(\dot{\sigma}_t - \frac{\dot{s}_t}{s_t}\sigma_t\right) \epsilon_\theta(s_t \boldsymbol{y}_t, t).$$

Canceling out common terms and rearranging gives,

$$\frac{d\boldsymbol{y}_t}{dt} = \frac{1}{s_t}\left(\dot{\sigma}_t - \frac{\dot{s}_t}{s_t}\sigma_t\right) \epsilon_\theta(s_t \boldsymbol{y}_t, t). \tag{13}$$

The noise-to-signal ratio is defined as

$$\rho_t := \frac{\sigma_t}{s_t}.$$

Differentiating ρ_t with respect to t,

$$\dot{\rho}_t = \frac{\dot{\sigma}_t s_t - \sigma_t \dot{s}_t}{s_t^2} = \frac{1}{s_t}\left(\dot{\sigma}_t - \frac{\dot{s}_t}{s_t}\sigma_t\right). \tag{14}$$

Using this identity in (13), we get

$$\frac{d\boldsymbol{y}_t}{dt} = \dot{\rho}_t\, \epsilon_\theta(s_t \boldsymbol{y}_t, t). \tag{15}$$

By assumption, $\rho : [0, T] \to \mathbb{R}_{>0}$ is strictly increasing, so the inverse map $t = t(\rho)$ exists. The ρ-reparameterized trajectory is defined as

$$\hat{\boldsymbol{y}}_\rho := \boldsymbol{y}_{t(\rho)}.$$

Applying the chain rule,

$$\frac{d\hat{\boldsymbol{y}}_\rho}{d\rho} = \frac{d\boldsymbol{y}_t}{dt}\Big|_{t=t(\rho)} \cdot \frac{dt}{d\rho}.$$

Since $\frac{dt}{d\rho} = \frac{1}{\dot{\rho}_{t(\rho)}}$, we get

$$\frac{d\hat{\boldsymbol{y}}_\rho}{d\rho} = \frac{1}{\dot{\rho}_{t(\rho)}} \frac{d\boldsymbol{y}_t}{dt}\Big|_{t=t(\rho)}.$$

Using (15) evaluated at $t = t(\rho)$,

$$\frac{d\boldsymbol{y}_t}{dt}\Big|_{t=t(\rho)} = \dot{\rho}_{t(\rho)}\, \epsilon_\theta\big(s_{t(\rho)} \boldsymbol{y}_{t(\rho)}, t(\rho)\big).$$

Substituting the above,

$$\frac{d\hat{\boldsymbol{y}}_\rho}{d\rho} = \frac{1}{\dot{\rho}_{t(\rho)}} \dot{\rho}_{t(\rho)} \epsilon_\theta\big(s_{t(\rho)} \boldsymbol{y}_{t(\rho)}, t(\rho)\big) = \epsilon_\theta\big(s_{t(\rho)} \boldsymbol{y}_{t(\rho)}, t(\rho)\big) = \epsilon_\theta\big(s_{t(\rho)} \hat{\boldsymbol{y}}_\rho, t(\rho)\big),$$

which proves the lemma.

A.2 Relation between noise/data/velocity parameterizations (6) in rectified coordinates

From the standard data-prediction parameterization, we have

$$\epsilon_\theta(\boldsymbol{x}_t, t) = \frac{\boldsymbol{x}_t - s_t \hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t)}{\sigma_t}.$$

Now substituting $\boldsymbol{x}_t = s_{t(\rho)} \hat{\boldsymbol{y}}_\rho$ and using $\rho = \frac{\sigma_{t(\rho)}}{s_{t(\rho)}}$, gives

$$\epsilon_\theta(s_{t(\rho)} \hat{\boldsymbol{y}}_\rho, t(\rho)) = \frac{s_{t(\rho)} \hat{\boldsymbol{y}}_\rho - s_{t(\rho)} \hat{\boldsymbol{x}}_\theta(s_{t(\rho)} \hat{\boldsymbol{y}}_\rho, t(\rho))}{\sigma_{t(\rho)}} = \frac{\hat{\boldsymbol{y}}_\rho - \hat{\boldsymbol{x}}_\theta(s_{t(\rho)} \hat{\boldsymbol{y}}_\rho, t(\rho))}{\rho}.$$

Similarly, starting from

$$\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) = \dot{s}_t\, \hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t) + \dot{\sigma}_t\, \epsilon_\theta(\boldsymbol{x}_t, t),$$

we substitute $\boldsymbol{x}_t = s_{t(\rho)}\hat{\boldsymbol{y}}_\rho$ and $\hat{\boldsymbol{x}}_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) = \hat{\boldsymbol{y}}_\rho - \rho\,\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho))$ to obtain

$$\begin{aligned}
\boldsymbol{v}_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) &= \dot{s}_{t(\rho)}\,\hat{\boldsymbol{x}}_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) + \dot{\sigma}_{t(\rho)}\,\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) \\
&= \dot{s}_{t(\rho)}\big(\hat{\boldsymbol{y}}_\rho - \rho\,\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho))\big) + \dot{\sigma}_{t(\rho)}\,\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) \\
&= \dot{s}_{t(\rho)}\,\hat{\boldsymbol{y}}_\rho + (\dot{\sigma}_{t(\rho)} - \dot{s}_{t(\rho)}\rho)\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) \\
&= \dot{s}_{t(\rho)}\,\hat{\boldsymbol{y}}_\rho + s_{t(\rho)}\dot{\rho}_{t(\rho)}\,\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)),
\end{aligned}$$

The last equality is obtained from the form of $\dot{\rho_{t(\rho)}}$ in (14). Rearranging terms, we get,

$$\epsilon_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) = \frac{\boldsymbol{v}_\theta(s_{t(\rho)}\hat{\boldsymbol{y}}_\rho, t(\rho)) - \dot{s}_{t(\rho)}\hat{\boldsymbol{y}}_\rho}{s_{t(\rho)}\dot{\rho}_{t(\rho)}}.$$

A.3 Derivation of CAB-2 and CAB-3 updates

The CAB-2 prediction step corresponds to the standard variable-step AB2 formula, whose derivation can be found, for example, in [46]. Likewise, the CAB-3 prediction step follows the standard variable-step AB3 construction; see [36]. For completeness, we present below the derivations of the CAB-2 predictor, the CAB-3 predictor, and the proposed correction term.

Prediction step for CAB-2. To derive the CAB-2 predictor, we approximate the velocity in (5) over the interval $[\rho_{t_i}, \rho_{t_{i+1}}]$ by the degree-1 Lagrange interpolant through the two most recent points (ρ_{t_i}, ϵ_i) and $(\rho_{t_{i-1}}, \epsilon_{i-1})$, namely

$$\mathcal{L}_1(\rho) = \frac{\rho - \rho_{t_{i-1}}}{\rho_{t_i} - \rho_{t_{i-1}}}\,\epsilon_i + \frac{\rho - \rho_{t_i}}{\rho_{t_{i-1}} - \rho_{t_i}}\,\epsilon_{i-1}.$$

Integrating the rectified ODE (5) with this approximation gives

$$\boldsymbol{y}_{t_{i+1}} \approx \boldsymbol{y}_{t_i} + \int_{\rho_{t_i}}^{\rho_{t_{i+1}}} \mathcal{L}_1(\rho)\,d\rho.$$

Since

$$\rho_{t_i} - \rho_{t_{i-1}} = h_{i-1}, \qquad \rho_{t_{i+1}} - \rho_{t_i} = h_i, \qquad r_i = \frac{h_i}{h_{i-1}},$$

we obtain

$$\begin{aligned}
\int_{\rho_{t_i}}^{\rho_{t_{i+1}}} \mathcal{L}_1(\rho)\,d\rho &= \left(\int_{\rho_{t_i}}^{\rho_{t_{i+1}}} \frac{\rho - \rho_{t_{i-1}}}{h_{i-1}}\,d\rho\right)\epsilon_i - \left(\int_{\rho_{t_i}}^{\rho_{t_{i+1}}} \frac{\rho - \rho_{t_i}}{h_{i-1}}\,d\rho\right)\epsilon_{i-1} \\
&= \left(h_i + \frac{h_i^2}{2h_{i-1}}\right)\epsilon_i - \frac{h_i^2}{2h_{i-1}}\epsilon_{i-1} \\
&= h_i\left[\left(1 + \frac{r_i}{2}\right)\epsilon_i - \frac{r_i}{2}\epsilon_{i-1}\right].
\end{aligned}$$

This is exactly the standard non-uniform Adams–Bashforth prediction as derived in [46].

Prediction step for CAB-3. Similarly, CAB-3 is obtained by approximating the velocity in (5) over the interval $[\rho_{t_i}, \rho_{t_{i+1}}]$ by the degree-2 Lagrange interpolant through the three most recent points (ρ_{t_i}, ϵ_i), $(\rho_{t_{i-1}}, \epsilon_{i-1})$, and $(\rho_{t_{i-2}}, \epsilon_{i-2})$, namely

$$\mathcal{L}_2(\rho) = \ell_i(\rho)\epsilon_i + \ell_{i-1}(\rho)\epsilon_{i-1} + \ell_{i-2}(\rho)\epsilon_{i-2},$$

where

$$\ell_i(\rho) = \frac{(\rho - \rho_{t_{i-1}})(\rho - \rho_{t_{i-2}})}{(\rho_{t_i} - \rho_{t_{i-1}})(\rho_{t_i} - \rho_{t_{i-2}})},$$

$$\ell_{i-1}(\rho) = \frac{(\rho - \rho_{t_i})(\rho - \rho_{t_{i-2}})}{(\rho_{t_{i-1}} - \rho_{t_i})(\rho_{t_{i-1}} - \rho_{t_{i-2}})}, \qquad \ell_{i-2}(\rho) = \frac{(\rho - \rho_{t_i})(\rho - \rho_{t_{i-1}})}{(\rho_{t_{i-2}} - \rho_{t_i})(\rho_{t_{i-2}} - \rho_{t_{i-1}})}.$$

Integrating the rectified ODE (5) with this approximation gives

$$\boldsymbol{y}_{t_{i+1}} \approx \boldsymbol{y}_{t_i} + \int_{\rho_{t_i}}^{\rho_{t_{i+1}}} \mathcal{L}_2(\rho)\,d\rho.$$

Using

$$h_i = \rho_{t_{i+1}} - \rho_{t_i}, \qquad h_{i-1} = \rho_{t_i} - \rho_{t_{i-1}}, \qquad h_{i-2} = \rho_{t_{i-1}} - \rho_{t_{i-2}},$$

The integral can be written as

$$\int_{\rho_{t_i}}^{\rho_{t_{i+1}}} \mathcal{L}_2(\rho)\, d\rho = h_i \left(\beta_0 \epsilon_i + \beta_1 \epsilon_{i-1} + \beta_2 \epsilon_{i-2}\right),$$

where

$$\beta_0 = \frac{\frac{h_i^2}{3} + \frac{h_i}{2}(2h_{i-1} + h_{i-2}) + h_{i-1}(h_{i-1} + h_{i-2})}{h_{i-1}(h_{i-1} + h_{i-2})},$$

$$\beta_1 = -\frac{h_i(2h_i + 3h_{i-1} + 3h_{i-2})}{6h_{i-1}h_{i-2}}, \qquad \beta_2 = \frac{h_i(2h_i + 3h_{i-1})}{6h_{i-2}(h_{i-1} + h_{i-2})}.$$

Equivalently, in terms of the step ratios

$$r_i := \frac{h_i}{h_{i-1}}, \qquad r_{i-1} := \frac{h_{i-1}}{h_{i-2}},$$

These coefficients become

$$\beta_0 = 1 + \frac{r_i(2r_{i-1} + 1)}{2(r_{i-1} + 1)} + \frac{r_{i-1}r_i^2}{3(r_{i-1} + 1)},$$

$$\beta_1 = -\frac{r_i}{6}\left(2r_{i-1}r_i + 3r_{i-1} + 3\right), \qquad \beta_2 = \frac{r_{i-1}^2 r_i(2r_i + 3)}{6(r_{i-1} + 1)}.$$

This yields the explicit variable-step AB3 predictor. The derivation follows the standard variable-step Adams construction on non-uniform grids, obtained by integrating the quadratic Lagrange interpolant of past drift evaluations; we include the resulting closed-form coefficients here for completeness and refer to [36] for the general derivation.

Correction step. To derive the correction term, we approximate the velocity at the current point ρ_{t_i} by the degree-1 Lagrange extrapolant built from the two previous evaluations $(\rho_{t_{i-1}}, \epsilon_{i-1})$ and $(\rho_{t_{i-2}}, \epsilon_{i-2})$. This extrapolant is

$$\mathcal{L}_1^{\text{ext}}(\rho) = \frac{\rho - \rho_{t_{i-2}}}{\rho_{t_{i-1}} - \rho_{t_{i-2}}}\, \epsilon_{i-1} + \frac{\rho - \rho_{t_{i-1}}}{\rho_{t_{i-2}} - \rho_{t_{i-1}}}\, \epsilon_{i-2}.$$

Evaluating at $\rho = \rho_{t_i}$ gives

$$\epsilon_i^{\text{ext}} = \mathcal{L}_1^{\text{ext}}(\rho_{t_i}) = \frac{\rho_{t_i} - \rho_{t_{i-2}}}{\rho_{t_{i-1}} - \rho_{t_{i-2}}}\, \epsilon_{i-1} - \frac{\rho_{t_i} - \rho_{t_{i-1}}}{\rho_{t_{i-1}} - \rho_{t_{i-2}}}\, \epsilon_{i-2}.$$

Using

$$\rho_{t_i} - \rho_{t_{i-1}} = h_{i-1}, \qquad \rho_{t_{i-1}} - \rho_{t_{i-2}} = h_{i-2}, \qquad \rho_{t_i} - \rho_{t_{i-2}} = h_{i-1} + h_{i-2},$$

we obtain

$$\epsilon_i^{\text{ext}} = \left(1 + \frac{h_{i-1}}{h_{i-2}}\right)\epsilon_{i-1} - \frac{h_{i-1}}{h_{i-2}}\,\epsilon_{i-2} = (1 + r_{i-1})\epsilon_{i-1} - r_{i-1}\epsilon_{i-2}, \qquad r_{i-1} = \frac{h_{i-1}}{h_{i-2}}.$$

The correction is then defined by the extrapolation error,

$$\epsilon_i - \epsilon_i^{\text{ext}},$$

scaled by the current step size h_i.

A.4 Proof of Theorem 3.2

We use a unified argument for CAB-2 and CAB-3. Let

$$\boldsymbol{z}_i := \hat{\boldsymbol{y}}(\rho_{t_i}), \qquad \boldsymbol{u}_i := \hat{\boldsymbol{y}}'(\rho_{t_i}), \qquad e_i := \boldsymbol{y}_{t_i} - \boldsymbol{z}_i.$$

The update in Algorithm 1 of the form

$$\boldsymbol{y}_{t_{i+1}} = \boldsymbol{y}_{t_i} + h_i \sum_{m=0}^{2} a_{m,i}\, \epsilon_{i-m}, \tag{16}$$

where CAB-2 and CAB-3 have different coefficients $a_{m,i}$. The corresponding exact-solution error is

$$\tau_{i+1} := \boldsymbol{z}_{i+1} - \boldsymbol{z}_i - h_i \sum_{m=0}^{q} a_{m,i} \boldsymbol{u}_{i-m}. \tag{17}$$

For CAB-2, the coefficients are

$$a_{0,i} = 1 + \frac{r_i}{2} + \gamma_i, \qquad a_{1,i} = -\Big(\frac{r_i}{2} + (1 + r_{i-1})\gamma_i\Big), \qquad a_{2,i} = r_{i-1}\gamma_i, \tag{18}$$

where

$$r_i = \frac{h_i}{h_{i-1}}, \qquad r_{i-1} = \frac{h_{i-1}}{h_{i-2}}.$$

For CAB-3, letting $\beta_{0,i}, \beta_{1,i}, \beta_{2,i}$ denote the variable-step AB3 coefficients in the companion predictor box of Algorithm 1, the corrected coefficients are

$$a_{0,i} = \beta_{0,i} + \gamma_i, \qquad a_{1,i} = \beta_{1,i} - (1 + r_{i-1})\gamma_i, \qquad a_{2,i} = \beta_{2,i} + r_{i-1}\gamma_i. \tag{19}$$

Step 1: local truncation error. Expand the exact increment about ρ_{t_i} using Taylor series:

$$\boldsymbol{z}_{i+1} - \boldsymbol{z}_i = h_i \boldsymbol{z}_i' + \frac{h_i^2}{2}\boldsymbol{z}_i'' + \frac{h_i^3}{6}\boldsymbol{z}_i''' + \frac{h_i^4}{24}\boldsymbol{z}_i'''' + O(h_i^5). \tag{20}$$

where $p = 2$ for CAB-2 and $p = 3$ for CAB-3. Next expand the past velocities $\boldsymbol{u}_{i-1}$ and $\boldsymbol{u}_{i-2}$ about ρ_{t_i}. Since

$$\rho_{t_{i-1}} - \rho_{t_i} = -\frac{h_i}{r_i}, \qquad \rho_{t_{i-2}} - \rho_{t_i} = -\frac{h_i(1 + r_{i-1})}{r_i r_{i-1}},$$

we obtain

$$\boldsymbol{u}_{i-1} = \boldsymbol{u}_i - \frac{h_i}{r_i}\boldsymbol{z}_i'' + \frac{h_i^2}{2r_i^2}\boldsymbol{z}_i''' - \frac{h_i^3}{6r_i^3}\boldsymbol{z}_i'''' + O(h_i^4), \tag{21}$$

$$\boldsymbol{u}_{i-2} = \boldsymbol{u}_i - \frac{h_i(1 + r_{i-1})}{r_i r_{i-1}}\boldsymbol{z}_i'' + \frac{h_i^2(1 + r_{i-1})^2}{2r_i^2 r_{i-1}^2}\boldsymbol{z}_i''' - \frac{h_i^3(1 + r_{i-1})^3}{6r_i^3 r_{i-1}^3}\boldsymbol{z}_i'''' + O(h_i^4). \tag{22}$$

Substituting (21)–(22) into (17) yields

$$\begin{aligned}\tau_{i+1} = {} & h_i\Big[1 - \sum_{m=0}^{2} a_{m,i}\Big]\boldsymbol{z}_i' + h_i^2\left[\frac{1}{2} + \frac{a_{1,i}}{r_i} + \frac{a_{2,i}(1 + r_{i-1})}{r_i r_{i-1}}\right]\boldsymbol{z}_i'' \\ & + h_i^3\left[\frac{1}{6} - \frac{a_{1,i}}{2r_i^2} - \frac{a_{2,i}(1 + r_{i-1})^2}{2r_i^2 r_{i-1}^2}\right]\boldsymbol{z}_i''' + h_i^4\left[\frac{1}{24} + \frac{a_{1,i}}{6r_i^3} + \frac{a_{2,i}(1 + r_{i-1})^3}{6r_i^3 r_{i-1}^3}\right]\boldsymbol{z}_i'''' + O(h_i^5)\end{aligned} \tag{23}$$

CAB-2. Using (18), we can verify that

$$a_{0,i} + a_{1,i} + a_{2,i} = 1,$$

and

$$\frac{a_{1,i}}{r_i} + \frac{a_{2,i}(1 + r_{i-1})}{r_i r_{i-1}} = -\frac{1}{2}.$$

Hence the $O(h_i)$ and $O(h_i^2)$ terms in (23) cancel, leaving

$$\tau_{i+1} = O(h_i^3).$$

Thus, CAB-2 has local truncation order 3.

CAB-3. For the variable-step AB3 predictor coefficients $\beta_{0,i}, \beta_{1,i}, \beta_{2,i}$, the third-order consistency conditions hold:

$$\begin{aligned}
\beta_{0,i} + \beta_{1,i} + \beta_{2,i} &= 1, \\
\frac{\beta_{1,i}}{r_i} + \frac{\beta_{2,i}(1+r_{i-1})}{r_i r_{i-1}} &= -\frac{1}{2}, \\
\frac{\beta_{1,i}}{2r_i^2} + \frac{\beta_{2,i}(1+r_{i-1})^2}{2r_i^2 r_{i-1}^2} &= -\frac{1}{6}.
\end{aligned}$$

The correction modifies the AB3 coefficients according to

$$a_{0,i} = \beta_{0,i} + \gamma_i, \qquad a_{1,i} = \beta_{1,i} - (1+r_{i-1})\gamma_i, \qquad a_{2,i} = \beta_{2,i} + r_{i-1}\gamma_i.$$

$O(h_i)$ and $O(h_i^2)$ terms still cancel since the additional terms satisfy

$$\begin{aligned}
\gamma_i - (1+r_{i-1})\gamma_i + r_{i-1}\gamma_i &= 0, \\
\frac{-(1+r_{i-1})\gamma_i}{r_i} + \frac{r_{i-1}\gamma_i(1+r_{i-1})}{r_i r_{i-1}} &= 0
\end{aligned}$$

Substituting $a_{1,i}, a_{2,i}$ into the coefficient of $h_i^3 \boldsymbol{z}_i'''$ in (23), we obtain

$$\frac{1}{6} - \frac{a_{1,i}}{2r_i^2} - \frac{a_{2,i}(1+r_{i-1})^2}{2r_i^2 r_{i-1}^2} = -\gamma_i \, \frac{1+r_{i-1}}{2r_i^2 r_{i-1}}.$$

Therefore,

$$\tau_{i+1} = \gamma_i \, h_i^3 \, \kappa_i \, \boldsymbol{z}_i''' + O(h_i^4), \qquad \kappa_i := -\frac{1+r_{i-1}}{2r_i^2 r_{i-1}}. \tag{24}$$

Since the step ratios are uniformly bounded away from 0 and ∞, κ_i is uniformly bounded. Consequently, if $\gamma_i = O(1)$, then $\tau_{i+1} = O(h_i^3)$, whereas if $\gamma_i = O(h_i)$, then $\tau_{i+1} = O(h_i^4)$. Hence CAB-3 has local truncation order 4 under the condition $\gamma_i = O(h_i)$.

Step 2: global error. The exact solution satisfies

$$\boldsymbol{z}_{i+1} = \boldsymbol{z}_i + h_i \sum_{m=0}^{2} a_{m,i} \boldsymbol{u}_{i-m} + \tau_{i+1},$$

while the numerical scheme satisfies

$$\boldsymbol{y}_{t_{i+1}} = \boldsymbol{y}_{t_i} + h_i \sum_{m=0}^{2} a_{m,i} \epsilon_{i-m}.$$

Subtracting gives

$$e_{i+1} = e_i + h_i \sum_{m=0}^{2} a_{m,i} \Delta_{i-m} - \tau_{i+1}, \tag{25}$$

where

$$\Delta_j := \epsilon_\theta(s_{t_j} \boldsymbol{y}_{t_j}, t_j) - \epsilon_\theta(s_{t_j} \boldsymbol{z}_j, t_j).$$

By the Lipschitz assumption,

$$\|\Delta_j\| \le L \|e_j\|.$$

Since the step ratios are uniformly bounded and the corrector weights are bounded, the coefficients $a_{m,i}$ are uniformly bounded. Thus there exists $B > 0$ such that

$$\sum_{m=0}^{2} |a_{m,i}| \le B.$$

Taking norms in (25), we obtain

$$\|e_{i+1}\| \le \|e_i\| + BLh_i \max\{\|e_i\|, \|e_{i-1}\|, \|e_{i-2}\|\} + \|\tau_{i+1}\|.$$

Define

$$E_i := \max_{0\le j\le i} \|e_j\|.$$

Then

$$E_{i+1} \le (1 + BLh_i)E_i + \|\tau_{i+1}\|.$$

Iterating this recurrence and using $1 + x \le e^x$ to form

$$\prod_{j=k+1}^{n-1} (1 + BLh_j) \le \exp\big(BL(\rho_{t_0} - \rho_{t_N})\big),$$

yields a discrete Grönwall bound of the form

$$E_n \le C_G \left(E_2 + \sum_{k=2}^{n-1} \|\tau_{k+1}\| \right). \tag{26}$$

CAB-2. For CAB-2, the local truncation bound is

$$\|\tau_{k+1}\| \le C_\tau h_k^3.$$

Substituting this into (26) gives

$$E_n \le C_G E_2 + C_G C_\tau \sum_{k=2}^{n-1} h_k^3.$$

Since $h_k \le h_{\max}$,

$$\sum_{k=2}^{n-1} h_k^3 \le h_{\max}^2 \sum_{k=2}^{n-1} h_k \le h_{\max}^2 (\rho_{t_N} - \rho_{t_0}).$$

By assumption, the starting iterates satisfy the target global-order bound, so we get

$$E_n \le C h_{\max}^2,$$

for some constant $C > 0$ independent of step sizes $\{h_i\}$.

CAB-3. For CAB-3, under the condition $\gamma_i = O(h_i)$, the local truncation bound is

$$\|\tau_{k+1}\| \le C_\tau h_k^4.$$

Similar to CAB-2, using $h_k \le h_{\max}$,

$$\sum_{k=2}^{n-1} h_k^4 \le h_{\max}^3 \sum_{k=2}^{n-1} h_k \le h_{\max}^3 (\rho_{t_N} - \rho_{t_0}).$$

and since the starting iterates are assumed accurate to the target global order, we get

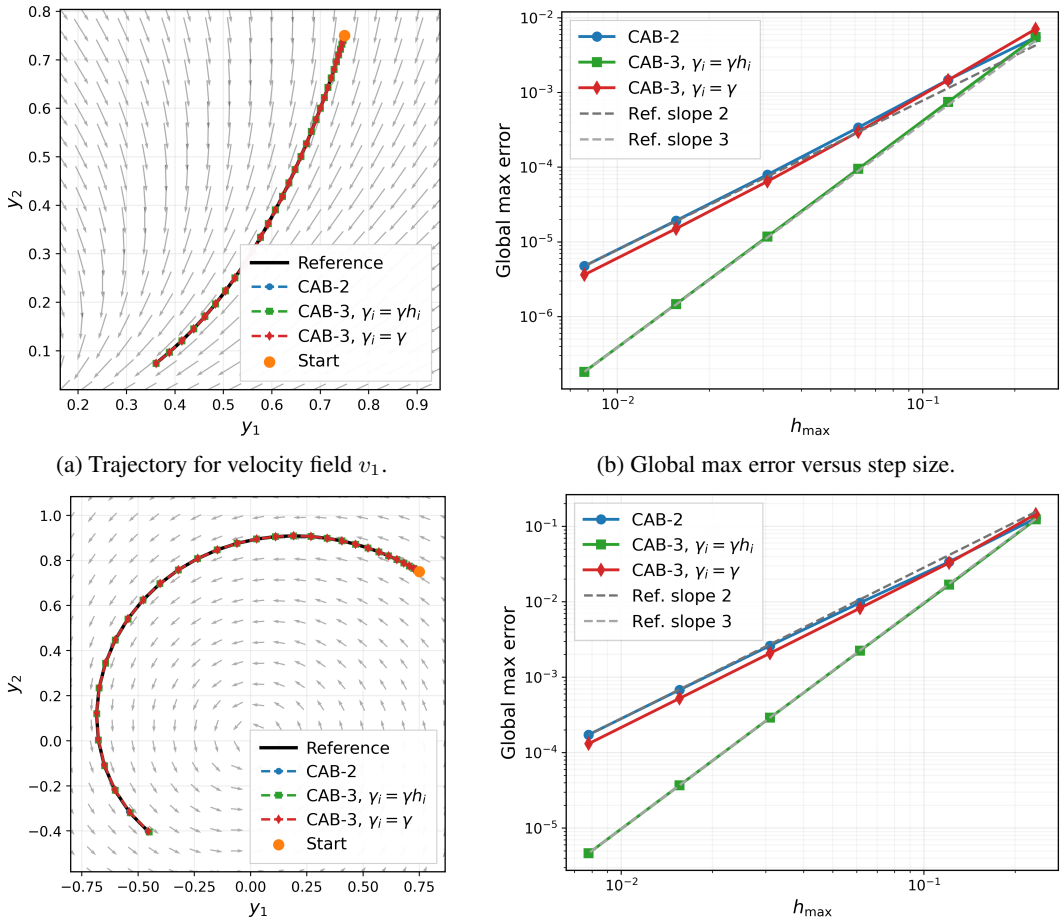
$$E_n \le C h_{\max}^3,$$

for some constant $C > 0$ independent of step sizes $\{h_i\}$.

A.5 Empirical demonstration of Theorem 3.2

We next verify the theoretical convergence results in Theorem 3.2 on two representative nonlinear test problems. Specifically, we apply CAB-2 and CAB-3 to the initial-value problem

$$\frac{dy}{d\rho} = v(y, \rho), \qquad y(\rho_0) = y_0, \tag{27}$$



(a) Trajectory for velocity field v_1.

(b) Global max error versus step size.

(c) Trajectory for velocity field v_2.

(d) Global max error versus step size.

Figure 7: Empirical verification of the accuracy results in Theorem 3.2 on two representative nonlinear velocity fields. (a) and (c) show sample trajectories and normalized velocity fields, while (b) and (d) plot the global maximum error against the maximum step size $h_{\max}$ on non-uniform ρ-grids. In both examples, the observed slopes match the theoretical predictions; CAB-2 exhibits second-order global accuracy, whereas CAB-3 exhibits third-order global accuracy .

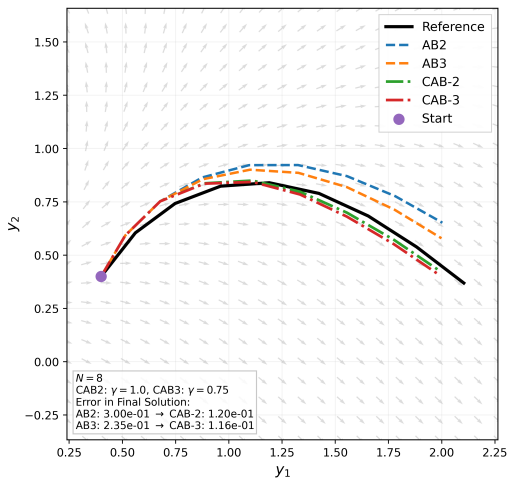
using the two velocity fields shown in Figure. 7(a) and Figure. 7(c). The first field is

$$v_1(y,\rho) = \begin{bmatrix} -1.5\,y_1 + 0.9\,y_2 + 0.2\,y_1 y_2 + 0.15\sin(3\rho) \\ -1.0\,y_2 - 0.7\,y_1 + 0.1\,y_1^2 - 0.08\,y_2^2 + 0.1\cos(2\rho) \end{bmatrix}, \tag{28}$$

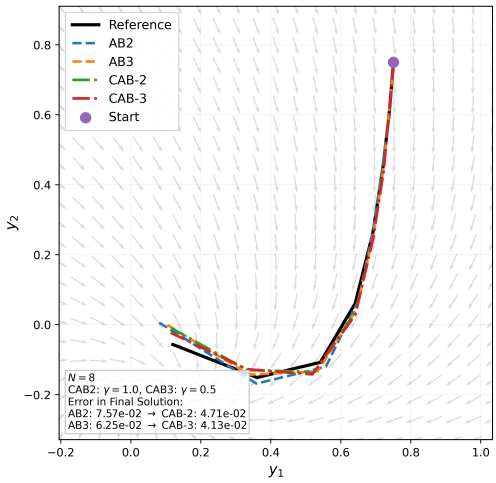
and the second is

$$v_2(y,\rho) = \begin{bmatrix} -(0.3 + 0.4(y_1^2 + y_2^2))\,y_1 - (3.0 + 0.2\sin\rho)\,y_2 \\ (3.0 + 0.2\sin\rho)\,y_1 - (0.3 + 0.4(y_1^2 + y_2^2))\,y_2 \end{bmatrix}. \tag{29}$$

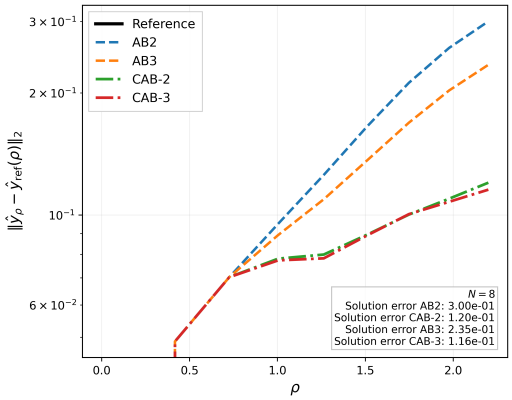
As the reference solution, we use a high-accuracy adaptive DOP853 integrator with stringent error tolerances using SciPy. Figures 7(b) and 7(d) plot the maximum pointwise trajectory error over the time steps as a function of the maximum step size $h_{\max}$. For CAB-2, we use $\gamma = 0.75$, while for CAB-3 we consider two choices: a constant parameter $\gamma = 0.25$ and a step-dependent parameter $\gamma = 0.75\,h_i$. In both settings, the empirical convergence behavior is consistent with the theory: the global error scales as $O(h_{\max}^2)$ for CAB-2 and for CAB-3 with constant γ, whereas it improves to $O(h_{\max}^3)$ for CAB-3 when $\gamma = \mathcal{O}(h_i)$.



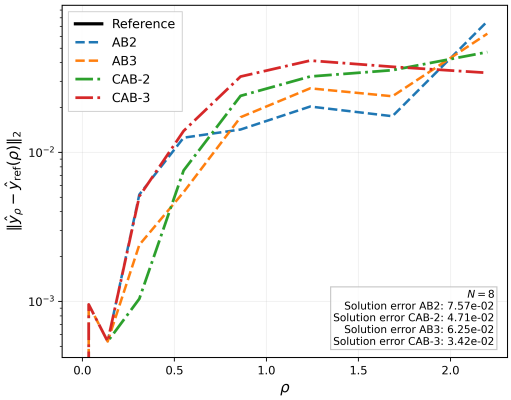
(a) Trajectory comparison for Example 1.



(b) Trajectory comparison for Example 2.



(c) Trajectory error for Example 1.



(d) Trajectory error for Example 2.

Figure 8: Comparison of AB2/AB3 and the proposed CAB-2/CAB-3 on two representative nonlinear rectified ODEs with a small sampling budget ($N = 8$). The top row shows the numerical trajectories overlaid on the corresponding velocity fields, and the bottom row shows the trajectory error $\|\hat{\boldsymbol{y}}_\rho - \hat{\boldsymbol{y}}_{\mathrm{ref}}(\rho)\|_2$ as a function of ρ. These visualizations highlight the benefit of the proposed correction in improving low-step integration accuracy to match the exact output.

A.6 Effect of rectification and correction.

CAB's gains come from both rectification and correction. Table 5 shows that the noise-to-signal coordinate yields a smoother learned field, supporting the role of rectification in making multistep extrapolation easier. The AB baselines in Tables 1–4 isolate the correction effect: CAB improves over the corresponding AB solver at the same NFE and under the same sampling setup, showing that the gains are not merely due to Adams–Bashforth history. The γ ablation further shows that correction strength controls the FID–NIQE trade-off, with moderate values giving robust perceptual behavior. Thus, rectification improves the integration geometry, while CAB's extrapolation-defect correction provides an additional zero-extra-NFE improvement.

Figure 8 compares AB2/AB3 and the proposed CAB-2/CAB-3 on two nonlinear rectified ODEs, both using ($N = 8$). The top row visualizes the trajectories and the underlying velocity fields, while the bottom row plots the trajectory errors $\|\hat{\boldsymbol{y}}_\rho - \hat{\boldsymbol{y}}_{\mathrm{ref}}(\rho)\|_2$ across the steps. In both examples, the corrected schemes track the reference trajectory more faithfully than their uncorrected AB counterparts. In particular, CAB-2 and CAB-3 attain lower final solution errors than AB2 and AB3, respectively. These plots illustrate the practical effect of the correction term: it suppresses the error accumulation visible in AB2/AB3 and provides near-exact output in the low-NFE regime.

A.7 Additional Experiments

In this section, we provide additional qualitative comparisons to complement the main results. The examples show that CAB often produces sharper and more coherent samples than competing tr aining-free solvers, especially in the low-NFE regime where visual artifacts are more pronounced. We also include Figure 9, which illustrates that aggressive higher-order corrections based on previous velocity evaluations can become unstable for strong baseline solvers, in contrast to the controlled correction used by CAB. For DPM-Solver++, UniPC, and STORK, we use implementations from the `diffusers` library whenever available; otherwise, we use the official code released by the authors, with only minimal changes required to match our evaluation protocol.

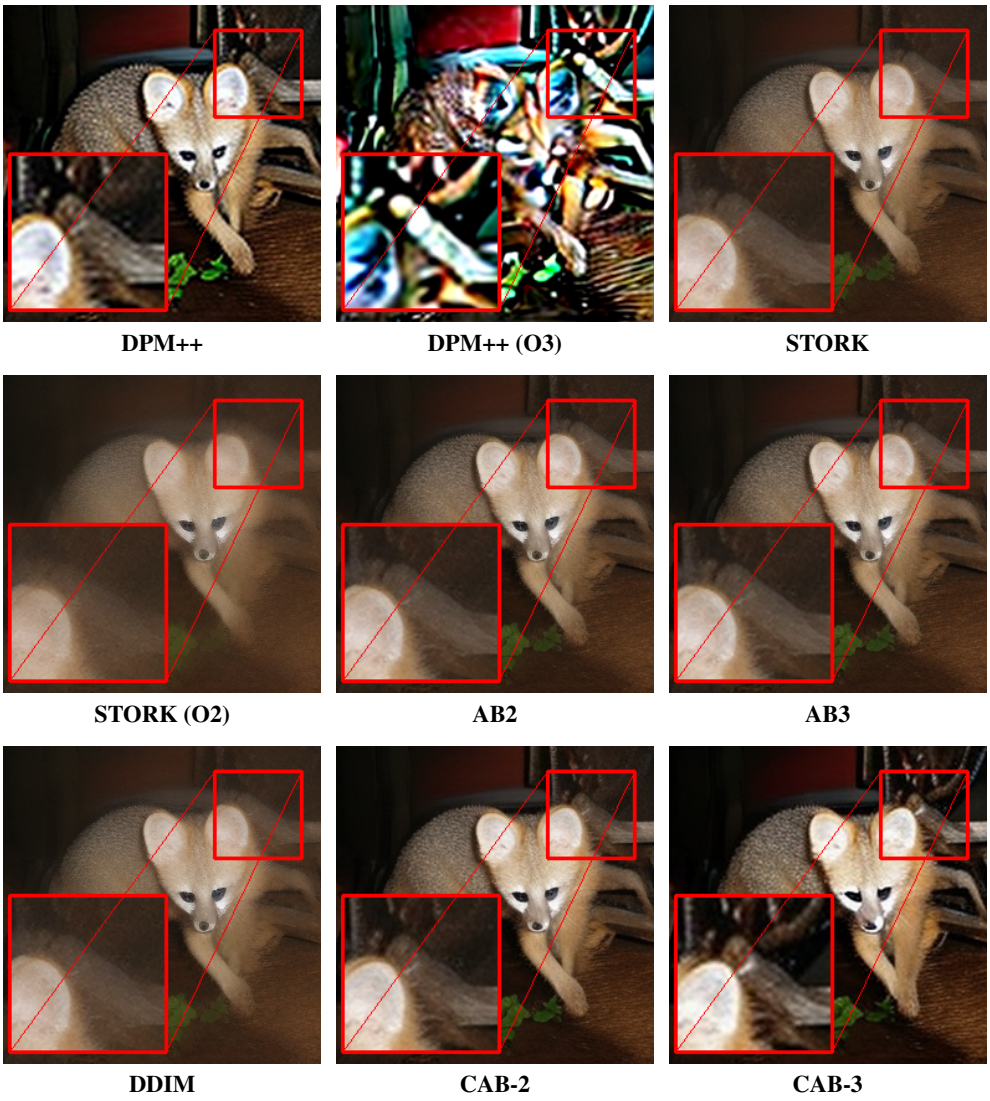


Figure 9: Qualitative comparison of samplers in the 8-NFE regime for DiT. Increasing the solver order of DPM++ and STORK can lead to over-correction of the predictor trajectory, producing amplified artifacts and structural distortions in the Grey Fox image, consistent with the quantitative trends in Table 2. AB predictor algorithms generate images that oversmooth, whereas CAB-based methods maintain improved visual consistency and cleaner structural fidelity under the same sampling budget.

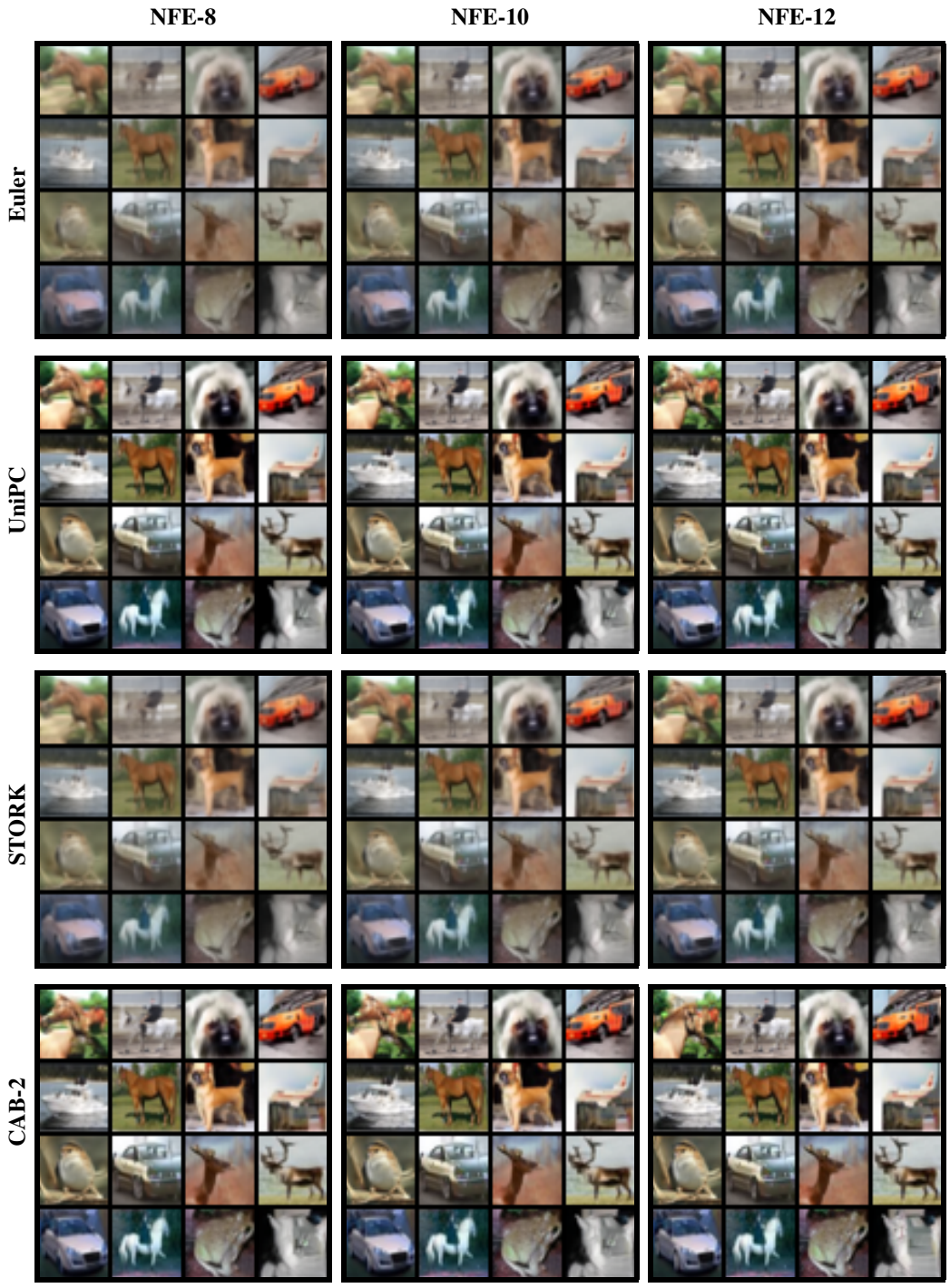


Figure 10: Unconditional generation on CIFAR-10 using EDM model.



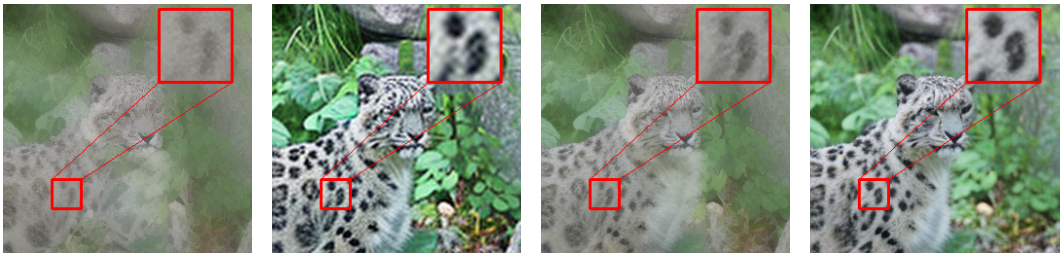
(a) DDIM (b) DPM++ (c) STORK (d) CAB-3

10 NFEs. CAB-3 produces the most faithful reconstruction with sharper beak structure, consistent texture, and reduced oversmoothing in just 10 NFEs.



(e) DDIM (f) DPM++ (g) STORK (h) CAB-2

8 NFEs. CAB-2 best preserves texture, shape, and color, while DDIM/STORK oversmooth and DPM++ introduces artifacts.



(i) DDIM (j) DPM++ (k) STORK (l) CAB-3

10 NFEs. DDIM oversmooths, DPM++ is noisy and over-sharpened, STORK generated image is blurred, while CAB-3 gives the most natural image.



(m) DDIM (n) DPM++ (o) STORK (p) CAB-3

10 NFEs. DDIM oversmooths and loses detail, DPM++ introduces noise and distorted text, STORK remains blurry, while CAB-3 preserves sharp structure, clean edges, and readable text.

Figure 11: Comparison of samplers on class-conditional ImageNet (256×256) generation using DiT.



(a) Euler (b) DPM++ (c) STORK (d) CAB-2

6 NFEs. *"Two men standing side by side and smiling."* CAB-2 better preserves facial detail and natural color, while the baselines show noise, oversmoothing, or loss of detail.



(e) Euler (f) DPM++ (g) STORK (h) CAB-2

6 NFEs. *"A black bear walks down a hill away from a tree."* CAB-2 gives clearer bear structure and surrounding scene content, while competing samplers show haze, noise, or oversmoothing.



(i) Euler (j) DPM++ (k) STORK (l) CAB-2

8 NFEs. *"Four men playing frisbee in a fenced park."* CAB-2 better preserves human structure, color, and scene coherence, while baselines exhibit color shifts, artifacts, or oversmoothing.



(m) Euler (n) DPM++ (o) STORK (p) CAB-2

8 NFEs. *"A person riding a bicycle on a deserted street."* CAB-2 recovers sharper subject structure and road geometry, while baselines remain noisy, blurred, or overly smooth.

Figure 12: Comparison of training-free samplers on QWEN-Image 1024×1024 text-to-image generation.

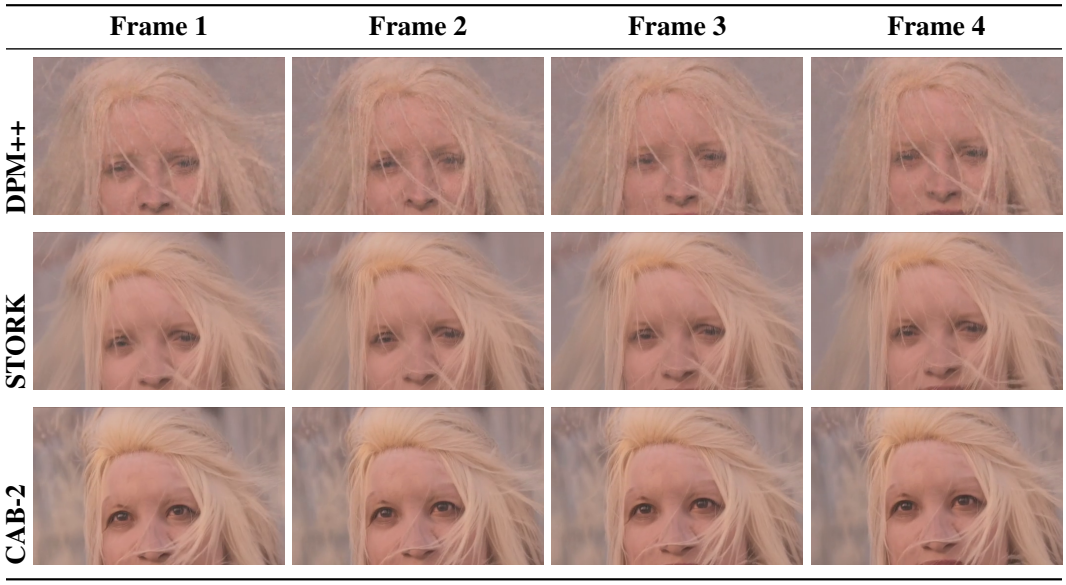


Figure 13: **Prompt:** *"light wind, feathers moving, she moves her gaze, 4k."* Temporal comparison of training-free samplers on HunyuanVideo-1.5 over four frames from the same generated video. CAB-2 better preserves appearance consistency and smoother motion progression, while DPM++ and STORK show stronger temporal drift and frame-to-frame variation.

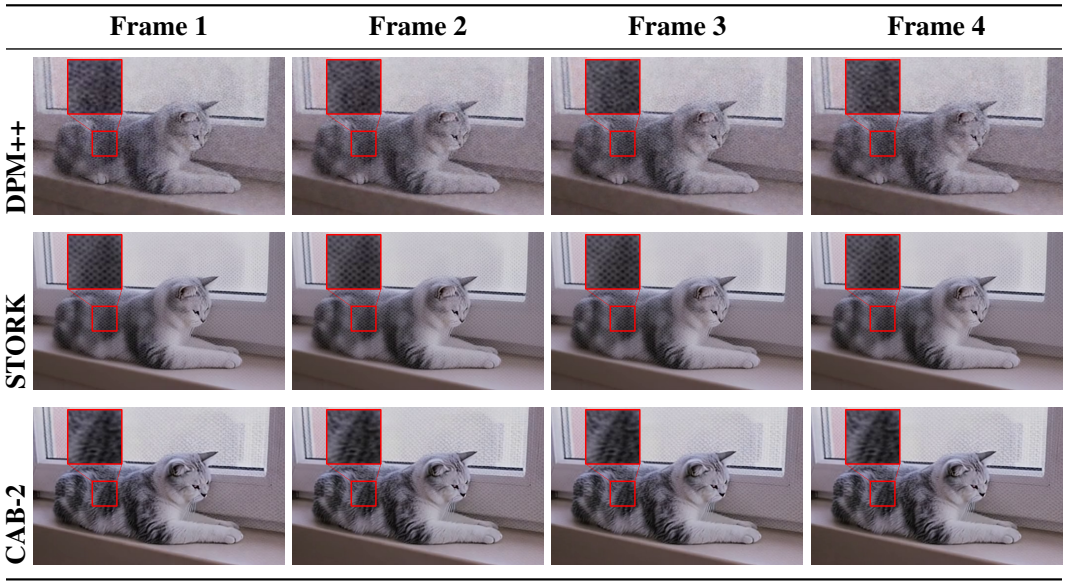


Figure 14: **Prompt:** *"A fluffy grey and white cat is lazily stretched out on a sunny window sill, enjoying a nap after a long day of lounging."* Temporal comparison of training-free samplers on HunyuanVideo-1.5 over four frames from the same generated video. CAB-2 preserves sharper fur texture, clearer facial structure, and more consistent window-side lighting, while DPM++ and STORK appear blurrier and show weaker appearance consistency across frames.

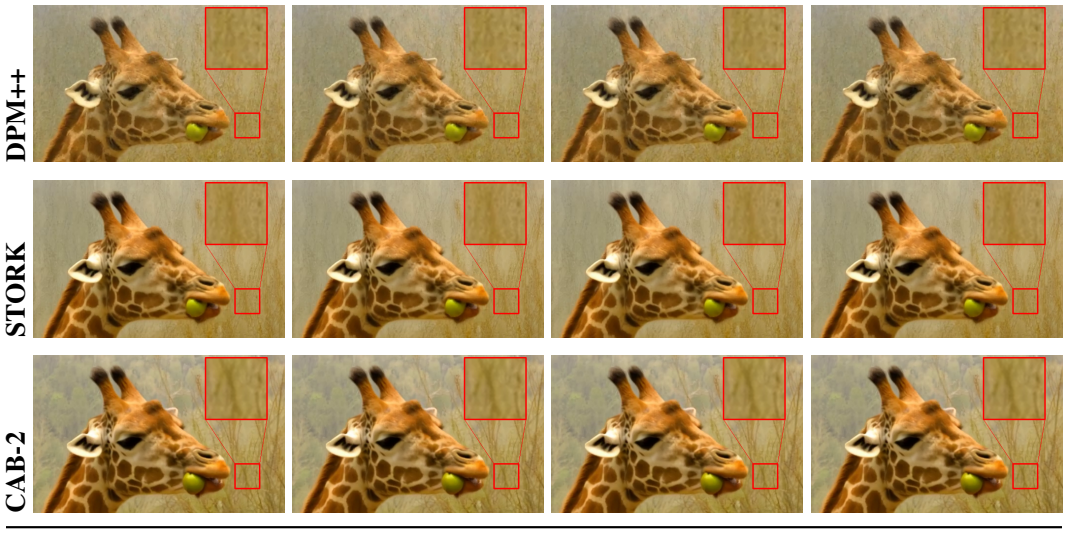


Figure 15: **Prompt:** *"a giraffe eating an apple."* Temporal comparison of training-free samplers on HunyuanVideo-1.5 over four frames from the same generated video. CAB-2 preserves cleaner appearance and more coherent scene context across frames, while DPM++ and STORK show weaker texture consistency and less stable background structure.

Table 9: Extension of Table. 4 including AB2 and AB3

Metric	Method	NFE					Metric	Method	NFE				
		4	6	8	10	20			4	6	8	10	20
Visual	DPM++	45.07	44.88	45.78	47.35	51.74	**Align.**	DPM++	30.87	36.88	40.13	41.04	42.50
	STORK	45.18	47.44	**49.35**	50.47	**53.09**		STORK	35.30	37.61	40.00	40.53	42.06
	AB2	45.32	47.43	**49.44**	50.52	**53.15**		AB2	34.93	37.65	39.61	40.40	42.48
	AB3	45.27	47.45	49.20	50.30	53.01		AB3	34.78	37.37	40.11	40.27	42.19
	CAB-2	**46.08**	**47.70**	49.03	**50.54**	52.71		CAB-2	**37.92**	**39.80**	40.97	42.12	**43.17**
	CAB-3	46.03	47.05	47.88	49.33	51.53		CAB-3	37.72	39.67	**41.41**	**42.34**	42.73
Temp.	DPM++	62.22	62.88	63.04	63.13	**63.13**	**Final**	DPM++	138	146	149	152	157
	STORK	**63.71**	**63.47**	**63.22**	**63.30**	63.03		STORK	144	149	152	154	158
	AB2	63.09	62.91	62.89	62.87	62.67		AB2	143	148	151	154	158
	AB3	63.14	62.95	62.92	62.91	62.61		AB3	143	148	152	153	158
	CAB-2	62.84	62.72	62.78	63.04	62.86		CAB-2	**147**	**150**	**153**	**156**	**159**
	CAB-3	62.73	62.42	62.45	62.73	62.72		CAB-3	146	149	151	154	157

A.7.1 Comparison with Distillation-Based Fast Sampling

To contextualize training-free acceleration against distillation-based sampling, we compare CAB-2 applied to the original Qwen-Image model with Qwen-Image-Lightning [47], a recent distribution-matching-distilled model optimized for few-step inference using an Euler sampler. The developers report using approximately 420,000 text prompts for the model-specific distillation process. The released distilled models target 4- and 8-step generation and require substantial additional training, whereas CAB is applied directly to the pretrained model without retraining.

Table 10 presents the quality–latency comparison between Qwen-Image-Lightning and training-free CAB-2, evaluated on the same prompt set. Qwen-Image-Lightning achieves higher quality and approximately 17–19% lower latency in the very-low-step regime, as expected for a model explicitly distilled for few-step generation. Nevertheless, CAB-2 becomes increasingly competitive as the NFE increases: the ImageReward gap decreases from 1.1981 at 4 NFEs to 0.2591 at 8 NFEs, where CAB-2 reaches an ImageReward of 1.0085 without additional training data or distillation cost.

Table 10: Quality–latency comparison of distilled Qwen-Image-Lightning and training-free CAB-2 on the same prompts.

	Qwen-Image-Lightning		Qwen-Image + CAB-2	
NFE	**ImageReward ↑**	**Time (s) ↓**	**ImageReward ↑**	**Time (s) ↓**
4	**1.2514**	**3.497**	0.0533	4.208
6	**1.2637**	**5.138**	0.7536	6.169
8	**1.2676**	**6.590**	1.0085	8.142

A.7.2 Compatibility with Cache-Based Acceleration

We further investigate whether CAB can be combined with cache-based acceleration methods such as TaylorSeer [48], which reduces computational cost by approximating intermediate transformer features instead of recomputing them at every step. Such approximations may introduce errors in the velocity evaluations stored in CAB's multistep history, potentially affecting its extrapolation-defect correction.

To assess compatibility, we integrate CAB-2 with TaylorSeer on Qwen-Image and compare it against TaylorSeer with the default Euler sampler. Both configurations use identical denoising steps, timestep grids, caching settings, prompts, and random seeds. As shown in Table 11, CAB-2 consistently improves ImageReward from -0.3115 to -0.1443 at 6 steps, 0.1991 to 0.3618 at 8 steps, and 0.7079 to 0.7785 at 10 steps. These results demonstrate that CAB's correction remains effective under the tested feature-caching approximations, supporting its compatibility with cache-based acceleration.

A.7.3 Applicability to Distilled Models

Although CAB is designed as a training-free acceleration method, its correction mechanism can also be applied to models already optimized for few-step generation. To examine this compatibility, we compare Euler and CAB-2 on Qwen-Image-Lightning across 2–8 NFEs. As shown in Table 12, CAB-2 consistently improves ImageReward at every evaluated budget, demonstrating that its multistep formulation can complement distillation-based acceleration.

The gains are modest at 4–8 NFEs, where the distilled model is already optimized for Euler sampling and only a few correction steps are available. At 2 NFEs, the larger improvement arises from the modified startup update rather than the full extrapolation-defect correction. Overall, these findings establish compatibility with distilled models and motivate integrating CAB into the student sampling trajectory during distribution-matching distillation, potentially combining solver-aware training with improved low-NFE integration.

Table 11: CAB-2 with TaylorSeer on Qwen-Image.

Steps	**Euler**	**CAB-2**	**Gain**
6	-0.3115	$\mathbf{-0.1443}$	$+0.1672$
8	0.1991	$\mathbf{0.3618}$	$+0.1627$
10	0.7079	$\mathbf{0.7785}$	$+0.0706$

Table 12: CAB-2 on Qwen-Image-Lightning.

Method	2	4	6	8
Euler	1.0917	1.2531	1.2699	1.2722
CAB-2	**1.2479**	**1.2684**	**1.2756**	**1.2769**

Table 13: CAB-2 rectification on DiT/ImageNet-256. FID ↓.

Method	6	8	10
θ-space	73.08	14.19	8.51
ρ-space	**15.13**	**9.21**	**7.99**

A.7.4 Native x_0- and v-Prediction Rectifications

The ϵ–ρ formulation is not unique. For the affine-Gaussian path $x_t = s_t x_0 + \sigma_t \epsilon$, an x_0-prediction model admits the rectification

$$z_\kappa = \frac{x_t}{\sigma_t}, \qquad \kappa_t = \frac{s_t}{\sigma_t} = \rho_t^{-1}, \qquad \frac{dz_\kappa}{d\kappa} = \hat{x}_{0,\theta}(x_t, t). \tag{30}$$

Thus, the predicted clean sample directly defines the rectified velocity field in κ-space.

For v-prediction, defining $r_t = \sqrt{s_t^2 + \sigma_t^2}$, $\theta_t = \operatorname{atan2}(\sigma_t, s_t)$, and $z_\theta = x_t / r_t$ yields

$$\frac{dz_\theta}{d\theta} = -\sin\theta_t \, x_0 + \cos\theta_t \, \epsilon = \frac{v_\theta(x_t, t) - \dot{r}_t z_\theta}{r_t \dot{\theta}_t}. \tag{31}$$

Here, v_θ denotes the predicted physical-time velocity. CAB's Adams–Bashforth predictor and extrapolation-defect correction apply to these rectified dynamics without additional NFEs.

Table 13 compares the ρ-space (ϵ-prediction) and θ-space (v-prediction) formulations on DiT/ImageNet-256. The ρ-space formulation consistently achieves lower FID, particularly in the low-NFE regime.

A.7.5 Third-Order Corrector for AB3

Following the Lagrange-extrapolation derivation, quadratic extrapolation from the previous three network evaluations gives, with $h_j = \rho_{j+1} - \rho_j$,

$$\begin{aligned}\epsilon_i^{\mathrm{ext},3} = &\frac{(h_{i-1}+h_{i-2})(h_{i-1}+h_{i-2}+h_{i-3})}{h_{i-2}(h_{i-2}+h_{i-3})}\epsilon_{i-1} \\ &- \frac{h_{i-1}(h_{i-1}+h_{i-2}+h_{i-3})}{h_{i-2}h_{i-3}}\epsilon_{i-2} \\ &+ \frac{h_{i-1}(h_{i-1}+h_{i-2})}{h_{i-3}(h_{i-2}+h_{i-3})}\epsilon_{i-3}.\end{aligned} \tag{32}$$

The corresponding extrapolation defect is

$$d_i^{(3)} = \epsilon_i - \epsilon_i^{\mathrm{ext},3}. \tag{33}$$

As shown in Table 14, the third-order corrector improves FID at 4–20 NFEs, as expected from higher-order extrapolation, while requiring one additional stored model evaluation but no additional NFEs. At larger sampling budgets, the second-order corrector performs slightly better.

Table 14: Second- vs. third-order correction for CAB-3 on CIFAR-10(VP).

Corrector	**NFE**							
	4	**6**	**8**	**10**	**20**	**30**	**40**	**50**
CAB-3 + 2nd-order	22.8001	7.6202	4.1510	2.9953	1.9157	**1.7906**	**1.7656**	**1.7708**
CAB-3 + 3rd-order	**22.7895**	**7.5886**	**4.1110**	**2.9353**	**1.8721**	1.8128	1.8282	1.8512

A.7.6 Robustness Across CFG Strengths

The optimal correction strength γ may depend on classifier-free guidance (CFG), since guidance changes the magnitude and variation of the learned ODE field. Nevertheless, a fixed $\gamma = 0.9$ remains effective across CFG values from 1.25 to 1.50, indicating robustness to moderate guidance changes. As shown in Table 15, the sampling performance remains stable across the evaluated NFEs, with CFG $= 1.50$ achieving the lowest FID throughout. This does not imply universal optimality, and tuning γ for individual CFG settings may further improve performance.

Table 15: CAB-2 with fixed $\gamma = 0.9$ across CFG strengths (FID $\downarrow$).

CFG	**NFE**							
	4	**6**	**8**	**10**	**20**	**30**	**40**	**50**
1.25	88.04	15.13	9.21	7.99	6.86	6.86	6.91	6.95
1.35	81.00	12.88	7.87	6.85	5.81	5.72	5.74	5.78
1.50	**71.68**	**10.56**	**6.77**	**6.08**	**5.31**	**5.19**	**5.19**	**5.20**

A.7.7 Effect of Discretization Time Grid

The original experiments use uniform spacing in t; after rectification, the corresponding noise-to-signal coordinate ρ generally follows a non-uniform grid. Since the discretization grid affects finite-step sampling performance, we further investigate the effect of non-uniform timestep selection.

As shown in Table 16, a non-uniform time grid improves ImageReward at every evaluated NFE, particularly in the low-NFE regime. For instance, ImageReward increases from 0.3393 to 0.7536 at

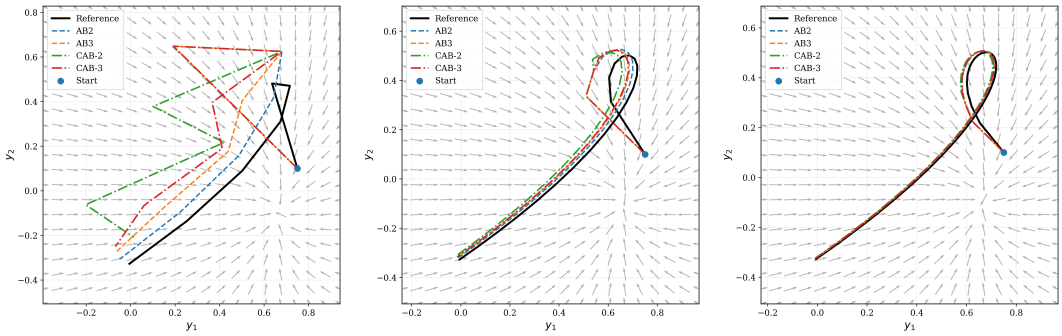
6 NFEs and from 0.6561 to 1.0085 at 8 NFEs. These results demonstrate that CAB is compatible with different discretization grids and can benefit substantially from improved timestep design.

Table 16: Uniform vs. non-uniform timestep grids for CAB-2 on Qwen-Image (ImageReward ↑).

Grid	**NFE**							
	4	**6**	**8**	**10**	**20**	**30**	**40**	**50**
Uniform	−0.3544	0.3393	0.6561	0.8689	1.2026	1.2655	1.2707	1.2866
Non-uniform	**0.0533**	**0.7536**	**1.0085**	**1.1240**	**1.2777**	**1.2867**	**1.2937**	**1.3104**

A.8 Limitations

As illustrated in Figure 16, although CAB-2 and CAB-3 perform well in flow and diffusion sampling settings after appropriate rectification, their behavior can degrade in strongly curved or poorly conditioned dynamical regimes, especially at very small step budgets. In particular, similar to AB2 and AB3, when the number of sampling steps is extremely limited (e.g., $N = 6$), both methods may produce wayward trajectories that deviate substantially from the reference path, indicating that the correction term does not fully prevent geometric mis-tracking caused by aggressive multistep extrapolation. These observations suggest that the proposed corrections improve robustness but do not completely eliminate failure modes associated with rapidly varying dynamics, large curvature, or stiffness-like effects. Developing more reliable adaptive corrections for such challenging low-step regimes remains an important direction for future work.



(a) $N = 6$ (b) $N = 20$ (c) $N = 50$

Figure 16: Trajectories produced by AB2, AB3, CAB-2, and CAB-3, compared with the reference solution, for different sampling budgets. At very low step counts, particularly $N = 6$, the numerical trajectories can deviate substantially from the reference path, revealing geometric mis-tracking in challenging regions of the velocity field. As the number of sampling steps increases, the trajectories become better aligned with the reference.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the proposed CAB sampler improves quality-NFE trade-off for flow and diffusion sampling and this is reflected in Section. 4 across multiple benchmarks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Refer to Subsection A.8, where we show a failure case for CAB solvers and reason the behavior.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Refer to Lemma. 3.1 and Theorem. 3.2 for the mild assumptions and A.1 and A.4 for their corresponding proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides implementation details and evaluation settings in 4 and in the Appendix, along with the main proposed algorithm to reproduce results. Source code with a detailed *Readme* file is also included.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Source code along with a detailed *Readme* file is provided during paper submission. All the data required to testing is mentioned in *Readme* as well.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Refer to 4 for the number of sampling steps and parameter γ values for every experiment.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All the metric scores shown in 4 are averaged over multiple runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Refer to 5 in Section 4 where we report the time taken for a specific setting, along with maximum GPU memory consumption.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the code of ethics and confirm that we followed the given code of conduct.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Refer to 5, where we have detailed the positive and negative impacts of our work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper proposes a sampling method and does not release new datasets or foundation models that require additional misuse safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Existing codebases, pretrained models, and datasets used in this work are properly cited and used according to their respective licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The modified sampling implementations and integration instructions are documented in *Readme* file in the source code.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our work does not involve human subjects, user studies, or crowdsourcing experiments.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We have used LLM only for editing purposes to find the right words to complete the text description.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.