Towards More Accurate Diffusion Model Acceleration with A Timestep Aligner
– *Supplementary Material* –

Mengfei Xia[1] Yujun Shen[2] Changsong Lei[1] Yu Zhou[1] Deli Zhao[3]
Ran Yi[4] Wenping Wang[5] Yong-Jin Liu[1*]
[1]Tsinghua University [2]Ant Group [3]Alibaba Group
[4]Shanghai Jiao Tong University [5]Texas A&M University

A. Proofs and Derivations

In this section, we will prove the theorems claimed in the main manuscript.

A.1. Proof of Theorem 1

Theorem 1. *Assume that $\boldsymbol{\epsilon}_\theta$ is the ground-truth noise prediction model, with $\|\boldsymbol{\epsilon}_\theta(\mathbf{x},t)-\boldsymbol{\epsilon}_\theta(\mathbf{y},t)\|_2 \geqslant \frac{1}{C}\|\mathbf{x}-\mathbf{y}\|_2$ for any t and some $C>0$. Denote by $\mathbf{x}_{t_i}^{gt}$ the ground-truth intermediate result at t_i starting from $\widetilde{\mathbf{x}}_{t_K}$, and by $f_{\theta,\tau}$ a deterministic sampler. We have the following inequality:*

$$\begin{aligned}&\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}[\|\widetilde{\mathbf{x}}_{t_{i-1}}-\mathbf{x}_{t_{i-1}}^{gt}\|_2]\\ \leqslant& C\Big(\sum_{n=i}^{K}\left(\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})-\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2\right]\right)^{\frac{1}{2}}\\ &+\sum_{l=i}^{K}\mathbb{E}[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_l}^{gt},t_l)-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{l-1}}^{gt},t_{l-1})\|_2]\Big).\end{aligned}\tag{S1}$$

Proof of Theorem 1. By the assumption, we have

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}[\|\widetilde{\mathbf{x}}_{t_{i-1}}-\mathbf{x}_{t_{i-1}}^{gt}\|_2]\leqslant C\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{i-1}}^{gt},t_{i-1})\|_2]\tag{S2}$$

Define $e_{i-1}=\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{i-1}}^{gt},t_{i-1})$. Then we can easily derive that

$$\begin{aligned}e_{i-1}=&\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})-\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)+\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_i}^{gt},t_i)\\ &+\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_i}^{gt},t_i)-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{i-1}}^{gt},t_{i-1})&\text{(S3)}\\ =&\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})-\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)+e_i+\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_i}^{gt},t_i)-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{i-1}}^{gt},t_{i-1})&\text{(S4)}\\ =&\sum_{n=i}^{K-1}\Big(\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_n},\tau_n),t_{n-1})-\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_n},t_n)\Big)+e_{K-1}\\ &+\sum_{l=i}^{K-1}\Big(\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_l}^{gt},t_l)-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{l-1}}^{gt},t_{l-1})\Big),&\text{(S5)}\end{aligned}$$

where Eq. (S4) is due to $\widetilde{\mathbf{x}}_{t_i}=f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_{i+1}},\tau_{i+1})$. Since $\mathbf{x}_{t_K}^{gt}=\widetilde{\mathbf{x}}_{t_K}$, we have

$$e_{K-1}=\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_K},\tau_K),t_{K-1})-\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_K},t_K)+\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_K}^{gt},t_K)-\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{K-1}}^{gt},t_{K-1}).\tag{S6}$$

*Corresponding author.

Then we have

$$e_{i-1} = \sum_{n=i}^{K} \Big(\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_n}, \tau_n), t_{n-1}) - \boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_n}, t_n) \Big) + \sum_{l=i}^{K} \Big(\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_l}^{gt}, t_l) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{l-1}}^{gt}, t_{l-1}) \Big), \tag{S7}$$

and

$$\begin{aligned}
&\mathbb{E}[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i}, \tau_i), t_{i-1}) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{i-1}}^{gt}, t_{i-1})\|_2] && \text{(S8)}\\
\leqslant &\sum_{n=i}^{K} \mathbb{E}[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_n}, \tau_n), t_{n-1}) - \boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_n}, t_n)\|_2] \\
&+ \sum_{l=i}^{K} \mathbb{E}[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_l}^{gt}, t_l) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{l-1}}^{gt}, t_{l-1})\|_2] && \text{(S9)}\\
\leqslant &\sum_{n=i}^{K} \mathcal{L}_n(\tau_n)^{\frac{1}{2}} + \sum_{l=i}^{K} \mathbb{E}[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_l}^{gt}, t_l) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_{l-1}}^{gt}, t_{l-1})\|_2], && \text{(S10)}
\end{aligned}$$

where Eq. (S10) is due to Cauchy inequality. Combine Eq. (S2) and Eq. (S10), we prove the theorem. □

A.2. Proof of Theorem 2

Theorem 2. *Assume that $\boldsymbol{\epsilon}_\theta$ is the ground-truth noise prediction model. The loss function of `TimeTuner` resembles that of the original DPM, i.e., for $i = K, K-1, \cdots, 1$, the optimal τ_i holds the following property:*

$$\begin{aligned}
&\arg\min_{\tau_i} \mathcal{L}_i(\tau_i) \\
=&\arg\min_{\tau_i} \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i}, \tau_i), t_{i-1}) \\
&\qquad - \frac{\widetilde{\mathbf{x}}_{t_i} - \alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\|_2^2]. && \text{(S11)}
\end{aligned}$$

We first claim the following lemmas which are crucial for the proof of Theorem 2.

Lemma 1. *Let $\mathbf{x}_0 \sim q_0(\mathbf{x}_0)$, and $q_{0t}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \alpha_t\mathbf{x}_0, \sigma_t^2\mathbf{I})$. Denote by $q_t(\mathbf{x}_t)$ the marginal distribution of $\mathbf{x}_t$. Then we have $\nabla \log q_t(\mathbf{x}_t) = -\mathbb{E}\left[\frac{\mathbf{x}_t - \alpha_t\mathbf{x}_0}{\sigma_t^2}|\mathbf{x}_t\right]$.*

Proof of Lemma 1. According to the definition of $q_t(\mathbf{x}_t)$, one can notice that $\nabla \log q_t(\mathbf{x}_t) = \nabla_{\mathbf{x}_t} \log \int q_0(\mathbf{x}_0) q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0$. Then we have

$$\begin{aligned}
\nabla \log q_t(\mathbf{x}_t) &= \frac{\int q_0(\mathbf{x}_0) \nabla_{\mathbf{x}_t} q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0}{\int q_0(\mathbf{x}_0) q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0} && \text{(S12)}\\
&= \frac{\int q_0(\mathbf{x}_0) q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \nabla_{\mathbf{x}_t} \log q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0}{q_t(\mathbf{x}_t)} && \text{(S13)}\\
&= \int \frac{q_0(\mathbf{x}_0) q_{0t}(\mathbf{x}_t|\mathbf{x}_0)}{q_t(\mathbf{x}_t)} \nabla_{\mathbf{x}_t} \log q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0 && \text{(S14)}\\
&= \int q(\mathbf{x}_0|\mathbf{x}_t) \nabla_{\mathbf{x}_t} \log q_{0t}(\mathbf{x}_t|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0 && \text{(S15)}\\
&= \mathbb{E}[\nabla_{\mathbf{x}_t} \log q_{0t}(\mathbf{x}_t|\mathbf{x}_0)|\mathbf{x}_t] && \text{(S16)}\\
&= -\mathbb{E}\left[\frac{\mathbf{x}_t - \alpha_t\mathbf{x}_0}{\sigma_t^2}|\mathbf{x}_t\right], && \text{(S17)}
\end{aligned}$$

where Eq. (S15) comes from Bayes' rule. □

Lemma 2. *Let $g(\mathbf{x}_t), h(\mathbf{x}_t, \mathbf{x}_0)$ be integrable functions, then the following equality holds.*

$$\mathbb{E}_{q(\mathbf{x}_t)}[\langle g(\mathbf{x}_t), \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}[h(\mathbf{x}_t, \mathbf{x}_0)|\mathbf{x}_t]\rangle] = \mathbb{E}_{q(\mathbf{x}_t)}[\langle g(\mathbf{x}_t), h(\mathbf{x}_t, \mathbf{x}_0)\rangle]. \tag{S18}$$

Proof of Lemma 2. Note that

$$\mathbb{E}_{q(\mathbf{x}_t)}[\langle g(\mathbf{x}_t), \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}[h(\mathbf{x}_t,\mathbf{x}_0)|\mathbf{x}_t]\rangle] = \int \langle g(\mathbf{x}_t), \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}[h(\mathbf{x}_t,\mathbf{x}_0)|\mathbf{x}_t]\rangle p(\mathbf{x}_t)\mathrm{d}\mathbf{x}_t \tag{S19}$$

$$= \int \langle g(\mathbf{x}_t), \int h(\mathbf{x}_t,\mathbf{x}_0)p(\mathbf{x}_0|\mathbf{x}_t)\mathrm{d}\mathbf{x}_0\rangle p(\mathbf{x}_t)\mathrm{d}\mathbf{x}_t \tag{S20}$$

$$= \int\int \langle g(\mathbf{x}_t), h(\mathbf{x}_t,\mathbf{x}_0)\rangle p(\mathbf{x}_0|\mathbf{x}_t)p(\mathbf{x}_t)\mathrm{d}\mathbf{x}_0\mathrm{d}\mathbf{x}_t \tag{S21}$$

$$= \int\int \langle g(\mathbf{x}_t), h(\mathbf{x}_t,\mathbf{x}_0)\rangle p(\mathbf{x}_0,\mathbf{x}_t)\mathrm{d}\mathbf{x}_0\mathrm{d}\mathbf{x}_t \tag{S22}$$

$$= E_{q(\mathbf{x}_t)}[\langle g(\mathbf{x}_t), h(\mathbf{x}_t,\mathbf{x}_0)\rangle]. \tag{S23}$$

where Eq. (S21) is by linearity of integral. □

Then we start to prove the Theorem 2 as below.

Proof of Theorem 2. Given the assumption that $\boldsymbol{\epsilon}_\theta$ is the ground-truth noise prediction model, we have $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t) = \mathbb{E}[\frac{\mathbf{x}_t-\alpha_t\mathbf{x}_0}{\sigma_t}|\mathbf{x}_t]$ from Lemma 1. Then we have

$$\mathcal{L}_i(\tau_i) = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1}) - \boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2\right] \tag{S24}$$

$$\begin{aligned}=&\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})\|_2^2 + \|\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2\right]\\ &- 2\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\langle\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1}),\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\rangle\right]\end{aligned} \tag{S25}$$

$$\begin{aligned}=&\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})\|_2^2 + \|\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2\right]\\ &- 2\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\langle\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1}),\mathbb{E}\left[\frac{\widetilde{\mathbf{x}}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}|\widetilde{\mathbf{x}}_{t_i}\right]\right\rangle\right]\end{aligned} \tag{S26}$$

$$\begin{aligned}=&\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1})\|_2^2 + \|\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2\right]\\ &- 2\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\langle\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1}),\frac{\widetilde{\mathbf{x}}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\right\rangle\right]\end{aligned} \tag{S27}$$

$$\begin{aligned}=&\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1}) - \frac{\widetilde{\mathbf{x}}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\|_2^2\right]\\ &, \quad + \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2 - \|\frac{\widetilde{\mathbf{x}}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\|_2^2\right],\end{aligned} \tag{S28}$$

where Eq. (S27) is due to Lemma 2. Since $\|\boldsymbol{\epsilon}_\theta(\widetilde{\mathbf{x}}_{t_i},t_i)\|_2^2 - \|\frac{\widetilde{\mathbf{x}}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\|_2^2$ is independent with τ_i, we have

$$\arg\min_{\tau_i}\mathcal{L}_i(\tau_i) = \arg\min_{\tau_i}\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(f_{\theta,\tau}(\widetilde{\mathbf{x}}_{t_i},\tau_i),t_{i-1}) - \frac{\widetilde{\mathbf{x}}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\|_2^2\right]. \tag{S29}$$

□

Remark 1. *Note that the objective of the original DPM has the following form:*

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_i},t_i) - \boldsymbol{\epsilon}\|_2^2\right] = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}_{t_i},t_i) - \frac{\mathbf{x}_{t_i}-\alpha_{t_i}\mathbf{x}_0}{\sigma_{t_i}}\|_2^2\right], \tag{S30}$$

which has a similar form as the objective in Theorem 2.

B. Pseudo-code of Training Process

Recall that we introduce two different training strategies for the proposed `TimeTuner`, *i.e.*, the *sequenatial strategy* and the *parallel strategy*. We have proved the equivalence of the two training strategies, and analyzed the performance difference between the two strategies upon DDIM [2]. In this part, we provide the pseudo-codes of the two training strategies in Algorithm S1 and Algorithm S2.

Algorithm S1 Pseudo-code of sequential training strategy of `TimeTuner` in a PyTorch-like style.

```
import torch


def sequential_training_loss(x_0, t_list, tau_list, i, tau_i, F, E):
    """Defines the forward process of one sequential training step.

    Args:
        x_0: Data inputs, with shape [B, C, H, W].
        t_list: The preset timestep trajectory from 0 to T.
        tau_list: The list consist of previously achieved re-aligned timesteps from tau_K to tau_ip1.
        i: The index of current timestep tau.
        tau_i: The timestep to re-align.
        F: The DE solver to denoise the input ’x’ from timestep ’t’ to timestep ’s’ using re-aligned input condition ’tau’.
        E: The noise prediction model with input ’x’ and ’t’.
    """
    # Compute the x_T at timestep T.
    z_T = torch.randn_like(x_0)
    x_T = alpha_T * x_0 + sigma_T * z_T

    # Compute the denoised intermediate x_t_i
    x = x_T
    for tau, t, t_prev in zip(tau_list, t_list[::-1], t_list[-2::-1]):
        x = F(x, t, t_prev, tau)
    x_t_i = x

    # Get the current and the previous timestep.
    t_i, t_im1 = t_list[i], t_list[i - 1]

    # Compute the denoised intermediate x_t_im1 with tau_i
    x_t_im1 = F(x_t_i, t_i, t_im1, tau_i)

    # Learn the translator.
    loss = (E(x_t_im1, t_im1) - E(x_t_i, t_i)).square().mean()

    return loss
```

Algorithm S2 Pseudo-code of parallel training strategy of `TimeTuner` in a PyTorch-like style.

```
import torch


def parallel_training_loss(x_0, t_list, i, tau_i, F, E):
    """Defines the forward process of one parallel training step.

    Args:
        x_0: Data inputs, with shape [B, C, H, W].
        t_list: The preset timestep trajectory from 0 to T.
        i: The index of current timestep tau.
        tau_i: The timestep to re-align.
        F: The DE solver to denoise the input ’x’ from timestep ’t’ to timestep ’s’ using re-aligned input condition ’tau’.
        E: The noise prediction model with input ’x’ and ’t’.
    """
    # Get the current and the previous timestep.
    t_i, t_im1 = t_list[i], t_list[i - 1]

    # Compute the x_t_i at timestep t_i.
    z_t_i = torch.randn_like(x_0)
    x_t_i = alpha_t_i * x_0 + sigma_t_i * z_t_i

    # Compute the denoised intermediate x_t_im1 with tau_i
    x_t_im1 = F(x_t_i, t_i, t_im1, tau_i)

    # Learn the translator.
    loss = (E(x_t_im1, t_im1) - E(x_t_i, t_i)).square().mean()

    return loss
```

Table S1. **Quantitative comparison** measured by IS ↑, FID ↓, sFID ↓, Precision ↑ and Recall ↑ on ImageNet 256. All are evaluated by drawing 50,000 samples via DDIM sampler upon LDM, with NFE $= 10$. We implement `TimeTuner` using the two equivalent loss functions, *i.e.*, Eq. (10) and Eq. (11).

NFE $= 10$	Method	IS ↑	FID ↓	sFID ↓	Precision ↑	Recall ↑
ImageNet	DDIM	324.52	10.13	12.52	0.91	0.28
	DDIM + Ours, Eq. (10)	336.94	9.63	7.29	0.92	0.30
	DDIM + Ours, Eq. (11)	330.42	9.13	7.72	0.92	0.30

Table S2. **Quantitative results** measured by FID ↓, Precision ↑ and Recall ↑ on LSUN Bedroom 256 and FFHQ 256. All are evaluated by drawing 50,000 samples via DDIM sampler upon LDM, with NFE $= 10$. We report the mean and variance of evaluation metrics with 5 independent sampling.

	Method	FID ↓	Precision ↑	Recall ↑
Bedroom	DDIM	9.46	0.55	0.34
	DDIM + Ours	5.84 $\pm$ 0.02	0.57 $\pm$ 0.00	0.44 $\pm$ 0.01
FFHQ	DDIM	23.58	0.63	0.21
	DDIM + Ours	14.92 $\pm$ 0.08	0.67 $\pm$ 0.00	0.32 $\pm$ 0.00

Table S3. **Quantitative comparison** measured by FID ↓, Precision ↑, and Recall ↑ on LSUN Bedroom 256, FFHQ 256, CelebA-HQ 256, and ImageNet 256. All are evaluated by drawing 50,000 samples via DDIM sampler upon LDM, with NFE $= 5$.

LSUN Bedroom 256x256, *unconditional* generation			
Method	FID ↓	Precision ↑	Recall ↑
DDIM	44.97	0.21	0.13
DDIM + Ours	**13.04**	**0.41**	**0.41**
FFHQ 256x256, *unconditional* generation			
Method	FID ↓	Precision ↑	Recall ↑
DDIM	65.85	0.36	0.04
DDIM + Ours	**28.73**	**0.61**	**0.32**
CelebA-HQ 256x256, *unconditional* generation			
Method	FID ↓	Precision ↑	Recall ↑
DDIM	50.02	0.44	0.03
DDIM + Ours	**39.50**	**0.45**	**0.19**
ImageNet 256x256, *conditional* generation			
Method	FID ↓	Precision ↑	Recall ↑
DDIM	16.87	0.68	0.26
DDIM + Ours	**9.74**	**0.80**	**0.30**

C. Additional Experiments

C.1. Experimental Comparison between Two Equivalent Loss Functions

Recall that we demonstrate the equivalence between the two loss functions of `TimeTuner` in Theorem 2. Note that the equivalence is proved under the assumption that the noise prediction model ϵ_θ is ground-truth. Therefore, in practice, the imperfect noise prediction model indeed results in inequivalence between the two loss functions. However, as demonstrated in Tab. S1, the empirical evidence suggests that the two variants achieve on-par performance (i.e., Eq. (10) better on some metrics while Eq. (11) on others), both better than the baseline.

C.2. Robustness of TimeTuner

Theoretically, `TimeTuner` optimizes the upper bound of the gap between real and sampling distributions, implemented by optimization of the τ_i on average across the whole real distribution. This may lead to potential non-robustness, especially for large-scale dataset like text-to-image DPMs. We compute the mean and variance of evaluation metrics with 5 independent

Table S4. **Comparison** between optimized timesteps regarding different solvers, datasets, and trajectories.

DDIM	Original	901	801	701	601	501	401	301	201	101	1
	Bedroom	855	758	662	567	477	377	284	190	95	1
	ImageNet	856	757	662	566	472	377	282	189	95	1
	MS-COCO	971	924	734	662	545	417	308	200	99	1
DPM-Solver-2	Original	999	885	754	597	401	194	69	21	5	1
	Bedroom	986	895	778	627	423	205	73	22	5	1
	FFHQ	1001	878	753	629	422	206	73	22	5	1
	CelebA-HQ	989	899	779	623	420	205	73	22	5	1

Table S5. **Comparison** between optimized timesteps on label- and text-conditioned generation regarding different CFG scales with NFE $= 10$.

ImageNet 256	Original	901	801	701	601	501	401	301	201	101	1
	CFG scale $= 3$	856	757	662	566	472	377	282	189	95	1
	CFG scale $= 5$	843	807	789	608	502	391	318	215	102	1
	CFG scale $= 7$	843	931	763	695	566	422	301	207	108	1
MS-COCO 256	Original	901	801	701	601	501	401	301	201	101	1
	CFG scale $= 3$	1012	740	647	527	507	404	302	206	89	1
	CFG scale $= 5$	971	924	734	662	545	417	308	200	99	1
	CFG scale $= 7$	954	935	705	597	581	412	298	180	107	1

sampling. Tab. S2 confirms the robust efficacy of `TimeTuner` convincingly.

C.3. Generation under extreme NFEs using DDIM

In addition to the quantitative comparison upon CD [3] in the manuscript, we also evaluate `TimeTuner` using DDIM sampler on LDM [1] with NFE $= 5$. As demonstrated in Tab. S3, our method achieves dramatic improvements over DDIM, which indeed help reveal the breaking point (*i.e.*, significant performance gain) of our method.

D. Analysis on Optimized Timesteps

Tab. S4 reports some optimized timesteps regarding different solvers, datasets, and trajectories. It is noteworthy that the optimized schedule varies across datasets, and hence is non-transferable. Thanks to the high efficiency of `TimeTuner` to optimize the timesteps for a new dataset (*e.g.*, around 1 hour for NFE $= 10$), our method is still highly applicable.

We also report the optimized timesteps on label- and text-conditioned generation with different CFG scales. As is demonstrated in Tab. S5, the optimized timesteps indeed varies for different scales, but `TimeTuner` can consistently improve the performance, which is reported in the manuscript.

References

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 10684–10695, 2022. 6

[2] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Int. Conf. Learn. Represent.*, 2021. 3

[3] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023. 6