TADA: Improved Diffusion Sampling with Training-free Augmented DynAmics

Tianrong Chen, Huangjie Zheng, David Berthelot, Jiatao Gu, Josh Susskind, Shuangfei Zhai
Apple
{tchen54,huangjie_zheng,dberthelot,jgu32, jsusskind,szhai}@apple.com

Abstract

Diffusion models have demonstrated exceptional capabilities in generating high-fidelity images but typically suffer from inefficient sampling. Many solver designs and noise scheduling strategies have been proposed to dramatically improve sampling speeds. In this paper, we introduce a new sampling method that is up to 186% faster than the current state of the art solver for comparative FID on ImageNet512. This new sampling method is training-free and uses an ordinary differential equation (ODE) solver. The key to our method resides in using higher-dimensional initial noise, allowing to produce more detailed samples with less function evaluations from existing pretrained diffusion models. In addition, by design our solver allows to control the level of detail through a simple hyper-parameter at no extra computational cost. We present how our approach leverages momentum dynamics by establishing a fundamental equivalence between momentum diffusion models and conventional diffusion models with respect to their training paradigms. Moreover, we observe the use of higher-dimensional noise naturally exhibits characteristics similar to stochastic differential equations (SDEs). Finally, we demonstrate strong performances on a set of representative pretrained diffusion models, including EDM, EDM2, and Stable-Diffusion 3, which cover models in both pixel and latent spaces, as well as class and text conditional settings. The code is available at https://github.com/apple/ml-tada.

1 Introduction

Diffusion Models (DMs; Song et al. [27]; Ho et al. [14]) and Flow Matching [20, 21] are foundational techniques in generative modeling, widely recognized for their impressive scalability and capability to generate high-resolution, high-fidelity images [4, 10, 24]. Both share a common underlying mathematical structure and generate data by iteratively denoising Gaussian noise through a solver.

Sampling from a diffusion model is typically discretized through a multi-step noise schedule. Denoising Diffusion Probabilistic Models sample new noise at every step resulting in a process interpretable as discretized solutions of diffusion Stochastic Differential Equations (SDEs). Meanwhile, Denoising Diffusion Implicit Models only sample initial noise which is reused at every step resulting in a process interpretable as discretized solutions of Ordinary Differential Equations (ODEs). Both ODE and SDE solvers should perform similarly since they merely represent different interpretations of the same Fokker-Planck Partial Differential Equation. In practice, however, ODE solvers often yield lower-fidelity results than SDE solvers as evidenced by the FID scores in [17] when sufficient function evaluations are performed and model capacity is a limiting factor. On the other hand, SDE solvers require finer steps due to the noise injection at every step, which gets amplified through discretization.

Generating high-quality images using DMs often necessitates a high number of steps and therefore a high Number of Function Evaluations (NFEs), substantially increasing computational costs com-

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

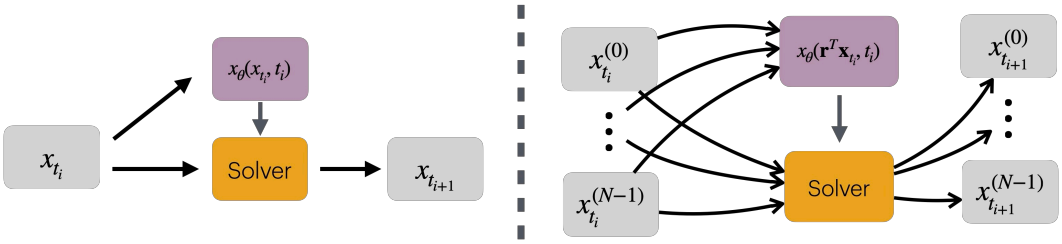

Figure 1: Here we show the distinctions between conventional diffusion models (*left*) and momentum diffusion models (*right*) during sampling. Leveraging Prop 3.1, we demonstrate that pretrained diffusion models with x_0-*prediction* $x_\theta(\cdot, \cdot)$ can be directly applied to propagate the momentum system with multiple varible $\{\mathbf{x}_{t_i}^{(}n)\}_0^{N-1}$. Moreover, the choice of numerical solver remains flexible. See Sec 3.2 for details.

pared to alternative generative approaches such as generative adversarial networks (GANs) [11]. Consequently, a large research effort has been focused on reducing the NFEs to reach a desired level of sample quality and this is also the focus of our paper. Recent efforts [17, 32] have significantly improved the efficiency of SDE solvers, achieving compelling results with a moderate number of function evaluations. Designing training-free faster solvers by leveraging numerical integration techniques has also received a lot of attention: exponential integrators [15] and multi-step methods [1], and hybrid combinations of these [22, 23, 33, 34]. Remarkably, faster solvers reach near-optimal quality with around 30 NFEs and retain acceptable image generation capabilities even with NFE < 10.

To further enhance performance, optimization-based solvers [35, 36] have advanced the capabilities of the aforementioned methods, achieving strong results with as few as 5 NFEs. In parallel, training-based dynamical distillation approaches [3, 25, 28, 30] have demonstrated effectiveness in reducing the number of function evaluations and can similarly benefit from the incorporation of fast solvers.

In this paper, we propose a new sampling method that is 186% faster than the current state of the art for generating 512x512 ImageNet samples with an FID of 2. Our method is orthogonal to existing diffusion model sampling techniques, allowing seamless integration with advanced solvers and classifier-free guidance (CFG) schedules simply by transitioning from standard ODE to momentum ODE frameworks. The proposed method is based on momentum diffusion models [6] and on the observation [7] that they can achieve competitive performance with a very low number of function evaluations. Specifically our paper extends the aforementioned works as follows:

1. We prove the training equivalence between momentum diffusion models [6, 9] and conventional diffusion models, modulo a transformation of the input variables to the neural network. Consequently, the equivalence shows that simple noise or noisy data augmentation together with diffusion loss, including but not limited to momentum diffusion, offers **no training benefit**. Readers may extend this reasoning to their own settings and verify the algorithms relevant to them.
2. We then shift our attention to the **sampling phase** of the momentum system. We propose a new sampling method named Training-free Augmented DynAmics (TADA) that uses higher-dimensional input noise and yet enables direct reuse of pretrained diffusion models.
3. Although no benefits are observed in the training phase in theory, we find that our proposed method, rooted in the momentum system, inherits advantageous SDE properties while employing ODE solvers, enabling both diverse generation and accelerated sampling through momentum dynamics.
4. We illustrate the superior performance of our approach through extensive experimentation.

2 Preliminary

First, we cover conventional diffusion models and flow matching, we then follow up by introducing momentum diffusion models and we finish this section with exponential integrators. In the rest of this paper, we follow the flow matching literature convention for the direction of distribution transport, e.g. from the prior distribution at time $t = 0$ to the data distribution at time $t = 1$.

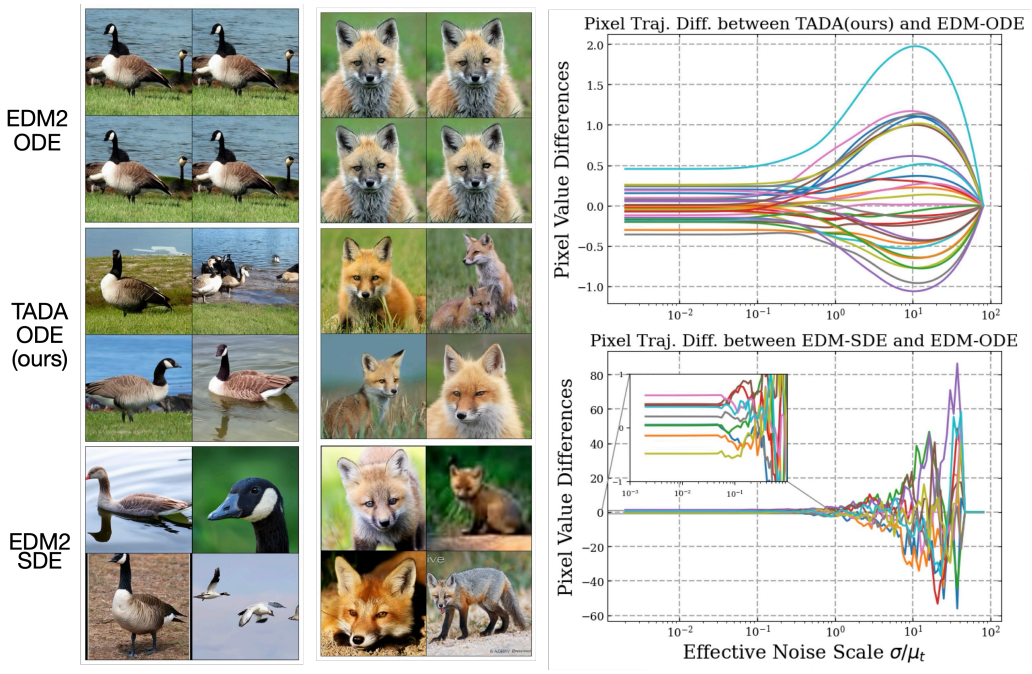

Figure 2: Demonstration of differences of TADA and probablistic ODE in terms of trajectories. We apply the same pretrained model with momentum system with same discretization in terms of SNR and the same initial prior samples. TADA can generate SDE-like property, such as generate different samples from same initial condition, but the system keeps deterministic ODE which can be solved more efficiently. See Prop.3.3 for more detail.

2.1 Diffusion Models and Flow Matching

In Diffusion Models (DM) and Flow Matching (FM), intermediate states $x_t \in \mathbb{R}^d$ are sampled from a tractable transition probability conditioned on an initial data point $x_1 \sim p_{\text{data}}$ during training:

$$x_t = \mu_t x_1 + \sigma_t \epsilon, \quad \epsilon \sim \mathcal{N}(0, \boldsymbol{I}_d). \tag{1}$$

The coefficients $\mu_t \in \mathbb{R}$ and $\sigma_t \in \mathbb{R}$ differ depending on the specific model parameterization: mean preserving, variance preserving or variance exploding. Regardless of these differences, the training objectives across various methods remain identical.

For the sake of clarity, we illustrate parameterizing the *noise prediction model* with a neural network with learnable weights θ. In this case the objective function can be expressed as:

$$\min_\theta \mathcal{L}_{\text{DM}}(\theta) := \mathbb{E}_{x_1,\epsilon,t} \|\epsilon_\theta(x_t, t) - \epsilon\|_2^2.$$

Alternatively, using the linear relationship given by eq.1, we could just as easily parameterize the *data prediction model* as $x_\theta(x_t, t) := (x_t - \sigma_t \epsilon_\theta(x_t, t))/\mu_t$, or even as a velocity prediction model.

Without loss of generality, we define a force term $F_\theta : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$ as a linear combination of the learned noise ϵ_θ and the state x_t, which pushes forward x_t from the prior towards the p_{data} from $t = 0$ to $t = 1$. During sampling, we use this force F_θ to define the following probabilistic flow:

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = a_t x_t + b_t F_\theta(x_t, t), \quad x_0 \sim \mathcal{N}(0, \sigma_0^2 \boldsymbol{I}_d), \tag{2}$$

where $a_t \in \mathbb{R}$, and $b_t \in \mathbb{R}$ are time-varying coefficients defined by specific parameterization of F_θ. For example, for a FM parameterization: $F_\theta(x_t, t) = (x_\theta(x_t, t) - x_t)/(1 - t)$, $a_t = 0$ and $b_t = 1$.

2.2 Phase Space Momentum Diffusion Model

Several recent studies have explored a more sophisticated diffusion process defined in the phase space [7, 9]. In this space there are $N = 2$ noise variables, consequently $\mathbf{x}_t, \boldsymbol{\epsilon} \in (\mathbb{R}^d)^2$. Owing to the linear

dynamics within this framework—similar to those in conventional diffusion models—the transition probability can also be derived analytically:

$$\mathbf{x}_t = (\boldsymbol{\mu}_t \otimes \mathbf{I}_d)x_1 + (\boldsymbol{L}_t \otimes \mathbf{I}_d)\boldsymbol{\epsilon}, \quad \boldsymbol{L}_t\boldsymbol{L}_t^{\mathsf{T}} = \boldsymbol{\Sigma}_t, \quad \boldsymbol{\epsilon} := \left[\epsilon^{(0)}, \epsilon^{(1)}\right]^{\mathsf{T}} \sim \mathcal{N}(0, \boldsymbol{I}_{2d}),$$

where $\boldsymbol{\mu}_t \in \mathbb{R}^2$ and $\boldsymbol{\Sigma}_t \in \mathbb{R}^{2\times 2}$ denote the mean and covariance matrix of the resulting multi-variable Gaussian distribution. $\boldsymbol{L}_t \in \mathbb{R}^{2\times 2}$ is the noise scaling matrix. The superscript denotes the i-th variable in the multi-variable setting, while the boldface notation $\mathbf{x}_t$ represents the aggregation of variables, i.e., $\mathbf{x}_t = [x_t^{(0)}, x_t^{(1)}]^{\mathsf{T}}$ in this setting.

Notation: In the rest of this document, we use of the Kronecker product to specify that all the weights are identical along the data dimension d. For example, the expression $(\mu_t \otimes \mathbf{I}_d)x_1$ denotes the stacking of the scaled versions of x_1 by $\boldsymbol{\mu}_t{}^{(n)}$, e.g. $(\boldsymbol{\mu}_t \otimes \mathbf{I}_d)x_1 = [\boldsymbol{\mu}_t{}^{(0)}x_1, \boldsymbol{\mu}_t{}^{(1)}x_1]^{\mathsf{T}}$.

The standard training objective for MDM minimizes the approximation error of $\epsilon^{(1)}$:

$$\min_\theta \mathcal{L}_{\text{MDM}}(\theta) \coloneqq \mathbb{E}_{x_1,\boldsymbol{\epsilon},t}\|\epsilon_\theta(\mathbf{x}_t, t) - \epsilon^{(1)}\|_2^2 \quad \text{with } \epsilon_\theta : (\mathbb{R}^d)^2 \times [0,1] \to \mathbb{R}^d \tag{3}$$

Note: It should be observed that such techniques require training since the function ϵ_θ takes two data inputs and cannot reuse pre-trained conventional diffusion models where ϵ_θ takes only one data input.

Just like for conventional diffusion models, various parameterizations of the learned noise ϵ_θ remain viable. In particular, AGM[6] and CLD[9] adopt distinct formulations of F_θ, defined as linear combinations of ϵ_θ and $\mathbf{x}_t$, specifically constructed to guide $x_t^{(0)}$ from the prior distribution to the target data distribution. Consequently, sampling reduces to solving a similar probabilistic flow:

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = (\boldsymbol{A}_t \otimes \mathbf{I}_d)\mathbf{x}_t + (\mathbf{b}_t \otimes \mathbf{I}_d)F_\theta(\mathbf{x}_t, t), \quad \mathbf{x}_0 \sim \mathcal{N}(0, \boldsymbol{\Sigma}_0). \tag{4}$$

where $\boldsymbol{A}_t \in \mathbb{R}^{2\times 2}$ and $\mathbf{b}_t \in \mathbb{R}^2$ denote time-dependent coefficients. Specifically, in the MDM framework, the function $F_\theta : (\mathbb{R}^d)^2 \times [0,1] \to \mathbb{R}^d$ accepts an aggregated input of two variables of dimension d and outputs a single vector of dimension d. The resulting d-dimensional output is subsequently broadcasted across the $N = 2$ variables through the coefficient $\mathbf{b}_t$, a key distinction from conventional DM. The explicit formulations for $\boldsymbol{A}_t$ and $\mathbf{b}_t$ are detailed in Appendix.F.1.

2.3 Sampling with Exponential Integrators

Once a pretrained diffusion model ϵ_θ is obtained, the dynamics specified by eq. 2 or 4 can be readily solved. A variety of advanced training-free fast sampling techniques exist, and many of them rely on exponential integrators [15] which presents as the form of eq.5:

$$x_t = \Phi(t,s)x_s + \underbrace{\int_s^t \Phi(t,\tau)b_\tau F_\theta(x_\tau, \tau)\mathrm{d}\tau}_{\text{Approximator } \Psi(s,t,x_s,F_\theta)}. \tag{5}$$

Where $\Phi(\cdot,\cdot)$ is the transition kernel induced by A_t. Since the first linear component of the ODE can be analytically integrated via the transition kernel, the primary challenge lies in accurately approximating the second nonlinear integration, such as the neural network parameterized function F_θ over discretized time steps $s < t$.

To improve the accuracy of approximating the nonlinear integral in the second term, advanced ODE solvers are employed as approximators $\Psi(\cdot,\cdot,\cdot,\cdot)$ in eq.5. Higher-order single-step methods such as Heun's method [12] and multi-step explicit methods like Adams–Bashforth [5] have been widely adopted in prior works [17, 22, 23, 33], often in conjunction with exponential integrators. To further enhance accuracy, implicit schemes such as the Adams–Moulton method [2] have also been introduced [34]. The current state-of-the-art solvers integrate these various techniques, and the specific combinations along with the resulting algorithms are summarized in Appendix.F.3.

This work, however, is specifically focused on designing the ODE itself. Therefore, our approach is entirely orthogonal to existing solver methodologies, allowing it to be seamlessly integrated with any solver type, see fig.1 for explanation and demonstration.

Algorithm 1 TADA sampling

Require: discretized times $t_i \in [t_0, t_1, ...t_T]$; ODE Solver as approximator $\Psi(\cdot,\cdot,\cdot,\cdot)$ (see eq. 5); Pretrained DM $x_\theta(\cdot,\cdot)$. Transition function: $\Phi(t,s) = \exp\int_s^t \mathbf{A}_\tau \mathrm{d}\tau$. Cache Q.

$\mathbf{x}_{t_0} \sim \mathcal{N}(0, \mathbf{\Sigma}_{t_0})$ ▷ draw prior sample

for $i = 0$ to $T-1$ **do**

  compute $\boldsymbol{\mu}_{t_i}, \mathbf{\Sigma}_{t_i}$, and $\mathbf{r}_{t_i} := \frac{\mathbf{\Sigma}_{t_i}^{-1}\boldsymbol{\mu}_{t_i}}{\boldsymbol{\mu}_{t_i}^\top \mathbf{\Sigma}_{t_i}^{-1}\boldsymbol{\mu}_{t_i}}$ ▷ obtain the reweighting for N variables

  $\hat{x} \leftarrow x_\theta\big((\mathbf{r}_{t_i}^\top \otimes I_d)\mathbf{x}_{t_i}, t_i\big)$ ▷ data prediction with pretrained DM

  $F_\theta \leftarrow N! \dfrac{\hat{x} - \sum_{n=0}^{N-1} \frac{x_{t_i}^{(n)}}{n!}(1-t_i)^n}{(1-t_i)^N}$ ▷ Compute force term, see Sec. 3.2

  $\Psi_{t_i} \approx \displaystyle\int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, \tau)\, \mathbf{b}_\tau F(\mathbf{x}_\tau, \tau)\, d\tau$ ▷ Approx. nonliear part using existing solver (with Q if solver is multistep)

  if Solver is multistep **then** $Q \xleftarrow{\text{cache}} \hat{x}$ ▷ store history

  $\mathbf{x}_{t_{i+1}} \leftarrow \Phi(t_{i+1}, t_i)\mathbf{x}_{t_i} + \Psi_{t_i}$ ▷ state update

end for

return $x_\theta\big((\mathbf{r}_{t_T}^\top \otimes I_d)\, \mathbf{x}_{t_T},\, t_T\big)$ ▷ return results with data prediction

3 Method

We start in Sec. 3.1 with a proof for the training equivalence between momentum diffusion models and conventional diffusion models for any $N \geq 1$. This condition is necessary both for the training-free property of our method as well as for the generalization to arbitrarily large N. We then introduce the proposed method itself in Sec. 3.2. Finally in Sec. 3.3, we study the dynamics of the proposed method and how they tie SDE and ODE formulations.

3.1 Training equivalence between Momentum Diffusion Models and Diffusion Models

In Momentum Diffusion Model case, typically, neural networks are parameterized with $N \in \{2, 3\}$ augmented variables as input. This method naively introduce certain problems. While state-of-the-art diffusion models have developed advanced signal-to-noise ratio (SNR) schedules achieving excellent results [16, 17, 19], defining an appropriate SNR within momentum systems is challenging. Each variable in a momentum system inherently has its own SNR and is coupled via a covariance matrix, complicating the definition and practical usage of SNR. Consequently, this complexity has hindered progress in momentum-based diffusion modeling. Here, we show that the SNR of the momentum diffusion model can be characterized as the optimal SNR achievable through a linear combination of multiple variables, as stated in the following proposition.

Proposition 3.1. *The training objective of general Momentum Diffusion Models (MDM) (i.e., eq. 3) can be equivalently reparameterized as:*

$$\mathcal{L}_{MDM}(\theta) \propto \mathbb{E}_{\mathbf{x}_t}||x_\theta(\mathbf{x}_t, t) - x_1||_2^2 \;\Rightarrow\; x_\theta^*(\mathbf{x}_t, t) = \mathbb{E}[x_1|\mathbf{x}_t]$$

$$\mathbb{E}[x_1|\mathbf{x}_t] = \mathbb{E}\left[x_1|(\mathbf{r}_t^\mathsf{T} \otimes \mathbf{I}_d)\mathbf{x}_t, \boldsymbol{\epsilon}\right] = \mathbb{E}\left[x_1|(\mathbf{r}_t^\mathsf{T} \otimes \mathbf{I}_d)\mathbf{x}_t\right] \quad \textit{where } \mathbf{r}_t := \frac{\mathbf{\Sigma}_t^{-1}\boldsymbol{\mu}_t}{\boldsymbol{\mu}_t^\mathsf{T}\mathbf{\Sigma}_t^{-1}\boldsymbol{\mu}_t}.$$

Moreover, the F_θ in eq. 4 can be recovered as a linear combination of x_θ and $\mathbf{x}_t$ (see section.3.2 for details). Here, $\boldsymbol{\mu}_t^\mathsf{T}\mathbf{\Sigma}_t^{-1}\boldsymbol{\mu}_t$ is the effective SNR of $(\mathbf{r}_t^\mathsf{T} \otimes \mathbf{I}_d)\mathbf{x}_t$ which simply is a weighted linear combination of $\mathbf{x}_t$ by $\mathbf{r}_t$.

Proof. See Appendix. B. □

This proposition holds significant implications despite its apparent conceptual simplicity: It demonstrates that MDM, even when involving multiple input variables to the neural network, can be trained using a single input constructed as a $\mathbf{r}_t$-weighted linear combination of the N input variables. As a consequence, the training objective becomes equivalent to that of a conventional diffusion model, addressing debates about potential advantages in training arising from momentum diffusion or trivial

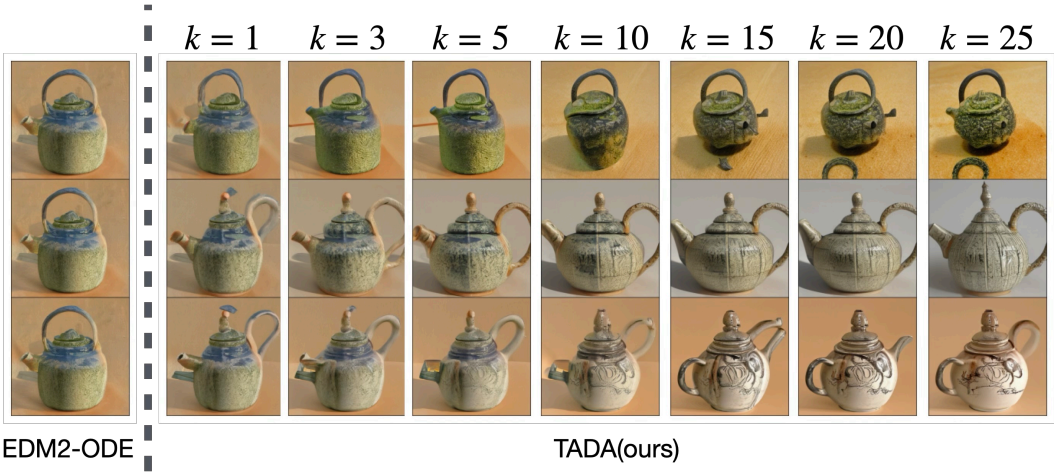

Figure 3: Generated samples under varying prior scales are shown, all initialized with the same initial condition of dynamics in Prop. 3.3:$y_0 := (\mathbf{r}_0^\mathsf{T} \otimes \boldsymbol{I}_d)\mathbf{x}_0 \equiv \epsilon$, using an identical time discretization over the SNR and the same pretrained model with 15 NFEs. It can be observed that the diversity of the generated results increases proportionally with the standard deviation of the final variable $x_0^{(N-1)}$, which is scaled by a factor k: $\boldsymbol{\Sigma}_0 = \text{diag}(1, 1, ..., k)$.

variable augmentations. Importantly, this also allows direct reuse of pretrained conventional diffusion models within the MDM framework and therefore the training-free property claimed by our method.

3.2 Momentum Diffusion Sampling Methodology

[6] highlighted that MDM can yield promising results with small numbers of function evaluations (NFE); however, the absolute performance at sufficient NFE lags behind traditional diffusion models, revealing fundamental training issues. Due to the observation in Prop. 3.1, such training issues can be trivially resolved, or, even more easily, we can simply plug in the pretrained diffusion model.

Training-free Augmented DynAmics (TADA)

We now present the sampling procedure after plugging the pretrained diffusion model into the system. Now the input of neural network becomes $(\mathbf{r}_t^\mathsf{T} \otimes \mathbf{I}_d)\mathbf{x}_t \in \mathbb{R}^d$ instead of $\mathbf{x}_t \in (\mathbb{R}^d)^N$ as is the case in vanilla MDM with $N = 2$. Since $\boldsymbol{\mu}_t^\mathsf{T}\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t$ is the effective SNR in the momentum system, one can simply map it to the time conditioning used in the pretrained model.

We extend the AGM [6] framework to an arbitrary N-variables augmented space. Similarly to Sec. 2.2, we reparameterize the data estimation x_θ to the force term F_θ, which drives the dynamics of $x_t^{(0)}$ toward $x_1^{(0)} \sim p_{\text{data}}$ explicitly. The matrices involved have the closed form:

$$\mathbf{A}_t = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}_{N\times N}, \mathbf{b}_t := \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}; F_\theta(\mathbf{x}_t, t) := N! \frac{x_\theta(\mathbf{r}^\mathsf{T}\mathbf{x}_t, t) - \sum_{n=0}^{N-1} \frac{x_t^{(n)}}{n!}(1-t)^n}{(1-t)^N}.$$

Remark 3.2. *When $N = 1$, this formulation simplifies precisely to vanilla flow matching. When $N = 2$, it is essentially same as the deterministic case of AGM [6] but with the added training-free property that our method carries. Please see Appendix. F.4 for details.*

To compute the reweighting term $\mathbf{r}_t$, the mean and covariance matrix at time t are required. These quantities can be obtained by analytically solving the coupled dynamics of the mean $\boldsymbol{\mu}_t$ and covariance $\boldsymbol{\Sigma}_t$, a standard approach [26] in both conventional DM and MDM. Due to space constraints, the explicit analytical expressions are presented in Appendix F.2.

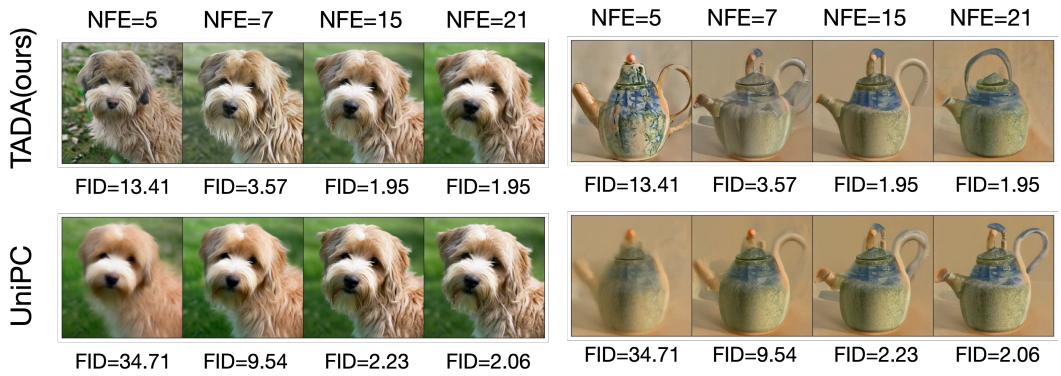

Figure 4: Qualitative comparison with UniPC, varying the NFEs, using the same initial condition and the same pretrained EDM2 model. More qualitative comparision can be found in Appendix. G.

3.3 Analysis of Sampling Dynamics

In conventional diffusion models, variations in generative dynamics largely stem from differences in time discretization schemes, as discussed in [17]; for example, Variance Preserving, Variance Exploding, and Flow Matching dynamics primarily differ in their time discretization over SNR during sampling. This naturally raises the question of whether momentum diffusion simply constitutes another form of time discretization. To address this, we conduct a detailed analysis of variable y_t which is fed into the neural network in Prop. 3.3.

Proposition 3.3. *The dynamics of neural network input $y_t \coloneqq (\mathbf{r}_t^\mathsf{T} \otimes \boldsymbol{I}_d)\mathbf{x}_t$ is given by:*

$$\frac{\mathrm{d}y_t}{\mathrm{d}t} = \underbrace{\sum_{i=0}^{N-1} w_t^{(i)} \epsilon^{(i)}(x_t^{(i)})}_{\textit{Pseudo Noise}} + \alpha_t y_t + \beta_t x_\theta(y_t, t).$$

For the coefficient of w_t^i, α_t and β_t, please refer to Appendix. C. $\epsilon_t^{(i)}$ denotes the estimated Gaussian noise for each variable x_t^i, induced by x_θ. This is analogous to the standard diffusion model but extended to the multi-variable case. Please see Appendix.C for more detail.

Proof. See Appendix. C. □

Remark 3.4. *There are two scenarios in which Prop 3.3 degenerates into a mere different time discretization of the conventional diffusion model, irrespective of the value of N. The first occurs when $N = 1$, and the second arises when $\mathbf{A}_t$ is a diagonal matrix, implying that each variable evolves independently. Further details and proof are provided in Appendix. F.5.*

As demonstrated in Prop 3.3, the dynamics of the neural network input y_t cannot be expressed solely as a function of y_t; rather, an additional *pseudo noise* term emerges due to interactions among variables, endowing the system with SDE properties, even when solving the deterministic ODE in Eq. 4. Notably, we empirically find that, the diversity of samples generated from the same prior can be explicitly manipulated by scaling the standard deviation of the final variable $x_0^{(N-1)}$ by a factor of k, corresponding to the last diagonal entry of the covariance matrix $\boldsymbol{\Sigma}_0$, as illustrated in Fig. 3.

4 Experiments

In this section, we evaluate the performance TADA in comparison with a range of ODE and SDE solvers based on conventional first-order diffusion probabilistic flows. Specifically, we assess diffusion models such as EDM [17], EDM2 [18], in both pixel and latent spaces on the ImageNet-64 and ImageNet-512 datasets [8]. Additionally, we evaluate the flow matching model Stable Diffusion 3 [10] in the latent space.

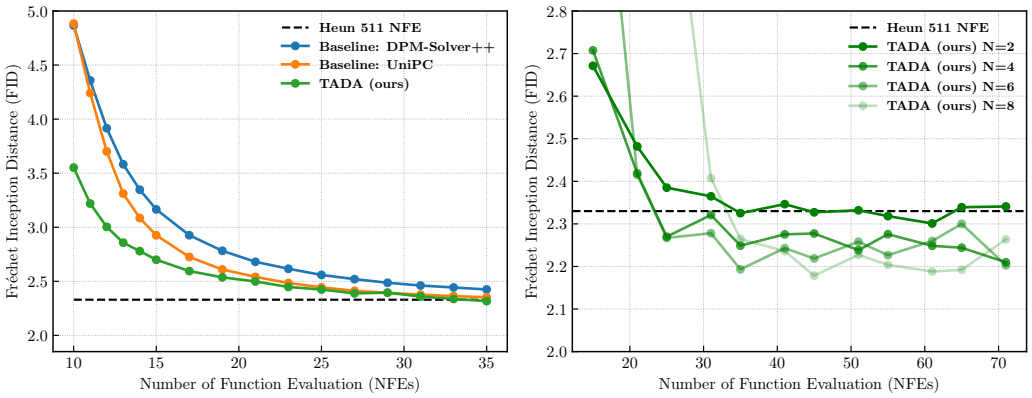

Figure 5: *left*: Comparison with baselines on ImageNet-64 using EDM pretrained model. *Right*: Performance under varying numbers of variables N while keeping the SNR-based time discretization same to the $N = 2$ setting;$N = 2$ is the default configuration reported throughout this paper.

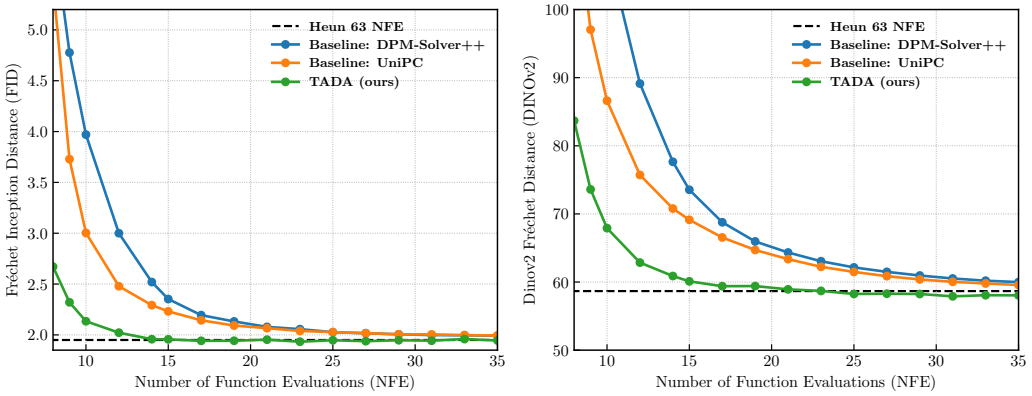

Figure 6: Comparison with baselines on ImageNet-512 with EDM2 pretrained model.

4.1 EDM and EDM2 Experiment

We begin by benchmarking our method against UniPC [34] and DPM-Solver++ [23] on both ImageNet-64 and ImageNet-512, using the EDM [17] and EDM2 [18] frameworks. To ensure fair and competitive evaluation, we conduct comprehensive ablations over all combinations of discretization schemes, solver variants, and solver orders available in the respective codebases, selecting the best-performing configurations for comparison. Full ablation results for the baselines are reported in the Supplementary Material. For our method, TADA , we consistently employ the simplest multistep exponential integrator with third-order solvers, using the same polynomial discretization across all experiments.

We evaluate performance using Fréchet Inception Distance score(FID [13]) for both ImageNet-64 and ImageNet-512, and additionally use Fréchet Distance-DINOv2 (FD-DINOv2 [29]) for ImageNet-512. Our results show that TADA consistently outperforms the baselines across all tested numbers of function evaluations (NFEs) in fig. 5 and fig. 6. We also examine the case with $N > 2$, which introduces additional pseudo noise as described in Prop. 3.3. In all setups, we use the same discretization over SNR and vary only the standard deviation of $x_0^{(N-1)}$. This configuration results in improved performance on ImageNet-64, as shown in fig 5. However, no consistent improvement is observed on ImageNet-512, which may be attributed to the high capacity of the neural network and limited exploration of time discretization, thereby diminishing the impact of the additional noise perturbation. For consistency, we therefore report results only for the $N = 2$ in fig. 6 and the rest of experiments.

Fig 4 presents qualitative results under varying NFE budgets. Unlike conventional ODE solvers, which tend to produce increasingly blurred outputs as NFEs decrease, TADA continues to generate

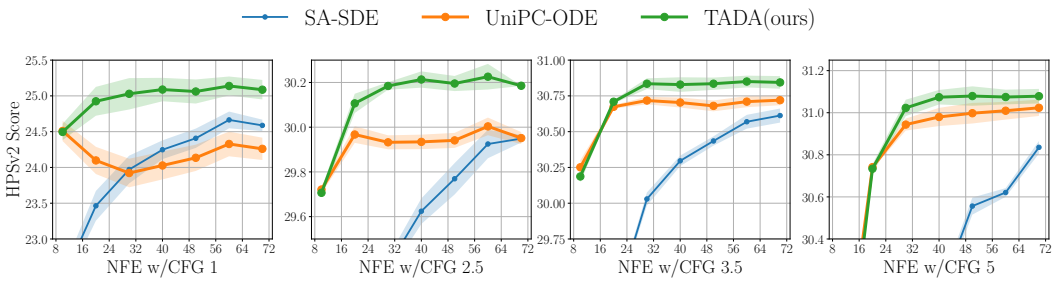

Figure 7: Comparison with baselines with HpsV2 metrics with SD3 pretrained model.

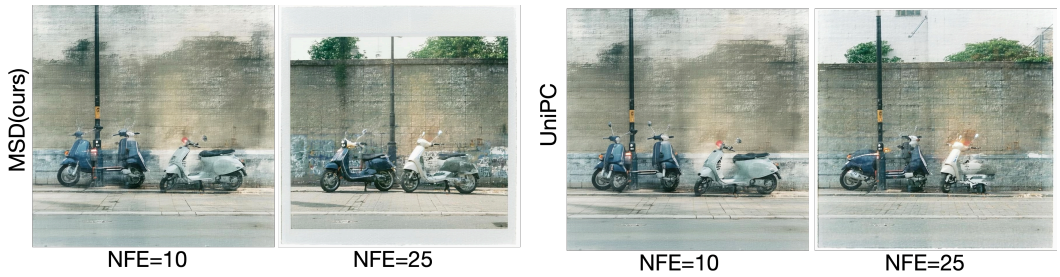

Figure 8: Qualitative Comparison with baselines with SD3 model w/o CFG. UniPC delivers no additional gains in image quality, whereas our method continues to improve. This trend is evident both in Fig. 7 and in the samples generated at NFE = 25.

plausible images with preserved details. Nevertheless, these images may still deviate from the data distribution p_{data}, as indicated by the larger FID metric. See Appendix. G for more examples.

4.2 SD3 Experiment

For evaluations on Stable Diffusion 3 [10], we exclude DPM-Solver++ from comparison, as UniPC has consistently outperformed it in Section 4.1. Additionally, we observe that the default Flow Matching solver in SD3 achieves performance comparable to UniPC. To enrich our baseline set, we include a recently proposed SDE-based solver, Stochastic Adam (SA) [32]. All baseline configurations follow the recommended settings from the official Diffusers implementation. For evaluation, we adopt HPSv2 [31] as our primary benchmark. Specifically, we generate images for all benchmark prompts, producing 3200 samples per seed, and report the mean and standard deviation over three random seeds, thus, totally 9600 images. To ensure consistency, our method uses the same SNR-based discretization scheme as SD3, which is also the default across all baseline implementations. For the sake of consistency, we employ a second-order multi-step solver, regardless of CFG scale.

Fig. 7 presents the quantitative performance of TADA compared to the baselines. Our proposed method consistently outperforms the baselines across all CFG strengths. However, the performance gap narrows as the CFG increases, suggesting that TADA is particularly effective at lower CFG values, where it achieves more pronounced improvements over existing methods. Fig 8 demonstrates that TADA produces images with more semantically coherent content at 25 NFEs than the baseline, an observation further supported by the quantitative metrics presented in Fig. 7.

5 Conclusions and Limitations

In this paper, we have identified and elucidated the underlying training principle of the momentum diffusion model, demonstrating that it fundamentally aligns with that of conventional diffusion models. This observation enables the direct integration of pretrained diffusion models into momentum-based systems without additional training. Furthermore, we conducted a thorough analysis of the sampling dynamics associated with this approach and discovered that the implicit system dynamics introduce additional pseudo-noise. Empirical evaluations confirmed that this characteristic indeed enhances the sampling quality across various datasets and pretrained models.

However, our approach also has limitations. On the theoretical side, Prop 3.3 does not disentangle the degrees of freedom associated with the pseudo noise dimension N and the weighting coefficients $w_t^{(i)}$. As a result, while Fig.5 demonstrates that increasing system stochasticity can lead to improved outcomes, our method still falls short of matching the SDE results reported in [17] with 511 NFEs. This shortfall stems from the limited control over the injected stochasticity due to the naive choice of $\mathbf{A}_t$. On the practical side, experiments with SD3 show that as model capacity and CFG strength increase, the performance differences among solvers, dynamical formulations, and between SDE and ODE sampling become increasingly negligible,a trend that also holds for TADA. Conversely, the results underscore TADA's strength in under-parameterized settings, where modest-capacity DM must tackle large scale, high-dimensional data such as those encountered in video generation. Lastly, TADA currently uses a basic exponential integrator. Evaluating further improvement with more advanced solvers, such as UniPC and DPM-Solver++, remains an critical direction for future work.

Broader Impacts: TADA boosts DM efficiency, accelerating the spread of generated content.

References

[1] Kendall Atkinson, Weimin Han, and David E Stewart. *Numerical solution of ordinary differential equations*. John Wiley & Sons, 2009.

[2] R Bank, RL Graham, J Stoer, and R Varga. Springer series in 8. 1987.

[3] David Berthelot, Arnaud Autef, Jierui Lin, Dian Ang Yap, Shuangfei Zhai, Siyuan Hu, Daniel Zheng, Walter Talbott, and Eric Gu. Tract: Denoising diffusion models with transitive closure time-distillation. *arXiv preprint arXiv:2303.04248*, 2023.

[4] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[5] John C Butcher. *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, 2008.

[6] Tianrong Chen, Jiatao Gu, Laurent Dinh, Evangelos A Theodorou, Joshua Susskind, and Shuangfei Zhai. Generative modeling with phase stochastic bridges. *arXiv preprint arXiv:2310.07805*, 2023.

[7] Tianrong Chen, Guan-Horng Liu, Molei Tao, and Evangelos A Theodorou. Deep momentum multi-marginal schr\" odinger bridge. *arXiv preprint arXiv:2303.01751*, 2023.

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

[9] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. *arXiv preprint arXiv:2112.07068*, 2021.

[10] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[12] Karl Heun et al. Neue methoden zur approximativen integration der differentialgleichungen einer unabhängigen veränderlichen. *Z. Math. Phys*, 45:23–38, 1900.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

[15] Marlis Hochbruck and Alexander Ostermann. Exponential integrators. *Acta Numerica*, 19: 209–286, 2010.

[16] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning*, pages 13213–13232. PMLR, 2023.

[17] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35: 26565–26577, 2022.

[18] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24174–24184, 2024.

[19] Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems*, 36:65484–65516, 2023.

[20] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[21] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[22] 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. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[23] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[24] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[25] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[26] Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

[27] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

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

[29] George Stein, Jesse Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L Caterini, Eric Taylor, and Gabriel Loaiza-Ganem. Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models. *Advances in Neural Information Processing Systems*, 36:3732–3784, 2023.

[30] Daniel Watson, William Chan, Jonathan Ho, and Mohammad Norouzi. Learning fast samplers for diffusion models by differentiating through sample quality. In *International Conference on Learning Representations*, 2021.

[31] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

[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. *Advances in Neural Information Processing Systems*, 36:77632–77674, 2023.

[33] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022.

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

[35] Wenliang Zhao, Haolin Wang, Jie Zhou, and Jiwen Lu. Dc-solver: Improving predictor-corrector diffusion sampler via dynamic compensation. In *European Conference on Computer Vision*, pages 450–466. Springer, 2024.

[36] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solverv3: Improved diffusion ode solver with empirical model statistics. *Advances in Neural Information Processing Systems*, 36: 55502–55542, 2023.

Appendix

A Notations

In this appendix, we follow the notation introduced before. However, in order to reduce the complexity of the derivation, we simplify all the coefficient with Kronecker product. All the variable which sololy depends on the time, will be broadcast by the kroneker product, For example $\mathbf{A}_t := \mathbf{A}_t \otimes \mathbf{I}_d$, $\boldsymbol{\mu}_t := \boldsymbol{\mu}_t \otimes \mathbf{I}_d$, $\mathbf{L}_t^v v = \mathbf{L}_t^v v \otimes \mathbf{I}_d$ etc.

B Proof of Proposition. 3.1

Proof. We first analyze the objective function of momentum diffusion model for $N = 2$ case, and it can be generalize to larger N. **For clarity in dimensional alignment and derivational correctness, we refer the reader to the simplified notation in Section A, which will also be adopted throughout this section.**

$$\min_\theta \mathcal{L}_{\text{MDM}}(\theta) = \mathbb{E}_{x_1,\boldsymbol{\epsilon},t}\|\epsilon_\theta(\mathbf{x}_t, t) - \epsilon^{(1)}\|_2^2 \tag{6}$$

$$= \mathbb{E}_{x_1,\boldsymbol{\epsilon},t}\frac{1}{{L_t^{vv}}^2}\|L_t^{vv}\epsilon_\theta(\mathbf{x}_t, t) - L_t^{vv}\epsilon^{(1)}\|_2^2 \tag{7}$$

$$= \mathbb{E}_{x_1,\boldsymbol{\epsilon},t}\frac{1}{{L_t^{vv}}^2}\|L_t^{vv}\epsilon_\theta(\mathbf{x}_t, t) - \left[x_t^{(1)} - \frac{L_t^{xv}}{L_t^{xx}}x_t^{(0)} - \left(\boldsymbol{\mu}_t{}^{(0)} - \frac{L_t^{xv}}{L_t^{xx}}\boldsymbol{\mu}_t{}^{(1)}\right)x_1\right]\|_2^2 \tag{8}$$

$$= \mathbb{E}_{x_1,\boldsymbol{\epsilon},t}\frac{1}{{L_t^{vv}}^2}\|L_t^{vv}\epsilon_\theta(\mathbf{x}_t, t) - x_t^{(1)} + \frac{L_t^{xv}}{L_t^{xx}}x_t^{(0)} - \left(\frac{L_t^{xv}}{L_t^{xx}}\boldsymbol{\mu}_t{}^{(1)} - \boldsymbol{\mu}_t{}^{(0)}\right)x_1\|_2^2 \tag{9}$$

$$= \mathbb{E}_{x_1,\boldsymbol{\epsilon},t}\frac{\left(\frac{L_t^{xv}}{L_t^{xx}}\boldsymbol{\mu}_t{}^{(1)} - \boldsymbol{\mu}_t{}^{(0)}\right)^2}{{L_t^{vv}}^2}\|\underbrace{\frac{L_t^{vv}\epsilon_\theta(\mathbf{x}_t, t) - x_t^{(1)} + \frac{L_t^{xv}}{L_t^{xx}}x_t^{(0)}}{\frac{L_t^{xv}}{L_t^{xx}}\boldsymbol{\mu}_t{}^{(1)} - \boldsymbol{\mu}_t{}^{(0)}}}_{\text{parameterized Neural Netowrk}} - x_1\|_2^2 \tag{10}$$

Following the same spirit, one can derive the case for N variable. See Appendix. J for details.

We know that,

$$\mathbf{x}_t \mid x_1 \sim \mathcal{N}\big(\boldsymbol{\mu}_t x_1,\ \boldsymbol{\Sigma}_t\big),$$

Define

$$\mathbf{r}_t := \frac{\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t}{\boldsymbol{\mu}_t{}^\top\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t}, \quad y_t := \mathbf{r}_t^\top \mathbf{x}_t,$$

and the residual ("noise")

$$\boldsymbol{\epsilon} := \mathbf{x}_t - y_t\,\boldsymbol{\mu}_t.$$

Since the operation is linear, one can transform the $\mathbf{x}_t$ by

$$T:\ \mathbf{x}_t \ \mapsto\ (y_t, \boldsymbol{\epsilon}) = \big(\mathbf{r}_t^\top \mathbf{x}_t,\ \mathbf{x}_t - \tfrac{(\mathbf{r}_t^\top \mathbf{x}_t)}{a_t}\boldsymbol{\mu}_t\big)$$

and it is *invertible* with linear inverse $\mathbf{x}_t = y_t\boldsymbol{\mu}_t + \boldsymbol{\epsilon}$. Hence the σ-algebras coincide:

$$\sigma(\mathbf{x}_t) = \sigma\big(y_t, \boldsymbol{\epsilon}\big).$$

For any integrable random variable Z, equal σ-fields imply $\mathbb{E}[Z \mid \mathbf{x}_t] = \mathbb{E}[Z \mid y_t, \boldsymbol{\epsilon}]$. Taking $Z = x_1$ yields

$$\mathbb{E}[x_1 \mid \mathbf{x}_t] = \mathbb{E}[x_1 \mid y_t, \boldsymbol{\epsilon}].$$

due to the fact that $\boldsymbol{\epsilon}$ is the independent gaussian, thus

$$\mathbb{E}[x_1 \mid \mathbf{x}_t] = \mathbb{E}[x_1 \mid y_t, \boldsymbol{\epsilon}] = \mathbb{E}[x_1 \mid y_t, \boldsymbol{\epsilon}] = \mathbb{E}[x_1 \mid y_t].$$

□

C Proof of Proposition. 3.3

For clarity in dimensional alignment and derivational correctness, we refer the reader to the simplified notation in Section A, which will also be adopted throughout this section.

Proof. The dynamics we considered reads

$$\frac{\mathrm{d}\mathbf{x}_t}{dt} = \mathbf{A}_t\,\mathbf{x}_t + \mathbf{b}_t F_t, \quad x_0 \sim \mathcal{N}(0, I). \tag{11}$$

and again, in this paper, we only consider,

$$\mathbf{A}_t = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}_{N\times N}, \text{and} \quad \mathbf{b}_t := \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \tag{12}$$

If we expand the system, it basically represent:

$$\mathrm{d}x_t^{(0)} = x_t^{(1)}\mathrm{d}t \tag{13}$$
$$\mathrm{d}x_t^{(1)} = x_t^{(2)}\mathrm{d}t \tag{14}$$
$$\cdots \tag{15}$$
$$\mathrm{d}x_t^{(N-1)} = F_t\mathrm{d}t \tag{16}$$
$$\tag{17}$$

In our case, the F_t function is:

$$F_t := N!\frac{x_1 - \sum_{i=0}^{N-1}\frac{x_t^{(i)}}{i!}(1-t)^i}{(1-t)^N} \tag{18}$$

One can easily verify that, when $N = 1$, it actually degenerate to flow matching:

$$F_t := \frac{x_1 - \mathbf{x}_t}{1-t}, \text{and} \tag{19}$$
$$\mathrm{d}x_t^{(0)} = F_t\mathrm{d}t \tag{20}$$

One can consider it as the higher augment dimension extension of flow matching model. And the magical part is that, we do not need to retrain model.

For better analysis, we rearange the system:

$$\frac{\mathrm{d}\mathbf{x}_t}{dt} = \mathbf{A}_t\,\mathbf{x}_t + \mathbf{b}_t F_t \tag{21}$$
$$= \mathbf{A}_t\,\mathbf{x}_t + \mathbf{b}_t N!\frac{x_1 - \sum_{i=0}^{N-1}\frac{x_t^{(i)}}{i!}(1-t)^i}{(1-t)^N} \tag{22}$$
$$= \mathbf{A}_t\,\mathbf{x}_t + \frac{\mathbf{b}_t N!}{(1-t)^N}x_1 - \frac{\mathbf{b}_t N!\sum_{i=0}^{N-1}\frac{x_t^{(i)}}{i!}(1-t)^i}{(1-t)^N} \tag{23}$$
$$= \mathbf{A}_t\,\mathbf{x}_t + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ \frac{N!}{(1-t)^N} \end{bmatrix}}_{\hat{\mathbf{b}}_t} x_1 - \underbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{(1-t)^0 N!}{0!(1-t)^N} & \frac{(1-t)^1 N!}{1!(1-t)^N} & \cdots & \frac{(1-t)^N N!}{(N-1)!(1-t)^N} \end{bmatrix}}_{\tilde{\mathbf{A}}_t} \mathbf{x}_t \tag{24}$$
$$:= \hat{\mathbf{A}}_t\mathbf{x}_t + \hat{\mathbf{b}}_t x_1, \quad (\hat{\mathbf{A}}_t := \mathbf{A}_t - \tilde{\mathbf{A}}_t) \tag{25}$$

Consider the linear time-varying system:

$$\frac{d\mathbf{x}_t}{dt} = \hat{\mathbf{A}}_t\,\mathbf{x}_t + \hat{\mathbf{b}}_t\,x_1, \quad \mathbf{x}_0 \sim \mathcal{N}(0, I). \tag{26}$$

Since (26) is linear and deterministic (apart from the random initial condition), the state remains Gaussian. Its mean and covariance evolve as follows [26].

Mean Dynamics. Let

$$\mathbf{m}_t = \mathbb{E}[\mathbf{x}_t].$$

Then

$$\dot{\mathbf{m}}_t = \hat{\mathbf{A}}_t\,\mathbf{m}_t + \hat{\mathbf{b}}_t\,x_1, \quad m_0 = 0. \tag{27}$$

We can write the mean in a factorized form as

$$\mathbf{m}_t = \boldsymbol{\mu}_t\,x_1,$$

so that by dividing by x_1, we obtain

$$\dot{\boldsymbol{\mu}}_t = \hat{\mathbf{A}}_t\,\boldsymbol{\mu}_t + \hat{\mathbf{b}}_t.$$

Covariance Dynamics. Similarly, the convariance follows dynamics

$$\dot{\boldsymbol{\Sigma}}_t = \hat{\mathbf{A}}_t\,\boldsymbol{\Sigma}_t + \boldsymbol{\Sigma}_t\,\hat{\mathbf{A}}_t^T \tag{28}$$

Recall that, Then we define the scalar quantity of interest as

$$y_t := \frac{\left(\boldsymbol{\Sigma}_t{}^{-1}\mathbf{m}_t\right)^T\mathbf{x}_t}{\left(\boldsymbol{\Sigma}_t{}^{-1}\mathbf{m}_t\right)^T\boldsymbol{\mu}_t} = \frac{\left(\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t\right)^T\mathbf{x}_t}{\boldsymbol{\mu}_t{}^T\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t} = \frac{\mathbf{r}_t^T\mathbf{x}_t}{\gamma_t}, \quad \text{with } \gamma_t := \boldsymbol{\mu}_t{}^T\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t. \tag{29}$$

We wish to compute the derivative of

$$y_t = \frac{\mathbf{r}_t^T\mathbf{x}_t}{\gamma_t},$$

Using the quotient rule,

$$\dot{y}_t = \frac{\frac{d}{dt}\left(\mathbf{r}_t^T\mathbf{x}_t\right)\,\gamma_t - \left(\mathbf{r}_t^T\mathbf{x}_t\right)\,\dot{\gamma}_t}{\gamma_t^2}.$$

Since $\mathbf{r}_t^T\mathbf{x}_t = y_t\gamma_t$, this becomes

$$\dot{y}_t = \frac{\dot{\mathbf{r}}_t^T\mathbf{x}_t + \mathbf{r}_t^T\dot{x}_t}{\gamma_t} - y_t\,\frac{\dot{\gamma}_t}{\gamma_t}.$$

Derivative of $\mathbf{r}_t$

Recall that

$$\mathbf{r}_t = \boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t.$$

Differentiating gives

$$\dot{\mathbf{r}}_t = -\boldsymbol{\Sigma}_t{}^{-1}\dot{\boldsymbol{\Sigma}}_t\,\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t + \boldsymbol{\Sigma}_t{}^{-1}\dot{\boldsymbol{\mu}}_t.$$

Substitute the known dynamics:

$$\dot{\boldsymbol{\Sigma}}_t = \hat{\mathbf{A}}_t\,\boldsymbol{\Sigma}_t + \boldsymbol{\Sigma}_t\,\hat{\mathbf{A}}_t^T,$$
$$\dot{\boldsymbol{\mu}}_t = \hat{\mathbf{A}}_t\,\boldsymbol{\mu}_t + \hat{\mathbf{b}}_t.$$

It follows that

$$\begin{aligned}
\dot{\mathbf{r}}_t &= -\boldsymbol{\Sigma}_t{}^{-1}\dot{\boldsymbol{\Sigma}}_t\,\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t + \boldsymbol{\Sigma}_t{}^{-1}\dot{\boldsymbol{\mu}}_t && (30)\\
&= -\boldsymbol{\Sigma}_t{}^{-1}\left(\hat{\mathbf{A}}_t\,\boldsymbol{\Sigma}_t + \boldsymbol{\Sigma}_t\,\hat{\mathbf{A}}_t^T\right)\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t + \boldsymbol{\Sigma}_t{}^{-1}\left(\hat{\mathbf{A}}_t\,\boldsymbol{\mu}_t + \hat{\mathbf{b}}_t\right) && (31)\\
&= -\boldsymbol{\Sigma}_t{}^{-1}\hat{\mathbf{A}}_t\boldsymbol{\mu}_t - \hat{\mathbf{A}}_t^{\mathsf{T}}\boldsymbol{\Sigma}_t{}^{-1}\boldsymbol{\mu}_t + \boldsymbol{\Sigma}_t{}^{-1}\hat{\mathbf{A}}_t\boldsymbol{\mu}_t + \boldsymbol{\Sigma}_t{}^{-1}\hat{\mathbf{b}}_t && (32)\\
&= -\,\hat{\mathbf{A}}_t^{\mathsf{T}}\mathbf{r}_t + \boldsymbol{\Sigma}_t{}^{-1}\,\hat{\mathbf{b}}_t. && (33)
\end{aligned}$$

Derivative of $\mathbf{x}_t$

From (26),

$$\dot{\mathbf{x}}_t = \hat{\mathbf{A}}_t\, \mathbf{x}_t + \hat{\mathbf{b}}_t\, \hat{x}_1.$$

Derivative of γ_t

Recall

$$\gamma_t = {\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t = \mathbf{r}_t^T\, \boldsymbol{\mu}_t.$$

Differentiating,

$$\dot{\gamma}_t = \dot{\mathbf{r}}_t^T\, \boldsymbol{\mu}_t + \mathbf{r}_t^T\, \dot{\boldsymbol{\mu}}_t.$$

Using (30) and $\dot{\boldsymbol{\mu}}_t = \hat{A}_t \boldsymbol{\mu}_t + \hat{\mathbf{b}}_t$, one obtains (after cancellation) the result:

$$\dot{\gamma}_t = \Big(-\hat{\mathbf{A}}_t^{\mathsf{T}} \mathbf{r}_t + {\boldsymbol{\Sigma}_t}^{-1}\, \hat{\mathbf{b}}_t\Big)\, \boldsymbol{\mu}_t + \mathbf{r}^{\mathsf{T}}\, \Big(\hat{\mathbf{A}}_t \boldsymbol{\mu}_t + \hat{\mathbf{b}}_t\Big) \tag{34}$$

$$= 2\, \hat{\mathbf{b}}_t^T\, \mathbf{r}_t \tag{35}$$

$$= 2\, \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1} \boldsymbol{\mu}_t \tag{36}$$

Combining Everything

Substitute the pieces into

$$\dot{y}_t = \frac{\dot{\mathbf{r}}_t^T \mathbf{x}_t + \mathbf{r}_t^T \dot{\mathbf{x}}_t}{\gamma_t} - y_t\, \frac{\dot{\gamma}_t}{\gamma_t}.$$

Using

$$\dot{\mathbf{r}}_t^T = -\,\mathbf{r}_t^T\, \hat{\mathbf{A}}_t + \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1},$$
$$\mathbf{r}_t^T \dot{\mathbf{x}}_t = \mathbf{r}_t^T \big(\hat{\mathbf{A}}_t \mathbf{x}_t + \hat{\mathbf{b}}_t\, \hat{x}_1\big),$$

we have:

$$\dot{\mathbf{r}}_t^T \mathbf{x}_t + \mathbf{r}_t^T \dot{\mathbf{x}}_t = \Big[-\,\mathbf{r}_t^T\, \hat{\mathbf{A}}_t\, \mathbf{x}_t + \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \mathbf{x}_t\Big] + \Big[\mathbf{r}_t^T\, \hat{\mathbf{A}}_t\, \mathbf{x}_t + \mathbf{r}_t^T\, \hat{\mathbf{b}}_t\, \hat{x}_1\Big]$$
$$= \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \mathbf{x}_t + \mathbf{r}_t^T\, \hat{\mathbf{b}}_t x_1$$

Thus,

$$\dot{y}_t = \frac{\hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \mathbf{x}_t + x_1\, \mathbf{r}_t^T\, \hat{\mathbf{b}}_t}{\gamma_t} - y_t\, \frac{\dot{\gamma}_t}{\gamma_t}. \tag{37}$$

Recall that $\gamma_t = {\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t$ and $\dot{\gamma}_t = 2\, \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t$. Also note that

$$\mathbf{r}_t^T\, \hat{\mathbf{b}}_t = {\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \hat{\mathbf{b}}_t.$$

Thus, the final expression becomes

$$\dot{y}_t = \frac{\hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \mathbf{x}_t + \hat{x}_1\, {\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \hat{\mathbf{b}}_t}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} - y_t\, \frac{2\, \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} \tag{38}$$

$$= \underbrace{\frac{\hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t}}_{\mathbf{e}_t}\, \mathbf{x}_t + \frac{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \hat{\mathbf{b}}_t}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} x_1 - \frac{2\, \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} y_t \tag{39}$$

The frist term is essentially one kind of linear combination of $\mathbf{x}_t$, and Recally that $y_t = \mathbf{r}^{\mathsf{T}} \mathbf{x}_t := \frac{\boldsymbol{\mu}_t^{\mathsf{T}} \boldsymbol{\Sigma}_t^{-1}}{\boldsymbol{\mu}_t^{\mathsf{T}} \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\mu}_t}$ which is another linear combination of $\mathbf{x}_t$. Assume that $\mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_t x_1, \boldsymbol{\Sigma}_t)$, thus, one can derive the relationship between the frist term and y_t. Thus, according to Lemma. I.1

$$\frac{\hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} \mathbf{x}_t = \mathbf{e}_t^{\mathsf{T}} \left[\mathbf{I} - \frac{\boldsymbol{\Sigma}_t \mathbf{r}_t \mathbf{r}_t^{\mathsf{T}}}{\mathbf{r}_t^{\mathsf{T}} \boldsymbol{\Sigma}_t \mathbf{r}_t}\right] \boldsymbol{\mu}_t x_1 + \frac{\mathbf{e}_t^{\mathsf{T}} \boldsymbol{\Sigma}_t \mathbf{r}_t}{\mathbf{r}_t^{\mathsf{T}} \boldsymbol{\Sigma}_t \mathbf{r}_t} y_t + \mathbf{e}_t^{\mathsf{T}} \mathbf{L}_t \boldsymbol{\epsilon}_\perp, \tag{40}$$

$$\boldsymbol{\epsilon}_\perp \sim \mathcal{N}\big(\mathbf{0},\ I_d - \mathbf{L}_t^{\mathsf{T}} \mathbf{r}_t \mathbf{r}_t^{\mathsf{T}} \mathbf{L}_t / \mathbf{r}_t^{\mathsf{T}} \boldsymbol{\Sigma}_t \mathbf{r}_t\big). \tag{41}$$

Thus, by plugging in the expression, one can get:

$$\dot{y}_t = \alpha_t y_t + \beta x_1 + \mathbf{e}_t^\mathsf{T} \mathbf{L}_t \boldsymbol{\epsilon}_\perp \tag{42}$$

$$\approx \alpha_t y_t + \beta x_\theta + \mathbf{e}_t^\mathsf{T} \mathbf{L}_t \boldsymbol{\epsilon}_\perp \tag{43}$$

Where,

$$\alpha_t = \frac{\mathbf{e}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t}{\mathbf{r}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t} - \frac{2\, \hat{\mathbf{b}}_t^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} \tag{44}$$

$$\beta_t = \frac{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \hat{\mathbf{b}}_t}{{\boldsymbol{\mu}_t}^T\, {\boldsymbol{\Sigma}_t}^{-1}\, \boldsymbol{\mu}_t} + \mathbf{e}_t^\mathsf{T} \left[\mathbf{I} - \frac{\boldsymbol{\Sigma}_t \mathbf{r}_t \mathbf{r}_t^\mathsf{T}}{\mathbf{r}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t} \right] \boldsymbol{\mu}_t \tag{45}$$

$$w_t^{(i)} = (\mathbf{e}_t^\mathsf{T} \mathbf{L}_t)^{(i)} \tag{46}$$

□

D Experiment Details

Here we elaborate more on experiment details.

D.1 EDM and EDM2

For the baselines on EDM and EDM2 codebase, we directly use the code provide in DPM-Solver-v3 [36]. For the fair comparision, for all baselines, we controlled $\sigma_{\text{min}} = 0.002$ and $\sigma_{\text{max}} = 80$ as suggested in the original EDM and EDM2 paper.

For DPM-Solver++, we did abalation search over order $\in [1, 2, 3]$, discretization $\in$ [logSNR, time uniform, , edm, time quadratic].

For UniPC, we did abalation search over order $\in [1, 2, 3]$, discretization $\in$ [logSNR, time uniform, , edm, time quadratic],variant $\in$ [bh1, bh2].

For all the ablation results, please see the supplementary material.

D.2 Stable Diffusion 3

For stable diffusion 3, we simply plug in the implementation of all the baselines provided in the Diffuser. We use latest HpsV2.1 to evaluate generate dresults.

E Additional Plots

E.1 General N variable dynamics

This section is not referenced in the main paper and will be removed soon; it is retained only for now to keep the appendix numbering aligned with the main paper.

F Detailed Explanations

F.1 Explicit form of $\mathbf{A}_t$ and $\mathbf{b}_t$

Here we demonstrate the $\mathbf{A}_t$ and $\mathbf{b}_t$ used in AGM [6] and CLD [9]. Here we abuse the notation and inherent the notation from CLD.

F.2 Explicit form of mean and convariance matrix

Here we first quickly derive how the F derived which is straight-forward. We know $x_t^{(0)} := x_t$ be the position and define higher derivatives recursively

$$x_t^{(k)} \;=\; \frac{d^k x_t^{(0)}}{dt^k}, \quad k = 1, \ldots, N-1.$$

Table 1: Comparison of different solvers

Algorithm	$\mathbf{A}_t$	$\mathbf{b}$	F_t
AGM [9]	$\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}$	$[0,1]^\mathsf{T}$	$\frac{-4}{t-1}\left(\frac{x1-\mathbf{x}_t^{(0)}}{1-t}-\mathbf{x}_t^{(1)}\right)$
CLD [9]	$\begin{bmatrix}0 & -M^{-1}\\ 1 & \Gamma M^{-1}\end{bmatrix}\beta$	$[0,\Gamma\beta]^\mathsf{T}$	$\nabla_{\mathbf{x}_t^{(1)}}\log p(\mathbf{x},t)$

The system dynamics form an *Nth–order chain of integrators* driven by a scalar input $F(t,\mathbf{x}_t)$:

$$\begin{aligned}\dot{x}_t^{(0)} &= x_t^{(1)},\\ \dot{x}_t^{(1)} &= x_t^{(2)},\\ &\vdots\\ \dot{x}_t^{(N-2)} &= x_t^{(N-1)},\\ \dot{x}_t^{(N-1)} &= F(t,\mathbf{x}_t).\end{aligned} \tag{47}$$

Equivalently, the position satisfies the scalar ODE

$$\boxed{\frac{d^N x_t^{(0)}}{dt^N} = F(t,\mathbf{x}_t)}.$$

Our goal is starting at some time $t\in[0,1)$ with known state $\left\{x_t^{(k)}\right\}_{k=0}^{N-1}$, choose F so that the position reaches a prescribed value at $t=1$:

$$x_1^{(0)} = x_1 \quad (\text{“hit the target”}).$$

Assume F is *held constant* over the *remaining* interval $[t,1]$. Repeated integration yields the degree-N Taylor polynomial about t:

$$x_1^{(0)} = \sum_{k=0}^{N-1}\frac{(1-t)^k}{k!}\,x_t^{(k)} + \frac{(1-t)^N}{N!}\,F. \tag{48}$$

Thus, one can simply solve the F by Rearranging (48) to isolate F:

$$F = \frac{N!}{(1-t)^N}\Big[x_1 - \sum_{k=0}^{N-1}\frac{(1-t)^k}{k!}\,x_t^{(k)}\Big].$$

Thanks to the simple form of F, one can readily write down the mean and covariance of the system.

Now we know

$$A \;=\; \begin{bmatrix}0 & 1 & 0 & \dots & 0\\ 0 & 0 & 1 & \dots & 0\\ \vdots & \vdots & \ddots & \ddots & \vdots\\ 0 & 0 & \dots & 0 & 1\\ 0 & 0 & \dots & 0 & 0\end{bmatrix}, \qquad \mathbf{b}_t \;=\; [0,\dots,0,1]^\top.$$

By rearraging the dynamics, gives the *linear* time-varying closed loop

$$\dot{\mathbf{x}}_t = \underbrace{\hat{\mathbf{A}}_t}_{A\text{ w/ control}}\mathbf{x}_t + \underbrace{\hat{\mathbf{b}}}_{b\text{ w/ control}}\,x_1 \tag{49}$$

We need first compute the transition matrix induced by $\hat{\mathbf{A}}_t$ and we call it controlled transition matrix. By Solving $\dot{\Phi} = \hat{\mathbf{A}}_t\Phi$ column-wise gives the polynomial matrix

$$\Phi(t,0) = \big[T_{k,m}(t)\big]_{k,m=0}^{N-1}, \quad T_{k,m}(t) = \begin{cases} \dfrac{t^{\,m-k}}{(m-k)!} - \dfrac{N!\,t^{\,N-k}}{(N-k)!\,m!}, & m \geq k, \\ -\dfrac{N!\,t^{\,N-k}}{(N-k)!\,m!}, & m < k. \end{cases}$$

Plugging this $\Phi(t,0)$ into the boxed formulas above supplies $\mu(t)$ and $\Sigma(t)$ explicitly for *every* order N.

Let $\mu(t) = \mathbb{E}[\mathbf{x}_t]$. Because $\hat{\mathbf{b}}_t x_1$ is deterministic,

$$\dot{\mu}(t) = \hat{\mathbf{A}}_t\,\mu(t) + \hat{\mathbf{b}}_t\,x_1, \qquad \mu(0) = \mu_0. \tag{50}$$

Define the state–transition matrix $\Phi(t,\tau)$ of $\hat{\mathbf{A}}_t$:

$$\dot{\Phi}(t,\tau) = \hat{\mathbf{A}}_t\,\Phi(t,\tau),\ \Phi(\tau,\tau) = I.$$

Then the standard variation-of-constants formula gives

$$\boldsymbol{\mu}_t = \Phi(t,0)\,\boldsymbol{\mu}_0 + \int_0^t \Phi(t,\tau)\,\hat{\mathbf{b}}_\tau\,x_1\,d\tau.$$

If the initial derivatives are i.i.d. $\mathcal{N}(0,1)$, then $\mu_0 = \mathbf{0}$ and only the integral term remains. Carrying out the integral (polynomials of τ) yields

$$\boxed{\boldsymbol{\mu}_t^{(k)} = \frac{N!\,t^{\,N-k}}{(N-k)!}\,x_1, \qquad k = 0,\ldots,N-1.}$$

Meanwhile, the propagation of covariance matrix is:

$$\dot{\boldsymbol{\Sigma}}_t = \hat{\mathbf{A}}_t\boldsymbol{\Sigma}_t + \boldsymbol{\Sigma}_t\hat{\mathbf{A}}_t^\top, \quad \Sigma(0) = \Sigma_0. \tag{51}$$

Eq. 51 is a homogeneous Lyapunov ODE whose unique solution is exactly (see Appendix. H for more details):

$$\boxed{\boldsymbol{\Sigma}_t = \Phi(t,0)\,\boldsymbol{\Sigma}_0\,\Phi(t,0)^\top.}$$

F.3 Previous Fast Solver

Table 2: Comparison of different solvers

	Order type	Order	Multistep type	Expansion term	Discretize space
Heun [17]	Single Step	2	N/A	N/A	σ_t
DEIS [33]	Multi-step	$2/3/4$	Adams–Bashforth	ϵ_θ	σ_t
DPM-Solver [22]	Multi/Single-step	$2/3/4$	Adams–Bashforth	ϵ_θ	Optional
DPM-Solver++ [23]	Multi/Single-step	$2/3/4$	Adams–Bashforth	x_θ	Optional
UniPC [34]	Multi-step	$3/4/5$	Adams–Moulton	x_θ	Optional
TADA(ours)	Multi-step	$2/3$	Adams–Bashforth	F_θ	t

F.4 Extended Flow Matching

In the framework of flow mathcing, one obtain the velocity by $v_t = \frac{x_1 - x_t}{1-t}$ because it is the linear interpolation between data x_1 and prior x_0. And meanwhile, it happens to be the solution of optimal control problem:

$$\min_{v_t} \int_t^1 \|v_t\|_2^2 \mathrm{d}t, \quad s.t \quad \mathrm{d}x_t = v_t \mathrm{d}t \tag{52}$$

For the detailed derivation, please see Sec.C.1 in [6].

For AGM, they consider a momentum system, which reads

$$\min_{a_t} \int_t^1 \|a_t\|_2^2 \mathrm{d}t, \quad s.t \quad \mathrm{d}x_t = v_t \mathrm{d}t, \quad \mathrm{d}v_t = a_t \mathrm{d}t + \mathrm{d}w_t \tag{53}$$

The differences is that, AGM consider the injection of stochasticity in the velocity channel. For our case, the F_θ derived in Sec. 3.2, is the solution for

$$\begin{aligned} \min_{F_t} \int_t^1 \|F_t\|_2^2 \mathrm{d}t \\ \mathrm{d}x_t^{(0)} = x_t^{(1)} \mathrm{d}t \\ \mathrm{d}x_t^{(1)} = x_t^{(2)} \mathrm{d}t \\ \cdots \\ \mathrm{d}x_t^{(N-1)} = F_t \mathrm{d}t \end{aligned}$$

and its spirit keeps same as previous formulation, move $x_t^{(0)}$ to $x_1^0 \sim p_{\mathrm{data}}$ from $t = 0$ to $t = 1$.

F.5 Degenerate Case of TADA

Here we discuss about the degenerated case of TADA. The reasoning behind it is rather simple. We show the dynamics of y_t (eq. 39) again here,

$$\dot{y}_t = \underbrace{\frac{\hat{\mathbf{b}}_t^T \, {\boldsymbol{\Sigma}_t}^{-1}}{{\boldsymbol{\mu}_t}^T \, {\boldsymbol{\Sigma}_t}^{-1} \, \boldsymbol{\mu}_t}}_{\mathbf{e}_t} \mathbf{x}_t + \frac{{\boldsymbol{\mu}_t}^T \, {\boldsymbol{\Sigma}_t}^{-1} \, \hat{\mathbf{b}}_t}{{\boldsymbol{\mu}_t}^T \, {\boldsymbol{\Sigma}_t}^{-1} \, \boldsymbol{\mu}_t} x_1 - \frac{2 \, \hat{\mathbf{b}}_t^T \, {\boldsymbol{\Sigma}_t}^{-1} \, \boldsymbol{\mu}_t}{{\boldsymbol{\mu}_t}^T \, {\boldsymbol{\Sigma}_t}^{-1} \, \boldsymbol{\mu}_t} y_t \tag{54}$$

and recall that

$$y_t = \frac{{\boldsymbol{\mu}_t}^\mathsf{T} \boldsymbol{\Sigma}_t^{-1}}{{\boldsymbol{\mu}_t}^\mathsf{T} \boldsymbol{\Sigma}_t^{-1} \boldsymbol{\mu}_t} \tag{55}$$

Thus, if the first term depends exclusively on y_t, the system reduces to the scalar ODE for y_t and becomes formally identical to other diffusion-model parameterizations such as VP, VE, or FM. More precisely, in order to degenerate TADA, one only requires

$$\boldsymbol{\mu}_t \;\propto\; \hat{\mathbf{b}}_t \tag{56}$$

where $\hat{\mathbf{b}}_t$ is defined in eq. 24. This proportionality holds in two scenarios:

1. When $N = 1$, so that $\boldsymbol{\mu}_t$ and $\hat{\mathbf{b}}_t$ are scalars. In that case, the framework collapses to flow matching—a mere reparameterization of the diffusion model.

2. When $\mathbf{A}_t$ is diagonal and its components evolve independently. Then every dimension of $\boldsymbol{\mu}_t$ and $\hat{\mathbf{b}}_t$ shares the same mean and variance, and proportionality follows directly.

A simple empirical check is to propagate the model from different random initializations using our formulation: it yields identical FID scores after generation, confirming the degeneracy.

G Additional Qualitative Comparision

Please see fig. 10 and fig. 9.

H Solution of the homogeneous Lyapunov ODE

Let $\Phi(t, \tau)$ be the *state–transition matrix* of the (possibly time–varying) coefficient A_t:

$$\dot{\Phi}(t, \tau) = A_t \, \Phi(t, \tau), \qquad \Phi(\tau, \tau) = I.$$

Throughout we abbreviate $\Phi(t, 0) \equiv \Phi(t)$.

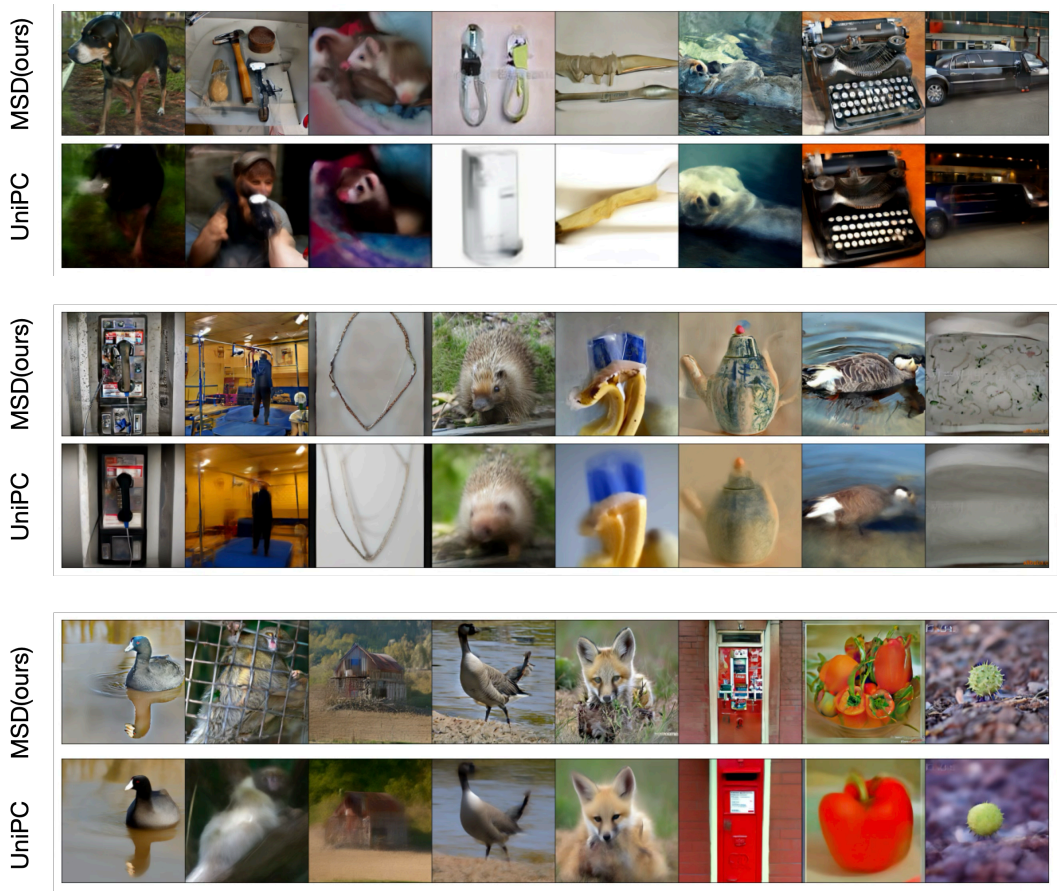

Figure 9: Additional Visual comparision with UniPC using EDM2 w/ 5 NFEs.

1. Candidate solution. Consider

$$\Sigma(t) \;=\; \Phi(t)\,\Sigma_0\,\Phi(t)^\top.$$

2. Verification. Differentiate previous equation and use the product rule together with $\dot{\Phi}(t) = A_t\Phi(t)$ and $\frac{d}{dt}\Phi(t)^\top = \Phi(t)^\top A_t^\top$:

$$\begin{aligned}\dot{\Sigma}(t) &= \dot{\Phi}\,\Sigma_0\Phi^\top + \Phi\,\Sigma_0\dot{\Phi}^\top \\ &= A_t\Phi\Sigma_0\Phi^\top + \Phi\Sigma_0\Phi^\top A_t^\top \\ &= A_t\Sigma(t) + \Sigma(t)A_t^\top.\end{aligned}$$

Hence $\Sigma(t)$ satisfies the differential equation in (Lyap), and $\Sigma(0) = \Phi(0)\Sigma_0\Phi(0)^\top = \Sigma_0$.

3. Uniqueness. Lyapunov Equation is linear in the matrix variable Σ; by the Picard–Lindelöf theorem its solution is unique. Therefore (S) is *the* solution.

I Correlation between two Gaussian Variable

Lemma I.1. *Let the random vector*

$$\mathbf{x}_t \;\sim\; \mathcal{N}\big(\boldsymbol{\mu}\,x_1,\;\boldsymbol{\Sigma}_t\big), \qquad \boldsymbol{\Sigma}_t = \mathbf{L}_t\mathbf{L}_t^\top \quad (\textit{Cholesky factorisation}).$$

For two fixed column-vectors $\mathbf{r}, \mathbf{e} \in \mathbb{R}^d$ *set*

$$y_t := \mathbf{r}_t^\top \mathbf{x}_t, \quad z_t := \mathbf{e}_t^\top \mathbf{x}_t.$$

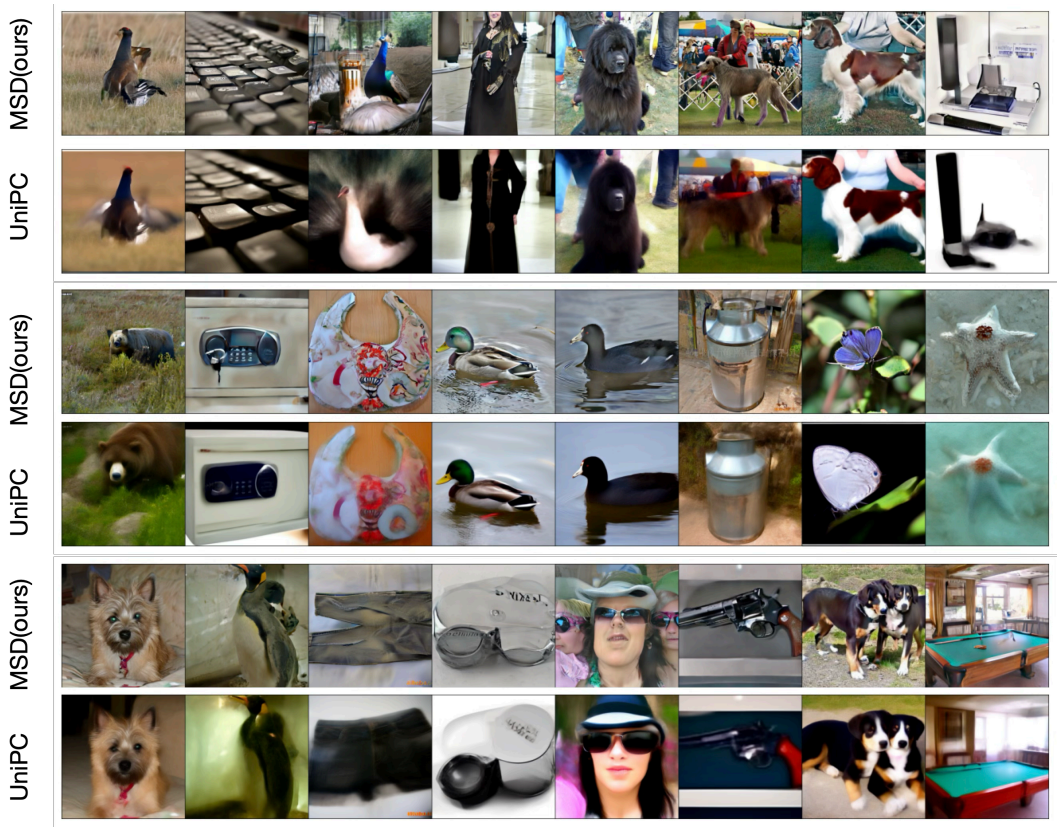

Figure 10: Additional Visual comparision with UniPC using EDM2 w/ 5 NFEs.

Write the convenient abbreviations

$$c_t := \mathbf{r}_t^\top \boldsymbol{\mu}_t \, x_1, \quad d_t := \mathbf{e}_t^\top \boldsymbol{\mu}_t \, x_1, \quad \mathbf{g}_t := \mathbf{L}_t^\top \mathbf{r}_t, \quad \mathbf{h}_t := \mathbf{L}_t^\top \mathbf{e}_t, \quad \sigma_y^2 := \|\mathbf{g}_t\|^2.$$

Then

$$\boxed{z_t = \mathbf{e}_t^\mathsf{T} \left[\mathbf{I} - \frac{\boldsymbol{\Sigma}_t \mathbf{r}_t \mathbf{r}_t^\mathsf{T}}{\mathbf{r}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t} \right] \boldsymbol{\mu}_t x_1 + \frac{\mathbf{e}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t}{\mathbf{r}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t} y_t + \mathbf{e}_t^\mathsf{T} \mathbf{L}_t \boldsymbol{\epsilon}_\perp, \qquad \boldsymbol{\epsilon}_\perp \sim \mathcal{N}\big(\mathbf{0},\ I_d - \mathbf{L}_t^\mathsf{T} \mathbf{r}_t \mathbf{r}_t^\mathsf{T} \mathbf{L}_t / \mathbf{r}_t^\mathsf{T} \boldsymbol{\Sigma}_t \mathbf{r}_t\big).}$$

Proof.

$$y_t = c_t + \mathbf{g}_t^\mathsf{T} \epsilon, \qquad z_t = d_t + \mathbf{h}_t^\mathsf{T} \epsilon.$$

Any vector can be decomposed into the component along $\mathbf{s}$ and the component orthogonal to $\mathbf{s}$:

$$\boldsymbol{\epsilon} = \frac{\mathbf{g}_t}{\sigma_y^2} \left(\mathbf{g}_t^\mathsf{T} \boldsymbol{\epsilon} \right) + \boldsymbol{\epsilon}_\perp, \qquad \mathbf{g}_t^\mathsf{T} \boldsymbol{\epsilon}_\perp = 0$$

Because $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I_d)$ and the projector onto $\mathbf{g}_t$ is orthogonal to the projector onto the complement, $\mathbf{g}_t^\mathsf{T} \boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}_\perp$ are independent Gaussian variables.

Insert $\mathbf{g}_t^\mathsf{T} \boldsymbol{\epsilon} = y_t - c_t$ to obtain

$$\boldsymbol{\epsilon} = \frac{\mathbf{g}_t}{\sigma_y^2} \left(y_t - c_t \right) + \boldsymbol{\epsilon}_\perp, \qquad \boldsymbol{\epsilon}_\perp \sim \mathcal{N}\big(\mathbf{0},\ I_d - \mathbf{g}_t \mathbf{g}_t^\mathsf{T} / \sigma_y^2\big).$$

Then we can plug this into z_t:

$$z_t = d_t + \mathbf{h}_t^\mathsf{T}\Big(\frac{\mathbf{g}_t}{\sigma_y^2}(y_t - c_t) + \boldsymbol{\epsilon}_\perp\Big) \tag{57}$$

$$= \mathbf{e}_t^\mathsf{T}\boldsymbol{\mu}_t x_1 + \frac{\mathbf{h}_t^\mathsf{T}\mathbf{g}_t}{\sigma_y^2}(y_t - c_t) + \mathbf{h}_t^\mathsf{T}\boldsymbol{\epsilon}_\perp \tag{58}$$

$$= \mathbf{e}_t^\mathsf{T}\boldsymbol{\mu}_t x_1 + \frac{\mathbf{h}_t^\mathsf{T}\mathbf{g}_t}{\sigma_y^2}y_t - \frac{\mathbf{h}_t^\mathsf{T}\mathbf{g}_t\mathbf{r}_t^\mathsf{T}\boldsymbol{\mu}_t}{\sigma_y^2}x_1 + \mathbf{h}_t^\mathsf{T}\boldsymbol{\epsilon}_\perp \tag{59}$$

$$= \mathbf{e}_t^\mathsf{T}\boldsymbol{\mu}_t x_1 + \frac{\mathbf{e}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}{\mathbf{r}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}y_t - \frac{\mathbf{e}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}{\mathbf{r}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}\left(\mathbf{r}_t^\mathsf{T}\boldsymbol{\mu}_t\right)x_1 + \mathbf{e}_t^\mathsf{T}\mathbf{L}_t\boldsymbol{\epsilon}_\perp \tag{60}$$

$$= \mathbf{e}_t^\mathsf{T}\left[\mathbf{I} - \frac{\boldsymbol{\Sigma}_t\mathbf{r}_t\mathbf{r}_t^\mathsf{T}}{\mathbf{r}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}\right]\boldsymbol{\mu}_t x_1 + \frac{\mathbf{e}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}{\mathbf{r}_t^\mathsf{T}\boldsymbol{\Sigma}_t\mathbf{r}_t}y_t + \mathbf{e}_t^\mathsf{T}\mathbf{L}_t\boldsymbol{\epsilon}_\perp \tag{61}$$

□

J General N variable MDM loss

At time t the N-variables are generated by

$$\boxed{\mathbf{x}_t \;=\; \boldsymbol{\mu}_t\, x_1 \;+\; \mathbf{L}_t\,\boldsymbol{\varepsilon}} \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, I_N),$$

with known $\boldsymbol{\mu}_t \in \mathbb{R}^N$ and *invertible* $\mathbf{L}_t \in \mathbb{R}^{N\times N}$. The goal is to predict $\varepsilon^{(N-1)}$.

By whitening trick, one can isolate $\boldsymbol{\epsilon}^{(N-1)}$. Let $\mathbf{e}_{N-1}^\top = [0, \ldots, 0, 1]$ be the row vector that selects the last coordinate. Left-multiplying by $\mathbf{e}_{N-1}^\top\mathbf{L}_t^{-1}$ gives

$$\mathbf{e}_{N-1}^\top\mathbf{L}_t^{-1}\mathbf{x}_t \;=\; \mathbf{e}_{N-1}^\top\mathbf{L}_t^{-1}\boldsymbol{\mu}_t\, x_1 \;+\; \mathbf{e}_{N-1}^\top\underbrace{\mathbf{L}_t^{-1}L_t}_{I_N}\boldsymbol{\varepsilon}.$$

Define the *time-dependent scalars*

$$\mathbf{a}_t^\top := \mathbf{e}_{N-1}^\top\mathbf{L}_t^{-1} \in \mathbb{R}^N, \quad b_t := \mathbf{a}_t^\top\boldsymbol{\mu}_t \neq 0,$$

then

$$\boldsymbol{\epsilon}^{(N-1)} \;=\; \mathbf{a}_t^\top\mathbf{x}_t \;-\; b_t\, x_1. \tag{62}$$

A neural network $\varepsilon_\theta(\mathbf{x}_t, t)$ is trained to approximate $\varepsilon^{(N-1)}$ with the standard

$$\mathcal{L}_{\text{MDM}}(\theta) := \mathbb{E}\|\varepsilon_\theta(\mathbf{x}_t, t) - \boldsymbol{\epsilon}^{(N-1)}\|_2^2$$

Insert eq. 62 and multiply the interior by b_t:

$$\begin{aligned}\mathcal{L}_{\text{MDM}}(\theta) &= \mathbb{E}\|\varepsilon_\theta(\mathbf{x}_t, t) - \mathbf{a}_t^\top\mathbf{x}_t + b_t\, x_1\|_2^2 \\ &\propto \mathbb{E}\|g_\theta(\mathbf{x}_t, t) - x_1\|_2^2\end{aligned}$$

where

$$g_\theta(\mathbf{x}_t, t) := -\frac{\varepsilon_\theta(\mathbf{x}_t, t) - \mathbf{a}_t^\top\mathbf{x}_t}{b_t}$$

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: we provide theocratically and empirically analysis in the main paper, which is aligned with what abstract as introduction described.

   Guidelines:

   - The answer NA 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 NA 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: We have a limitation section in the end.

Guidelines:

- The answer NA 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: We have rigorous proof with assumption in the appendix, and the statement of the proof is accurate.

   Guidelines:

   - The answer NA 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: We provide the detailed pseudo code in the paper and We will release the code later.

   Guidelines:

- The answer NA 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: We do not provide code in the paper, but we denote the open access link for the data we used.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We discussed the experiment detail and setup in the paper.

   Guidelines:

   - The answer NA 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: Yes, we provide the standard deviation over seeds in the experiments.

   Guidelines:

   - The answer NA 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: We will disclose the computation resource in the paper.

   Guidelines:

   - The answer NA 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 do.

   Guidelines:

   - The answer NA 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: We have a short broader impacts discussion in the main paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

Justification: This is the initial exploration of a brand new method. It has not reached the level of safeguards, so we did not describe it.

Guidelines:

- The answer NA 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: We took care of the liscence and terms of use.

Guidelines:

- The answer NA 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: [NA]

Justification: We did not release a new assets. The assets will be documented once we release it.

Guidelines:

- The answer NA 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: [No]

Justification: we do not have crowdsourcing experiments.

Guidelines:

- The answer NA 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: [No]

Justification: we do not have human subjects in this work.

Guidelines:

- The answer NA 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 rigorousness, or originality of the research, declaration is not required.

Answer: [No]

Justification: We did not use LLM for important, original component of the core methods in the paper.

Guidelines:

- The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.