A Proof

A.1 Proof of Proposition 3.1

When $q(x_0)$ is a mixture of Dirac distribution which means that $q(x_0) = \sum_{i=1}^{M} w_i \delta(x - x_i), \sum_{i=1}^{M} w_i = 1$, which has total M components, and when the forward process $q(x_t|x_0)$ is a Gaussian distribution as Eq. (1), the backward process $q(x_s|x_t)$ would be:

$$q(x_s|x_t) = \int q(x_s|x_t, x_0)q(x_0|x_t)dx_0 = \int q(x_s|x_t, x_0)q(x_0)q(x_t|x_0)/q(x_t)dx_0$$
$$= 1/q(x_t) \int q(x_s|x_t, x_0)q(x_0)q(x_t|x_0)dx_0 = 1/q(x_t) \sum_{i=1}^{M} w_i q(x_s|x_t, x_0^i)q(x_t|x_0^i)$$

According to the definition of forward process, the distribution $q(x_t|x_0^i) = \mathcal{N}(x_t|\sqrt{\bar{a}_t}x_0^i, (1 - \bar{a}_t)I)$. Due to the Markov property of forward process, when $t > s$, we have $q(x_s, x_t|x_0)q(x_s|x_0)q(x_t|x_s)$. The term $q(x_s|x_t, x_0)$ would be viewed as a Bayesian posterior resulting from a prior $q(x_s|x_0)$, updated with a likelihood term $q(x_t|x_s)$. And therefore

$$q(x_s|x_t, x_0^i) = N(x_s|\mu_q(x_t, x_0), \Sigma_q(x_t, x_0)I) = \mathcal{N}(x_s|\frac{a_{t|s}\sigma_s^2}{\sigma_t^2}x_t + \frac{a_s\sigma_{t|s}^2}{\sigma_t^2}x_0, \frac{\sigma_s^2\sigma_{t|s}^2}{\sigma_t^2}I)$$

It is easy to prove that, the distribution $q(x_s|x_t)$ is a mixture of Gaussian distribution:

$$q(x_s|x_t) \propto \sum_{i=1}^{M} w_i q(x_s|x_t, x_0^i)q(x_t|x_0^i)$$
$$= \sum_{i=1}^{M} w_i \mathcal{N}(x_t|\sqrt{\bar{a}_t}x_0^i, (1 - \bar{a}_t)I) * \mathcal{N}(x_s|\mu_q(x_t, x_0), \Sigma_q(x_t, x_0))$$

When t is large, s is small, $\sigma_{t|s}^2$ would be large, meaning that the influence of x_0^i would be large.

Secondly, when $q(x_0)$ is a mixture of Gaussian distribution which means that $q(x_0) = \sum_{i=1}^{M} w_i \mathcal{N}(x_0^i|\mu_i, \Sigma_i), \sum_{i=1}^{M} w_i = 1$. For simplicity of analysis, we assume that this distribution is a one-dimensional distribution or that the covariance matrix is a high-dimensional Gaussian distribution with a diagonal matrix $\Sigma_i = \text{diag}_i(\sigma^2)$. Similar to the situation above, for each dimension in the backward process:

$$q(x_s|x_t) = 1/q(x_t) \int q(x_s|x_t, x_0)q(x_0)q(x_t|x_0)dx_0$$
$$= \sum_{i=1}^{M} w_i/q(x_t) \int q(x_s|x_t, x_0)\mathcal{N}(x_0|\mu_i, \Sigma_i)q(x_t|x_0)dx_0$$
$$= \sum_{i=1}^{M} w_i/q(x_t) \int \frac{1}{\sqrt{2\pi}\sigma_q} e^{-\frac{(x_0^i - \mu_q)^2}{\sigma_q^2}} \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(x_0^i - \mu_i)^2}{\sigma_i^2}} \frac{1}{\sqrt{2\pi}\sqrt{1 - \bar{a}_t}} e^{-\frac{(x_t - \sqrt{\bar{a}_t}x_0)^2}{1 - \bar{a}_t}} dx_0$$
$$= \sum_{i=1}^{M} w_i/q(x_t) \int Z_i \frac{1}{\sqrt{2\pi}\sigma_x} e^{-\frac{(x_0^i - \mu_x)^2}{\sigma_x^2}} e^{-\frac{(x_s - \mu(x_t))^2}{\sigma(x_t)}} dx_0$$

And $q(x_s|x_t)$ could be a Gaussian mixture which has M component.

A.2 Proof of Proposition 3.2

Recall that our objective function is to find optimal parameters:

$$\hat{\theta} = \underset{\theta}{\text{argmin}}[\underbrace{Q_{N_c}(\theta)}_{1\times 1}] = \underset{\theta}{\text{argmin}}[\underbrace{g_{N_c}(\theta)^T}_{1\times N} \underbrace{W_{N_c}}_{N\times N} \underbrace{g_{N_c}(\theta)}_{1\times N}] \tag{13}$$

where $g_M(\theta)$ is the moment conditions talked about in Sec. 3.2, W is the weighted Matrix, N_c is the sample size. When solving such problems using optimizers, which are equivalent to the one we used when selecting θ_{GMM} such that $\frac{\partial Q_T(\theta_{GMM})}{\partial \theta} = 0$, and its first derivative:

$$\underbrace{\frac{\partial Q_{N_c}(\theta)}{\partial \theta}}_{d\times 1} = \begin{pmatrix} \frac{\partial Q_{N_c}(\theta)}{\partial \theta_1} \\ \frac{\partial Q_{N_c}(\theta)}{\partial \theta_2} \\ \frac{\partial Q_{N_c}(\theta)}{\partial \theta_3} \end{pmatrix}, \frac{\partial Q_{N_c}(\theta)}{\partial \theta_m} = 2\underbrace{[\frac{1}{N_c}\sum_{i=1}^{N_c}\frac{\partial g(x_i,\theta)}{\partial \theta_m}]^T}_{1\times N}\underbrace{W_{N_c}}_{N\times N}\underbrace{[\frac{1}{N_c}\sum_{i=1}^{M} g(x_i,\theta)]}_{N\times 1} \tag{14}$$

its second derivative (the Hessian matrix):

$$\underbrace{\frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta^2}}_{d\times d} = \begin{pmatrix} \frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta_1 \theta_1} & \frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta_1 \theta_2} & \dots & \frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta_1 \theta_d} \\ \frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta_2 \theta_1} & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta_d \theta_d} \end{pmatrix} \tag{15}$$

$$\begin{aligned} , \frac{\partial^2 Q_{N_c}(\theta)}{\partial \theta_i \theta_j} &= 2[\frac{1}{M}\sum_{i=1}^{N_c}\frac{\partial g(x_i,\theta)}{\partial \theta_i}]^T W_{N_c}[\frac{1}{N_c}\sum_{i=1}^{N_c}\frac{\partial g(x_i,\theta)}{\partial \theta_j}] \\ &\quad + 2[\frac{1}{N_c}\sum_{i=1}^{N_c}\frac{\partial^2 g(x_i,\theta)}{\partial \theta_i \theta_j}]W_{N_c}[\frac{1}{N_c}\sum_{i=1}^{N_c} g(x_i,\theta)]. \end{aligned}$$

By Taylor's expansion of the gradient around optimal parameters θ_0, we have:

$$\begin{aligned} &\frac{\partial Q_{N_c}(\theta_{GMM})}{\partial \theta} - \frac{\partial Q_{N_c}(\theta_0)}{\partial \theta} \approx \frac{\partial^2 Q_{N_c}(\theta_0)}{\partial \theta \partial \theta^T}(\theta_{GMM} - \theta) \\ &\longmapsto (\theta_{GMM} - \theta) \approx -(\frac{\partial^2 Q_{N_c}(\theta_0)}{\partial \theta \partial \theta^T})^{-1}\frac{\partial Q_{N_c}(\theta_0)}{\partial \theta}. \end{aligned} \tag{16}$$

Consider one element of the gradient vector $\frac{\partial Q_{N_c}(\theta_0)}{\partial \theta_m}$

$$\frac{\partial Q_{N_c}(\theta_0)}{\partial \theta_m} = 2[\underbrace{\frac{1}{M}\sum_{i=1}^{N_c}\frac{\partial g(x_i,\theta_0)}{\partial \theta_m}}_{\xrightarrow{p}\mathbb{E}(\frac{\partial g(x_i,\theta_0)}{\partial \theta_m})=\Gamma_{0,m}}]^T \underbrace{W_{N_c}}_{\xrightarrow{p}W}[\underbrace{\frac{1}{N_c}\sum_{i=1}^{N_c} g(x_i,\theta_0)}_{\xrightarrow{p}\mathbb{E}[g(x_i,\theta_0)]=0}]. \tag{17}$$

Consider one element of the Hessian matrix $\frac{\partial^2 Q_{N_c}(\theta_0)}{\partial \theta_i \partial \theta_j^T}$

$$\begin{aligned} \frac{\partial^2 Q_{N_c}(\theta_0)}{\partial \theta_i \partial \theta_j^T} &= 2[\frac{1}{N_c}\sum_{i=1}^{N_c}\frac{\partial g(x_i,\theta_0)}{\partial \theta_i}]^T W_{N_c}[\frac{1}{N_c}\sum_{i=1}^{N_c}\frac{\partial g(x_i,\theta_0)}{\partial \theta_j}] \\ &\quad + 2[\frac{1}{N_c}\sum_{i=1}^{N_c}\frac{\partial^2 g(x_i,\theta_0)}{\partial \theta_i \theta_j}]^T W_{N_c}[\frac{1}{N_c}\sum_{i=1}^{N_c} g(x_i,\theta_0)] \xrightarrow{p} 2\Gamma_{0,i}^T W \Gamma_{0,j}. \end{aligned} \tag{18}$$

Therefore, it is easy to prove that $\theta_{GMM} - \theta_0 \xrightarrow{p} 0$, and uses law of large numbers we could obtain,

$$\sqrt{T}\frac{\partial Q_T(\theta_0)}{\partial \theta_m} = 2[\underbrace{\frac{1}{M}\sum_{i=1}^{M}\frac{\partial g(x_i,\theta_0)}{\partial \theta_m}}_{\xrightarrow{p}\mathbb{E}(\frac{\partial g(X,\theta_0)}{\partial \theta_m})=\Gamma_{0,m}}]^T \underbrace{W_M}_{\xrightarrow{p}W}[\underbrace{\frac{1}{\sqrt{T}}\sum_{i=1}^{M} g(x_i,\theta_0)}_{\xrightarrow{d}\mathcal{N}(0,\mathbb{E}(g(X,\theta_0)g(X,\theta_0)^T))}] \xrightarrow{d} 2\Gamma_{0,m}^T W \mathcal{N}(0,\Phi_0), \tag{19}$$

therefore, we have

$$\begin{aligned} \sqrt{T}(\theta_{GMM} - \theta_0) &\approx -(\frac{\partial^2 Q_T(\theta_0)}{\partial \theta \partial \theta^T})^{-1}\sqrt{T}\frac{\partial Q_T(\theta_0)}{\partial \theta} \\ &\xrightarrow{d} \mathcal{N}(0,(\Gamma_0^T W \Gamma_0)^{-1}\Gamma_0^T W \Phi_0 W \Gamma_0 (\Gamma_0^T W \Gamma_0)^{-1}). \end{aligned} \tag{20}$$

When the number of parameters d equals the number of moment conditions N, Γ_0 becomes a (nonsingular) square matrix, and therefore,

$$\begin{aligned}\sqrt{T}(\theta_{GMM}-\theta_0) &\xrightarrow{d} \mathcal{N}(0,(\Gamma_0^T W\Gamma_0)^{-1}\Gamma_0^T W\Phi_0 W\Gamma_0(\Gamma_0^T W\Gamma_0)^{-1}) \\ &= \mathcal{N}(0,\Gamma_0^{-1}W^{-1}(\Gamma_0^T)^{-1}\Gamma_0^T W\Phi_0 W\Gamma_0\Gamma_0^{-1}W^{-1}(\Gamma_0^T)^{-1}) \\ &= \mathcal{N}(0,\Gamma_0^{-1}\Phi_0(\Gamma_0^T)^{-1}),\end{aligned} \tag{21}$$

which means that the foregoing observation suggests that the selection of W_{N_c} has no bearing on the asymptotic variance of the GMM estimator. Consequently, it implies that regardless of the specific method employed to determine W_{N_c}, provided the moment estimates are asymptotically consistent, W_{N_c} serves as the optimal weight matrix, even when dealing with small samples.

The moments utilized for fitting via the generalized method of moments in each step are computed based on the noise network's first n-th order. To ensure adherence to the aforementioned proposition, it is necessary to assume that the n-th order of the noise network converges with probability to $\mathbb{E}_{q(x_0|x_t)}[\text{diag}(\epsilon \otimes^{n-1} \epsilon)]$, details in Appendix. B. Consequently, the n-th order moments derived from the noise networks converge with probability to the true moments. Therefore, any choice of weight matrix is optimal.

B Calculation of the first order moment and higher order moments

Suppose the forward process is a Gaussian distribution same with the Eq. (1) as $q(x_t|x_s) = N(x_t|a(t)x_{t-1},\sigma(t)I)$.

And let $1 \geq t > s \geq 0$ always satisfy, $q(x_t|x_s) = N(x_t|a_{t|s}x_s,\beta_{t|s}I)$, where $a_{t|s} = a_t/a_s$ and $\beta_{t|s} = \sigma_t^2 - a_{t|s}^2\sigma_s^2$, $\sigma_s = \sqrt{1-a(s)}$,$\sigma_t = \sqrt{1-a(t)}$. It's easy to prove that the distribution $q(x_t|x_0)$, $q(x_s|x_t,x_0)$ are also a Gaussian distribution [19]. Therefore, the mean of x_s under the measure $q(x_s|x_t)$ would be

$$\begin{aligned}\mathbb{E}_{q(x_s|x_t)}[x_s] &= \mathbb{E}_{q(x_0|x_t)}E_{q(x_s|x_t,x_0)}[x_s] \\ &= \mathbb{E}_{q(x_0|x_t)}[\frac{1}{a_{t|s}}(x_t - \frac{\beta_{t|s}}{\sigma_t}\epsilon_t)] \\ &= \frac{1}{a_{t|s}}(x_t - \frac{\beta_{t|s}}{\sigma_t}\mathbb{E}_{q(x_0|x_t)}[\epsilon_t]).\end{aligned} \tag{22}$$

And for the second order central moment $\text{Cov}_{q(x_s|x_t)}[x_s]$, we use the total variance theorem, refer to [2], and similar with [2], we only consider the diagonal covariance.

$$\begin{aligned}\text{Cov}_{q(x_s|x_t)}[x_s] &= \mathbb{E}_{q(x_0|x_t)}\text{Cov}_{q(x_s|x_t,x_0)}[x_s] + \text{Cov}_{q(x_0|x_t)}\mathbb{E}_{q(x_s|x_t,x_0)}[x_s] \\ &= \lambda_t^2 I + \text{Cov}_{q(x_0|x_t)}\tilde{\mu}(x_n,\mathbb{E}_{q(x_0|x_n)}[x_0]) \\ &= \lambda_t^2 I + \frac{a_s\beta_{t|s}^2}{\sigma_t^4}\text{Cov}_{q(x_0|x_t)}[x_0] \\ &= \lambda_t^2 I + \frac{a_{s|0}\beta_{t|s}^2}{\sigma_t^4}\frac{\sigma_t^2}{a_{t|0}}\text{Cov}_{q(x_0|x_t)}[\epsilon_t] \\ &= \lambda_t^2 I + \frac{\beta_{t|s}^2}{\sigma_t^2 a_{t|s}}(\mathbb{E}_{q(x_0|x_t)}[\epsilon_t \odot \epsilon_t] - \mathbb{E}_{q(x_0|x_t)}[\epsilon_t] \odot \mathbb{E}_{q(x_0|x_t)}[\epsilon_t]),\end{aligned} \tag{23}$$

since the higher-order moments are diagonal matrix, we use $\text{diag}(M)$ to represent the diagonal elements that have the same dimensions as the first-order moments, such as $\text{diag}(x_s \otimes x_s) = \text{Cov}_{q(x_s|x_t)}[x_s]$ and $\text{diag}(x_s \otimes x_s \otimes x_s) = \hat{M}_3$ have the same dimensions as x_s

Moreover, for the diagonal elements of the third-order moments, we have $\mathbb{E}_{q(x_s|x_t)}[\text{diag}(x_s \otimes x_s \otimes x_s)] = \mathbb{E}_{q(x_0|x_t)}\mathbb{E}_{q(x_s|x_t,x_0)}[x_s \odot x_s \odot x_s]$, we could use the fact that $\mathbb{E}_{q(x_s|x_t,x_0)}[(x_s - \mu(x_t,x_0)) \odot$

$(x_s - \mu(x_t, x_0)) \odot (x_s - \mu(x_t, x_0))] = 0$

$$\hat{M}_3 = \mathbb{E}_{q(x_s|x_t)}[\text{diag}(x_s \otimes x_s \otimes x_s)] = \mathbb{E}_{q(x_0|x_t)}\mathbb{E}_{q(x_s|x_t,x_0)}[\text{diag}(x_s \otimes x_s \otimes x_s)] = \tag{24}$$

$$\underbrace{[(\frac{a_{t|s}\sigma_s^2}{\sigma_t^2})^3 \text{diag}(x_t \otimes x_t \otimes x_t) + 3\lambda_t^2 \frac{a_{t|s}\sigma_s^2}{\sigma_t^2} x_t]}_{\text{Constant term}}$$

$$+ \underbrace{[\frac{3a_{t|s}^2\sigma_s^4 a_{s|0}^2 \beta_{t|s}^2}{\sigma_t^8}(\text{diag}(x_t \otimes x_t)) + \frac{a_{s|0}\beta_{t|s}}{\sigma_t^2} I] \odot \mathbb{E}_{q(x_0|x_t)}[x_0]}_{\text{Linear term in } x_0}$$

$$+ \underbrace{3\frac{a_{t|s}\sigma_s^2}{\sigma_t^2}(\frac{a_{s|0}\beta_{t|s}}{\sigma_t^2})^2 x_t \odot \mathbb{E}_{q(x_0|x_t)}[\text{diag}(x_0 \otimes x_0)]}_{\text{Quadratic term in } x_0} + \underbrace{(\frac{a_{s|0}\beta_{t|s}}{\sigma_t^2})^3 \mathbb{E}_{q(x_0|x_t)}[\text{diag}(x_0 \otimes x_0 \otimes x_0)]}_{\text{Cubic term in } x_0},$$

What's more, for the three-order moment and higher-order moment, we only consider the diagonal elements, and therefore all outer products can be transformed into corresponding element multiplications and we have:

$$\begin{aligned}
\mathbb{E}_{q(x_0|x_t)}[\text{diag}(x_0 \otimes^2 x_0)] &= \frac{1}{\alpha^{\frac{3}{2}}(t)} \mathbb{E}_{q(x_0|x_t)}[\text{diag}((x_t - \sigma(t)\epsilon) \otimes^2 (x_t - \sigma(t)\epsilon))] \\
&= \frac{1}{\alpha^{\frac{3}{2}}(t)} \mathbb{E}_{q(x_0|x_t)}[\text{diag}(x_t \otimes^2 x_t - 3\sigma(t)(x_t \otimes x_t) \otimes \epsilon \\
&+ 3\sigma^2(t) x_t \otimes (\epsilon \otimes \epsilon) - \sigma^3(t)(\epsilon \otimes^2 \epsilon))] \\
&= \frac{1}{\alpha^{\frac{3}{2}}(t)} [\mathbb{E}_{q(x_0|x_t)}[x_t \odot^2 x_t] - 3\sigma(t)(x_t \odot x_t) \odot \mathbb{E}_{q(x_0|x_t)}[\epsilon] + \\
&+ 3\sigma^2(t) x_t \mathbb{E}_{q(x_0|x_t)}[\epsilon \odot \epsilon] - \sigma^3(t)[\epsilon \odot^2 \epsilon]],
\end{aligned} \tag{25}$$

Therefore, when we need to calculate the third-order moment, we only need to obtain $\mathbb{E}_{q(x_0|x_t)}[\epsilon_t], \mathbb{E}_{q(x_0|x_t)}[\epsilon_t \odot \epsilon_t]$ and $\mathbb{E}_{q(x_0|x_t)}[\epsilon_t \odot \epsilon_t \odot \epsilon_t]$. Similarly, when we need to calculate the n-order moment, we will use $\mathbb{E}_{q(x_0|x_t)}[\epsilon_t], ..., \mathbb{E}_{q(x_0|x_t)}[\epsilon_t \odot^{n-1} \epsilon_t]$. Bao et al. [2] put forward using a sharing network and using the MSE loss to estimate the network to obtain the above information about different orders of noise.

C Modeling reverse transition kernel via exponential family

Analysis in Sec. 3.1 figures out that modeling reverse transition kernel via Gaussian distribution is no longer sufficient in fast sampling scenarios. In addition to directly proposing the use of Gaussian Mixture for modeling, we also analyze in principle whether there are potentially more suitable distributions i.e., the feasibility of using them for modeling.

We would turn back to analyzing the original objective function of DPMs to find a suitable distribution. The forward process $q(x_t|x_s) = N(x_t|a_{t|s}x_s, \beta_{t|s}I)$, consistent with the definition in Appendix. B. And DPMs' goal is to optimize the modeled backward process parameters to maximize the variational bound L in Ho et al. [12]. And the ELBO in Ho et al. [12] can be re-written to the following formula:

$$L = D_{\text{KL}}(q(x_T)||p(x_T)) + \mathbb{E}_q[\sum_{t \geq 1} D_{\text{KL}}(q(x_s|x_t)||p(x_s|x_t))] + H(x_0), \tag{26}$$

where $q_t \doteq q(x_t)$ is the true distribution and $p_t \doteq p(x_t)$ is the modeled distribution, and the minimum problem could be transformed into a sub-problem, proved in Bao et al. [3]:

$$\min_{\{\theta\}} L \Leftrightarrow \min_{\{\theta_{s|t}\}_{t=1}^T} D_{\text{KL}}(q(x_s|x_t)||p_{\theta_{s|t}}(x_s|x_t)). \tag{27}$$

We have no additional information besides when the reverse transition kernel is not Gaussian. But Lemma. C.3 proves that when the reverse transition kernel $p_{\theta_{s|t}}(x_s|x_t)$ is exponential family $p_{\theta_t}(x_s|x_t) = p(x_t, \theta_{s|t}) = h(x_t)\exp\Big(\theta_{s|t}^T t(x_t) - \alpha(\theta_{s|t})\Big)$, solving the sub-problem Eq. (27) equals

to solve the following equations, which is to match moments between the modeled distribution and true distribution:

$$\mathbb{E}_{q(x_s|x_t)}[t(x_s)] = \mathbb{E}_{p(x_t,\theta_{s|t})}[t(x_s)]. \tag{28}$$

When $t(x) = (x, .., x^n)^T$, solving Eq.(28) equals to match the moments of true distribution and modeled distribution.

Meanwhile, Gaussian distribution belongs to the exponential family with $t(x) = (x, x^2)^T$ and $\theta_t = (\frac{\mu_t}{\sigma_t^2}, \frac{-1}{2\sigma_t^2})^T$, details in Lemma. C.2. Therefore, when modeling the reverse transition kernel as Gaussian distribution, the optimal parameters are that make its first two moments equal to the true first two moments of the real reverse transition kernel $q(x_s|x_t)$, which is consistent with the results in Bao et al. [3] and Bao et al. [2].

The aforementioned discussion serves as a motivation to acquire higher-order moments and identify a corresponding exponential family, which surpasses the Gaussian distribution in terms of complexity. However, proposition C.1 shows that finding such exponential family distribution with higher-order moments is impossible.

Proposition C.1 (Infeasibility of exponential family with higher-order moments.)**.** *Given the first n-th order moments. It's non-trivial to find an exponential family distribution for* $\min D_{\mathrm{KL}}(q||p)$ *when n is odd. And it's hard to solve* $\min D_{\mathrm{KL}}(q||p)$ *when n is even.*

C.1 Proof of Proposition C.1

Lemma C.2. *(Gaussian Distribution belongs to Exponential Family). Gaussian distribution* $p(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ *is exponential family with* $t(x) = (x, x^2)^T$ *and* $\theta = (\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2})^T$

Proof. For simplicity, we only prove one-dimensional Gaussian distribution. We could obtain:

$$\begin{aligned}
p(x) &= \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \\
&= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2\sigma^2}(x^2 - 2\mu x + \mu^2)\right) \\
&= \exp\left(\log\left(2\pi\sigma^2\right)^{-1/2}\right)\exp\left(-\frac{1}{2\sigma^2}(x^2 - 2\mu x) - \frac{\mu^2}{\sigma^2}\right) \\
&= \exp\left(\log\left(2\pi\sigma^2\right)^{-1/2}\right)\exp\left(-\frac{1}{2\sigma^2}(-2\mu \quad 1)(x \quad x^2)^T - \frac{\mu^2}{\sigma^2}\right) \\
&= \exp\left((\frac{\mu}{\sigma^2} \quad \frac{-1}{2\sigma^2})(x \quad x^2)^T - (\frac{\mu^2}{2\sigma^2} + \frac{1}{2}\log\left(2\pi\sigma^2\right))\right),
\end{aligned} \tag{29}$$

where $\theta = (\frac{\mu}{\sigma^2}, \frac{-1}{2\sigma^2})^T$ and $t(x) = (x, x^2)^T$ □

Lemma C.3. *(The Solution for Exponential Family in Minimizing the KL Divergence). Suppose that $p(x)$ belongs to exponential family $p(x, \theta) = h(x)\exp\left(\theta^T t(x) - \alpha(\theta)\right)$, and the solution for minimizing the $E_q[\log p]$ is $E_q[t(x)] = E_{p(x,\theta)}[t(x)]$.*

Proof. An exponential family $p(x, \eta) = h(x)\exp\left(\eta^T t(x) - \alpha(\eta)\right) \propto f(x, \eta) = h(x)\exp\left(\eta^T t(x)\right)$ with log-partition $\alpha(\eta)$. And we could obtain its first order condition on $E_q[\log p]$ as:

$$\nabla_\eta \log f(x, \eta) = \nabla_\eta(\log h(x) + \eta^T t(x)) = t(x) \tag{30}$$

$$\begin{aligned}
\nabla_\eta \alpha(\eta) &= \nabla_\eta \log\left(\int f(x, \eta)dx\right) = \frac{\int \nabla_\eta f(x, \eta)dx}{\int f(x, \eta)dx} \\
&= e^{-\alpha(\eta)}\int t(x)f(x, \eta)dx = \int t(x)p(x, \eta)dx = \mathbb{E}_{p(x,\eta)}[t(x)]
\end{aligned} \tag{31}$$

In order to minimize the $D_{\mathrm{KL}}(q||p) = \int q \log(q/p) = -\mathbb{E}_q[\log p]$, we have:

$$\begin{aligned}
&\mathbb{E}_q[\log p] = \int dq \log(h(x)) + \int dq(\eta^T t(x) - \alpha(\eta) \\
&\implies \frac{\partial}{\partial \eta}\mathbb{E}_q[\log p] = \int dq[\frac{\partial}{\partial \eta}(\eta^T x - \alpha(\eta))] = 0 \\
&\implies \int dq(x - \mathbb{E}_{p(x,\eta)}[t(x)]) = \mathbb{E}_q[t(x)] - \mathbb{E}_{p(x,\eta)}[t(x)] = 0 \\
&\implies \mathbb{E}_q[t(x)] = \mathbb{E}_{p(x,\eta)}[t(x)]
\end{aligned}$$

And for the second-order condition, we have the:

$$\begin{aligned}
\frac{\partial^2}{\partial \eta^2}\alpha(\eta) &= \frac{\partial}{\partial \eta}\int dp(x,\eta)t(x) \\
&= \int \frac{\partial}{\partial \eta}h(x)\exp\big(\eta^T t(x) - \alpha(\eta)\big)t(x)dx \\
&= \int h(x)t(x)\frac{\partial}{\partial \eta}\exp\big(\eta^T t(x) - \alpha(\eta)\big)dx \\
&= \int h(x)t(x)\exp\big(\eta^T t(x) - \alpha(\eta)\big)dx(t(x) - \mathbb{E}_p[t(x)]) \\
&= \int p(x,\eta)dx(t^2(x) - t(x)\mathbb{E}_p[t(x)]) \\
&= \mathbb{E}_{p(x,\eta)}[t^2(x)] - \mathbb{E}_{p(x,\eta)}[t(x)]^2 = Cov_{p(x,\eta)}[t(x)] \geq 0
\end{aligned} \tag{32}$$

Therefore, the second-order condition for the cross entropy would be:

$$\begin{aligned}
\frac{\partial^2}{\partial \eta^2}\mathbb{E}_q[\log p] &= \frac{\partial}{\partial \eta}(\mathbb{E}_q[t(x)] - \mathbb{E}_{p(x,\eta)}[t(x)]) \\
&= -\int \frac{\partial}{\partial \eta}p(x,\eta)t(x)dx \\
&= -\frac{\partial^2}{\partial \eta^2}\alpha(\eta) = -Cov_{p(x,\eta)}[t(x)] \leq 0
\end{aligned} \tag{33}$$

When we assume that the backward process is Gaussian, the solution to Eq. (27) equals to match the moment of true distribution and modeled distribution $\mu = E_q[x], \Sigma = Cov_q[x]$. □

Lemma C.4. *(Infeasibility of the exponential family with higher-order moments). Suppose given the first N-th order moments $M_i, i = 1, .., N$ and modeled p as an exponential family. It is nontrivial to solve the minimum problem $E_q[\log p]$ when N is odd and it's difficult to solve when N is even.*

Proof. While given the mean, covariance, and skewness of the data distribution, assume that we could find an exponential family that minimizes the KL divergence, so that the distribution would satisfy the following form:

$$\begin{aligned}
&L(p,\hat{\lambda}) = D_{\mathrm{KL}}(q||p) - \hat{\lambda}^T(\int pt - m) \Rightarrow \frac{\partial}{\partial p}L(p,\hat{\lambda}) = \log\frac{p(x)}{h(x)} + 1 - \hat{\lambda}^T t = 0 \\
&\Rightarrow p(x) = h(x)\exp\Big(\hat{\lambda}^T t - 1\Big)
\end{aligned} \tag{34}$$

where, $t(x) = (x, x^2, x^3)$, $p = h(x)\exp\big(\lambda_0 + \lambda_1 x + \lambda_2 x^2 + \lambda_3 x^3\big)$ and $\int dp x^3 = M_3$. However, when λ_3 is not zero, $\int p = \infty$ and density can't be normalized. The situation would be the same given an odd-order moment.

Similarly, given a more fourth-order moment, we could derive that $\lambda_3 = 0$ above, and we should solve an equation $\int dp x^4 = M_4$ and $p = h(x)\exp\big(\lambda_0 + \lambda_1 x + \lambda_2 x^2 + \lambda_4 x^4\big)$. Consider such function:

$$Z(\lambda) = \int_{-\infty}^{\infty} dx \exp\big(-x^2 - \lambda x^4\big), \lambda > 0 \tag{35}$$

When $\lambda \longrightarrow 0$, we could obtain $\lim_{\lambda \to 0} Z(\lambda) = \sqrt{\pi}$ For other cases, the lambda can be expanded and then integrated term by term, which gives $Z(\lambda) \sim \sum_{n=0}^{\infty} \frac{(-\lambda)^n}{n!} \Gamma(2n + 1/2)$, but this function However, the radius of convergence of this level is 0, so when the λ takes other values, we need to propose a reasonable expression for the expansion after the analytic extension. Therefore, for solving the equation $\int dp x^4 = M_4$, there is no analytical solution first, and the numerical solution also brings a large computational effort. □

D More information about Fig. 1 and Fig. 2

D.1 Experiment in Toy-data

To illustrate the effectiveness of our method, we first compare the results of different solvers on one-dimensional data.

The distribution of our toy-data is $q(x_0) = 0.4\mathcal{N}(-0.4, 0.12^2) + 0.6\mathcal{N}(0.3, 0.05^2)$ and we define our solvers in each step as $p(x_s|x_t) = \frac{1}{3}\mathcal{N}(\mu_t^{(1)}, \sigma_t^2) + \frac{2}{3}\mathcal{N}(\mu^{(1)}t, \sigma_t^2)$ with vectors $\mu_t^{(1)}$, $\mu_t^{(2)}$ and σ_t^2, which can not overfit the ground truth.

We then train second and third-order noise networks on the one dimension Gaussian mixture whose density is multi-modal. We use a simple MLP neural network with Swish activation [31].

Moreover, we experiment with our solvers in 8-Gaussian. The result is shown in Table 2. GMS outperforms Extended AnalyticDPM (SN-DDPM) [2] as presented in Tab. 2, with a bandwidth of $1.05\sigma L^{-0.25}$, where σ is the standard deviation of data and L is the number of samples.

Table 2: **Comparison with Extended Analytic-DPMs w.r.t. Likelihood $\mathbb{E}_q[\log p_\theta(x)] \uparrow$ on 8-Gaussian.** GMDDPM outperforms Extended AnalyticDPMs.

	8-GAUSSIAN			
# K	5	10	20	40
SN-DDPM	-0.7885	0.0661	0.0258	0.1083
GMDDPM	**-0.6304**	**0.0035**	**0.0624**	**0.1127**

D.2 Experiment in Fig. 2

In this section, we will provide a comprehensive explanation of the procedures involved in computing the discrepancy between two third-order moment calculation methods, as depicted in Fig. 2.

The essence of the calculation lies in the assumption that the reverse transition kernel follows a Gaussian distribution. By employing the following equations (considering only the diagonal elements of higher-order moments), we can compute the third-order moment using the first two-order moments:

$$\mathbb{E}_{q(x_{t_{i-1}}|x_{t_i})}[x_{t_{i-1}} \odot x_{t_{i-1}} \odot x_{t_{i-1}}]_{\mathrm{G}} \doteq M_G = \mu \odot \mu \odot \mu + 3\mu \odot \Sigma, \tag{36}$$

where μ is the first-order moment and Σ is the diagonal elements of second order moment, which can be calculated by the Eq. (22) and Eq. (23). Meanwhile, we can calculate the estimated third-order moment $\hat{M}_3$ by Eq. (24).

We use the pre-trained noise network from Ho et al. [12] and the second-order noise network form Bao et al. [2] and train the third-order noise network in CIFAR10 with the linear noise schedule.

Given that all higher-order moments possess the same dimension as the first-order moment μ, we can directly compare the disparity between different third-order moment calculation methods using the Mean Squared Error (MSE).

Thus, to quantify the divergence between the reverse transition kernel $q(x_s|x_t)$ and the Gaussian distribution, we can utilize the following equation:

$$\mathrm{D}_{s|t} = \log\left(\mathbb{E}_{q(x_s|x_t)}[x_s \odot x_s \odot x_s]_{\mathrm{G}} - \hat{M}_3\right)^2, \tag{37}$$

where $\hat{M}_3$ is obtained via Eq. (24), and we can start at different time step t and choose a corresponding s to calculate the $\mathrm{D}_{s|t}$ and draw different time step and step size $t - s$ and we can derive Fig. 2.

E Experimental details

E.1 More discussion on weight of Gaussian mixture

From Proposition 3.2, we know that when the number of parameters in the Gaussian mixture equals the number of moment conditions, any choice of weight matrix is optimal. Therefore, we will discuss the choice of parameters to optimize in this section. As we have opted for a Gaussian mixture with two components $q(x_s|x_t) = \omega_1\mathcal{N}(\mu_{s|t}^{(1)}, \Sigma_{s|t}^{(1)}) + \omega_2\mathcal{N}(\mu_{s|t}^{(2)}, \Sigma_{s|t}^{(2)})$ as our foundational solvers, there exist five parameters (considered scalar, with the vector cases being analogous) available for optimization.

Our primary focus is on optimizing the mean and variance of the two components, as optimizing the weight term would require solving the equation multiple times. Additionally, we have a specific requirement that our Gaussian mixture can converge to a Gaussian distribution at the conclusion of optimization, particularly when the ground truth corresponds to a Gaussian distribution. In Tab. 3, we show the result of different choices of parameters in the Gaussian mixture.

Table 3: Results among different parameters in CIFAR10 (LS), the number of steps is 50. The weight of Gaussian mixture is $\omega_1 = \frac{1}{3}$ and $\omega_2 = \frac{2}{3}$

	$\mu_{s\|t}^{(1)}$,$\mu_{s\|t}^{(2)}$,$\Sigma_{s\|t}$	$\mu_{s\|t}^{(1)}$,$\Sigma_{s\|t}^{(1)}$,$\Sigma_{s\|t}^{(2)}$	$\mu_{s\|t}$,$\Sigma_{s\|t}^{(1)}$,$\Sigma_{s\|t}^{(2)}$
CIFAR10 (LS)	**4.17**	10.12	4.22

When a parameter is not accompanied by a superscript, it implies that both components share the same value for that parameter. On the other hand, if a parameter is associated with a superscript, and only one moment contains that superscript, it signifies that the other moment directly adopts the true value for that parameter.

It is evident that the optimization of the mean value holds greater significance. Therefore, our subsequent choices for optimization are primarily based on the first set of parameters $\mu_{s|t}^{(1)}$,$\mu_{s|t}^{(2)}$,$\Sigma_{s|t}$. Another crucial parameter to consider is the selection of weights ω_i. In Tab. 4, we show the result while changing the weight of the Gaussian mixture and the set of weight $\omega_1 = \frac{1}{3}, \omega_2 = \frac{1}{2}$ performs best among different weight.

Table 4: Results among different weight choices in CIFAR10 (LS), the number of steps is 50.

	$\omega_1 = \frac{1}{100}, \omega_2 = \frac{99}{100}$	$\omega_1 = \frac{1}{5}, \omega_2 = \frac{4}{5}$	$\omega_1 = \frac{1}{3}, \omega_2 = \frac{2}{3}$	$\omega_1 = \frac{1}{2}, \omega_2 = \frac{1}{2}$
CIFAR10 (LS)	4.63	4.20	**4.17**	4.26

E.2 Details of pre-trained noise networks

In Table 5, we list details of pre-trained noise prediction networks used in our experiments.

Table 5: Details of noise prediction networks used in our experiments. LS means the linear schedule of $\sigma(t)$ [12] in the forward process of discrete time step (see Eq. (1)). CS means the cosine schedule of $\sigma(t)$ [28] in the forward process of discrete timesteps (see Eq. (1)).

	# TIMESTEPS N	NOISE SCHEDULE	OPTIMIZER FOR GMM
CIFAR10 (LS)	1000	LS	ADAN
CIFAR10 (CS)	1000	CS	ADAN
IMAGENET 64X64	4000	CS	ADAN

E.3 Details of the structure of the extra head

In Table 6, we list structure details of NN_1, NN_2 and NN_3 of prediction networks used in our experiments.

Table 6: NN_1 represents noise prediction networks and NN_2, NN_3 represent networks for estimating the second- and the third-order of noise, which used in our experiments. Conv denotes the convolution layer. Res denotes the residual block.

	NN_1	NN_2 (NOISE)	NN_3 (NOISE)
CIFAR10 (LS)	NONE	CONV	RES+CONV
CIFAR10 (CS)	NONE	CONV	RES+CONV
IMAGENET 64X64	NONE	RES+CONV	RES+CONV

E.4 Training Details

We use a similar training setting to the noise prediction network in [28] and [2]. On all datasets, we use the ADAN optimizer [39] with a learning rate of 10^{-4}; we train 2M iterations in total for a higher order of noise network; we use an exponential moving average (EMA) with a rate of 0.9999. We use a batch size of 64 on ImageNet 64X64 and 128 on CIFAR10. We save a checkpoint every 50K iterations and select the models with the best FID on 50k generated samples. Training one noise network on CIFAR10 takes about 100 hours on one A100. Training on ImageNet 64x64 takes about 150 hours on one A100.

E.5 Details of Parameters of Optimizer in Sampling

In Tab. 7, we list details of the learning rate, learning rate schedule, and warm-up steps for different experiments.

Table 7: Details of Parameters of Optimizer used in our experiments. lr Schedule means the learning rate schedule. min lr means the minimum learning rate while using the learning rate schedule, ι_t is a function with the second order growth function of sampling steps t.

	LEARNING RATE	LR SCHEDULE	MIN LR	WARM-UP STEPS
CIFAR10	MAX(0.16-ι_t*0.16,0.12)	COS	0.1	18
IMAGENET 64×64	MAX(0.1-ι_t*0.1,0.06)	COS	0.04	18

where COS represents the cosine learning rate schedule [5]. We find that the cosine learning rate schedule works best. The cos learning rate could be formulated as follows:

$$\alpha_{i+1} = \begin{cases} \frac{i}{I_w}\alpha_i & \text{if} \quad i \leq I_w \\ \max((0.5\cos\left(\frac{i-I_w}{I-I_w}\pi\right)+1)\alpha_t, \alpha_{\min}) & \text{else} \end{cases} \tag{38}$$

where, a_t is the learning rate after t steps, I_w is the warm-up steps, $\alpha_{\min}$ is the minimum learning rate, I is the total steps.

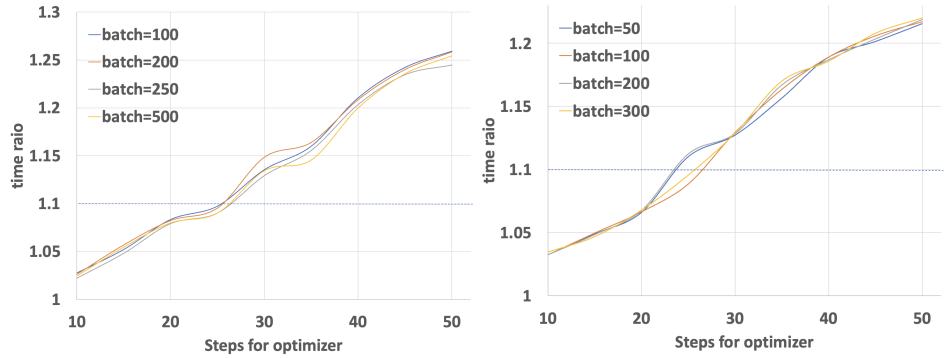
E.6 Details of memory and time cost

In Table 8, we list the memory of models (with the corresponding methods) used in our experiments. The extra memory cost higher-order noise prediction network is negligible.

Table 8: Model size (MB) for different models. The model of SNDDPM denotes the model that would predict noise and the square of noise; The model of GMDDPM denotes the model that would predict noise, the square of noise, and the third power of noise.

	NOISE PREDICTION NETWORK (ALL BASELINES)	NOISE & SN PREDICTION NETWORKS SNDDPM	NOISE & SN PREDICTION NETWORKS (GMDDPM)
CIFAR10 (LS)	50.11 MB	50.11 MB	50.52 MB (+0.8%)
CIFAR10 (CS)	50.11 MB	50.11 MB	50.52 MB (+0.8%)
IMAGENET 64×64	115.46	115.87 MB	116.28 (+0.7%)

In Fig. 5, we report the time ratio on CIFAR10 and ImageNet, which is defined by the time GMS required for one iteration divided by the time Extended AnalyticDPM (SN-DDPM) for one step. The optimizer is ADAN, and ADAM would be faster than ADAN[39]. We could see that for CIFAR10 or ImageNet 64×64, 25 steps of ADAN to estimate the parameters of Gaussian Mixture requires 10% extra time to compute, 40 steps require about 20% extra time, therefore, we would make other solvers to run 10% more steps (sampling steps) than GMS to keep the same computation time.



(a) time cost on CIFAR10(LS) (b) time cost on ImageNet 64X64

Figure 5: Generated samples on CIFAR10 (LS), using ADAN (40 steps) to solve the Gaussian mixture.

Since many parts in the GMS introduce additional computational effort, Fig. 6 reports the distribution of the additional computational effort of the GMS and Extended AnalyticDPM (SN-DDPM) relative to the DDPM, assuming that the computational time of the network predicting the noise be unit one.

It should be emphasized that the additional time required serves purely as a reference, as our observation indicates that the majority of pixels do not necessitate optimization, and employing a Gaussian distribution is satisfactory. Consequently, when we establish a threshold value and the disparity between the reverse transition kernel and Gaussian surpasses the threshold pixel prior to optimization, we can conserve approximately 4% of computational resources without compromising the quality of the results.

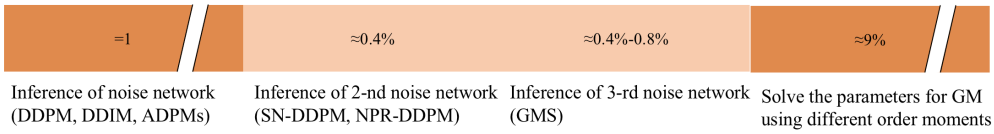


Figure 6: Time cost distribution for GMS.

E.7 Additional results with same calculation cost

Since GMS will cost more computation in the process of fitting the Gaussian mixture, we use the maximum amount of computation required (i.e., an additional 10% of computation is needed) for comparison, and for a fair comparison, we let the other solvers take 10% more sampling steps.

Table 9: **Fair comparison with competitive SDE-based solvers w.r.t. FID score ↓ on CIFAR10 and ImageNet 64×64 with the same computation cost.** Our GMS still outperforms existing SDE-based solvers with the same (maximum) computation cost. SN-DDPM denotes Extended AnalyticDPM from Bao et al. [2]. The number of sampling steps for the GMS is indicated within parentheses, while for other solvers, it is represented outside of parentheses.

	CIFAR10 (LS)							
# TIMESTEPS K	11(10)	22(20)	28(25)	44(40)	55(50)	110(100)	220(200)	1000(1000)
SN-DDPM*	17.56	7.74	6.76	4.81	4.23	3.60	3.20	3.65
GMS (OURS)	**17.43**	**7.18**	**5.96**	**4.52**	**4.16**	**3.26**	**3.01**	**2.76**

	CIFAR10 (CS)					
# TIMESTEPS K	11(10)	28(25)	55(50)	110(100)	220(200)	1000(1000)
SN-DDPM*	**13.26**	5.61	4.13	3.69	3.83	**4.07**
GMS (OURS)	13.80	**5.48**	**4.00**	**3.46**	**3.34**	4.23

	IMAGENET 64 × 64					
# TIMESTEPS K	28(25)	55(50)	110(100)	220(200)	440(400)	4000(4000)
SN-DDPM*	**25.49**	20.80	17.88	16.97	16.18	16.22
GMS (OURS)	26.50	**20.13**	**17.29**	**16.60**	**15.98**	**15.79**

For completeness, we compare the sampling speed of GMS and non-improved reverse transition kernel in Tab. 10, and it can be seen that within 100 steps, our method greatly outperforms Gotta Go Fast [15]. It is worth noting that the results of Gotta Go Fast are based on Song et al. [35]'s pre-trained model, while ours is based on Ho et al. [12]'s pre-trained model.

Table 10: **Comparison with GOTTA GO FAST [15] w.r.t. FID score ↓ on CIFAR10 and ImageNet.** The number of sampling steps inside the parentheses is our sampling step, while the number outside the parentheses is GOTTA GO FAST's sampling step, in order to ensure that the total time consumption is the same for both methods.

	CIFAR10 (VP SDE)						
# TIMESTEPS K	11(10)	29(27)	49(45)	145(135)	179 (163)	274 (250)	329 (300)
GOTTA GO FAST	325.33	247.79	72.29	3.03	**2.59**	**2.74**	**2.70**
GMS (OURS)	**17.43**	**5.45**	**4.22**	**3.00**	3.06	3.08	2.98

E.8 Codes and License

In Tab.11, we list the code we used and the license.

Table 11: codes and license.

URL	CITATION	LICENSE
HTTPS://GITHUB.COM/W86763777/PYTORCH-DDPM	HO ET AL. [12]	WTFPL
HTTPS://GITHUB.COM/OPENAI/IMPROVED-DIFFUSION	NICHOL AND DHARIWAL [28]	MIT

F SDEdit

Fig. 7 illustrates one of the comprehensive procedures of SDEdit. Given a guided image, SDEdit initially introduces noise to t_0. Subsequently, using this noisy image and then discretizes the inverse SDE to generate the final image. Fig. 7 shows that the choice of t_0 will can greatly will greatly affect the realism of sample images. With the increase of t_0, the similarity between sample images and the real image is decreasing. Hence, apart from conducting quantitative evaluations to assess the fidelity of the generated images, it is also crucial to undertake qualitative evaluations to examine the outcomes associated with different levels of fidelity. Taking all factors into comprehensive consideration, we have selected the range of t_0 from 0.3T to 0.5T in our experiments.

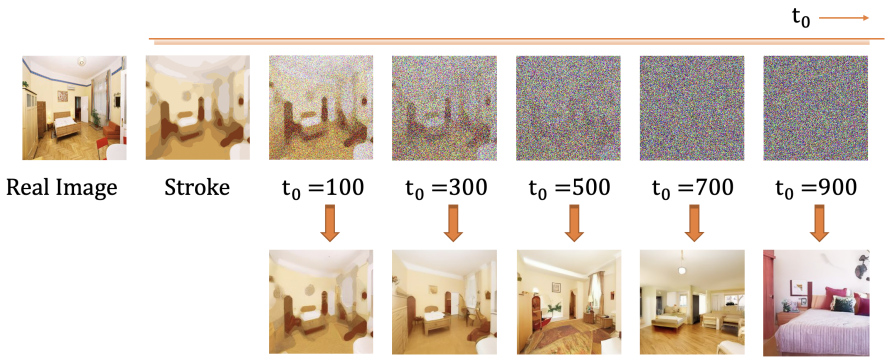


Figure 7: t_0 denotes the timestep to noise the stroke

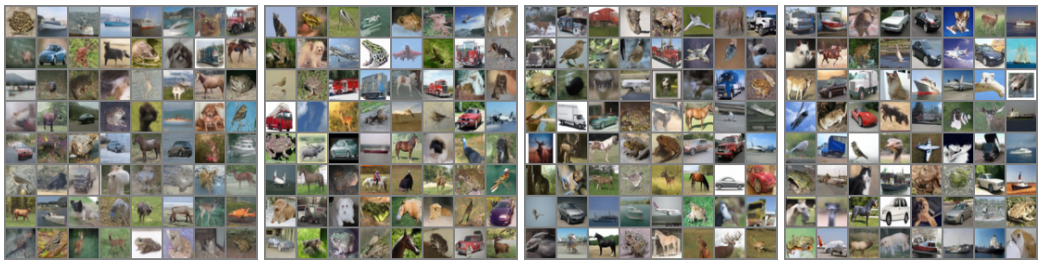
Besides experiments on LSUN 256×256, we also carry out the SDEdit on Imagenet 64×64. In Table 12, we show the FID score for different methods in different t_0 and different sample steps. And our method outperforms other SDE-based solvers as well.

Table 12: **Comparison with competitive methods in SDEdit w.r.t. FID score ↓ on ImageNet64×64.** ODE-based solver is worse than all SDE-based solvers. With nearly the same computation cost, our GMS outperforms existing methods in most cases.

	IMAGENET 64X64, $t_0 = 1200$			
# K	26(28)	51(55)	101(111)	201(221)
DDPM,$\tilde{\beta}_n$	21.37	19.15	18.85	18.15
DDIM	21.87	21.81	21.95	21.90
SN-DDPM	20.76	18.67	17.50	16.88
GMS	**20.50**	**18.37**	**17.18**	**16.83**

G Samples

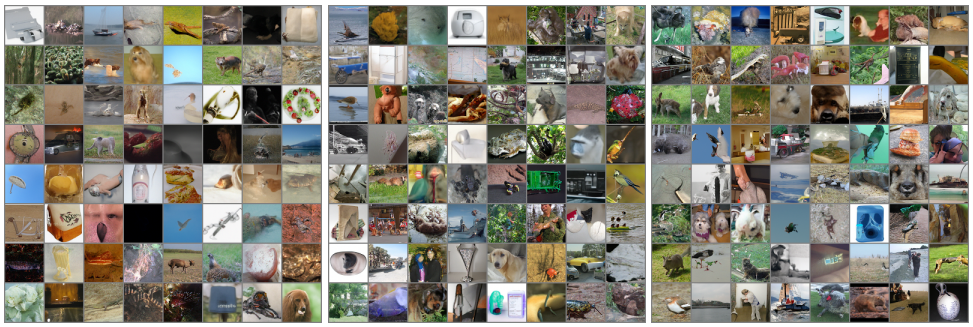
From Fig. 8 to Fig. 10, we show generated samples of GMS under a different number of steps in CIFAR10 and Imagenet 64×64. Here we use K to denote the number of steps for sampling.



(a) GMS ($K = 10$) (b) GMS ($K = 20$) (c) GMS ($K = 25$) (d) GMS ($K = 40$)

(e) GMS ($K = 50$) (f) GMS ($K = 100$) (g) GMS ($K = 200$) (h) GMS ($K = 1000$)

Figure 8: Generated samples on CIFAR10 (LS)



(a) GMS ($K = 25$) (b) GMS ($K = 50$) (c) GMS ($K = 100$)

Figure 9: Generated samples on Imagenet 64×64.

(a) GMS ($K = 200$) (b) GMS ($K = 400$) (c) GMS ($K = 4000$)

Figure 10: Generated samples on Imagenet 64×64.