Lookahead Diffusion Probabilistic Models for Refining Mean Estimation

Guoqiang Zhang
University of Technology Sydney
guoqiang.zhang@uts.edu.au

Kenta Niwa
NTT Communication Science Laboratories
kenta.niwa.bk@hco.ntt.co.jp

W. Bastiaan Kleijn
Victoria University of Wellington
bastiaan.kleijn@vuw.ac.nz

Abstract

We propose lookahead diffusion probabilistic models (LA-DPMs) to exploit the correlation in the outputs of the deep neural networks (DNNs) over subsequent timesteps in diffusion probabilistic models (DPMs) to refine the mean estimation of the conditional Gaussian distributions in the backward process. A typical DPM first obtains an estimate of the original data sample $\boldsymbol{x}$ by feeding the most recent state $\boldsymbol{z}_i$ and index i into the DNN model and then computes the mean vector of the conditional Gaussian distribution for $\boldsymbol{z}_{i-1}$. We propose to calculate a more accurate estimate for $\boldsymbol{x}$ by performing extrapolation on the two estimates of $\boldsymbol{x}$ that are obtained by feeding $(\boldsymbol{z}_{i+1}, i+1)$ and $(\boldsymbol{z}_i, i)$ into the DNN model. The extrapolation can be easily integrated into the backward process of existing DPMs by introducing an additional connection over two consecutive timesteps, and fine-tuning is not required. Extensive experiments showed that plugging in the additional connection into DDPM, DDIM, DEIS, S-PNDM, and high-order DPM-Solvers leads to a significant performance gain in terms of Fréchet inception distance (FID) score. Our implementation is available at https://github.com/guoqiang-zhang-x/LA-DPM.

1. Introduction

As one type of generative model, diffusion probabilistic models (DPMs) have made significant progress in recent years. The pioneering work [17] applied non-equilibrium statistical physics to estimating probabilistic data distributions. In doing so, a Markov forward diffusion process is constructed by systematically inserting additive noise in the data until essentially only noise remains. The data distribution is then gradually restored by a reverse diffusion process starting from a simple parametric distribution. The main advantage of DPMs over classic tractable models (e.g., HMMs, GMMs, see [5]) is that they can accurately model both the high and low likelihood regions of the data distribution via the progressive estimation of noise-perturbed data distributions. In comparison to generative adversarial networks (GANs) [1, 8, 9], DPMs exhibit more stable training dynamics by avoiding adversarial learning.

The work [10] focuses on a particular type of DPM, namely a denoising diffusion probabilistic model (DDPM), and shows that after a sufficient number of timesteps (or equivalently iterations) in the backward process, DDPM can achieve state-of-the-art performance in image generation tasks by the proper design of a weighted variational bound (VB). In addition, by inspection of the weighted VB, it is found that the method *score matching with Langevin dynamics* (SMLD) [19, 20] can also be viewed as a DPM. The recent work [21] interprets DDPM and SMLD as search of approximate solutions to stochastic differential equations. See also [15] and [7] for improved DPMs that lead to better log-likelihood scores and sampling qualities, respectively.

One inconvenience of a standard DPM is that the associated deep neural network (DNN) needs to run for a sufficient number of timesteps to achieve high sampling quality while the generative model of a GAN only needs to run once. This has led to an increasing research focus on reducing the number of reverse timesteps in DPMs while retaining a satisfactory sampling quality (see [22] for a detailed overview). Song et al. proposed the so-called denoising diffusion implicit model (DDIM) [18] as an extension of DDPM from a non-Markov forward process point of view. The work [11] proposed to learn a denoising schedule in the reverse process by explicitly modeling the signal-to-noise ratio in the image generation task. [6] and [12] considered effective audio generation by proposing different noise scheduling schemes in DPMs. Differently from the above methods, the recent works [4] and [3] proposed to estimate the optimal variance of the backward conditional Gaussian distribution to improve sampling qualities for both small and large numbers of timesteps.

Another approach for improving the sampling quality of DPMs with a limited computational budget is to exploit high-order methods for solving the backward ordinary differential equations (ODEs) (see [21]). The authors of [13] proposed pseudo numerical methods for diffusion models (PNDM), of which high-order polynomials of the estimated Gaussian noises $\{\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+j}, i+j)|r \geq j \geq 0\}$ are introduced to better estimate the latent variable $\boldsymbol{z}_{i-1}$ at iteration i, where $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ represents a pre-trained neural network model for predicting the Gaussian noises. The work [23] further extends [13] by refining the coefficients of the high-order polynomials of the estimated Gaussian noises, and proposes the diffusion exponential integrator sampler (DEIS). Recently, the authors of [14] considered solving the ODEs of a diffusion model differently from [23]. In particular, a high-order Taylor expansion of the estimated Gaussian noises was employed to better approximate the continuous solutions of the ODEs, where the developed sampling methods are referred to as DPM-Solvers.

We note that the computation of $\boldsymbol{z}_{i-1}$ at timestep i in the backward process of existing DPMs (including the high-order ODE solvers) can always be reformulated in terms of an estimate $\hat{\boldsymbol{x}}$ for the original data sample $\boldsymbol{x}$ in combination with other terms. In principle, as the timestep i decreases, the estimate $\hat{\boldsymbol{x}}$ would become increasingly accurate. In this paper, we aim to improve the estimation accuracy of $\boldsymbol{x}$ at each timestep i in computation of the mean vector for the latent variable $\boldsymbol{z}_{i-1}$. To do so, we propose to make an extrapolation from the two most recent estimates of $\boldsymbol{x}$ obtained at timestep i and $i+1$. The extrapolation allows the backward process to look ahead towards a noisy direction targeting $\boldsymbol{x}$, thus improving the estimation accuracy. The extrapolation can be realized by simply introducing additional connections between two consecutive timesteps, which can be easily plugged into existing DPMs with negligible computational overhead. We refer to the improved diffusion models as Lookahead-DPMs (LA-DPMs).

We conducted an extensive evaluation by plugging in the additional connection into the backward process of DDPM, DDIM, DEIS, S-PNDM, and DPM-Solver. Interestingly, it is found that the performance gain of LA-DPMs is more significant for a small number of timesteps. This makes it attractive for practical applications as it is computationally preferable to run the backward process in a limited number of timesteps.

2. Background of Markov Diffusion Models

We revisit the standard Markov DPMs being studied in [11]. In the following, we first briefly review the forward diffusion process. We then investigate its backward process. The notation in this paper is in line with that of [11].

2.1. Forward diffusion process

Suppose we have a set of observations of $\boldsymbol{x}$ that are drawn from a data distribution $q(\boldsymbol{x})$. A forward diffusion process can be defined as a sequence of increasingly noisy versions $\boldsymbol{z}_t$, $t \in [0, 1]$, of $\boldsymbol{x}$, where $\boldsymbol{z}_{t=1}$ indicates the noisiest version (we will discretize t later on). For a Gaussian-driven process, the latent variable $\boldsymbol{z}_t$ can be represented in terms of $\boldsymbol{x}$ being contaminated by a Gaussian noise as

$$\boldsymbol{z}_t = \alpha_t \boldsymbol{x} + \sigma_t \boldsymbol{\epsilon}_t, \tag{1}$$

where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \boldsymbol{I})$, and α_t and σ_t are strictly positive scalar-valued functions of t, and $\boldsymbol{I}$ is the identity matrix. To formalise the notion of $\boldsymbol{z}_t$ being increasingly noisy, $\boldsymbol{z}_t$ can be alternatively represented in terms of $\boldsymbol{z}_s$, $s < t$, as

$$\boldsymbol{z}_t = \alpha_{t|s} \boldsymbol{z}_s + \sigma_{t|s} \boldsymbol{\epsilon}_{t|s}, \tag{2}$$

where $\boldsymbol{\epsilon}_{t|s}$ is the additional Gaussian noise being added to a scaled version of $\boldsymbol{z}_s$, and $(\alpha_{t|s}, \sigma^2_{t|s})$ are given by

$$\alpha_{t|s} = \alpha_t / \alpha_s \;\text{ and }\; \sigma^2_{t|s} = \sigma^2_t - \alpha^2_{t|s}\sigma^2_s, \tag{3}$$

where the conditional variance $\sigma^2_{t|s}$ is assume to be positive, i.e., $\sigma^2_{t|s} > 0$. One major advantage of the above formulation is that it includes both the variance-preserving process with $\alpha_t = \sqrt{1 - \sigma^2_t}$ [10, 17] and variance-exploding process with $\alpha_t = 1$ [20, 21].

It is immediate that the process (1)-(3) is Markov. That is, the conditional distribution $q(\boldsymbol{z}_u|\boldsymbol{z}_t, \boldsymbol{z}_s) = q(\boldsymbol{z}_u|\boldsymbol{z}_t) = \mathcal{N}(\alpha_{u|t}\boldsymbol{z}_t, \sigma^2_{u|t}\boldsymbol{I})$, where $0 \leq s < t < u \leq 1$. Consequently, it can be shown that $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x})$, $s < t$, is Normal distributed by using Bayes rule (see Appendix A of [11]),

$$q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x}) = \mathcal{N}\left(\frac{\sigma^2_s}{\sigma^2_t}\alpha_{t|s}\boldsymbol{z}_t + \frac{\sigma^2_{t|s}}{\sigma^2_t}\alpha_s \boldsymbol{x}, \frac{\sigma^2_s \sigma^2_{t|s}}{\sigma^2_t}\boldsymbol{I}\right). \tag{4}$$

As will be discussed later on, the backward process heavily relies on the relation between $(\boldsymbol{z}_t, \boldsymbol{x})$ and $\boldsymbol{z}_{\boldsymbol{s}}$ in the formulation (4).

As one example, the above process includes the forward process of a DDPM as a special case. One can simply discretize $t \in [0, 1]$ into N uniform timesteps, i.e., $t_i = i/N$, and let $\{\alpha_{t_i} = \sqrt{1 - \sigma^2_{t_i}}|N \geq i \geq 0\}$ be a strictly decreasing sequence.

2.2. Backward diffusion process

In general, a backward process is designed to reverse the forward process introduced earlier for the purpose of approximating the data distribution $q(\boldsymbol{x})$. Without loss of generality, we denote a discrete backward process as

$$p(\boldsymbol{x}, \boldsymbol{z}_{0:N}) = p(\boldsymbol{z}_N)\prod_{i=1}^{N} p(\boldsymbol{z}_{i-1}|\boldsymbol{z}_{i:N})p(\boldsymbol{x}|\boldsymbol{z}_{0:N}), \tag{5}$$

where the support region $[0, 1]$ for t is discretized into N uniform timesteps, i.e., $t_i = i/N$, and t_i is replaced by i to simplify notation. The objective is to find a specific form of the backward process such that its marginal distribution with regard to $\boldsymbol{x}$ approaches $q(\boldsymbol{x})$:

$$q(\boldsymbol{x}) \approx \int p(\boldsymbol{x}, \boldsymbol{z}_{0:N}) d\boldsymbol{z}_0 \dots d\boldsymbol{z}_N. \tag{6}$$

To facilitate computation, DDPM makes the following approximation to the backward process of (4):

$$\begin{aligned} & p(\boldsymbol{z}_{i-1}|\boldsymbol{z}_{i:N}) \\ & = p(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i) \qquad (7) \\ & \approx q(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, \boldsymbol{x} = \hat{\boldsymbol{x}}(\boldsymbol{z}_i, i)) \\ & = \mathcal{N}\left(\underbrace{\frac{\sigma_{i-1}^2}{\sigma_i^2}\alpha_{i|i-1}\boldsymbol{z}_i + \frac{\sigma_{i|i-1}^2}{\sigma_i^2}\alpha_{i-1}\hat{\boldsymbol{x}}(\boldsymbol{z}_i, i)}_{\boldsymbol{\mu}(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, i)}, \underbrace{\frac{\sigma_{i-1}^2\sigma_{i|i-1}^2}{\sigma_i^2}}_{\varphi_i}\boldsymbol{I} \right), \end{aligned} \tag{8}$$

where $\alpha_{i|i-1}$ and $\sigma_{i|i-1}^2$ follow from (3) with $(t, s) = (i/N, (i-1)/N)$, and $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, i)$ denotes the predicted sample for $\boldsymbol{x}$ by using $\boldsymbol{z}_i$ and timestep i. The marginal distribution of $\boldsymbol{z}_N$ is approximated to be a spherical Gaussian, i.e., $p(\boldsymbol{z}_N) \approx p(0, \beta\boldsymbol{I})$, where $\beta = 1$ for variance-preserving DPMs. A nice property of (8) is that the conditional distribution is Gaussian. As a result, the computation in the backward process only needs to focus on a sequence of means $\boldsymbol{\mu}(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, i)$ and variances φ_i from $i = N$ to $i = 0$.

Next, we briefly discuss the computation for $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, i)$ in [10], which is followed by recent, more advanced, DPM models such as DDIM [18] and DPM-Solver [14]. In [10], a DNN model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ is designed to make a direct prediction of the added Gaussian noise $\boldsymbol{\epsilon}_t$ to $\boldsymbol{x}$ in a latent variable $\boldsymbol{z}_t$ of (1). In particular, the model is trained to minimize a summation of expected squared errors:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \mathbf{E}_{\boldsymbol{x}, \boldsymbol{\epsilon}_i}\left[\|\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\alpha_i\boldsymbol{x} + \sqrt{1-\alpha_i^2}\boldsymbol{\epsilon}_i, i) - \boldsymbol{\epsilon}_i\|^2 \right]. \tag{9}$$

As $\boldsymbol{\epsilon}$ and $\boldsymbol{z}$ share the same dimensionality, the architecture of the model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ is often selected to be a variant of UNet [16]. In the sampling process, an approximation of $\boldsymbol{x}$ can be easily obtained in terms of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)$ by following (1) under the condition $\sigma_i = \sqrt{1-\alpha_i^2}$:

$$\begin{aligned} \hat{\boldsymbol{x}}(\boldsymbol{z}_i, i) &= \hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) \\ &= \boldsymbol{z}_i/\alpha_i - \sqrt{1-\alpha_i^2}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)/\alpha_i. \end{aligned} \tag{10}$$

The expression (10) for $\boldsymbol{x}$ can then be plugged into $\boldsymbol{\mu}(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, i)$ in (8).

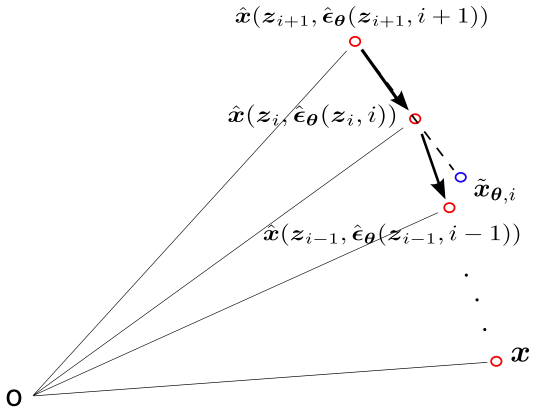
$\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1))$
$\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$
$\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}$
$\hat{\boldsymbol{x}}(\boldsymbol{z}_{i-1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i-1}, i-1))$
$\boldsymbol{x}$
o

Figure 1. Illustration of the extrapolation operation for refining the mean estimation in the backward process of DDPM. At timestep i, the estimate $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}$ is computed by extrapolating from the two traditional estimates $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ and $\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1))$. $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}$ is taken to replace $\boldsymbol{x}$ in the conditional Gaussian distribution $q(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, \boldsymbol{x})$.

In practice, different approximations have been made to the variance φ_i of the conditional Gaussian distribution in (8). For instance, it has been found in [10] that two different setups for φ_i lead to similar sampling performance. As mentioned earlier, the two recent works [4] and [3] propose to train DNNs to optimally estimate the time-dependent variance in (8) under different conditions, which is found to produce considerably high sampling quality.

3. Basic Lookahead Diffusion Models

In this section, we first consider the correlations carried in $\{\hat{\boldsymbol{x}}(\boldsymbol{z}_j, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j)) | N \geq j \geq 0\}$ over consecutive timesteps. Then, we propose to refine the estimate for $\boldsymbol{x}$ by performing extrapolation in the backward process of DDPM, DDIM, and DPM-Solvers, respectively. We refer to the improved generative models as LA-DPMs. Finally, we conduct an analysis to study the strengths of the extrapolations.

3.1. Inspection of the estimates for $\boldsymbol{x}$

From the earlier presentation, it is clear that the latent variables $\{\boldsymbol{z}_i | N \geq i \geq 0\}$ form a sequence of progressively noisier versions of the data sample $\boldsymbol{x}$ as index i increases from 0 to N. It is therefore reasonable to assume that as the index i decreases from N until 0, the estimates $\{\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) | N \geq i \geq 0\}$ in (10) are increasingly accurate. As shown in Fig. 1, as i decreases, the estimate $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ becomes increasingly close to $\boldsymbol{x}$. As the Gaussian noise $\boldsymbol{\epsilon}_i$ in $\boldsymbol{z}_i$ is a random variable, the estimate $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)$ should also be treated as following a certain distribution. If the model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ is well trained, the variances of the estimates should be upper-bounded. By following the above

Algorithm 1 Sampling of an LA-DDPM

```
Input: $\boldsymbol{z}_N$ and $\hat{\boldsymbol{x}}(\boldsymbol{z}_{N+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{n+1}, N+1)) = 0$, $\lambda_N = 0$
for $i = N, \ldots, 1$ do
    Compute $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$
    $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i) = (1+\lambda_i)\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$
        $-\lambda_i \hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1))$
    $\boldsymbol{\mu}(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, i, \boldsymbol{z}_{i+1}, i+1) = \frac{\sigma_{i-1}^2}{\sigma_i^2}\alpha_{i|i-1}\boldsymbol{z}_i$
        $+\frac{\sigma_{i|i-1}^2}{\sigma_i^2}\alpha_{i-1}\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)$
    $\boldsymbol{z}_{i-1} = \boldsymbol{\mu}(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, i, \boldsymbol{z}_{i+1}, i+1) + \varphi_i \boldsymbol{\epsilon}$
end for
output: $\hat{\boldsymbol{x}}(\boldsymbol{z}_0, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_0, 0))$
```

guidelines, we make an assumption to the estimates for $\boldsymbol{x}$ below. We will use the assumption later on to investigate the strengths of the extrapolation introduced in LA-DPMs.

Assumption 1 *The estimates $\{\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))|N \geq i \geq 0\}$ are assumed to be represented in terms of $\boldsymbol{x}$ as*

$$\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j) = \gamma_j \boldsymbol{x} + \phi_i \boldsymbol{\epsilon}_{b,j}, \tag{11}$$

where $\phi_i < M$, and for simplicity, the residual noise $\boldsymbol{\epsilon}_{b,j}$ is assumed to follow a spherical Gaussian distribution, i.e., $\boldsymbol{\epsilon}_{b,j} \sim \mathcal{N}(0, \boldsymbol{I})$, and for $0 \leq j < k \leq N$, we have

$$1 > \gamma_j > \gamma_k \geq 0, \quad 0 \leq \varphi_j < \varphi_k. \tag{12}$$

Furthermore, the estimate $\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1))$ can be represented in terms of $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ as

$$\begin{aligned}&\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1)) \\ &= \gamma_{i+1|i}\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) + \phi_{i+1|i}\boldsymbol{\epsilon}_{b,i+1|i},\end{aligned} \tag{13}$$

where $\gamma_{i+1|i} = \gamma_{i+1}/\gamma_i \in (0,1)$, $\phi_{i+1|i}^2 = \phi_{i+1}^2 - \gamma_{i+1|i}^2\phi_i^2 > 0$, and $\boldsymbol{\epsilon}_{b,i+1|i} \sim \mathcal{N}(0, \boldsymbol{I})$. That is, the estimates $\{\hat{\boldsymbol{x}}(\boldsymbol{z}_j, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j))|N \geq j \geq 0\}$ form a Markov process.

Next, we briefly consider the two consecutive estimates $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ and $\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1))$. It is clear from (11)-(12) that as j decreases from $i+1$ to i, the estimate $\hat{\boldsymbol{x}}(\boldsymbol{z}_j, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j))$ becomes more accurate. By applying (11)-(13), the difference of the two estimates can be represented as

$$\begin{aligned}\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i} &= \hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) - \hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1)) \\ &= (1-\gamma_{i+1|i})\gamma_i\boldsymbol{x} + (1-\gamma_{i+1|i})\phi_i\boldsymbol{\epsilon}_{b,i} \\ &\quad - \phi_{i+1|i}\boldsymbol{\epsilon}_{b,i+1|i},\end{aligned} \tag{14}$$

where $\boldsymbol{\epsilon}_{b,i}$ and $\boldsymbol{\epsilon}_{b,i+1|i}$ are independent variables. Because of the term $(1-\gamma_{i+1|i})\gamma_i\boldsymbol{x}$ in (14), the difference $\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i}$ provides additional information about $\boldsymbol{x}$ in comparison to $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$. As demonstrated in Fig. 1, $\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i}$ can be viewed as a noisy vector towards $\boldsymbol{x}$ at timestep i. From a high level point of view, it provides additional gradient-descent information that could be exploited to refine the estimate for $\boldsymbol{x}$ at timestep i.

3.2. LA-DDPM

In this subsection, we incorporate the additional gradient information $\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i}$ of (14) into the backward update expression for $\boldsymbol{z}_{i-1}$ in the DDPM model. In particular, (8) is modified to be

$$\begin{aligned}&p(\boldsymbol{z}_{i-1}|\boldsymbol{z}_{i:N}) \\ &\approx q(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, \boldsymbol{x} = \tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)) \\ &= \mathcal{N}\left(\underbrace{\frac{\sigma_{i-1}^2}{\sigma_i^2}\alpha_{i|i-1}\boldsymbol{z}_i + \frac{\sigma_{i|i-1}^2}{\sigma_i^2}\alpha_{i-1}\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)}_{\boldsymbol{\mu}(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, i, \boldsymbol{z}_{i+1}, i+1)}, \underbrace{\frac{\sigma_{i-1}^2\sigma_{i|i-1}^2}{\sigma_i^2}}_{\varphi_i}\boldsymbol{I}\right),\end{aligned} \tag{15}$$

where $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)$ is computed in the form of

$$\begin{aligned}&\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i) \\ &= \hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) + \lambda_i\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i} \\ &= (1+\lambda_i)\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) - \lambda_i\hat{\boldsymbol{x}}(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1)),\end{aligned} \tag{16}$$

where $\lambda_i \geq 0$ denotes the stepsize for incorporating the difference $\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i}$, and $\lambda_i = 0$ reduces to the original update procedure for DDPM. It is noted from (16) that the new estimate $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)$ is obtained by conducting extrapolation over the two consecutive vectors $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ and $\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+1}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+1}, i+1))$. As demonstrated in Fig. 1, the new estimate $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)$ is closer to $\boldsymbol{x}$. Conceptually speaking, the extrapolation operation allows the backward process to look ahead toward a noisy direction targeting $\boldsymbol{x}$. This improves the estimation accuracy for $\boldsymbol{x}$ when the parameter λ_i is properly selected. See Alg. 1 for a summary of the sampling procedure of an LA-DDPM.

3.3. LA-DDIM

It is known that DDIM extends DDPM by considering a non-Markov forward process $q(\boldsymbol{z}_N|\boldsymbol{x})\prod_{i=1}^N q(\boldsymbol{z}_{i-1}|\boldsymbol{z}_i, \boldsymbol{x})$ while keeping the marginal distribution $q(\boldsymbol{z}_i|\boldsymbol{x})$ the same as that of DDPM. Consequently, in the backward process of DDIM, the latent variable $\boldsymbol{z}_{i-1}$ can be estimated with higher accuracy from $\boldsymbol{z}_i$ than DDPM. Specially, $\boldsymbol{z}_{i-1}$ in DDIM is computed in the form of

$$\boldsymbol{z}_{i-1} = \alpha_{i-1}\underbrace{\left(\frac{\boldsymbol{z}_i - \sigma_i\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)}{\alpha_i}\right)}_{\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))} + \sigma_{i-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i). \tag{17}$$

It is clear from (17) that $\boldsymbol{z}_{i-1}$ can be viewed as a linear combination of $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ and $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)$.

To obtain the update expression for LA-DDIM, we simply modify (17) by replacing $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ with $\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i)$ in (16), which can be represented as

$$\boldsymbol{z}_{i-1} = \alpha_{i-1}\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i) + \sigma_{i-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i). \tag{18}$$

3.4. LA-DPM-Solver

In this subsection, we first briefly explain how DPM-Solver of [14] is motivated. We then consider incorporating the difference vector $\Delta\hat{\boldsymbol{x}}_{\boldsymbol{\theta},i}$ into the update expressions of DPM-Solver.

In [14], the authors attempted to solve the following ODE derived from the forward process (1):

$$\frac{d\boldsymbol{z}_t}{dt} = f(t)\boldsymbol{z}_t + \frac{g^2(t)}{2\sigma_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t) \quad \boldsymbol{z}_{T=1} \sim \mathcal{N}(0, \tilde{\sigma}\boldsymbol{I}), \tag{19}$$

where $f(t) = \frac{d\log\alpha_t}{dt}$ and $g^2(t) = \frac{d\sigma_t^2}{dt} - 2\frac{d\log\alpha_t}{dt}\sigma_t^2$. By applying the “variation of constants” formula [2] to (19), the exact solution $\boldsymbol{z}_{i-1}$ given $\boldsymbol{z}_i$ can be represented as

$$\begin{aligned}\boldsymbol{z}_{i-1} = e^{\int_{t_i}^{t_{i-1}} f(\tau)d\tau}\boldsymbol{z}_i \\ + \int_{t_i}^{t_{i-1}} \left(e^{\int_{t_i}^{t_{i-1}} f(r)dr} \frac{g^2(\tau)}{2\sigma_\tau}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_\tau, \tau)\right) d\tau,\end{aligned} \tag{20}$$

which involves an integration over the predicted Gaussion noise vector $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_\tau, \tau)$.

The authors of [14] then propose discrete high-order solutions to approximate the integration in (20). Before presenting the solutions, we first introduce two functions. Let $\lambda_t = \log(\alpha_t/\sigma_t)$ denote the logarithm of the SNR-ratio α_t/σ_t. λ_t is a monotonic decreasing function over time t. Therefore, one can also define an inverse function from λ to t, denoted as $t_\lambda(\cdot) : \mathbb{R} \to \mathbb{R}$. Upon introducing the above functions, the update expression for the 2nd order discrete solution (referred to as DPM-Solver-2 in [14]) takes the form of

$$t_{i-\frac{1}{2}} = t_\lambda(\frac{\lambda_{t_{i-1}} + \lambda_{t_i}}{2}), \tag{21}$$

$$\boldsymbol{z}_{i-\frac{1}{2}} = \frac{\alpha_{i-\frac{1}{2}}}{\alpha_i}\boldsymbol{z}_i - \sigma_{i-\frac{1}{2}}(e^{\frac{h_i}{2}} - 1)\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i), \tag{22}$$

$$\boldsymbol{z}_{i-1} = \frac{\alpha_{i-1}}{\alpha_i}\boldsymbol{z}_i - \sigma_{i-1}(e^{h_i} - 1)\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}\left(\boldsymbol{z}_{i-\frac{1}{2}}, i - \frac{1}{2}\right), \tag{23}$$

where $h_i = \lambda_{t_{i-1}} - \lambda_{t_i}$. The subscript $i - \frac{1}{2}$ indicates that the time $t_{i-\frac{1}{2}}$ is in between t_{i-1} and t_i. The latent variable $\boldsymbol{z}_{i-\frac{1}{2}}$ at time $t_{i-\frac{1}{2}}$ is firstly estimated in preparation for computing $\boldsymbol{z}_{i-1}$. By using the property that $\lambda_{t_{i-\frac{1}{2}}} = (\lambda_{t_{i-1}} + \lambda_{t_i})/2$, the update expression (22) for $\boldsymbol{z}_{i-\frac{1}{2}}$ can be simplified to be

$$\boldsymbol{z}_{i-\frac{1}{2}} = \alpha_{i-\frac{1}{2}}\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) + \sigma_{i-\frac{1}{2}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i), \tag{24}$$

which in fact coincides with the update expression (17) for DDIM.

We are now in a position to design LA-DPM-Solver-2. Similarly to LA-DDIM, we modify (24) by replacing $\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$ with an extrapolated term:

$$\begin{aligned}\boldsymbol{z}_{i-\frac{1}{2}} = &\alpha_{i-\frac{1}{2}}\Big[(1+\lambda_i)\hat{\boldsymbol{x}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)) \\ &-\lambda_i\hat{\boldsymbol{x}}(\boldsymbol{z}_{i+\frac{1}{2}}, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+\frac{1}{2}}, i+\frac{1}{2}))\Big] + \sigma_{i-\frac{1}{2}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i).\end{aligned} \tag{25}$$

Once $\boldsymbol{z}_{i-\frac{1}{2}}$ is computed, The computation for $\boldsymbol{z}_{i-1}$ follows directly from (23).

Remark 1 *In [14], the authors further propose DPM-Solver-3, the 3rd order discrete solution for approximating (20). Correspondingly, we propose LA-DPM-Solver-3. See Appendix C.2 for the update expressions.*

3.5. Analysis of estimation accuracy for $\boldsymbol{x}$

In this subsection, we derive the optimal setup λ_i^* for λ_i under Assumption 1. Our objective is to find out under what condition, λ_i^* is positive, indicating that the extrapolation operation improves the estimation accuracy for $\boldsymbol{x}$. To do so, we minimize the expected squared error $\|\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i) - \boldsymbol{x}\|)^2$ conditioned on $\boldsymbol{x}$ in terms of λ_i:

$$\lambda_i^* = \arg\min_{\lambda_i} \mathbb{E}[\|\tilde{\boldsymbol{x}}_{\boldsymbol{\theta},i}(\lambda_i) - \boldsymbol{x}\|^2|\boldsymbol{x}]. \tag{26}$$

By using (14)-(16) and the property that $\{\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j))\}$ follows a Gaussian distribution as stated in Assumption 1, (26) can be simplified to be

$$\begin{aligned}\lambda_i^* = &\arg\min_{\lambda_i}((1 + \lambda_i - \lambda_i\gamma_{i+1|i})\gamma_i - 1)^2\|\boldsymbol{x}\|^2 \\ &+ (1 + \lambda_i - \lambda_i\gamma_{i+1|i})^2\phi_i^2 + \lambda_i^2\phi_{i+1|i}^2.\end{aligned} \tag{27}$$

It is clear that the RHS of (27) is a quadratic function of λ_i. The optimal solution λ_i^* can be derived easily and can be expressed as

$$\lambda_i^* = \frac{(1 - \gamma_{i+1|i})(\gamma_i(1-\gamma_i)\|\boldsymbol{x}\|^2 - \phi_i^2)}{(1-\gamma_{i+1|i})^2\gamma_i^2\|\boldsymbol{x}\|^2 + (1-\gamma_{i+1|i})^2\phi_i^2 + \phi_{i+1|i}^2}. \tag{28}$$

With (28), one can obtain the condition that leads to $\lambda_i^* > 0$. We present the results in a proposition below:

Proposition 1 *Suppose the conditions for $\{\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))\}$ in Assumption 1 hold. The optimal setup λ_i^* is positive (i.e., $\lambda_i^* > 0$) when the noise amplitude ϕ_i satisfies the following inequality*

$$\phi_i^2 < \gamma_i(1-\gamma_i)\|\boldsymbol{x}\|^2. \tag{29}$$

The condition (29) indicates that if the outputs of the DNN model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ are not too noisy, namely $\{\phi_i\}$ are small in comparison to $\|\boldsymbol{x}\|^2$, then it is desirable to apply the extrapolation operation for the purpose of refining the estimate of $\boldsymbol{x}$. In other words, if the model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ is well designed and trained, one should introduce the additional connections in the sampling procedure of a DPM model. It is noted that the analysis above is based on approximations of Markov Gaussian distributions made in Assumption 1. In practice, it is suggested to find the optimal values of $\{\lambda_j^*\}_{j=0}^N$ by training an additional DNN instead of relying on the expression (28). As will be demonstrated later on, it is found empirically that a constant λ value in LA-DPMs leads to significant performance gain over traditional DPMs even though it may not be optimal.

4. Advanced Lookahead Diffusion Models

In this section, we explain how to introduce additional extrapolation into DEIS and S-PNDM. These methods already employ high-order polynomials of the historical estimated Gaussian noises $\{\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+j}, i+j)|r \geq j \geq 0\}$ in the estimation of the latent variable $\boldsymbol{z}_{i-1}$ at iteration i.

For simplicity, we first consider extending DEIS to obtain LA-DEIS. By following [23], the update expression for $\boldsymbol{z}_{i-1}$ in the backward process takes the form

$$\boldsymbol{z}_{i-1} = \frac{\alpha_{i-1}}{\alpha_i}\boldsymbol{z}_i + \sum_{j=0}^{r} c_{ij}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+j}, i+j), \tag{30}$$

where the $\{c_{ij}\}_{j=0}^r$ are pre-computed hyper-parameters for the purpose of more accurately approximating an integration of the ODE (19) for (1)-(3).

Next, we reformulate (30) into an expression similar to (17) for DDIM:

$$\boldsymbol{z}_{i-1} = \alpha_{i-1}\underbrace{\left(\frac{\boldsymbol{z}_i - \sigma_i\tilde{\boldsymbol{\epsilon}}_{[i:i+r]}}{\alpha_i}\right)}_{\ddot{\boldsymbol{x}}_{[i:i+r]}} + \sigma_{i-1}\tilde{\boldsymbol{\epsilon}}_{[i:i+r]}, \tag{31}$$

where $\tilde{\boldsymbol{\epsilon}}_{[i:i+r]}$ is given by

$$\tilde{\boldsymbol{\epsilon}}_{[i:i+r]} = \sum_{j=0}^{r} c_{ij}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i+j}, i+j) \Big/ \left(\sigma_{i-1} - \frac{\alpha_{i-1}\sigma_i}{\alpha_i}\right). \tag{32}$$

The quantity $\tilde{\boldsymbol{\epsilon}}_{[i:i+r]}$ can be viewed as a more accurate estimate of the Gaussian noise than $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)$ in DDIM. With $\tilde{\boldsymbol{\epsilon}}_{[i:i+r]}$, we can compute an estimate $\ddot{\boldsymbol{x}}_{[i:i+r]}$ of the original data sample $\boldsymbol{x}$.

Upon obtaining (31)-(32), we can easily design LA-DEIS by introducing an additional extrapolation into (31), which can be represented by

$$\begin{aligned}\boldsymbol{z}_{i-1} = &\alpha_{i-1}[(1+\lambda_i)\ddot{\boldsymbol{x}}_{[i:i+r]} - \lambda_i\ddot{\boldsymbol{x}}_{[i+1:i+r+1]}] \\ &+ \sigma_{i-1}\tilde{\boldsymbol{\epsilon}}_{[i:i+r]},\end{aligned} \tag{33}$$

where the estimate $\ddot{\boldsymbol{x}}_{[i+1:i+r+1]}$ is from the previous timestep $i+1$. Our intention with the additional extrapolation is to provide a better estimate for $\boldsymbol{x}$ at timestep i. In principle, the estimate $\ddot{\boldsymbol{x}}_{[i:i+r]}$ should be less noisy than $\ddot{\boldsymbol{x}}_{[i+1:i+r+1]}$. As a result, the difference of the two vectors would approximately point towards $\boldsymbol{x}$, thus providing additional gradient information in computing $\boldsymbol{z}_{i-1}$.

Similarly to the design of LA-DEIS, S-PNDM can also be easily extended to obtain LA-S-PNDM. See Appendix D for the details.

5. Numerical Experiments

In the 1st experiment, we used as basis variants of DDPM and DDIM in [3] that obtain the optimal covariances of the conditional Gaussian distributions in the backward process by employing additional pre-trained DNN models. We will show that the sampling quality can be significantly improved by introducing the proposed extrapolations in the above methods. We also conduct an ablation study for a particular LA-DDIM that investigates how the parameters $\{\lambda_i\}$ affect the FID scores.

In the 2nd experiment, we evaluate LA-DEIS and LA-S-PNDM and the corresponding original methods. The evaluation for LA-DPM-Solvers can be found in Appendix C.3.

5.1. Evaluation of covariance-optimised DPMs

Experimental setup: In this experiment, we evaluated the improved DPM models developed in [3], which make use of trained neural networks to estimate the optimal covariances of the conditional Gaussian distributions in the backward process. Four types of improved DPM models from [3] were tested, which are NPR-DDPM, SN-DDPM, NPR-DDIM, and SN-DDIM, respectively. The two notations "NPR" and "SN" refer to two different approaches for designing DNN models to estimate the optimal covariances under different conditions.

Similarly to [3], we conducted experiments over three datasets: CIFAR10, ImageNet64, and CelebA64. For each dataset, two pre-trained models were downloaded from the open-source link[1] provided in [3], one for the "NPR" approach and the other for the "SN" approach.

In the experiment, the strengths of the extrapolations were set to $\lambda_i = 0.1$ for all $i \in \{0, 1, \ldots, N-1\}$ in all pre-trained models. The tested timesteps for the three datasets were set to $\{10, 25, 50, 100, 200, 1000\}$. For each configuration, 50K artificial images were generated for the computation of FID score.

Performance comparison: The sampling qualities for the three datasets are summarized in Table 1. It is clear that for CIFAR10 and Celeba64, the LA-DPM models outperform

[1] https://github.com/baofff/Extended-Analytic-DPM

Table 1. Comparison of FID score for CIFAR10, CelebA64, and ImageNet64. The notation "LA" stands for "lookahead", where the associated DPM models are obtained by introducing extrapolation accordingly. **Lower** is better.

Data sets	CIFAR10						CelebA64						ImageNet64					
Timesteps	10	25	50	100	200	1000	10	25	50	100	200	1000	10	25	50	100	200	1000
NPR-DDPM	32.64	10.48	6.18	4.46	3.70	4.04	28.32	15.51	10.70	8.28	7.01	5.26	53.22	28.41	21.05	18.26	**16.75**	16.30
LA-NPR-DDPM	**25.59**	**8.48**	**5.28**	**4.07**	**3.47**	**3.90**	**25.08**	**13.92**	**9.58**	**7.43**	**6.32**	**5.01**	**48.71**	**25.42**	**20.27**	**18.16**	16.83	**16.27**
gain (%)	21.6	19.1	14.6	8.7	6.2	3.5	11.4	10.3	10.4	10.3	9.8	4.75	8.5	10.5	3.7	0.5	-0.5	0.2
SN-DDPM	23.75	6.88	4.58	3.67	3.31	3.65	20.55	11.85	7.58	5.95	4.96	4.44	51.09	27.77	20.65	18.07	**16.70**	16.30
LA-SN-DDPM	**19.75**	**5.92**	**4.31**	**3.55**	**3.24**	**3.55**	**17.43**	**10.08**	**6.41**	**5.12**	**4.41**	**4.21**	**46.13**	**24.67**	**19.83**	**17.95**	16.76	**16.28**
gain (%)	16.8	14.0	5.9	3.3	2.1	2.7	15.2	14.9	15.4	13.9	11.1	5.2	9.7	11.2	4.0	0.7	-0.4	0.1
NPR-DDIM	13.41	5.43	3.99	3.53	3.40	3.67	14.94	9.18	6.17	4.40	3.67	3.12	97.27	28.75	19.79	**17.71**	**17.15**	**17.59**
LA-NPR-DDIM	**10.74**	**4.71**	**3.64**	**3.33**	**3.29**	**3.49**	**14.25**	**8.83**	**5.67**	**3.76**	**2.95**	**2.95**	**71.98**	**25.39**	**19.47**	18.11	17.89	18.41
gain (%)	19.9	13.3	8.8	5.7	3.2	4.9	4.6	3.8	8.1	14.5	19.61	5.4	26.0	11.7	1.6	-2.3	-4.3	-4.7
SN-DDIM	12.19	4.28	3.39	3.22	4.22	3.65	10.17	5.62	3.90	3.21	2.94	2.84	91.29	27.74	19.51	**17.67**	**17.14**	**17.60**
LA-SN-DDIM	**8.48**	**3.15**	**2.93**	**2.92**	**3.08**	**3.47**	**8.05**	**4.56**	**2.93**	**2.39**	**2.19**	**2.48**	**69.11**	**25.07**	**19.32**	18.06	17.89	18.57
gain (%)	30.4	26.4	13.6	9.3	27.0	4.9	20.8	18.9	24.9	25.5	25.5	12.7	24.3	9.6	9.7	-2.2	-4.4	-5.5

the original DPM models significantly for both small and large numbers of timesteps. Roughly speaking, as the number of timesteps decreases from 1000 to 10, the performance gain of LA-DPM increases. That is, it is more preferable to introduce the extrapolation operations when sampling with a limited computational budget. This might be because for a large number of timesteps, the original DPM models are able to maximally exploit the gradient information provided by the DNN model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ and generate high quality samples accordingly. On the other hand, with a small number of timesteps, limited access to the DNN model makes it difficult for the original PDM models to acquire detailed structural information of the data sample $\boldsymbol{x}$. As a result, for a small number of timesteps, the proposed extrapolation operation plays a more important role by improving the mean estimation of the backward conditional Gaussian distributions in the sampling procedure.

Next, we consider the results obtained for ImageNet64. As shown in Table 1, the introduction of extrapolation operations leads to better performance only for a small number of steps (e.g., 10, 25, 50). When the number of steps is large, we observe slightly degraded performance. This is because ImageNet64 is a very large dataset that covers many classes of objects compared to CIFAR10 and CelebA64. As a result, the estimate $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j))$ may be noisier than the corresponding estimates over CIFAR10 and CelebA. In other words, the setups $\{\lambda_i = 0.1\}_{i=0}^{N-1}$ are not appropriate for ImageNet64. In this case, one can simply reduce the strengths (i.e., $\lambda_i \downarrow$) of the extrapolation operations.

5.2. Ablation study of LA-SN-DDIM over CIFAR10

In subsection 5.1, the strengths of the extrapolations in LA-SN-DDIM were set to $\{\lambda_i = \lambda = 0.1\}_{i=0}^{N-1}$, which led to significant performance gain for small numbers of stepsizes in comparison to SN-DDIM. We now consider how the FID scores change for different setups of $\lambda \in \{0, 0.05, 0.1, 0.15, 0.2, 0.3, 0.4\}$ over timesteps of 10 and 20, where $\{\lambda_i = \lambda\}_{i=0}^{N-1}$ for each λ value.

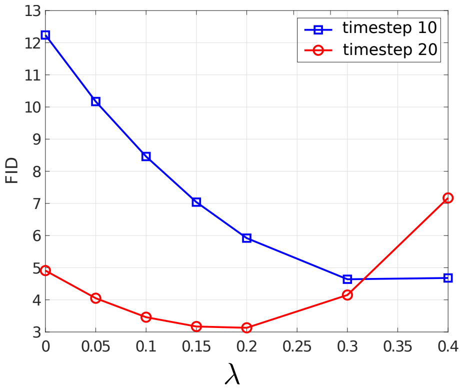

Figure 2. FID scores versus λ values for LA-SN-DDIM over CIFAR10.

Fig. 2 displays two FID curves over different λ values, one for timestep 10 and the other for timestep 20. It is clear that for timestep 10, FID score of around 4.65 can be achieved when $\lambda = 0.3$. On the other hand, for timestep 20, FID score of around 3.1 can be achieved when $\lambda = 0.2$. This suggests that the setup $\lambda = 0.1$ in the first experiment is far from optimality for a small number of timesteps. In other words, the FID scores in Table 1 can be improved significantly if the λ value is tuned for different timesteps.

5.3. Evaluation of LA-DEIS and LA-S-PNDM

Experimental setup: As noted earlier, DEIS and S-PNDM exploit high-order polynomials of the estimated Gaussian noises per timestep in the backward process for better sam-

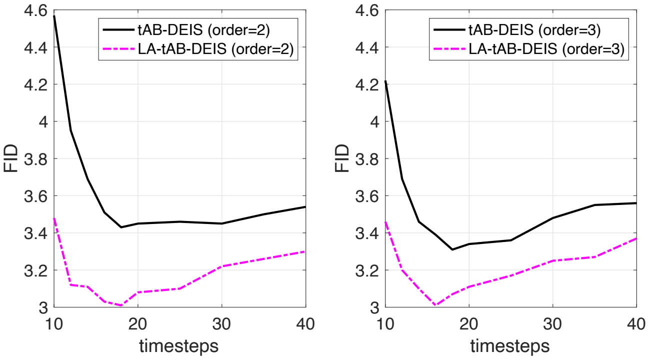

Figure 3. Performance of tAB-DEIS and LA-tAB-DEIS in terms of FID scores versus timesteps over CIFAR10. The two subplots are for polynomials of order $r = 2$ and $r = 3$ in (30), respectively. The setup $\lambda = 0.1$ was employed in LA-tAB-DEIS.

pling quality. In this experiment, we demonstrate that their sampling performance can be further improved by introducing additional extrapolation on the estimates of $\boldsymbol{x}$.

We note that the authors of [23] proposed different versions of DEIS depending on how the parameters $\{c_{ij}\}_{j=i}^{r}$ in (30) are computed. For our experiments, we used tAB-DEIS and our new method LA-tAB-DEIS. Furthermore, we also evaluated S-PNDM and LA-S-PNDM (see the update procedure of Alg. 2 in Appendix D).

The tested pre-trained models are summarized in Table 3 in Appendix E. In particular, we evaluated LA-tAB-DEIS by utilizing a pre-trained model of VPSDE for CIFAR10 in [21]. On the other hand, LA-S-PNDM was evaluated by utilizing four pre-trained models over four popular datasets in [13]. The tested timesteps for each sampling method are within the range of $[10, 40]$.

Performance comparison: Fig. 3 visualizes the FID scores versus tested timesteps for tAB-DEIS and LA-tAB-DEIS. It is clear from this figure that the introduction of additional extrapolation on the estimates of $\boldsymbol{x}$ significantly improves the sampling quality of tAB-DEIS for polynomials of both order $r = 2$ and $r = 3$. Similarly to the gain in Table 1, the performance gain in Fig. 3 is relatively large for small timesteps, which is desirable for practical applications.

The performance of S-PNDM and LA-S-PNDM is summarized in the four subplots of Fig. 4, one subplot per dataset. It is seen that LA-S-PNDM outperforms S-PNDM consistently over different timesteps and across different datasets, which is consistent with the results of Table 1 in the 1st experiment. It can also be seen from the figure that the performance gain is more significant for CelebA64 and LSUN church than for CIFAR10 and LSUN bedroom. This might be because different DNN models have different fitting errors when they are being trained.

The above positive results indicate that extrapolation on the estimates of $\boldsymbol{x}$ and the high-order polynomials of the estimated Gaussian noises are compatible. In practice, one should incorporate both techniques in the sampling procedure of DPMs.

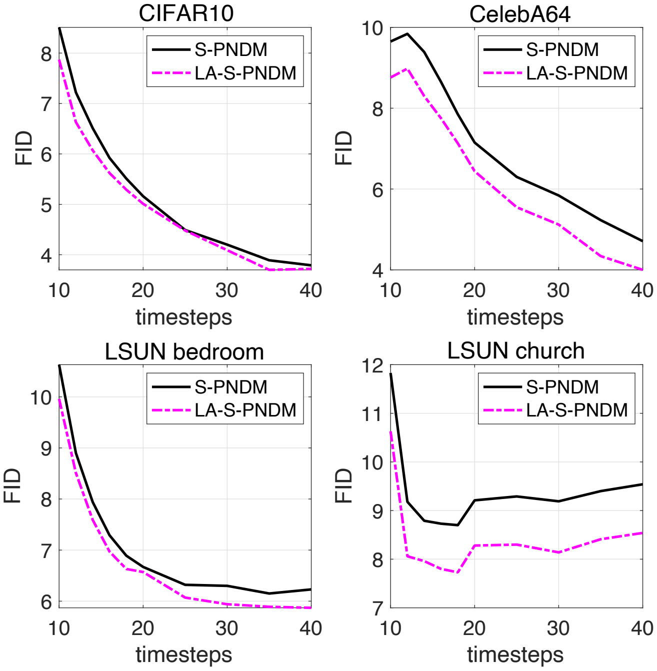

Figure 4. Performance of S-PNDM and LA-S-PNDM over 4 different datasets. The parameter λ in LA-S-PNDM was set to $\lambda = 0.1$ for {CIFAR10, CelebA64, LSUN church} and $\lambda = 0.05$ for LSUN bedroom.

Remark 2 *Due to limited space, we put the experimental results for LA-DPM-Solver-2 and LA-DPM-Solver-3 in Appendix C.3.*

6. Conclusions

In this paper, we proposed a simple approach for improving the estimation accuracy of the mean vectors of a sequence of conditional Gaussian distributions in the backward process of a DPM. A typical DPM model (even including high-order ODE solvers like DEIS and PNDM) first makes a prediction $\hat{\boldsymbol{x}}$ of the data sample $\boldsymbol{x}$ at each timestep i, and then uses it in the computation of the mean vector for $\boldsymbol{z}_{i-1}$. We propose to perform extrapolation on the two most recent estimates of $\boldsymbol{x}$ obtained at times i and $i+1$. In principle, the difference vector of the two estimates approximately points towards $\boldsymbol{x}$, thus providing certain type of gradient information. The extrapolation makes use of the gradient information to obtain a more accurate estimation of $\boldsymbol{x}$, thus improving the estimation accuracy for $\boldsymbol{z}_{i-1}$.

Extensive experiments showed that the extrapolation operation improves the sampling qualities of variants of DDPM and DDIM, DEIS, S-PNDM, and high-order DPM solvers. It was found that the performance gain is generally more significant for a small number of timesteps. This makes the new technique particularly attractive for settings with limited computational resources.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. arXiv:1701.07875 [stat.ML], 2017. 1

[2] K. Atkinson, W. Han, and D. E. Stewart. Numerical solution of ordinary differential equations. *John Wiley Sons*, 108, 2011. 5

[3] F. Bao, C. Li, J. Sun, J. Zhu, and B. Zhang. Estimating the Optimal Covariance with Imperfect Mean in Diffusion Probabilistic Models. In *ICML*, 2022. 1, 3, 6

[4] F. Bao, Chongxuan Li, J. Zhu, and B. Zhang. Analytic-DPM: an Analytic Estimate of the Optimal Reverse Variance in Diffusion Probabilistic Models. In *ICLR*, 2022. 1, 3

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. 1

[6] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan. WaveGrad: Estimating Gradients for Waveform Generation. arXiv:2009.00713, September 2020. 1

[7] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. arXiv:2105.05233 [cs.LG], 2021. 1

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Nets. In *Proceedings of the International Conference on Neural Information Processing Systems*, pages 2672–2680, 2014. 1

[9] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017. 1

[10] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 1, 2, 3

[11] D. P. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. arXiv: preprint arXiv:2107.00630, 2021. 1, 2

[12] M. W. Y. Lam, J. Wang, D. Su, and D. Yu. BDDM: Bilateral Denoising Diffusion Models for Fast and High-Quality Speech Synthesis. In *ICLR*, 2022. 1

[13] L. Liu, Yi Ren, Z. Lin, and Z. Zhao. Pseudo Numerical Methods for Diffusion Models on Manifolds. In *ICLR*, 2022. 2, 8, 14

[14] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Sampling in Around 10 Steps. In *NeurIPS*, 2022. 2, 3, 5, 11, 12, 13

[15] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. arXiv preprint arXiv:2102.09672, 2021. 1

[16] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597 [cs.CV], 2015. 3

[17] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. ICML, 2015. 1, 2

[18] J. Song, C. Meng, and S. Ermon. Denoising Diffusion Implicit Models. In *ICLR*, 2021. 1, 3

[19] Y. Song, C. Durkan, I. Murray, and S. Ermon. Maximum likelihood training of score-based diffusion models. In *Advances in neural information processing systems (NeurIPS)*, 2021. 1

[20] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in neural information processing systems (NeurIPS)*, page 11895–11907, 2019. 1, 2

[21] Y. Song, J. S.-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-Based Generative Modeling Through Stochastic Differential Equations. In *ICLR*, 2021. 1, 2, 8

[22] L. Yang, Z. Zhang, S. Hong, R. Xu, Y., Y. Shao, W. Zhang, M.-H. Yang, and B. Cui. Diffusion models: A comprehensive survey of methods and applications. arXiv preprint arXiv:2102.09672, 2021. 1

[23] Q. Zhang and Y. Chenu. Fast Sampling of Diffusion Models with Exponential Integrator. arXiv:2204.13902 [cs.LG], 2022. 2, 6, 8