Exact Diffusion Inversion via Bidirectional Integration Approximation

Guoqiang Zhang[1], J.P. Lewis[2], and W. Bastiaan Kleijn[3]

[1] University of Exeter, United Kingdom, g.z.zhang@exeter.ac.uk
[2] NVIDIA Research, USA, jpl@nvidia.com
[3] Victoria Univ. of Wellington, New New Zealand, bastiaan.kleijn@vuw.ac.nz

Abstract. Recently, various methods have been proposed to address the inconsistency issue of DDIM inversion to enable image editing, such as EDICT [40] and Null-text inversion [24]. However, the above methods introduce considerable computational overhead. In this paper, we propose a new technique, *bidirectional integration approximation* (BDIA), to perform exact diffusion inversion with negligible computational overhead. We consider a family of second order integration algorithms obtained by averaging forward and backward DDIM steps. The resulting approach estimates the next diffusion state as a linear combination of the estimated Gaussian noise at the current step and the previous and current diffusion states. This allows for exact backward computation of previous state given the current and next ones, leading to exact diffusion inversion. We perform a convergence analysis for BDIA-DDIM that includes the analysis for DDIM as a special case. It is demonstrated with experiments that BDIA-DDIM is effective for (round-trip) prompt-driven image editing. Our experiments further show that BDIA-DDIM produces markedly better image sampling quality than DDIM and EDICT for text-to-image generation and conventional image sampling.[4]

1 Introduction

As one type of generative models [1, 5, 10, 11, 31], diffusion probabilistic models (DPMs) have made significant progress in recent years. Following the work of [34], various learning and/or sampling strategies have been proposed to improve the performance of DPMs, which include, for example, denoising diffusion probabilistic models (DDPMs) [12], denoising diffusion implicit models (DDIMs) [35], improved DDIMs [7, 25], latent diffusion models (LDMs) [27], score matching with Langevin dynamics (SMLD) [36–38], analytic-DPMs [3, 4], optimized denoising schedules [6, 19, 20], guided diffusion strategies [17, 26], and classifier-free guided diffusion [13]. It is worth noting that DDIM can be interpreted as a first-order ordinary differential equation (ODE) solver. As an extension of DDIM, various high-order ODE solvers have been proposed, such as EDM [16], DEIS [44], PLMS and PNDM [21], DPM-Solvers++ [23], and IIA-EDM and IIA-DDIM [42].

[4] BDIA can also be applied to improve the performance of other ODE solvers in addition to DDIM. In particular, it is found that applying BDIA to the EDM sampling procedure produces consistently better performance over four pre-trained models (see Alg. 4 and Table 3 in Appendix C).

Fig. 1: Comparison of images generated by using DDIM, BDIA-DDIM and EDICT with 10 timesteps over StableDiffusion V2. EDICT needs to perform 20 NFEs, which is two times of the NFEs needed in BDIA-DDIM. See Appendix I for an explanation why EDICT does not perform well for one-way noise-to-image generation. See Table 1 for FID and CLIP evaluation, and Figs 7-8 for more image comparisons. Please also see Appendix K in the supplementary material for additional text-based and ControlNet-based image-editing results.

In recent years, image-editing via diffusion models has attracted increasing attention in both academia and industry. One important operation for editing a real image is to first perform forward process on the image to obtain the final noise representation and then perform a backward process with embedded editing to generate the desired image [28, 30]. DDIM inversion has been widely used to perform the above forward and backward processes [32]. A major issue with DDIM inversion is that the intermediate diffusion states in the forward and backward processes may be inconsistent due to the inherent approximations (see Subsection 4.1). This issue becomes significant when utilizing classifier-free guidance in text-to-image editing [32]. The newly generated images are often perceptually far away from the original ones, which is undesirable.

Recently, two methods have been proposed to address the inconsistency issue of DDIM inversion. Specifically, the work of [24] proposed a technique named *null-text inversion* to push the diffusion states of the backward process to be optimally close to those of the forward process via iterative optimization. The null-text inputs to the score neural network are treated as free variables in the optimization procedure. In [40], the authors proposed the EDICT technique to enforce exact DDIM inversion. Their basic idea is to introduce an auxiliary diffusion state and then perform alternating updates on the primal and auxiliary diffusion states, which is inspired by the flow generative framework [8, 9, 18]. One drawback of EDICT is that the number of neural function evaluations (NFEs) has to be doubled in comparison to DDIM inversion (See Subsection 4.2). Another related line of research work is DDPM inversion (see [14]).

In this paper, we propose a new technique to enforce exact DDIM inversion with negligible computational overhead, reducing the number of NFEs required in EDICT by half. Suppose we are in a position to estimate the next diffusion state $\boldsymbol{z}_{i-1}$ at timestep t_i by utilizing the two most recent states $\boldsymbol{z}_i$ and $\boldsymbol{z}_{i+1}$. With the estimated Gaussian noise $\hat{\boldsymbol{\epsilon}}(\boldsymbol{z}_i, i)$, we perform the DDIM update procedure

twice for approximating the ODE integration over the next time-slot $[t_i, t_{i-1}]$ in the forward manner and the previous time-slot $[t_i, t_{i+1}]$ in the backward manner. Note that in this paper *forward* refers to the denoising direction. The DDIM for the previous time-slot is employed to refine the integration approximation made earlier when computing $\boldsymbol{z}_i$. As a result, the expression for $\boldsymbol{z}_{i-1}$ becomes a linear combination of $(\boldsymbol{z}_{i+1}, \boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}(\boldsymbol{z}_i, i))$, and naturally facilitates exact diffusion inversion[5]. We refer to the above technique as *bidirectional integration approximation (BDIA)*. We provide a convergence analysis for BDIA-DDIM that follows an approach similar to that used for the linear multi-step methods in the literature of numerical integration [2]. Experiments demonstrate that BDIA-DDIM produces promising results on both image sampling and image editing (Figs. 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, Table 1, 2).

2 Related Work

From a high-level point of view, BDIA-DDIM can be viewed as a variant of linear multistep methods in the literature of numerical integration [2]. A typical linear multistep method employs multiple historical states and gradients in estimation of the next state when solving an ODE. One main difference between BDIA-DDIM and the linear multistep methods is that BDIA-DDIM performs bidirectional semi-linear integration approximation (see [22] for viewing the DDIM update expression as a semi-linear integration approximation). As a result, the convergence analysis for BDIA-DDIM is slightly different from that for linear multi-step methods as presented in Theorem 6.6 of [2].

BDIA-DDIM is also related to the existing work of time-reversable ODE solvers. For instance, Verlet integration is a time-reversible method for solving 2nd-order ODEs [39]. Leapfrog integration is another time-reversible method also developed for solving 2nd-order ODEs [33].

The recent work [41] proposed a new diffusion sampling method, named Restart sampling, that adds Gaussian noise in a backward manner from time to time. The main difference is that BDIA solves an ODE without adding any Gaussian noise in the sampling process whereas Restart alternates the ODE solver step and the insertion of Gaussian noise.

3 Preliminary

Forward and reverse diffusion processes: Suppose the data sample $\boldsymbol{x} \in \mathbb{R}^d$ follows a data distribution $p_{data}(\boldsymbol{x})$ with a bounded variance. A forward diffusion process progressively adds Gaussian noise to the data samples $\boldsymbol{x}$ to obtain $\boldsymbol{z}_t$ as t increases from 0 until T. The conditional distribution of $\boldsymbol{z}_t$ given $\boldsymbol{x}$ can be represented as

$$q_{t|0}(\boldsymbol{z}_t|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}_t|\alpha_t\boldsymbol{x}, \sigma_t^2\boldsymbol{I}) \quad \boldsymbol{z}_t = \alpha_t\boldsymbol{x} + \sigma_t\boldsymbol{\epsilon}, \tag{1}$$

[5] In principle, one can utilize more previous diffusion states when computing $\boldsymbol{z}_{i-1}$ to improve accuracy (see A.2 for a discussion). However, this would incur additional memory overhead, which may be undesirable.

where α_t and σ_t are assumed to be differentiable functions of t with bounded derivatives. It is assumed that the three functions α_t, σ_t, and $\lambda_t = \frac{\sigma_t}{\alpha_t}$, are non-increasing, non-decreasing, and strictly increasing (see [22]) as t increases, respectively. We use $q(\boldsymbol{z}_t; \alpha_t, \sigma_t)$ to denote the marginal distribution of $\boldsymbol{z}_t$. The samples of the distribution $q(\boldsymbol{z}_T; \alpha_T, \sigma_T)$ should be practically indistinguishable from pure Gaussian noise if $\sigma_T \gg \alpha_T$.

The reverse process of a diffusion model firstly draws a sample $\boldsymbol{z}_T$ from $\mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$, and then progressively denoises it to obtain a sequence of diffusion states $\{\boldsymbol{z}_{t_i} \sim p(\boldsymbol{z}; \alpha_{t_i}, \sigma_{t_i})\}_{i=0}^N$, where we use the notation $p(\cdot)$ to indicate that the reverse sample distribution might not be identical to $q(\cdot)$ because of practical approximations. It is expected that the final sample $\boldsymbol{z}_{t_0}$ is roughly distributed according to $p_{data}(\boldsymbol{x})$, i.e., $p_{data}(\boldsymbol{x}) \approx p(\boldsymbol{z}_{t_0}; \alpha_{t_0}, \sigma_{t_0})$ where $t_0 = 0$.

ODE formulation: In [38], Song et al. present a so-called *probability flow* ODE which shares the same marginal distributions as $\boldsymbol{z}_t$ in (1). Specifically, with the formulation (1) for a forward diffusion process, its reverse ODE form can be represented as [22, 42]

$$d\boldsymbol{z} = \overbrace{\left[\frac{d\log\alpha_t}{dt}\boldsymbol{z}_t - 1/2\left(\frac{d\sigma_t^2}{dt} - 2\frac{d\log\alpha_t}{dt}\sigma_t^2\right)\nabla_{\boldsymbol{z}}\log q(\boldsymbol{z}_t; \alpha_t, \sigma_t)\right]}^{\boldsymbol{d}(\boldsymbol{z}_t, t)} dt, \quad (2)$$

where $\boldsymbol{d}(\boldsymbol{z}_t, t)$ denotes the gradient vector at time t. $\nabla_{\boldsymbol{z}}\log q(\boldsymbol{z}; \alpha_t, \sigma_t)$ in (2) is the score function [15] pointing towards higher density of data samples at the given noise level (α_t, σ_t). One nice property of the score function is that it does not depend on the generally intractable normalization constant of the underlying density function $q(\boldsymbol{z}; \alpha_t, \sigma_t)$.

As t decreases, the probability flow ODE (2) continuously reduces the noise level of the data samples in the reverse process. In the ideal scenario where no approximations are introduced in (2), the sample distribution $p(\boldsymbol{z}; \alpha_t, \sigma_t)$ approaches the original data distribution $p_{data}(\boldsymbol{x})$ as t goes from T to 0. As a result, the sampling process of a diffusion model boils down to solving the ODE form (2), where randomness is only introduced in the initial sample at time T. This has opened up the research opportunity of exploiting different ODE solvers in diffusion-based sampling processes.

Denoising score matching: To be able to utilize (2) for sampling, one needs to specify a particular form of the score function $\nabla_{\boldsymbol{z}}\log q(\boldsymbol{z}; \alpha_t, \sigma_t)$. One common approach is to train a noise estimator $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ by minimizing the expected L_2 error for samples drawn from q_{data} (see [12, 35, 38]):

$$\mathbb{E}_{\boldsymbol{x}\sim p_{data}}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\sigma_t^2\boldsymbol{I})}\|\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\alpha_t\boldsymbol{x} + \sigma_t\boldsymbol{\epsilon}, t) - \boldsymbol{\epsilon}\|_2^2, \quad (3)$$

where (α_t, σ_t) are from the forward process (1). The common practice in diffusion models is to utilize a neural network of U-Net architecture [29] to represent the noise estimator $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$. With (3), the score function can then be represented in terms of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t; t)$ as (see also (229) of [16])

$$\nabla_{\boldsymbol{z}}\log q(\boldsymbol{z}_t; \alpha_t, \sigma_t) = \frac{-(\boldsymbol{z}_t - \alpha_t\boldsymbol{x})}{\sigma_t^2} = -\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t; t)/\sigma_t. \quad (4)$$

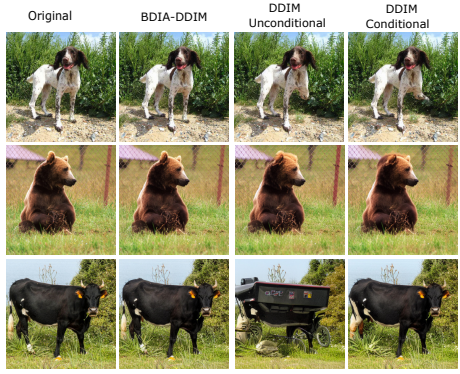

Fig. 2: BDIA-DDIM produces perceptually better image reconstruction (via round-trip) than DDIM. EDICT produces very similar results as BDIA-DDIM with two times of NFEs needed by BDIA-DDIM.

4 Bidirectional Integration Approximation (BDIA)

In this section, we first review DDIM inversion and EDICT as an extension of DDIM inversion. We then present our BDIA technique to enable exact diffusion inversion. As an example, Fig. 2 shows the performance of BDIA-DDIM for image reconstruction in comparison to that of DDIM. Convergence analysis for BDIA-DDIM is performed in the end.

4.1 Review of DDIM inversion

We first consider the update expression of DDIM for sampling, which is in fact a first-order solver for the ODE formulation (2) (see [22, 44]), given by

$$\boldsymbol{z}_{i-1} = \alpha_{i-1}\left(\frac{\boldsymbol{z}_i - \sigma_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)}{\alpha_i}\right) + \sigma_{i-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i) \tag{5}$$

$$= a_i \boldsymbol{z}_i + b_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i) \approx \boldsymbol{z}_i + \int_{t_i}^{t_{i-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau, \tag{6}$$

where $a_i = \alpha_{i-1}/\alpha_i$ and $b_i = \sigma_{i-1} - \sigma_i \alpha_{i-1}/\alpha_i$. It is clear from (5)-(6) that the integration $\int_{t_i}^{t_{i-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau$ is approximated by the forward DDIM update. That is, only the diffusion state $\boldsymbol{z}_i$ at the starting timestep t_i is used in the integration approximation.

To perform DDIM inversion, $\boldsymbol{z}_i$ can be approximated in terms of $\boldsymbol{z}_{i-1}$ as

$$\boldsymbol{z}_i = \alpha_i\left(\frac{\boldsymbol{z}_{i-1} - \sigma_{i-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)}{\alpha_{i-1}}\right) + \sigma_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i) \tag{7}$$

$$\approx \alpha_i\left(\frac{\boldsymbol{z}_{i-1} - \sigma_{i-1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i-1}, i)}{\alpha_{i-1}}\right) + \sigma_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{i-1}, i), \tag{8}$$

where $\boldsymbol{z}_i$ in the RHS of (7) is replaced with $\boldsymbol{z}_{i-1}$ to facilitate explicit computation. This naturally introduces approximation errors, leading to inconsistency of the diffusion states between the forward and backward processes.

4.2 Review of EDICT for exact diffusion inversion

Inspired by the flow generative framework [18], the recent work [40] proposed EDICT to enforce exact diffusion inversion. The basic idea is to introduce an auxiliary diffusion state $\boldsymbol{y}_i$ to be coupled with $\boldsymbol{z}_i$ at every timestep i. The next pair of diffusion states $(\boldsymbol{z}_{i-1}, \boldsymbol{y}_{i-1})$ is computed in an alternating fashion as

$$\boldsymbol{z}_i^{\text{inter}} = a_i \boldsymbol{z}_i + b_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{y}_i, i) \tag{9}$$

$$\boldsymbol{y}_i^{\text{inter}} = a_i \boldsymbol{y}_i + b_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i^{\text{inter}}, i) \tag{10}$$

$$\boldsymbol{z}_{i-1} = p\boldsymbol{z}_i^{\text{inter}} + (1-p)\boldsymbol{y}_i^{\text{inter}} \tag{11}$$

$$\boldsymbol{y}_{i-1} = p\boldsymbol{y}_i^{\text{inter}} + (1-p)\boldsymbol{z}_{i-1}, \tag{12}$$

where $p \in [0, 1]$ is the weighting factor and the pair $(\boldsymbol{z}_i^{\text{inter}}, \boldsymbol{y}_i^{\text{inter}})$ represents the intermediate diffusion states. According to [40], the two averaging operations (11)-(12) are introduced to make the update procedure stable.

Due to the alternating update formalism in (9)-(12), the computation can be inverted to obtain $(\boldsymbol{z}_i, \boldsymbol{y}_i)$ in terms of $(\boldsymbol{z}_{i-1}, \boldsymbol{y}_{i-1})$ as

$$\boldsymbol{y}_i^{\text{inter}} = (\boldsymbol{y}_{i-1} - (1-p)\boldsymbol{z}_{i-1})/p \tag{13}$$

$$\boldsymbol{z}_i^{\text{inter}} = (\boldsymbol{z}_{i-1} - (1-p)\boldsymbol{y}_i^{\text{inter}})/p \tag{14}$$

$$\boldsymbol{y}_i = (\boldsymbol{y}_i^{\text{inter}} - b_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i^{\text{inter}}, i))/a_i \tag{15}$$

$$\boldsymbol{z}_i = (\boldsymbol{z}_i^{\text{inter}} - b_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{y}_i, i))/a_i \tag{16}$$

Unlike (7)-(8), the inversion of (9)-(12) does not involve any approximation, thus enabling exact diffusion inversion.

However, it is clear from the above equations that the NFEs needed in EDICT is two times the NFEs required for DDIM. This makes the method computationally expensive. It is highly desirable to reduce the NFE in EDICT while retaining exact diffusion inversion. We provide such a method in the next subsection.

4.3 BDIA-DDIM for exact diffusion inversion

Reformulation of DDIM update expression: In this section, we present our new technique BDIA to assist DDIM in achieving exact diffusion inversion. To do so, we first reformulate the update expression for $\boldsymbol{z}_{i-1}$ in (6) in terms of all the historical diffusion states $\{\boldsymbol{z}_j\}_{j=N}^{i}$ as

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}_N + \sum_{j=N}^{i} \Delta(t_j \to t_{j-1} | \boldsymbol{z}_j) \tag{17}$$

$$\approx \boldsymbol{z}_N + \sum_{j=N}^{i} \int_{t_j}^{t_{j-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau, \tag{18}$$

where we use $\Delta(t_j \to t_{j-1}|\boldsymbol{z}_j)$ to denote approximation of the integration $\int_{t_j}^{t_{j-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau$ via the forward DDIM step, given by

$$\Delta(t_j \to t_{j-1}|\boldsymbol{z}_j) = \boldsymbol{z}_{j-1} - \boldsymbol{z}_j = a_j \boldsymbol{z}_j + b_j \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_j, j) - \boldsymbol{z}_j. \tag{19}$$

Algorithm 1 BDIA-DDIM for improving sampling quality

Input: number of time steps N, $\gamma \in [0,1]$
Sample $\boldsymbol{z}_N \sim \mathcal{N}(\boldsymbol{0}, \sigma_{t_N}^2 \boldsymbol{I})$
$\boldsymbol{z}_{N-1} = a_N \boldsymbol{z}_N + b_N \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_N, N)$
for $i \in \{N-1, \ldots, 1\}$ **do**
  $\boldsymbol{z}_{i-1} = \gamma(\boldsymbol{z}_{i+1} - \boldsymbol{z}_i) - \gamma\left(\frac{\boldsymbol{z}_i}{a_{i+1}} - \frac{b_{i+1}}{a_{i+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i) - \boldsymbol{z}_i\right) + \left[a_i \boldsymbol{z}_i + b_i \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)\right]$
end for
Output: $\boldsymbol{z}_0$

Extending forward DDIM by average of forward and backward DDIM: We argue that, in principle, each integration $\int_{t_j}^{t_{j-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau$ in (18) can be alternatively approximated by taking the average of both the forward and backward DDIM updates, expressed as

$$\int_{t_j}^{t_{j-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau \approx (1-\phi_{i,j}) \Delta(t_j \to t_{j-1} | \boldsymbol{z}_j) - \phi_{i,j} \Delta(t_{j-1} \to t_j | \boldsymbol{z}_{j-1}), \quad (20)$$

where the scalar $\phi_{i,j} \in [0,1]$, and the notation $\Delta(t_{j-1} \to t_j | \boldsymbol{z}_{j-1})$ denotes the backward DDIM step from t_{j-1} to t_j. The minus sign in front of $\Delta(t_{j-1} \to t_j | \boldsymbol{z}_{j-1})$ is due to integration over reverse time. The update expression for the backward DDIM step can be represented as

$$\Delta(t_{j-1} \to t_j | \boldsymbol{z}_{j-1}) = \frac{\boldsymbol{z}_{j-1}}{a_j} - \frac{b_j}{a_j} \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_{j-1}, j-1) - \boldsymbol{z}_{j-1}, \quad (21)$$

where the expressions for (a_j, b_j) are similar to (a_i, b_i) in (6). Note that in practice, we first need to perform a forward DDIM step over $[t_j, t_{j-1}]$ to obtain $\boldsymbol{z}_{j-1}$, and then we are able to perform the backward DDIM step computing $\Delta(t_{j-1} \to t_j | \boldsymbol{z}_{j-1})$.

We expect the averaged integration approximation (20) to be more accurate than the forward integration approximation alone. In particular, the averaged integration approximation should have a similar effect as the updates (see (57)-(58) in the appendix) of the improved Euler method which reduces the integration approximation error of the Euler method by utilizing the gradients at the two end points of a time-interval.

Bidirectional integration approximation (BDIA): We now present our new BDIA technique. Our primary goal is to develop an update expression for each $\boldsymbol{z}_{i-1}$ as a linear combination of $(\boldsymbol{z}_{i+1}, \boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i))$. As will be explained in the following, the summation of the integrations $\sum_{j=N}^{i} \int_{t_j}^{t_{j-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau) d\tau$ for $\boldsymbol{z}_{i-1}$ will involve both forward and backward DDIM updates as in (20).

We first draw insights from the updates for $\boldsymbol{z}_{N-1}$ and $\boldsymbol{z}_{N-2}$, and then propose a general update expression for $\boldsymbol{z}_i$, $i < N-1$. Suppose we are at the initial time step t_N with state $\boldsymbol{z}_N$. Then the next state $\boldsymbol{z}_{N-1}$ is computed by applying the forward DDIM (see (19)):

$$\boldsymbol{z}_{N-1} = a_N \boldsymbol{z}_N + b_N \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_N, N) = \boldsymbol{z}_N + \Delta(t_N \to t_{N-1} | \boldsymbol{z}_N). \quad (22)$$

Next we consider computing $\boldsymbol{z}_{N-2}$. With $\boldsymbol{z}_{N-1}$, we are able to compute $\Delta(t_{N-1} \to t_N|\boldsymbol{z}_{N-1})$ over the previous time-slot $[t_{N-1}, t_N]$ and $\Delta(t_{N-1} \to t_{N-2}|\boldsymbol{z}_{N-1})$ over the next time-slot $[t_{N-1}, t_{N-2}]$. When computing $\boldsymbol{z}_{N-2}$, the integration $\int_{t_N}^{t_{N-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau)d\tau$ can be approximated by averaging both $-\Delta(t_{N-1} \to t_N|\boldsymbol{z}_{N-1})$ and $(\boldsymbol{z}_{N-1} - \boldsymbol{z}_N) = \Delta(t_N \to t_{N-1}|\boldsymbol{z}_N)$. Other diffusion states $\boldsymbol{z}_i$, $i < N-2$, can be computed similarly as $\boldsymbol{z}_{N-2}$. Based on the above analysis, we propose to compute $\boldsymbol{z}_{i-1}$ for $i \leq N-1$ as:

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}_{i+1} \underbrace{-(1-\gamma)(\boldsymbol{z}_{i+1} - \boldsymbol{z}_i) - \gamma\Delta(t_i \to t_{i+1}|\boldsymbol{z}_i)}_{\approx \int_{t_{i+1}}^{t_i} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau)d\tau} + \underbrace{\Delta(t_i \to t_{i-1}|\boldsymbol{z}_i)}_{\approx \int_{t_i}^{t_{i-1}} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau)d\tau} \tag{23}$$

$$= \gamma\boldsymbol{z}_{i+1} - \gamma\left[\frac{\boldsymbol{z}_i}{a_{i+1}} - \frac{b_{i+1}}{a_{i+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)\right] + \left[a_i\boldsymbol{z}_i + b_i\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i)\right], \tag{24}$$

where $\gamma \in [0, 1]$. For the special case that $\gamma = 1$, (24) takes a simple form

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}_{i+1} - \Delta(t_i \to t_{i+1}|\boldsymbol{z}_i) + \Delta(t_i \to t_{i-1}|\boldsymbol{z}_i). \tag{25}$$

It is clear from (23) that in the computation of $\boldsymbol{z}_{i-1}$, the integration approximation $\int_{t_{i+1}}^{t_i} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau)d\tau$ for the previous time-slot $[t_{i+1}, t_i]$ is approximated by taking average of the backward DDIM update $-\Delta(t_i \to t_{i+1}|\boldsymbol{z}_i)$ and the integration approximation $\boldsymbol{z}_i - \boldsymbol{z}_{i+1}$ made earlier for the same time-slot. When $\gamma = 1$, the integration $\int_{t_{i+1}}^{t_i} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau)d\tau$ is simply approximated by the backward DDIM update $-\Delta(t_i \to t_{i+1}|\boldsymbol{z}_i)$. On the other hand, when $\gamma = 0$, (24) reduces to the regular DDIM update.

In principle, the computational complexities of DDIM and BDIA-DDIM sampling procedures should be roughly the same. BDIA-DDIM only incurs additional memory consumption to store the latest diffusion state $\boldsymbol{z}_{i+1}$.

Next we derive the explicit expression for $\boldsymbol{z}_i$ in terms of all the historical forward and backward DDIM updates, which is summarized in the following:

Proposition 1. *Let $\boldsymbol{z}_{N-1}$ and $\{\boldsymbol{z}_i | i \leq N-2\}$ be computed by following (22) and (24) sequentially. Then for each timestep $i \leq N-2$, $\boldsymbol{z}_i$ can be represented in the form of*

$$\boldsymbol{z}_i = \boldsymbol{z}_N + \sum_{j=i+2}^{N}\left[\frac{1-(-\gamma)^{j-i}}{1+\gamma}\Delta(t_j \to t_{j-1}|\boldsymbol{z}_j) - \frac{\gamma+(-\gamma)^{j-i}}{1+\gamma}\Delta(t_{j-1} \to t_j|\boldsymbol{z}_{j-1})\right] \tag{26}$$

$$+ \Delta(t_{i+1} \to t_i|\boldsymbol{z}_{i+1}). \tag{27}$$

Proof. See Appendix A for proof.

We now investigate (26)-(27). It is immediate from (26) that for each time-slot $[t_j, t_{j-1}]$, the summation of the two coefficients $(1-(-\gamma)^{j-i})/(1+\gamma)$ and $(\gamma+(-\gamma)^{j-i})/(1+\gamma)$ is equal to 1. This implies that the usage of BDIA indeed leads to the averaged forward and backward DDIM updates per time-slot as we planned

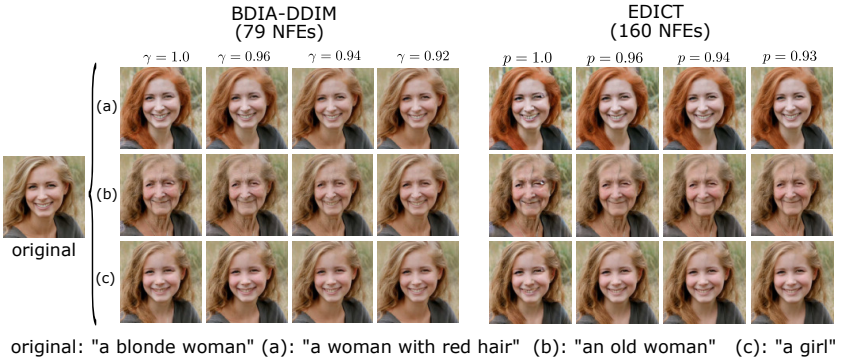

Fig. 3: Demonstration of the impact of γ and p values in (round-trip) image-editing performance of BDIA-DDIM and EDICT, respectively. The number of timesteps was set to 40 in both methods. The plots for BDIA-DDIM indicate that as γ decreases from 1 to 0.92, the edited images tend to be visually closer to the original image. That is, the γ parameter in BDIA-DDIM provides one more degree of freedom to allow for flexible image-editing than DDIM. The figure also indicates that the performance of EDICT with $p = 1$ has noticeable distortions while BDIA-DDIM with $\gamma = 1$ produces better image quality, and at the same time, only consumes half of the NFEs needed by EDICT. The recommended setup for p in EDICT is $p = 0.93$. See Appendix L for additional experiments by utilizing Null-text inversion.

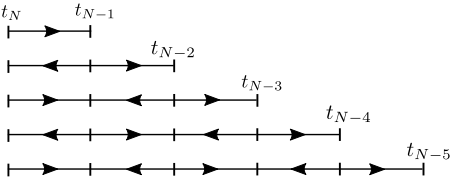

Fig. 4: Schematic illustration of the BDIA-DDIM integration formulation for $\gamma = 1$. Right and left arrows denote forward and backward updates, respectively.

earlier in (20). For the special case of $\gamma = 1$, the update expression (40) becomes much simpler (see Corollary 1 for the simplified expression). Fig. 4 demonstrates how the entire integration $\int_{t_N}^{t_i} \boldsymbol{d}(\boldsymbol{z}_\tau, \tau)d\tau$ for different $\boldsymbol{z}_i$ is approximated when $\gamma = 1$. It can be seen from the figure that the directions of the integration approximation for neighbouring time-slots are always opposite.

Next, we briefly discuss the averaging operations in EDICT and the proposed BDIA technique. It is seen from (11)-(12) that EDICT performs averaging via the parameter p over the two paired diffusion states $\boldsymbol{z}$ and $\boldsymbol{y}$, both of which are computed via forward DDIM updates (see Appendix I for an analysis of the update inconsistency in EDICT). On the other hand, BDIA proposes to average via the parameter γ over the DDIM forward and backward updates. See Fig. 3 for how the averaging operations affect the performance of EDICT and BDIA-DDIM for (round-trip) image editing.

BDIA-DDIM inversion: In brief, to allow for round-trip data manipulation such as image editing, the first step is to start from a clean image $\boldsymbol{z}_0 = \boldsymbol{x}$ and perform BDIA-DDIM inversion iteratively (except the computation of $\boldsymbol{z}_1$) to

Algorithm 2 Round-trip image-editing procedure of BDIA-DDIM

Input: $\boldsymbol{z}_0=\boldsymbol{x}$, N, ϕ $\tilde{\phi}$, $\gamma\in(0,1]$
$\boldsymbol{z}_1=\left(\frac{\boldsymbol{z}_0}{a_1}-\frac{b_1}{a_1}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_0,\phi,0)\right)$
for $i=1,\ldots,N-1$ **do**
  $\boldsymbol{z}_{i+1}=\boldsymbol{z}_{i-1}/\gamma-\left[a_i\boldsymbol{z}_i+b_i\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i,\phi,i)\right]/\gamma+\left[\frac{\boldsymbol{z}_i}{a_{i+1}}-\frac{b_{i+1}}{a_{i+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i,\phi,i)\right]$
end for
Set $(\tilde{\boldsymbol{z}}_N,\tilde{\boldsymbol{z}}_{N-1})=(\boldsymbol{z}_N,\boldsymbol{z}_{N-1})$
for $i=N-1,\ldots,1$ **do**
  $\tilde{\boldsymbol{z}}_{i-1}=\gamma\tilde{\boldsymbol{z}}_{i+1}-\gamma\left[\frac{\tilde{\boldsymbol{z}}_i}{a_{i+1}}-\frac{b_{i+1}}{a_{i+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{z}}_i,\tilde{\phi},i)\right]+\left[a_i\tilde{\boldsymbol{z}}_i+b_i\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{z}}_i,\tilde{\phi},i)\right]$
end for
Output: $\tilde{\boldsymbol{z}}_0$

compute a final pair of noise representations $(\boldsymbol{z}_N,\boldsymbol{z}_{N-1})$, conditioned on "reasonable" side information ϕ, such as text input correctly describing the image. Upon obtaining $(\boldsymbol{z}_N,\boldsymbol{z}_{N-1})$, we apply BDIA-DDIM based on the modified side information $\tilde{\phi}$ to acquire the final edited data $\tilde{\boldsymbol{z}}_0=\tilde{\boldsymbol{x}}$.

We first explain the process for obtaining $(\boldsymbol{z}_N,\boldsymbol{z}_{N-1})$ given $\boldsymbol{z}_0$ and ϕ. To start, we apply DDIM inversion to compute $\boldsymbol{z}_1$ with $\boldsymbol{z}_0$ and ϕ. After that, we compute $\boldsymbol{z}_{i+1}$ iteratively given $(\boldsymbol{z}_i,\boldsymbol{z}_{i-1})$ by following BDIA-DDIM inversion, where $i=1,\ldots,N-1$. It follows from (24) that the diffusion state $\boldsymbol{z}_{i+1}$ can be computed in terms of $(\boldsymbol{z}_i,\boldsymbol{z}_{i-1})$ as

$$\boldsymbol{z}_{i+1}=\boldsymbol{z}_{i-1}/\gamma-\left[a_i\boldsymbol{z}_i+b_i\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i,\phi,i)\right]/\gamma+\left(\frac{\boldsymbol{z}_i}{a_{i+1}}-\frac{b_{i+1}}{a_{i+1}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i,\phi,i)\right), \tag{28}$$

where ϕ is the additional conditioning input to the neural network $\boldsymbol{\theta}$. Similarly to the computation (28), EDICT also does not involve any approximation and results in exact diffusion inversion. However, in contrast to EDICT, (28) does not require a doubling of the NFE. Intuitively speaking, the efficiency of BDIA is because our new technique exploits time-reversibility while EDICT enforces reversibility by introducing an auxiliary diffusion state per timestep. For the special case of $\gamma=1$, both (25) and (28) are symmetric in time (provided the conditioning input ϕ is also symmetric). That is, switching the timestep t_{i+1} and t_{i-1} in (25) or (28) with $\gamma=1$ inverts the diffusion direction. We summarize the above property of time-symmetry in a lemma:

Proposition 2 (time-symmetry). *Switching the timestep t_{i-1} and t_{i+1} in (25) produces the reverse update (28) for the case $\gamma=1$, and vice versa.*

Once we obtain $(\boldsymbol{z}_N,\boldsymbol{z}_{N-1})=(\tilde{\boldsymbol{z}}_N,\tilde{\boldsymbol{z}}_{N-1})$, the remaining step is to apply BDIA-DDIM iteratively until reaching $\tilde{\boldsymbol{z}}_0$ by using the modified side information $\tilde{\phi}$. To do so, one can simply apply (24) by replacing $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{z}}_i,i)$ with $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{z}}_i,\tilde{\phi},i)$. Note that when $\tilde{\phi}=\phi$, the output $\tilde{\boldsymbol{z}}_0$ becomes a reconstruction of the original clean image $\boldsymbol{z}_0=\boldsymbol{x}$. See Alg. 2 for the (round-trip) image-editing procedure of BDIA-DDIM.

Remark 1. We have also designed coupled BDIA (CBDIA) as an extension of EDICT. CBDIA is also reversable and requires two NFEs per timestep in general. CBDIA includes BDIA with $\gamma = 1$ as a special case. See Appendix D for details.

Convergence analysis of BDIA-DDIM: Before we formally present the convergence results for BDIA-DDIM, we first introduce some notations and present two assumptions for $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$.We use $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}[m]$ to denote the mth component of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$. To facilitate the analysis later on, we denote the discrete stepsize at time t_i to be

$$h = \lambda_i - \lambda_{i-1} > 0 \quad \text{where } \lambda_j = \lambda_{t_j} = \frac{\sigma_{t_j}}{\alpha_{t_j}} \quad j = i, i-1 \tag{29}$$

for all $i = 1, \ldots, N$, over the time interval $t \in [T, \epsilon]$, where in practice, ϵ is set to be $\epsilon > 0$. For consistency, we let $\lambda_N = \lambda_T$ and $\lambda_0 = \lambda_\epsilon$. The two assumptions regarding $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ are summarized as below:

Assumption 1 *Assume $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}, t)$ satisfies the Lipschitz condition*

$$\|\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}, t) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{y}, t)\|_1 \le K_1 \|\boldsymbol{z} - \boldsymbol{y}\|_1 \quad T \ge t \ge \epsilon \quad \forall \boldsymbol{z}, \boldsymbol{y} \in \mathbb{R}^d, \tag{30}$$

where $K_1 > 0$, and d denotes the dimensionality of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$.

Assumption 2 *Let $\hat{\boldsymbol{\epsilon}}'_{\boldsymbol{\theta}}(\boldsymbol{z}, t)$ and λ'_t denote the differentials of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}, t)$ and λ_t with respect to t for $T \ge t \ge \epsilon$. Assume the magnitudes of $\left\{(1/\lambda'_t)\hat{\boldsymbol{\epsilon}}'_{\boldsymbol{\theta}}(\boldsymbol{z}, t)[m]\right\}_{m=1}^{d}$ are upper-bounded, represented as*

$$\left| \frac{1}{\lambda'_t} \hat{\boldsymbol{\epsilon}}'_{\boldsymbol{\theta}}(\boldsymbol{z}, t)[m] \right| \le K_2 \quad T \ge t \ge \epsilon \quad \forall \boldsymbol{z} \in \mathbb{R}^d \quad m = 1, \ldots, d. \tag{31}$$

Theorem 1. *Under the two Assumptions 1-2, consider using BDIA-DDIM to solve the ODE as specified by (2) and (4) with a uniform discrete step h as defined in (29). Denote the ground truth of the diffusion states of the ODE as $z_i^* = z_{t_i}^*$ over the time interval $[T, \epsilon]$. Let the initial errors with regard to the initial states $\{\boldsymbol{z}_i | i = N, N-1\}$ satisfy*

$$\eta(h) = \max_{N \ge i \ge N-1} \|\boldsymbol{z}_i^* - \boldsymbol{z}_i\|_1 \to 0 \quad \text{as} \quad h \to 0. \tag{32}$$

Then the error $\|\boldsymbol{z}_i^* - \boldsymbol{z}_i\|_1$, $N - 2 \ge i \ge 0$, is upper-bounded by

$$\|\boldsymbol{z}_i^* - \boldsymbol{z}_i\|_1 \le e^{\beta(\lambda_N - \lambda_0)} \eta(h) + \frac{\alpha_0 + \gamma\alpha_2}{2\beta} K_2 d (1 - e^{\beta(\lambda_N - \lambda_0)}) h, \tag{33}$$

where $\beta = \frac{2}{\lambda_0} + (\alpha_0 + \gamma\alpha_2) K_1$ and d denotes the dimensionality of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$. It is straightforward from (32) and (33) that the update expression of BDIA-DDIM converges in the time limit, i.e.,

$$\|\boldsymbol{z}_i^* - \boldsymbol{z}_i\|_1 \to 0 \text{ as } h \to 0 \quad \forall i = 0, \ldots, N-2. \tag{34}$$

Proof. See proof in Appendix B. Our convergence argument is inspired by the proof for Theorem 6.6 in [2] for linear multi-step methods.

When setting $\gamma = 0$ in Theorem 1, we obtain the convergence results for DDIM. Considering the initial condition (32), one can easily show by following Appendix B that it holds when applying DDIM in computing $\boldsymbol{z}_{N-1}$ given $\boldsymbol{z}_N$.

5 Experiments

We conducted two types of experiments: (1) image sampling; (2) image-editing. It was found that our new technique BDIA produces promising results for both tasks.

5.1 Evaluation of image sampling

Text-to-image generation: In this task, we tested five sampling methods by using StableDiffusion,V2[6] which are DDIM, DPM-Solver++, PLMS, EDICT, and BDIA-DDIM. COCO2014 validation set was utilized for this task. For each method, 20K images of size 512×512 were generated by feeding 20K text prompts to the diffusion model. All the five methods share the same seed of the random noise generator and the same text prompts. The FID score for each method was computed by resizing the generated images to size of 256×256 due to the fact that the original images in COCO2014 in general have lower resolution than the size of 512×512.

Table 1: FID (↓) and CLIP (↑) evaluation of five methods for text-to-image generation over StableDiffusion V2. See also Appendix H for how p and γ affect the sampling performance of EDICT and BDIA-DDIM, respectively. For each number of timesteps, EDICT requires twice the NFEs needed in each of other four methods. The results of EDICT at 80 NFEs are better than the results of DDIM at 40 NFEs, which are consistent with those of [40]. The values denoted with $*$ are taken from [42].

		EDICT	BDIA-DDIM	DDIM	DPM-Solver++	PLMS
10 NFEs	FID	-	**12.62**	14.78*	15.82*	24.42*
	CLIP	-	**25.05**	24.86*	24.83*	23.85*
20 NFEs	FID	30.91	**13.44**	14.65*	13.85*	14.30*
	CLIP	24.98	**25.12**	25.00*	25.02*	24.80*
40 NFEs	FID	15.51	**13.87**	14.69*	14.29*	15.05*
	CLIP	**25.44**	25.05	25.01*	25.05*	24.95*
80 NFEs	FID	14.34	**13.53**	15.16	14.83	14.97
	CLIP	**25.11**	24.85	24.99	25.04	25.00

It is clear from Table 1 that BDIA-DDIM has a remarkable FID performance gain over the other four methods. This indicates that the introduced backward DDIM update per timestep indeed helps to improve the accuracy of the corresponding integration approximation. EDICT, on the other hand, produces better CLIP scores for 40 and 80 NFEs. The visual improvement of BDIA-DDIM over DDIM is also demonstrated in Figs. 1, 7, 8.

Conventional image generation: In this experiment, we consider the task of noise-to-image generation without conditioning text. The parameter γ in BDIA-DDIM was set to $\gamma = 1.0$. The tested pre-trained models can be found in Appendix G. Given a pre-trained model, 50K artificial images were generated for a particular NFE, and the corresponding FID score was computed.

Table 2 summarize the computed FID scores. It is clear that by incorporating BDIA into DDIM, the FID scores are improved considerably. This can be explained by the fact that BDIA introduces the additional backward integration approximation per time-step in the sampling process. This makes the

[6] https://github.com/Stability-AI/stablediffusion

Table 2: FID comparison for conventional sampling. See Appendix I for how p affects the sampling performance of EDICT. Recall that for each number of timesteps, EDICT requires twice the number of NFEs needed by DDIM and BDIA-DDIM.

NFEs		EDICT	DDIM	BDIA -DDIM		EDICT	DDIM	BDIA -DDIM
10	CIFAR10	-	14.38	**10.03**	Celeba	-	13.41	**10.86**
20		203.86	7.51	**6.29**		190.62	9.45	**8.86**
40		116.72	4.95	**4.63**		81.44	6.93	**6.50**
80		51.17	3.82	**3.70**		43.93	4.87	**4.65**

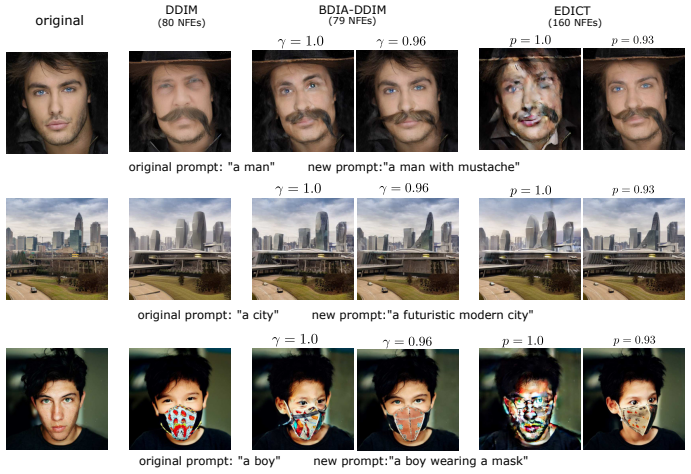

Fig. 5: Round-trip image editing via DDIM, BDIA-DDIM and EDICT. $p = 0.93$ is the recommended setup in [40] for EDICT. Similarly to Fig. 3, the number of timesteps was set to 40 in all the three methods. See Appendix K for additional text-based and ControlNet-based image-editing results.

resulting final integration approximation more accurate. We have also provided an explanation in appendix I why EDICT does not perform well in Table 2.

5.2 Evaluation of image-editing

In this second experiment, we evaluated BDIA-DDIM for round-trip image-editing by utilizing the open-source repository of EDICT,[7] and for single-trip ControlNet-based image-editing by using the repository of ControlNet.[8] ControlNet [43] allows the spatial layout of a synthesized image to be controlled with conditioning in the form of edges, "scribbles", and other modalities. In the experiment, we utilize the scribble format for image editing. Fig. 5 and 6 visualize the obtained results. We point out that in Fig. 5, BDIA-DDIM produces very similar results to EDICT while reducing by half the NFE compared to EDICT. Fig. 6 demonstrates that BDIA-DDIM produces better image quality than DDIM for ControlNet-based image editing.

[7] https://github.com/salesforce/EDICT
[8] https://github.com/lllyasviel/ControlNet

Fig. 6: Single-trip ControlNet-based image editing by using DDIM and BDIA-DDIM at 10 timesteps. The hyper-parameter γ in BDIA-DDIM was set to 0.5. See Appendix K for additional ControlNet-based image-editing results.

5.3 Editability and limitations of BDIA-DDIM

Figs. 3 and 5 demonstrate different effects that are obtained by changing the γ parameter. As image editing is inherently subjective and the algorithm cannot guess what effect is desired, the availability of this parameter can be viewed as an advantage of BDIA-DDIM. In practice, the γ parameter can be adjusted by the artist to produce different image-editing effects. Since GPUs allow the processing of a batch of images with different γ values simultaneously, the artist can thus choose the generated image with the desired effect from the batch.

In comparison to DDIM, BDIA-DDIM requires additional memory to store $\boldsymbol{z}_{i+1}$ in computing $\boldsymbol{z}_{i-1}$. However, it produces a better image quality. The additional computational overhead in BDIA-DDIM can be ignored since only simple linear algebra is involved. When performing round-trip image editing, the update expression for BDIA-DDIM inversion requires two initial diffusion states $(\boldsymbol{z}_0, \boldsymbol{z}_1)$, where $\boldsymbol{z}_1$ must be computed by following other methods, such as DDIM.

6 Conclusions

In this paper, we have proposed a new technique BDIA, to assist DDIM in achieving exact diffusion inversion. The key step of BDIA-DDIM is to perform DDIM update procedure twice at each time step t_i: one over the previous time-slot $[t_i, t_{i+1}]$ and the other over the next time-slot $[t_i, t_{i-1}]$ in computing $\boldsymbol{z}_{i-1}$. By doing so, the expression for $\boldsymbol{z}_{i-1}$ becomes a linear combination of $(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i), \boldsymbol{z}_{i+1})$. As a result, $\boldsymbol{z}_{i+1}$ can be computed exactly as a linear function of $(\boldsymbol{z}_i, \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i), \boldsymbol{z}_{i-1})$, enabling exact diffusion inversion. Note that although the DDIM update is evaluated twice at each step, this is inexpensive since the costly neural functional evaluation is performed only once. Convergence results for BDIA-DDIM are provided by following an analysis similar to that for linear multi-step methods. Extensive experiments show that BDIA-DDIM not only enables effective image editing but also improves over the image sampling quality of DDIM and EDICT. Please see Appendix K in the supplementary material for additional text-based and ControlNet-based image-editing results.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN. arXiv:1701.07875 [stat.ML] (2017) 1
2. Atkinson, K.E.: An Introduction to Numerical Analysis. John Wiley Sons, 2nd edn. (1988) 3, 11, 21
3. Bao, F., Li, C., Sun, J., Zhu, J., Zhang, B.: Estimating the Optimal Covariance with Imperfect Mean in Diffusion Probabilistic Models. In: ICML (2022) 1
4. Bao, F., Li, C., Zhu, J., Zhang, B.: Analytic-DPM: an Analytic Estimate of the Optimal Reverse Variance in Diffusion Probabilistic Models. In: ICLR (2022) 1
5. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006) 1
6. Chen, N., Zhang, Y., Zen, H., Weiss, R.J., Norouzi, M., Chan, W.: WaveGrad: Estimating Gradients for Waveform Generation. arXiv:2009.00713 (September 2020) 1
7. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. arXiv:2105.05233 [cs.LG] (2021) 1
8. Dinh, L., Krueger, D., Bengio, Y.: Nice: Non-linear independent components estimation. arXiv preprint arXiv:1410.8516 (2014) 2
9. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. arXiv preprint arXiv:1605.08803 (2016) 2
10. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. In: Proceedings of the International Conference on Neural Information Processing Systems. pp. 2672–2680 (2014) 1
11. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: Advances in neural information processing systems. pp. 5767–5777 (2017) 1
12. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS (2020) 1, 4
13. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022) 1
14. Huberman-Spiegelglas, I., Kulikov, V., Michaeli, T.: An Edit Friendly DDPM Noise Space: Inversion and Manipulations. arXiv:2304.06140v2 [cs.CV] (2023) 2
15. Hyvarinen, A.: Estimation of non-normalized statistical models by score matching. Journal of Machine Learning Research **24**, 695–709 (2005) 4
16. Karras, T., Aittala, M., Alia, T., Laine, S.: Elucidating the Design Space of Diffusion-Based Generative Models. In: 36th Conference on Nueral Information Processing Systems (NeurIPS) (2022) 1, 4, 25, 26
17. Kim, D., Kim, Y., Kwon, S.J., Kang, W., Moon, I.C.: Refining Generative Process with Discriminator Guidance in Score-based Diffusion Models. arXiv preprint arXiv:2211.17091 [cs.CV] (2022) 1
18. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In: Advances in neural information processing systems (2018) 2, 6
19. Kingma, D.P., Salimans, T., Poole, B., Ho, J.: Variational diffusion models. arXiv: preprint arXiv:2107.00630 (2021) 1
20. Lam, M.W.Y., Wang, J., Su, D., Yu, D.: BDDM: Bilateral Denoising Diffusion Models for Fast and High-Quality Speech Synthesis. In: ICLR (2022) 1
21. Liu, L., Ren, Y., Lin, Z., Zhao, Z.: Pseudo Numerical Methods for Diffusion Models on Manifolds. In: ICLR (2022) 1

22. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Sampling in Around 10 Steps. In: NeurIPS (2022) 3, 4, 5, 21
23. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models. arXiv preprint arXiv:2211.01095 [cs.LG] (November 2022) 1
24. Mokady, R., Hertz, A., Aberman, K., Pritch, Y., Cohen-Or, D.: Null-text Inversion for Editing Real Images using Guided Diffusion Models. In: CVPR (2023) 1, 2
25. Nichol, A., Dhariwal, P.: Improved denoising diffusion probabilistic models. arXiv preprint arXiv:2102.09672 (2021) 1
26. Nichol, A., Dharwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., Chen, M.: GLIDE: Towards Photorealistic image generation and editing with text-guided diffusion models. In: ICML (2022) 1
27. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022) 1
28. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: On High-resolution image ¨ synthesis with latent diffusion models. In: CVPR. p. 10684–10695 (2022) 2
29. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597 [cs.CV] (2015) 4
30. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S.K.S., Ayan, B.K., Mahdavi, S.S., Lopes, R.G., Salimans, T., Ho, J., Fleet, D.J., Norouzi, M.: Photorealistic text-to-image diffusion models with deep language understanding. arXiv preprint arXiv:2205.11487 (2022) 2
31. Sauer, A., Schwarz, K., Geiger, A.: StyleGAN-XL: Scaling StyleGAN to large diverse datasets. In: SIGGRAPH (2022) 1
32. Shi, Y., Xue, C., Pan, J., Zhang, W.: DragDiffusion: Harnessing Diffusion Models for Interactive Point-based Image Editing. arXiv:2306.14435v2 (2023) 2
33. Skeel, R.D.: Variable Step Size Destabilizes the Stömer/Leapfrog/Verlet Method. BIT Numerical Mathematics **33**, 172–175 (1993) 3
34. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. ICML (2015) 1
35. Song, J., Meng, C., Ermon, S.: Denoising Diffusion Implicit Models. In: ICLR (2021) 1, 4
36. Song, Y., Durkan, C., Murray, I., Ermon, S.: Maximum likelihood training of score-based diffusion models. In: Advances in neural information processing systems (NeurIPS) (2021) 1
37. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. In: Advances in neural information processing systems (NeurIPS). p. 11895–11907 (2019) 1
38. Song, Y., S.-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-Based Generative Modeling Through Stochastic Differential Equations. In: ICLR (2021) 1, 4
39. Verlet, L.: Computer Experiments on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. Physical Review **159**(1), 98–103 (1967) 3
40. Wallace, B., Gokul, A., Naik, N.: EDICT: Exact Diffusion Inversion via Coupled Transformations. In: CVPR (2023) 1, 2, 6, 12, 13, 31, 37, 38
41. Xu, Y., Deng, M., Cheng, X., Tian, Y., Liu, Z., Jaakkola, T.: 3
42. Zhang, G., Niwa, K., Kleijn, W.B.: On Accelerating Diffusion-Based Sampling Processes by Improved Integration Approximation. ICLR (2024) 1, 4, 12

43. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: IEEE International Conference on Computer Vision (ICCV) (2023) 13
44. Zhang, Q., Chenu, Y.: Fast Sampling of Diffusion Models with Exponential Integrator. arXiv:2204.13902 [cs.LG] (2022) 1, 5