Distilled Diffusion Models

July 3, 2024

Contents

1 Consistency Models

1.1 Motivation

Why Diffusion Models cannot generate plausible samples in the few-step (1-5) regime?

We consider the following reverse process to generate data using Diffusion Model:

$$\mathrm{d}\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] \mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t, \ t \in [1, 0]. \tag{1}$$

If the forward process satisfies $p_t(\boldsymbol{x}|\boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{x}_0, s_t^2\right)$, where $\boldsymbol{x}_0$ is a sample from data distribution. Then, the score function has the following form:

$$\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) = -\frac{1}{s_t^2}\left(\boldsymbol{x} - \mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right]\right). \tag{2}$$

We can find the form of the conditional expectation assuming that

$$p_{\text{data}}(\boldsymbol{x}_0) = \frac{1}{N}\sum_{j=1}^{N} \delta(\boldsymbol{x}_0 - \boldsymbol{x}_0^j). \tag{3}$$

Then,

$$\begin{aligned} \mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right] &= \int \boldsymbol{x}_0 p_t(\boldsymbol{x}_0|\boldsymbol{x})\mathrm{d}\boldsymbol{x}_0, \\ p_t(\boldsymbol{x}_0|\boldsymbol{x}) &= \frac{p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)}{\int p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0}, \end{aligned} \tag{4}$$

$$\begin{aligned} \mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right] &= \int \boldsymbol{x}_0 \frac{p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)}{\int p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0}\mathrm{d}\boldsymbol{x}_0 \\ &= \frac{1}{\int p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0}\int \boldsymbol{x}_0 p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0 \\ &= \frac{1}{\int p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0}\int \boldsymbol{x}_0 \mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{x}_0, s_t^2\right)\frac{1}{N}\sum_{j=1}^{N}\delta(\boldsymbol{x}_0 - \boldsymbol{x}_0^j)\mathrm{d}\boldsymbol{x}_0 \\ &= \frac{1}{\int p_t(\boldsymbol{x}|\boldsymbol{x}_0)p(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0}\frac{1}{N}\sum_{j=1}^{N}\boldsymbol{x}_0^j \mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{x}_0^j, s_t^2\right) \end{aligned} \tag{5}$$

Finally,

$$\boxed{\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right] = \frac{\sum_{j=1}^{N}\boldsymbol{x}_0^j \,\mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{x}_0^j, s_t^2\right)}{\sum_{j=1}^{N}\mathcal{N}\left(\boldsymbol{x} \mid \boldsymbol{x}_0^j, s_t^2\right)} \approx \boldsymbol{f}_\theta(\boldsymbol{x}, t)} \tag{6}$$

For large t we obtain the mean of the dataset. Diffusion Models provides progressive denoising and thus cannot produce a sample in a few steps. We have to make samples less and less averaged.

The main idea behind consistency models:

> How to train $\boldsymbol{f}_\theta(\boldsymbol{x}, t)$ in such a way it produces a concrete sample, $\boldsymbol{x}_0^j$, instead of the mixture. That is, $\boldsymbol{f}_\theta(\boldsymbol{x}, t) \approx \boldsymbol{x}_0^j$.

1.2 Consistency function

So, we want $\boldsymbol{f}_\theta(\boldsymbol{x}, t) \approx \boldsymbol{x}_0 \ \forall t$, where $\boldsymbol{x}$ - solution of the ODE at timestep t. A function that satisfies this property called **consistency function**. But how to obtain such function in practice?

Lets consider a discrete case, $t_1, ..., t_n$. Note that $\boldsymbol{f}_\theta(\boldsymbol{x}, t) \approx \boldsymbol{x}_0 \ \forall t$ leads to $\boldsymbol{f}_\theta(\boldsymbol{x}_1, t_1) = \boldsymbol{f}_\theta(\boldsymbol{x}_2, t_2) = ... = \boldsymbol{f}_\theta(\boldsymbol{x}_n, t_n)$. We can train a consistency model by enforcing this property.

$$\|\boldsymbol{f}_\theta(\boldsymbol{x}_{n-1}, t_{n-1}) - \boldsymbol{f}_\theta(\boldsymbol{x}_n, t_n)\| \to \min_\theta \tag{7}$$

Note that this optimization problem may lead to the trivial solution: $\boldsymbol{f}_\theta(\boldsymbol{x}, t) = 0$. To avoid this, the repamaterization is introduced:

$$\boldsymbol{f}_\theta(\boldsymbol{x}, t_1) = \boldsymbol{x}, \tag{8}$$

where t_1 is small. In practice, to make the consistency model to be continuous-time, we consider $\hat{\boldsymbol{f}}_\theta(\boldsymbol{x}, t) = c_1(t)\boldsymbol{x} + c_2(t)\boldsymbol{f}_\theta(\boldsymbol{x}, t)$, where c_1, c_2 are differentiable functions, $c_1(t_1) = 1, c_2(t_2) = 0$.

Can we ensure that Eq. 27 lead to the desired consistency property?

Proposition 1. *The optimal solution to the consistency distillation problem Eq. (27) has the following form:*

$$\boldsymbol{f}_\theta(\boldsymbol{x}, t) = \boldsymbol{x}_0^k. \tag{9}$$

Proof.

$$\begin{gathered}\|\boldsymbol{f}_\theta(\boldsymbol{x}_{n-1}, t_{n-1}) - \boldsymbol{f}_\theta(\boldsymbol{x}_n, t_n)\| = 0 \\ \to \boldsymbol{f}_\theta(\boldsymbol{x}_{n-1}, t_{n-1}) = \boldsymbol{f}_\theta(\boldsymbol{x}_n, t_n)\end{gathered} \tag{10}$$

Let $n = 2$, then

$$\boldsymbol{f}_\theta(\boldsymbol{x}_1, t_1) = \boldsymbol{x}_1 = \boldsymbol{x}_0 + s_1 \cdot \boldsymbol{z} = \boldsymbol{f}_\theta(\boldsymbol{x}_2, t_2), \boldsymbol{z} \sim \mathcal{N}(0, 1) \tag{11}$$

Assuming that $s_1 \approx 0$, we obtain $\boldsymbol{f}_\theta(\boldsymbol{x}_2, t_2) \approx \boldsymbol{x}_0$. □

We have two strategies to train the consistency model.

1.3 Consistency via distillation

. In this case, we initialize the consistency model from the diffusion model and solve the following optimization problem.

$$\mathbb{E}_{\mathcal{U}(n|2,N)}\, \mathbb{E}_{p_{\text{data}}(\boldsymbol{x}_0) p_{t_n}(\boldsymbol{x}|\boldsymbol{x}_0)}\, w(t_n) \|\boldsymbol{f}_{\theta^-}(\hat{\boldsymbol{x}}, t_{n-1}) - \boldsymbol{f}_\theta(\boldsymbol{x}, t_n)\|, \tag{12}$$

where $\hat{\boldsymbol{x}}$ is one step of ODE solver with parameters ϕ, from t_n to t_{n-1}, starting from $\boldsymbol{x}$; $\theta^- = \text{stopgrad}(\mu\theta^- + (1-\mu)\theta)$; $w(t)$ - weighting function. $\boldsymbol{f}_{\theta^-}(\hat{\boldsymbol{x}}, t_{n-1})$ and $\boldsymbol{f}_\theta(\boldsymbol{x}, t_n)$ are called as a teacher and student, respectively

Questions.

Q. *Why do we need a solver? Why using solver is better than simply* $\hat{\boldsymbol{x}} \sim p_{t_{n-1}}(\hat{\boldsymbol{x}}|\boldsymbol{x}_0)$

A. Intuition in Figure 1. If we do $\hat{\boldsymbol{x}} \sim p_{t_{n-1}}(\hat{\boldsymbol{x}}|\boldsymbol{x}_0)$ and $\boldsymbol{x} \sim p_{t_n}(\boldsymbol{x}|\boldsymbol{x}_0)$ then $\boldsymbol{f}_\theta(\boldsymbol{x})$ and $\boldsymbol{f}_\theta(\hat{\boldsymbol{x}})$ will look at slightly different modes, $\boldsymbol{x}_0$, because different trajectories intersect. However, we need that $\boldsymbol{f}_\theta$ look at the same mode for different t. I think, it potentially leads to slower convergence and worse final performance.

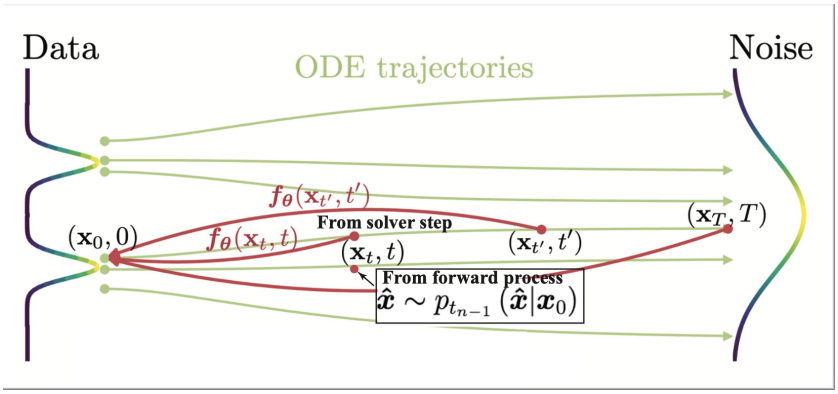


Figure 1

Q. *Why do we need EMA?*

A. Possible because we need the student and the teacher to be different models in order to avoid trivial solution.

Q. ***What is the right rule for distillation?*** *Now we need much training steps and full dataset to perform distillation. Is it the right way?*

A. Intuition: distillation is not a training, the model already knows the distribution. Thus, we do not need to learn a whole distribution. **Which subset of data we need?**; **How to change the model weights?**

P. *Possible ablations to try:*

- align not only neighboring t_n and t_{n-1} but also more distant steps. Concretely:

$$\|\boldsymbol{f}_\theta(\boldsymbol{x}_k, t_k) - \boldsymbol{f}_\theta(\boldsymbol{x}_j, t_j)\| \to \min_\theta \tag{13}$$

  where $k \neq j$. Intuition: in the definition of the consistency function we said that the function should be the same for different time steps, t.
  In practice we can do: **1)** make more ODE steps, not only a single step. However this is a time-consuming. But maybe will convergence much faster; **2)** Combine distillation and training loss of consistency:

$$\begin{aligned}&\mathbb{E}_{\mathcal{U}(k,j|1,N)}\,\mathbb{E}_{p_{\text{data}}(\boldsymbol{x}_0)p(\boldsymbol{z})}\, w(t_n)\|\boldsymbol{f}_\theta(\boldsymbol{x}_0 + t_k\boldsymbol{z}, t_k) - \boldsymbol{f}_\theta(\boldsymbol{x}_0 + t_j\boldsymbol{z}, t_j)\| \\ &+ \mathbb{E}_{\mathcal{U}(n|2,N)}\,\mathbb{E}_{p_{\text{data}}(\boldsymbol{x}_0)p_{t_n}(\boldsymbol{x}|\boldsymbol{x}_0)}\, w(t_n)\|\boldsymbol{f}_{\theta^-}(\hat{\boldsymbol{x}}, t_{n-1}) - \boldsymbol{f}_\theta(\boldsymbol{x}, t_n)\|\end{aligned} \tag{14}$$

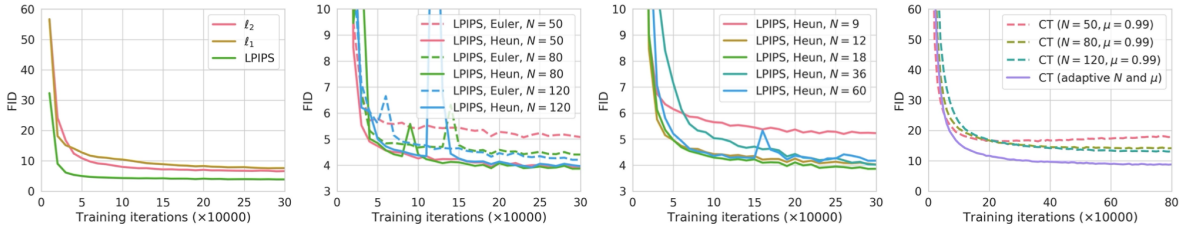
Or maybe combine 1) and 2). For example, if k and j are close - make 1) otherwise 2).

- try solvers of different order (DPM, DDIM, EDM), does it really matter?
- ablate EMA, does we really need this?
- ablate different weighting schemes.

1.3.1 Training details

Timesteps schedule, N (?). Song writes the intuition for the consistency training: *For improved practical performance, we propose to progressively increase N during training according to a schedule function. The consistency training loss has less "variance" but more "bias" when N is small, which facilitates faster convergence at the beginning of training. On the contrary, it has more "variance" but less "bias" when N is large (i.e., Δt is small), which is desirable when closer to the end of training.* **Is it true for CD?**

Ablation for N and solver order. The best performing configuration of N corresponds to the best sampling configuration of diffusion model (Figure 2c). That is, for EDM cifar is 18 steps. **Larger N better results does not work?**; As shown in Figure 2d, adaptive N and μ works good for CT, **Is it true for CD?**.
Think about it: As demonstrated in Fig. 3d, the convergence of CT is highly sensitive to N —smaller N leads to faster convergence but worse samples, whereas larger N leads to slower convergence but better samples upon convergence. This matches our analysis in Section 5, and motivates our practical choice of progressively growing N and μ for CT to balance the trade-off between convergence speed and sample quality.



(a) Metric functions in CD. (b) Solvers and N in CD. (c) N with Heun solver in CD. (d) Adaptive N and μ in CT.

Figure 2

Table 3: Hyperparameters used for training CD and CT models

Hyperparameter	CIFAR-10		ImageNet 64 × 64		LSUN 256 × 256	
	CD	CT	CD	CT	CD	CT
Learning rate	4e-4	4e-4	8e-6	8e-6	1e-5	1e-5
Batch size	512	512	2048	2048	2048	2048
μ	0		0.95		0.95	
μ_0		0.9		0.95		0.95
s_0		2		2		2
s_1		150		200		150
N	18		40		40	
ODE solver	Heun		Heun		Heun	
EMA decay rate	0.9999	0.9999	0.999943	0.999943	0.999943	0.999943
Training iterations	800k	800k	600k	800k	600k	1000k
Mixed-Precision (FP16)	No	No	Yes	Yes	Yes	Yes
Dropout probability	0.0	0.0	0.0	0.0	0.0	0.0
Number of GPUs	8	8	64	64	64	64

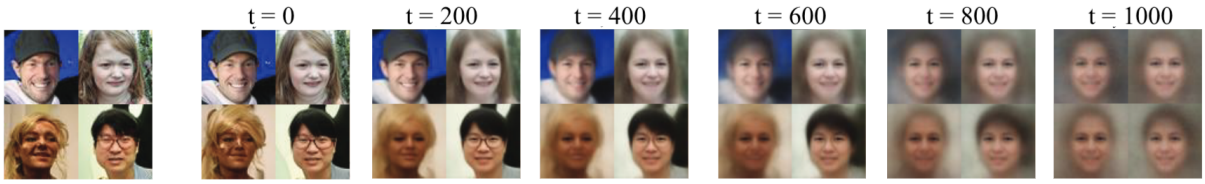
Figure 3

1.4 The difference between DM and CM

1. *Approximation difference*

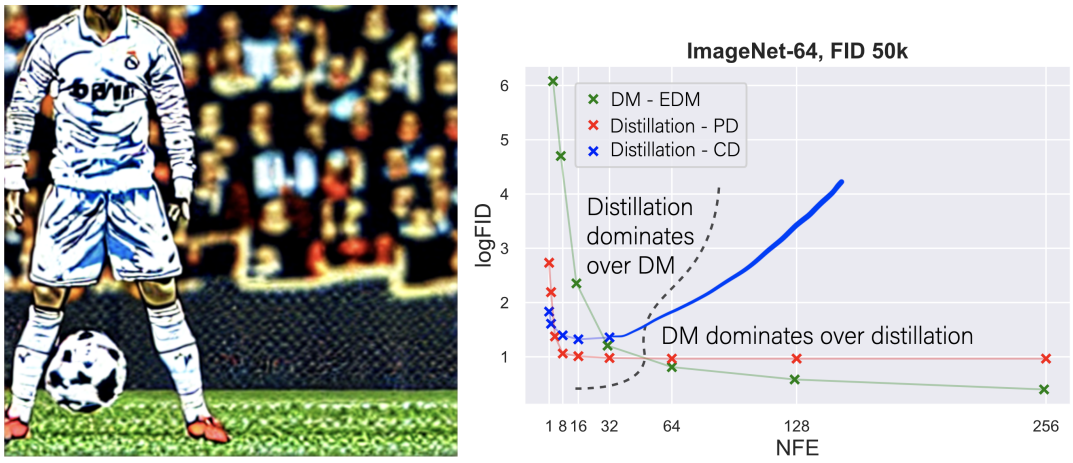
DM: $\|\boldsymbol{f}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\| \rightarrow \boxed{\boldsymbol{f}_\theta(\boldsymbol{x}_t, t) = \mathbb{E}\left[\boldsymbol{x}_0 | \boldsymbol{x}_t\right]} \rightarrow \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) = -\frac{1}{t}\left(\boldsymbol{x} - \mathbb{E}\left[\boldsymbol{x}_0 | \boldsymbol{x}\right]\right)$

CM: $\|\boldsymbol{f}_\theta(\boldsymbol{x}_{t-1}, t-1) - \boldsymbol{f}_\theta(\boldsymbol{x}_t, t)\| \rightarrow \boxed{\boldsymbol{f}_\theta(\boldsymbol{x}_t, t) = \boldsymbol{x}_0} \not\rightarrow \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$



2. *Sampling difference*

DM: Score function $\rightarrow$ Solvers
CM: Consistency function $\not\rightarrow$ Solvers // $\rightarrow$ Stochastic sampler



Q. Why CM degrades with the NFE increasing?

1.5 Consistency from scratch

Here, we also have to solve the same problem

$$\|\boldsymbol{f}_\theta(\boldsymbol{x}_{n-1}, t_{n-1}) - \boldsymbol{f}_\theta(\boldsymbol{x}_n, t_n)\| \to \min_\theta \tag{15}$$

We sample $\boldsymbol{x}_n \sim p_{t_n}(\boldsymbol{x}_n|\boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_n \mid \boldsymbol{x}_0, t_n^2\right)$, but how to obtain $\boldsymbol{x}_{n-1}$ if we do not have a diffusion model that approximates the score, $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$. In other words, we cannot solve the equation (or even approximate it) without the known score. However, lets look at this in more detail.

$$\begin{aligned} \mathrm{d}\boldsymbol{x} &= -t\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\mathrm{d}t, \\ \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) &= -\frac{1}{t^2}\left(\boldsymbol{x} - \mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right]\right). \end{aligned} \tag{16}$$

The unknown component, that we approximate via diffusion, it is $\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right]$. This is an integral that we cannot solve for the experimental data only. The authors consider a Monte-Carlo approximation:

$$\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right] = \int \boldsymbol{x}_0 p_t(\boldsymbol{x}_0|\boldsymbol{x})\mathrm{d}\boldsymbol{x}_0 \approx \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_0^i|\boldsymbol{x},\ \boldsymbol{x}_0^i \sim p_t(\boldsymbol{x}_0^i|\boldsymbol{x}) \tag{17}$$

The authors consider $n = 1$. Thus, $\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}\right] \approx \boldsymbol{x}_0^i$ and we have the following.

$$\mathrm{d}\boldsymbol{x} = -t\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\mathrm{d}t \approx \frac{1}{t}\left(\boldsymbol{x} - \boldsymbol{x}_0\right)\mathrm{d}t \tag{18}$$

Interestingly, now we can solve the equation analytically:

$$\begin{aligned} \mathrm{d}\boldsymbol{x} &= \frac{1}{t}\left(\boldsymbol{x} - \boldsymbol{x}_0\right)\mathrm{d}t \\ \frac{\mathrm{d}\boldsymbol{x}}{\left(\boldsymbol{x} - \boldsymbol{x}_0\right)} &= \frac{1}{t}\mathrm{d}t \\ \int \frac{\mathrm{d}\left(\boldsymbol{x} - \boldsymbol{x}_0\right)}{\left(\boldsymbol{x} - \boldsymbol{x}_0\right)} &= \frac{\mathrm{d}t}{t} \\ \ln\left|\boldsymbol{x}_{t_n} - \boldsymbol{x}_0\right| - \ln\left|\boldsymbol{x}_{t_{n+1}} - \boldsymbol{x}_0\right| &= \ln\left|t_n\right| - \ln\left|t_{n+1}\right| \\ \ln\left|\frac{\boldsymbol{x}_{t_n} - \boldsymbol{x}_0}{\boldsymbol{x}_{t_{n+1}} - \boldsymbol{x}_0}\right| &= \ln\left|\frac{t_n}{t_{n+1}}\right| \\ \boldsymbol{x}_{t_n} &= \boldsymbol{x}_0 + \frac{t_n}{t_{n+1}}\left(\boldsymbol{x}_{t_{n+1}} - \boldsymbol{x}_0\right) \\ \boldsymbol{x}_{t_n} &= \frac{t_n}{t_{n+1}}\boldsymbol{x}_{t_{n+1}} - \boldsymbol{x}_0\left(\frac{t_n - t_{n+1}}{t_{n+1}}\right) \end{aligned} \tag{19}$$

Note that we can take arbitrary t_n and t_{n+1}. So, in practice we sample $\boldsymbol{x}_{t_{n+1}} = \boldsymbol{x}_0 + t_{n+1} \cdot \boldsymbol{z},\ \boldsymbol{z} \sim \mathcal{N}(0, I)$. And calculate $\boldsymbol{x}_{t_n}$ according to the equation:

$$\begin{aligned} \boldsymbol{x}_{t_n} &= \frac{t_n}{t_{n+1}}\boldsymbol{x}_{t_{n+1}} - \boldsymbol{x}_0\left(\frac{t_n - t_{n+1}}{t_{n+1}}\right) \\ &= \frac{t_n}{t_{n+1}}\left(\boldsymbol{x}_0 + t_{n+1} \cdot \boldsymbol{z}\right) - \boldsymbol{x}_0\left(\frac{t_n - t_{n+1}}{t_{n+1}}\right) \\ &= \boldsymbol{x}_0 + t_n\boldsymbol{z} \end{aligned} \tag{20}$$

Finally, our optimization problem becomes:

$$\|\boldsymbol{f}_\theta(\boldsymbol{x}_0 + t_{n+1} \cdot \boldsymbol{z}, t_{n+1}) - \boldsymbol{f}_\theta(\boldsymbol{x}_0 + t_n \cdot \boldsymbol{z}, t_n)\| \to \min_\theta \tag{21}$$

Questions.

Q. *What happens if we take more accurate estimation of the conditional expectation? Or even analytical solution?*
A. **Case 1.** More accurate estimation.

$$\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}_t\right] \approx \boldsymbol{x}_0^i \to \mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}_t\right] \approx \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_0^i|\boldsymbol{x}_t, \tag{22}$$

The number n depends on the current time t. For large t, e.g. 1, we have to take large n (approximately all dataset). For small $t \approx 0$ one sample is enough. To sample from dataset, we can use k-nn to find corresponding x_0. Then,

$$\mathbb{E}\left[\boldsymbol{x}_0|\boldsymbol{x}_t\right] \approx \frac{1}{n(t, \boldsymbol{x}_t)} \sum_{i=1}^{n(t,\boldsymbol{x}_t)} \boldsymbol{x}_0^i|\boldsymbol{x}_t, \tag{23}$$

The equation and its solution becomes.

$$\begin{aligned} \mathrm{d}\boldsymbol{x} &= \frac{1}{t}\left(\boldsymbol{x} - \frac{1}{m(t, \boldsymbol{x}_t)} \sum_{i=1}^{m(t,\boldsymbol{x}_t)} \boldsymbol{x}_0^i\right) \mathrm{d}t \\ \boldsymbol{x}_{t_n} &= \frac{t_n}{t_{n+1}}\boldsymbol{x}_{t_{n+1}} - \left(\frac{t_n - t_{n+1}}{t_{n+1}}\right)\frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_0^i \end{aligned} \tag{24}$$

Taking into account $\boldsymbol{x}_{t_{n+1}} = \boldsymbol{x}_0 + t_{n+1} \cdot \boldsymbol{z}$, the consistency loss transforms to

$$\begin{aligned} \boldsymbol{x}_{t_n} &= \frac{t_n}{t_{n+1}}\boldsymbol{x}_0 + t_n \cdot \boldsymbol{z} - \left(\frac{t_n - t_{n+1}}{t_{n+1}}\right)\frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_0^i \\ \boldsymbol{x}_{t_n} &= \boldsymbol{x}_0\left(\frac{1 + t_n/t_{n+1}\,(m-1)}{m}\right) - \left(\frac{t_n - t_{n+1}}{t_{n+1}}\right)\frac{1}{m}\sum_{\boldsymbol{x}_0^i \setminus \boldsymbol{x}_0} \boldsymbol{x}_0^i + t_n \cdot \boldsymbol{z} \end{aligned} \tag{25}$$

Looks creepy.

Case 2. Analytical form of the expectation. No way :(, i'm full rubbish, biotrash.

Q. *Now we take a single step of the ODE using approximation of the conditional expectation. What if we do more?*

2 Improved CT

2.1 EMA

Previously, Song uses the following loss to train the Consistency Models

$$\mathcal{L}^N(\theta, \theta^-) = \|\boldsymbol{f}_\theta(\boldsymbol{x}_{t_{n+1}}, t_{n+1}) - \boldsymbol{f}_{\theta^-}(\boldsymbol{x}_{t_n}, t_n)\| \to \min_\theta, \tag{26}$$

where $\theta^- = \text{stopgrad}(\mu\theta^- + (1-\mu)\theta)$. For Consistency Training, Song uses an Euler step, approximating the score as a 1-sample Monte Carlo estimation.

$$\mathcal{L}^N_{\text{CT}}(\theta, \theta^-) = \|\boldsymbol{f}_\theta(\boldsymbol{x}_0 + t_{n+1}\cdot\boldsymbol{z}, t_{n+1}) - \boldsymbol{f}_{\theta^-}(\boldsymbol{x}_0 + t_n\cdot\boldsymbol{z}, t_n)\| \to \min_\theta, \tag{27}$$

However, he states that the EMA degrades the quality of the learned network. To confirm this, lets consider the following toy example.
Let

$$\begin{aligned} &p_{\text{data}}(\boldsymbol{x}) = \delta(\boldsymbol{x} - \boldsymbol{\epsilon}), \\ &\operatorname*{argmin}_\theta \mathcal{L}^N(\theta, \theta^-) - ? \end{aligned} \tag{28}$$

The question is the following: what is the optimum of $\mathcal{L}^N(\theta, \theta^-)$. To this end, we solve the ODE, using the Variance Exploding scheme.

$$\begin{aligned} &\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\sigma} = -\sigma\nabla_x \log p_\sigma(\boldsymbol{x}),\ \sigma \in [\sigma_{\min}, \sigma_{\max}], \\ &\quad \boldsymbol{x}_\sigma \sim p_\sigma(\boldsymbol{x}), p_{\sigma_{\min}}(\boldsymbol{x}) \approx p_{\text{data}}(\boldsymbol{x}) \end{aligned} \tag{29}$$

Firstly,

$$\begin{aligned} p_\sigma(\boldsymbol{x}) &= \int p_\sigma(\boldsymbol{x}|\boldsymbol{x}_0) p_{\text{data}}(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0 \\ &= \int \mathcal{N}(\boldsymbol{x}|\boldsymbol{x}_0, \sigma^2)\delta(\boldsymbol{x}_0 - \boldsymbol{\epsilon})\mathrm{d}\boldsymbol{x}_0 \\ &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{\epsilon}, \sigma^2) \end{aligned} \tag{30}$$

Secondly,

$$\nabla_x \log p_\sigma(\boldsymbol{x}) = -\frac{\boldsymbol{x} - \boldsymbol{\epsilon}}{\sigma^2} \tag{31}$$

Then, we can solve the equation.

$$\begin{aligned} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\sigma} &= -\sigma\nabla_x \log p_\sigma(\boldsymbol{x}) = \frac{\boldsymbol{x} - \boldsymbol{\epsilon}}{\sigma} \\ \int_{\boldsymbol{x}_{\sigma_{i+1}}}^{\boldsymbol{x}_{\sigma_i}} \frac{\mathrm{d}\boldsymbol{x}}{\boldsymbol{x} - \boldsymbol{\epsilon}} &= \int_{\sigma_{i+1}}^{\sigma_i} \frac{\mathrm{d}\sigma}{\sigma} \\ \ln \frac{\boldsymbol{x}_{\sigma_{i+1}} - \boldsymbol{\epsilon}}{\boldsymbol{x}_{\sigma_i} - \boldsymbol{\epsilon}} &= \frac{\sigma_{i+1}}{\sigma_i} \end{aligned} \tag{32}$$

Finally, the solution is the following.

$$\boldsymbol{x}_{\sigma_i} = \boldsymbol{\epsilon} + \frac{\sigma_i}{\sigma_{i+1}}\left(\boldsymbol{x}_{\sigma_{i+1}} - \boldsymbol{\epsilon}\right) \tag{33}$$

One more thing that should be noted is that the real consistency function has the following form in this case.

$$\boldsymbol{f}(\boldsymbol{x}, \sigma) = \frac{\sigma_{\min}}{\sigma}\boldsymbol{x} + \left(1 - \frac{\sigma_{\min}}{\sigma}\right)\boldsymbol{\epsilon} \tag{34}$$

It is clear that this function is satisfied the requirements consistency function has.

$$\begin{aligned} &1)\ \boldsymbol{f}(\boldsymbol{x}, \sigma_{\min}) = \boldsymbol{x} \\ &2)\ \text{If } \boldsymbol{x} \sim p_\sigma(\boldsymbol{x}),\ \text{then } \boldsymbol{f}(\boldsymbol{x}, \sigma) = \boldsymbol{x}_{\sigma_{\min}} \end{aligned} \tag{35}$$

Moreover, we can approximate the real consistency function using the following parameterization: $\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma) = \frac{\sigma_{\min}}{\sigma}\boldsymbol{x} + \left(1 - \frac{\sigma_{\min}}{\sigma}\right)\boldsymbol{\theta}$.
To sum it up, if $p_{\text{data}}(\boldsymbol{x}) = \delta(\boldsymbol{x} - \boldsymbol{\epsilon})$, then

$$\boxed{\begin{aligned} p_\sigma(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{\epsilon}, \sigma^2) \\ \nabla_x \log p_\sigma(\boldsymbol{x}) &= -\frac{\boldsymbol{x} - \boldsymbol{\epsilon}}{\sigma^2} \\ \boldsymbol{x}_{\sigma_i} &= \boldsymbol{\epsilon} + \frac{\sigma_i}{\sigma_{i+1}}\left(\boldsymbol{x}_{\sigma_{i+1}} - \boldsymbol{\epsilon}\right) \\ \boldsymbol{f}(\boldsymbol{x}, \sigma) &= \frac{\sigma_{\min}}{\sigma}\boldsymbol{x} + \left(1 - \frac{\sigma_{\min}}{\sigma}\right)\boldsymbol{\epsilon} \\ \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma) &= \frac{\sigma_{\min}}{\sigma}\boldsymbol{x} + \left(1 - \frac{\sigma_{\min}}{\sigma}\right)\boldsymbol{\theta} \end{aligned}} \tag{36}$$

Then we can back to the main question about the optimum of the loss function.

$$\mathcal{L}^N(\theta, \theta^-) = \|\boldsymbol{f}_\theta(\boldsymbol{x}_{\sigma_{i+1}}, \sigma_{i+1}) - \boldsymbol{f}_{\theta^-}(\boldsymbol{x}_{\sigma_i}, \sigma_i)\| \tag{37}$$

We have that $\boldsymbol{x}_{\sigma_{i+1}} = \boldsymbol{\epsilon} + \sigma_{i+1} \cdot \boldsymbol{z}$. Using the solution of the ODE, we can find that $\boldsymbol{x}_{\sigma_i} = \boldsymbol{\epsilon} + \sigma_i \cdot \boldsymbol{z}$.

$$\mathcal{L}^N(\theta, \theta^-) = \|\boldsymbol{f}_\theta(\boldsymbol{\epsilon} + \sigma_{i+1} \cdot \boldsymbol{z}, \sigma_{i+1}) - \boldsymbol{f}_{\theta^-}(\boldsymbol{\epsilon} + \sigma_i \cdot \boldsymbol{z}, \sigma_i)\| \tag{38}$$

which means that $\mathcal{L}^N(\theta, \theta^-) = \mathcal{L}^N_{\text{CT}}(\theta, \theta^-)$ in this case.

$$\begin{aligned} \mathcal{L}^N_{\text{CT}}(\theta, \theta^-) &= \mathbb{E}\left[(\boldsymbol{f}_\theta(\boldsymbol{\epsilon} + \sigma_{i+1} \cdot \boldsymbol{z}, \sigma_{i+1}) - \boldsymbol{f}_{\theta^-}(\boldsymbol{\epsilon} + \sigma_i \cdot \boldsymbol{z}, \sigma_i))\right] \\ &= \mathbb{E}\left[\left(\frac{\sigma_{\min}}{\sigma_{i+1}}\left(\epsilon + \sigma_{i+1} \cdot \boldsymbol{z}\right) + \left(1 - \frac{\sigma_{\min}}{\sigma_{i+1}}\right)\boldsymbol{\theta} - \frac{\sigma_{\min}}{\sigma_i}\left(\epsilon + \sigma_i \cdot \boldsymbol{z}\right) + \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\boldsymbol{\theta}^-\right)^2\right] \\ &= \mathbb{E}\left[\left(\frac{\sigma_{\min}}{\sigma_{i+1}}\epsilon + \left(1 - \frac{\sigma_{\min}}{\sigma_{i+1}}\right)\boldsymbol{\theta} - \frac{\sigma_{\min}}{\sigma_i}\epsilon - \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\boldsymbol{\theta}^-\right)^2\right] \end{aligned} \tag{39}$$

We know that $\sigma_{i+1} = \sigma_i + \Delta\sigma$. Assuming that $\Delta\sigma$ is small, we can write the following decomposition

$$\frac{\sigma_{\min}}{\sigma_{i+1}} = \frac{\sigma_{\min}}{\sigma_i + \Delta\sigma} = \frac{\sigma_{\min}}{\sigma_i\left(1 + \Delta\sigma/\sigma_i\right)} = \frac{\sigma_{\min}}{\sigma_i}\left(1 - \frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right) \tag{40}$$

Then,

$$\begin{aligned} &\mathcal{L}^N_{\text{CT}}(\theta, \theta^-) = \\ &= \mathbb{E}\left[\left(\frac{\sigma_{\min}}{\sigma_i}\left(1 - \frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right)\epsilon + \left(1 - \frac{\sigma_{\min}}{\sigma_i}\left(1 - \frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right)\right)\boldsymbol{\theta} - \frac{\sigma_{\min}}{\sigma_i}\epsilon - \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\boldsymbol{\theta}^-\right) \\ &= \mathbb{E}\left[\left(\frac{\sigma_{\min}}{\sigma_i}\left(-\frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right)\epsilon + \left(1 - \frac{\sigma_{\min}}{\sigma_i}\left(1 - \frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right)\right)\boldsymbol{\theta} - \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\boldsymbol{\theta}^-\right)^2\right] \\ &= \mathbb{E}\left[\left(\frac{\sigma_{\min}}{\sigma_i}\left(-\frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right)\epsilon + \left(1 - \frac{\sigma_{\min}}{\sigma_i}\left(1 - \frac{\Delta\sigma}{\sigma_i} + O(\Delta\sigma^2)\right)\right)\boldsymbol{\theta} - \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\boldsymbol{\theta}^-\right)^2\right] \\ &= \mathbb{E}\left[\left(\frac{\sigma_{\min}\Delta\sigma}{\sigma_i^2}\left(\boldsymbol{\theta} - \boldsymbol{\epsilon}\right) + \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\left(\boldsymbol{\theta} - \boldsymbol{\theta}^-\right) + O(\Delta\sigma^2)\right)^2\right] \end{aligned} \tag{41}$$

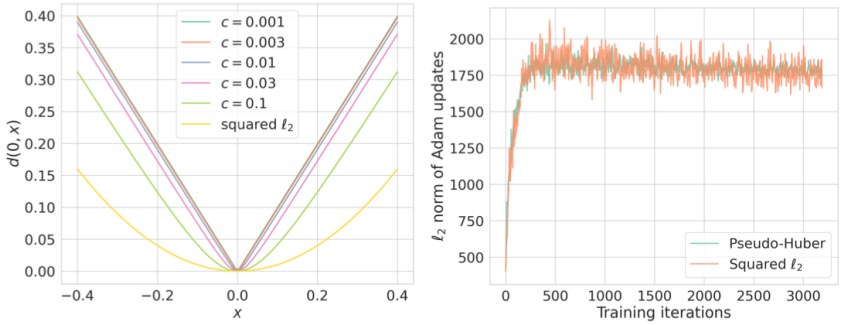
We can use simple formula,$(a+b)^2 = a^2 + 2ab + b^2$, to obtain the final result.

$$\boxed{\mathcal{L}_{\mathrm{CT}}^{N}(\theta, \theta^{-}) = \mathbb{E}\left[\left(\frac{\sigma_{\min}\Delta\sigma}{\sigma_i^2}\left(\boldsymbol{\theta} - \boldsymbol{\epsilon}\right) + \left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{-}\right)\right)^2\right] + O(\Delta\sigma^2)} \tag{42}$$

So, it can be seen that while minimizing this loss we have two components affecting our optimum: $\frac{\sigma_{\min}\Delta\sigma}{\sigma_i^2}\left(\boldsymbol{\theta} - \boldsymbol{\epsilon}\right)$ and $\left(1 - \frac{\sigma_{\min}}{\sigma_i}\right)\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{-}\right)$. The former is our goal since the optimum is $\boldsymbol{\theta} = \boldsymbol{\epsilon}$. However, if $\boldsymbol{\theta} \neq \boldsymbol{\theta}^{-}$, the latter moves us towards the wrong direction ($\boldsymbol{\theta} = \boldsymbol{\theta}^{-}$). Moreover, the first component is multiplied by small value $\Delta\sigma$ and thus has less influence than the second component.
Motivating by these arguments, Song propose to remove EMA, that is, use $\boldsymbol{\theta} = \boldsymbol{\theta}^{-}$.

2.2 Techinical improvements

2.2.1 Pseudo-Huber



(a) $d(\mathbf{0}, \boldsymbol{x})$ as a function of $\boldsymbol{x}$. (b) Comparing Adam updates.

Figure 6: (a) The shapes of various metric functions. (b) The ℓ_2 norms of parameter updates in Adam optimizer. Curves are rescaled to have the same mean. The Pseudo-Huber metric has lower variance compared to the squared ℓ_2 metric.

Instead of L2 loss, Song propose to use the following function

$$d(x, y) = \sqrt{\|x - y\|^2 + c^2} - c \tag{43}$$

The intuition behind is that compared to the squared l2 metric, the Pseudo-Huber metric is more robust to outliers as it imposes a smaller penalty for large errors than the squared l2 metric does, yet behaves similarly for smaller errors. We posit that this added robustness can reduce variance during training. **Apparently, consistency loss has large variance during training.**

3 Multistep CM

3.1 Multistep CD

3.1.1 VP case

This work splits the solution interval into m segments and train the consistency model on each of these segments separately. They parameterize consistency function as

$$\boldsymbol{f}_{\boldsymbol{\theta}}^{m}(\boldsymbol{x}_t, t, s_t^m, w) = \text{DDIM}(\boldsymbol{x}_t, t, s_t^m, w), \tag{44}$$

$$\boldsymbol{x}_s^w = \text{DDIM}(\boldsymbol{x}_t, t, s, \boldsymbol{c}, w) = \sqrt{\frac{\alpha_s}{\alpha_t}}\boldsymbol{x}_t + \boldsymbol{\epsilon}_{\boldsymbol{\theta}}^w(\boldsymbol{x}_t, t, \boldsymbol{c})\left(\sqrt{1-\alpha_s} - \sqrt{\frac{\alpha_s}{\alpha_t}}\sqrt{1-\alpha_t}\right), \tag{45}$$

The CD loss in this case

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}) = \mathbb{E}\left[\|\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{t_{n-1}}^w, t_{n-1}, s_{t_{n-1}}^m) - \boldsymbol{f}_{\boldsymbol{\theta}}^m(\boldsymbol{x}_{t_n}, t_n, s_{t_n}^m)\|\right] \to \min_{\boldsymbol{\theta}}, \tag{46}$$

Note that $s_{t_{n-1}} = s_{t_n}$. We parameterize consistency function by ϵ_θ. Thus, we can rewrite this loss directly via ϵ_θ.

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}) = \mathbb{E}\left[\left\|\frac{\boldsymbol{x}_{s_{t_n}}^w - \sqrt{\frac{\alpha_{t_n}}{\alpha_{s_{t_n}}}}\boldsymbol{x}_{t_n}}{\sqrt{1-\alpha_{t_n}} - \sqrt{\frac{\alpha_{t_n}}{\alpha_{s_{t_n}}}}\sqrt{1-\alpha_{s_{t_n}}}} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_{t_n}, t_n)\right\|\right] \to \min_{\boldsymbol{\theta}}, \tag{47}$$

3.1.2 VE case

Basic equations for the VE case:

$$\begin{gathered} dx = -t\nabla_x \log p_t(x) dt \\ \nabla_x \log p_t(x) = -\frac{1}{t^2}\left(x - \mathbb{E}\left[x_0|x\right]\right) \\ \mathbb{E}\left[x_0|x\right] \approx x^\phi(x,t) - \text{teacher network} \end{gathered} \tag{48}$$

Euler integrator:

$$\begin{gathered} dx = \frac{1}{t}\left(x - x^\phi(x,t)\right), \\ x_s^\phi = x_t + \int_t^s (x_\tau - x^\phi(x_\tau, \tau))\frac{d\tau}{\tau}, \\ x_s^\phi \approx x_t + \frac{s-t}{t}(x_t - x^\phi(x_t, t)) \end{gathered} \tag{49}$$

Consistency loss:

$$\begin{gathered} \mathcal{L}_{\text{CD}}(\theta) = \mathbb{E}\left[\|f_\theta(x_{t_{n-1}}^\phi, t_{n-1}) - f_\theta(x_{t_n}, t_n)\|\right] \to \min_{\boldsymbol{\theta}}, \\ x_{t_{n-1}}^\phi = x_{t_n} + \frac{t_{n-1} - t_n}{t_n}(x_{t_n} - x^\phi(x_{t_n}, t_n)) \end{gathered} \tag{50}$$

Multistep case
Parameterization:

$$\begin{gathered} f_\theta(x_{t_n}, t_n, b_{t_n}) = x_{t_n} + \frac{b_{t_n} - t_n}{t_n}(x_{t_n} - x^\theta(x_{t_n}, t_n)), \\ x^\theta - \text{student}, b_{t_n} - \text{boundary} \end{gathered} \tag{51}$$

Updated consistency loss

$$\mathcal{L}_{\text{CD}}(\theta) = \mathbb{E}\left[\|f_\theta(x^\phi_{t_{n-1}}, t_{n-1}, b_{t_{n-1}}) - f_\theta(x_{t_n}, t_n, b_{t_n})\|\right] \tag{52}$$

Note that, $b_{t_n} = b_{t_{n-1}}$. Lets consider in more detail the optimum

$$\boxed{f_\theta(x_{t_n}, t_n, b_{t_n}) = x_{t_n} + \frac{b_{t_n} - t_n}{t_n}(x_{t_n} - x^\theta(x_{t_n}, t_n))}$$

$$f_\theta(x^\phi_{t_{n-1}}, t_{n-1}, b_{t_{n-1}}) = x^\phi_{t_{n-1}} + \frac{b_{t_n} - t_{n-1}}{t_{n-1}}(x^\phi_{t_{n-1}} - x^\theta(x^\phi_{t_{n-1}}, t_{n-1})) \tag{53}$$

$$x^\phi_{t_{n-1}} = x_{t_n} + \frac{t_{n-1} - t_n}{t_n}(x_{t_n} - x^\phi(x_{t_n}, t_n))$$

Merge the second and the third:

$$\boxed{\begin{aligned} f_\theta(x^\phi_{t_{n-1}}, t_{n-1}, b_{t_{n-1}}) &= x_{t_n} + \frac{t_{n-1} - t_n}{t_n}(x_{t_n} - x^\phi(x_{t_n}, t_n)) + \\ &+ \frac{b_{t_n} - t_{n-1}}{t_{n-1}}\left(x_{t_n} + \frac{t_{n-1} - t_n}{t_n}(x_{t_n} - x^\phi(x_{t_n}, t_n)) - x^\theta(x^\phi_{t_{n-1}}, t_{n-1})\right) \end{aligned}} \tag{54}$$

Q. *Can we achieve any insigths from there reparameterization?*

4 Pagoda

This is a combination of Knowledge Distillation (but in the different way, we collect noise and images using inversion) and GAN. Without CFG:

$$\begin{aligned}\mathcal{L}_{rec}(\theta) &= \mathbb{E}_{p_{\text{data}}(x,c)}\left[\|x - G_\theta(E(x,c),c)\|\right] \\ &= \mathbb{E}_{p_{\text{data}}(x|c)p(c)}\left[\|x - G_\theta(E(x,c),c)\|\right]\end{aligned} \tag{55}$$

$$\begin{aligned}\mathcal{L}_{adv}(\theta) &= \mathbb{E}_{p_{\text{data}}(x,c)}\left[\log D_\psi(x,c)\right] + \mathbb{E}_{p(z,c)}\left[1 - \log D_\psi(G_\theta(z,c),c)\right] \\ &= \mathbb{E}_{p_{\text{data}}(x|c)p(c)}\left[\log D_\psi(x,c)\right] + \mathbb{E}_{p(z)p(c)}\left[1 - \log D_\psi(G_\theta(z,c),c)\right]\end{aligned}$$

With CFG:

$$\begin{aligned}\mathcal{L}_{rec}(\theta) &= \mathbb{E}_{p_{\text{data}}(x,c,w)}\left[\|x - G_\theta(E(x,c,w),c,w)\|\right] \\ &= \mathbb{E}_{p_{\text{data}}(x|c,w)p(c)p(w)}\left[\|x - G_\theta(E(x,c,w),c,w)\|\right]\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{adv}(\theta) &= \mathbb{E}_{p_{\text{data}}(x,c,w)}\left[\log D_\psi(x,c,w)\right] + \mathbb{E}_{p(z,c,w)}\left[1 - \log D_\psi(G_\theta(z,c,w),c,w)\right] \\ &= \mathbb{E}_{p_{\text{data}}(x|c,w)p(c)p(w)}\left[\log D_\psi(x,c,w)\right] + \mathbb{E}_{p(z)p(c)p(w)}\left[1 - \log D_\psi(G_\theta(z,c,w),c,w)\right]\end{aligned} \tag{56}$$

Bayes rule:

$$\begin{aligned}p_{\text{data}}(x,c,w) &= p_{\text{data}}(x|c,w)p(c)p(w) \\ p_{\text{data}}(x,c,w) &= p(w|x,c)p_{\text{data}}(x|c)p_{\text{data}}(c)\end{aligned} \tag{57}$$

With CFG and Bayes rule:

$$\begin{aligned}\mathcal{L}_{rec}(\theta) &= \mathbb{E}_{p_{\text{data}}(x,c,w)}\left[\|x - G_\theta(E(x,c,w),c,w)\|\right] \\ &= \mathbb{E}_{p(w|x,c)p_{\text{data}}(x|c)p(c)}\left[\|x - G_\theta(E(x,c,w),c,w)\|\right]\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{adv}(\theta) &= \mathbb{E}_{p_{\text{data}}(x,c,w)}\left[\log D_\psi(x,c,w)\right] + \mathbb{E}_{p(z,c,w)}\left[1 - \log D_\psi(G_\theta(z,c,w),c,w)\right] \\ &= \mathbb{E}_{p(w|x,c)p_{\text{data}}(x|c)p(c)}\left[\log D_\psi(x,c,w)\right] + \mathbb{E}_{p(z)p(c)p(w)}\left[1 - \log D_\psi(G_\theta(z,c,w),c,w)\right]\end{aligned} \tag{58}$$

5 Score distillation

5.1 DMD

In DMD, the authors consider Variational Score Distillation:

$$\begin{aligned}\mathrm{KL}(p_\theta||p_{real}) &= \mathbb{E}_{p_\theta}\log\left(\frac{p_\theta}{p_{real}}\right) = \int p_\theta(x)\left(\log p_\theta(x) - \log p_{real}(x)\right)dx \\ &= \mathbb{E}_{x=G_\theta(z),z\sim\mathcal{N}(0,1)}\left(\log p_\theta(x) - \log p_{real}(x)\right) \\ &= \int \mathcal{N}(z|0,1)\left[\log p_\theta(G_\theta(z)) - \log p_{real}(G_\theta(z))\right]dz\end{aligned} \tag{59}$$

We sample from the generator $x \sim p_\theta(x)$ and calculate the difference between real and fake scores. We cannot calculate this reverse KL-divergence, since we do not know the densities. We can take the gradient only:

$$\begin{aligned}\nabla_\theta\mathrm{KL}(p_\theta||p_{real}) &= \nabla_\theta\int \mathcal{N}(z|0,1)\left[\log p_\theta(G_\theta(z)) - \log p_{real}(G_\theta(z))\right]dz \\ &= \int \mathcal{N}(z|0,1)\nabla_\theta\left[\log p_\theta(G_\theta(z)) - \log p_{real}(G_\theta(z))\right]dz \\ \nabla_\theta\left[\log p_\theta(G_\theta(z)) - \log p_{real}(G_\theta(z))\right] &= \nabla_\theta\log p_\theta(G_\theta(z)) - \nabla_\theta\log p_{real}(G_\theta(z))\end{aligned} \tag{60}$$

The second term is easy:

$$\nabla_\theta\log p_{real}(G_\theta(z)) = \frac{1}{p_{real}(x)}\nabla_x\log p_{real}(x)\cdot\nabla_\theta G_\theta(z) \tag{61}$$

However, the first one is more tricky. We need to consider it as a function of two arguments.

$$\begin{aligned}\nabla_\theta\log p_\theta(G_\theta(z)) &= \frac{1}{p_\theta(x)}\nabla_\theta\left[p_\theta(G_\theta(z))\right] \\ &= \frac{1}{p_\theta(x)}\left[\nabla_x p_\theta(x)\cdot\nabla_\theta G_\theta(z) + \nabla_\theta p_\theta(x)\nabla_\theta\theta\right] \\ &= \frac{1}{p_\theta(x)}\left[\nabla_x p_\theta(x)\cdot\nabla_\theta G_\theta(z) + \nabla_\theta p_\theta(x)\right] \\ &= \nabla_x\log p_\theta(x)\cdot\nabla_\theta G_\theta(z) + \nabla_\theta\log p_\theta(x)\end{aligned} \tag{62}$$

The first term is the same as in the previous formula but with the fake score $\nabla_x\log p_\theta(x)$. However, the blue one is intractable. To solve this issue, the authors consider the fake score as a function independent of θ, that is $p_\theta(x) = p_{fake}(x)$, then $\nabla_\theta\log p_\theta(x) = 0$. So, we obtain the following:

$$\boxed{\nabla_\theta\mathrm{KL}(p_\theta||p_{real}) = \mathbb{E}_{x=G_\theta(z),z\sim\mathcal{N}(0,1)}\left[\nabla_x\log p_{fake}(x) - \nabla_x\log p_{real}(x)\right]\nabla_\theta G_\theta(z)} \tag{63}$$

The authors note: *Computing this gradient is still challenging for two reasons: first, the scores diverge for samples with low probability — in particular preal vanishes for fake samples, and second, our intended tool for estimating score, namely the diffusion models, only provide scores of the diffused distribution. Score-SDE [71, 72] provides an answer to these two issues. By perturbing the data distribution with random Gaussian*

noise of varying standard deviations, we create a family of "blurred" distributions that are fully-supported over the ambient space, and therefore overlap, so that the gradient in Eq. (2) is well-defined (Figure 4).

$$\nabla_\theta \mathrm{KL}(p_\theta||p_{real}) = \mathbb{E}_{x=G_\theta(z),z\sim\mathcal{N}(0,1)} \left[\nabla_x \log p_{fake}(x) - \nabla_x \log p_{real}(x)\right] \nabla_\theta G_\theta(z) + \\ + \mathbb{E}_{x=G_\theta(z),z\sim\mathcal{N}(0,1)} \nabla_\theta \log p_\theta(x) \tag{64}$$

Want but cannot:

$$\mathbb{E}_{x=G_\theta(z),z\sim\mathcal{N}(0,1)} \nabla_\theta \log p_\theta(x) = -\nabla_\theta \mathbb{E}_z \log \frac{p_{real}(x)}{p_\theta(x)} = \mathrm{KL}(p_{real}||p_\theta) \tag{65}$$

The real score is a pretrained teacher diffusion. The fake score trains dynamically on generated samples using diffusion loss since the fake distribution changes during distillation:

$$\boxed{\mathcal{L}_{\text{diffusion}} = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \|x_\phi(x_t, t) - G_\theta(z)\|^2, x_t \sim p_t(x_t|G_\theta(z), \sigma_t^2)} \tag{66}$$

$x_\phi(x_t, t)$ approximates the fake score. Moreover, authors consider the classical regression loss (knowledge distillation) calling it as a regression term.
The authors note that without the regression term, the only reverse-KL does not converge, maybe because of the $p_\theta(x) = p_{fake}(x)$? This question is considered in the 'score identity distillation' paper.

5.2 ADD

In Adversarial Diffusion Distillation there is also a score distillation loss but in a slightly different form.

$$\mathcal{L}_{ADD} = \mathbb{E}_{z,t} \|G_\theta(z) - x_\psi(G_\theta^t(z), t)\|^2, G_\theta^t(z) = G_\theta(z) + \sigma_t^2 \epsilon \tag{67}$$

The provide "the same" gradients:

$$\nabla_\theta \mathcal{L}_{ADD} = \mathbb{E}_{z,t}\, 2 \left[G_\theta(z) - x_\psi(G_\theta^t(z), t)\right] \nabla_\theta G_\theta(z), \nabla_x x_\psi(x, t) = 1??? \tag{68}$$

The loss is written in terms of x_0 estimates, x_ψ plays the role of real score. We can rewrite in terms of scores:

$$\nabla_x \log p_{real}(G_\theta^t(z)) = -\frac{G_\theta^t(z) - x_\psi(G_\theta^t(z), t)}{\sigma^2} \\ \nabla_x \log p_{fake}(G_\theta^t(z)) = -\frac{G_\theta^t(z) - x_\phi(G_\theta^t(z), t)}{\sigma^2} \tag{69}$$

If we set that $x_\phi(G_\theta^t(z), t) = G_\theta(z)$, then we obtain the same gradient as in the DMD:

$$\nabla_\theta \mathcal{L}_{ADD} = \mathbb{E}_{z,t}\, 2 \left[\nabla_x \log p_{fake}(x) - \nabla_x \log p_{real}(x)\right] \nabla_\theta G_\theta(z) \tag{70}$$

In other words, in ADD the authors does not learn the fake score while in the DMD it is happening dynamycally during training. Instead they approximate the fake score using the distilled model.

5.3 Score identity distillation

In this work, the authors consider the score matching loss between fake and real losses.

$$\mathcal{L}_\theta = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \|\nabla_{G^t_\theta(z)} \log p_{real}(G^t_\theta(z)) - \nabla_{G^t_\theta(z)} \log p_\theta(G^t_\theta(z))\|^2, G^t_\theta(z) = G_\theta(z) + \sigma^2_t\epsilon \tag{71}$$

As it can be seen, this is a sligthly different loss, we match scores instead of densities. The fake score can be trained similarly via diffusion loss using an additional network.

$$\mathcal{L}_{\text{diffusion}}(\phi) = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \|x_\phi(x_t, t) - G_\theta(z)\|^2, x_t \sim p_t(x_t|G_\theta(z)) \tag{72}$$

In this work, the authors reason what happens if the real score depends on the generator $\phi = \phi(\theta)$. So, they firstly assume, as previously, $\phi = \phi$. Thus,

$$\begin{aligned}\mathcal{L}_\theta \approx \mathcal{L}^1_\theta &= \mathbb{E}_{z\sim\mathcal{N}(0,1)} \|x_\phi(G^t_\theta(z), t) - x_\psi(G^t_\theta(z), t)\|^2 \\ &= \mathbb{E}_{z\sim\mathcal{N}(0,1)} \|\delta_{\phi,\psi}(G^t_\theta(z))\|^2\end{aligned} \tag{73}$$

Lets consider the difference between optimal solution $\phi^*(\theta)$ depending on the θ and our approximation ϕ.

$$\begin{aligned}\Delta_{\phi^*(\theta),\phi}(G^t_\theta(z)) &= x_{\phi^*(\theta)}(G^t_\theta(z), t) - x_\phi(G^t_\theta(z), t) \\ \delta_{\phi,\psi}(G^t_\theta(z)) &= \delta_{\phi^*(\theta),\psi}(G^t_\theta(z)) - \Delta_{\phi^*(\theta),\phi}(G^t_\theta(z))\end{aligned} \tag{74}$$

Thus, we obtain

$$\begin{aligned}\mathcal{L}^1_\theta = \mathcal{L}_\theta + \mathbb{E}_{z\sim\mathcal{N}(0,1)} \|\Delta_{\phi^*(\theta),\phi}(G^t_\theta(z))\|^2 - \\ -2\Delta_{\phi^*(\theta),\phi}(G^t_\theta(z))^T \cdot \delta_{\phi^*(\theta),\psi}(G^t_\theta(z))\end{aligned} \tag{75}$$

The second term we minimize via our diffusion loss. However, the third one requires that the fake score, $\phi^*(\theta)$, equals to the real score, ψ, controlled by $\delta_{\phi^*(\theta),\psi}(G^t_\theta(z))$. This loss is large especially in the initial training steps, when p_θ does not match p_{data}.

5.3.1 1D Gaussian

Let $p_{\text{data}}(x_0) = \mathcal{N}(x_0|0, 1)$, $p_t(x_t|G_\theta(z)) = \mathcal{N}(x_t|G_\theta(z), \sigma^2_t)$, $p_\theta(G_\theta(z)) = \mathcal{N}(G_\theta(z)|\theta, 1)$. Find the optimal θ for both $\mathcal{L}^1_\theta$ and $\mathcal{L}_\theta$.

For $\mathcal{L}^1_\theta$ and $\mathcal{L}_\theta$ we need to know the conditional expectation. We know that (from the toy examples in SDE/ODE framework file)
$p_t(x_t) = \mathcal{N}(x_t|0, \sigma^2_t + 1) \to \nabla_{x_t} \log p_t(x_t) = -\frac{x_t}{1+\sigma^2_t} \to \mathbb{E}\, x_0|x_t = \frac{x_t}{1+\sigma^2_t} = x_\psi(x_t, t)$
$p^\theta_t(x_t) = \mathcal{N}(x_t|\theta, \sigma^2_t + 1) \to \nabla_{x_t} \log p^\theta_t(x_t) = -\frac{x_t-\theta}{1+\sigma^2_t} \to \mathbb{E}\, G_\theta|x_t = \frac{x_t+\theta\sigma^2_t}{1+\sigma^2_t} = x_{\phi^*(\theta)}(x_t, t)$;
Let's take $x_\phi(x_t, t) = \frac{x_t+\phi\sigma^2_t}{1+\sigma^2_t}$, then $\phi^*(\theta) = \theta$. Finally, we have

$$\boxed{\mathcal{L}^1_\theta = \frac{\phi\sigma^2_t}{1+\sigma^2_t},\ \mathcal{L}_\theta = \frac{\theta\sigma^2_t}{1+\sigma^2_t}} \tag{76}$$

We can see that $\mathcal{L}^1_\theta$ does not provide meaningful gradients.
Let's also consider the DMD case from the toy example perspective.

$$\begin{aligned}\nabla_\theta \text{KL}(p_\phi||p_{real}) = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \left(\nabla_{G^t_\theta(z)} \log p^\phi_t(G^t_\theta(z)) - \nabla_{G^t_\theta(z)} \log p_t(G^t_\theta(z))\right) \nabla_\theta G_\theta(z) \\ \nabla_{x_t} \log p^\phi_t(x_t) = -\frac{x_t - \phi}{1+\sigma^2_t}; \nabla_{x_t} \log p_t(x_t) = -\frac{x_t}{1+\sigma^2_t}\end{aligned} \tag{77}$$

So, we do not have the problem here(?)

$$\nabla_\theta \mathrm{KL}(p_\phi || p_{real}) = \mathbb{E}_{z\sim\mathcal{N}(0,1)} \left(\frac{\phi}{1+\sigma_t^2}\right) \nabla_\theta G_\theta(z) \tag{78}$$

Authors method.
Note 1.

$$\begin{gathered}
\mathbb{E}_z \left[u^T(G_\theta^t)\nabla_{G_\theta^t} \log p_\theta(G_\theta^t)\right] = \mathbb{E}_z \left[u^T(G_\theta^t)\frac{\nabla_{G_\theta^t} p_\theta(G_\theta^t)}{p_\theta(G_\theta^t)}\right] \\
= \int u^T(G_\theta^t)\nabla_{G_\theta^t} p_\theta(G_\theta^t) dG_\theta^t \\
p_\theta(G_\theta^t) = \int p_t(G_\theta^t|G_\theta) p_\theta(G_\theta) dz \\
\int u^T(G_\theta^t)\nabla_{G_\theta^t} p_\theta(G_\theta^t) dz = \int u^T(G_\theta^t) \left[\int p_t(G_\theta^t|G_\theta)\nabla_{G_\theta^t} \log p_t(G_\theta^t|G_\theta) p_\theta(G_\theta) dz\right] dG_\theta^t \\
\boxed{\mathbb{E}_z \left[u^T(G_\theta^t)\nabla_{G_\theta^t} \log p_\theta(G_\theta^t)\right] = \mathbb{E}_{G_\theta^t|G_\theta,z} \left[u^T(G_\theta^t)\nabla_{G_\theta^t} \log p_t(G_\theta^t|G_\theta)\right]}
\end{gathered} \tag{79}$$

Note 2.

$$\begin{gathered}
\mathcal{L}_\theta = \mathbb{E}_z \|\nabla_{G_\theta^t(z)} \log p_{real}(G_\theta^t) - \nabla_{G_\theta^t} \log p_\theta(G_\theta^t)\|^2 \\
= \mathbb{E}_z \|\frac{1}{\sigma_t^2}(x_\psi(G_\theta^t, t)) - x_{\phi^*(\theta)}(G_\theta^t, t)\|^2 \\
= \frac{1}{\sigma_t^2} \mathbb{E}_z \left[\left(x_\psi(G_\theta^t, t)) - x_{\phi^*(\theta)}(G_\theta^t, t)\right)^T \left(\nabla_{G_\theta^t(z)} \log p_{real}(G_\theta^t) - \nabla_{G_\theta^t} \log p_\theta(G_\theta^t)\right)\right]
\end{gathered} \tag{80}$$

We need to expand the brackets and use note 1 (see app.)

$$\boxed{\mathcal{L}_\theta = \frac{1}{\sigma_t^2} \mathbb{E}_{G_\theta^t|G_\theta,z} \left[\delta_{\psi,\phi^*(\theta)}(G_\theta^t)^T (x_\psi(G_\theta^t, t) - G_\theta)\right]} \tag{81}$$

Let's consider the same approximation, $\phi^*(\theta) = \phi$

$$\begin{gathered}
\mathcal{L}_\theta^2 = \frac{1}{\sigma_t^2} \mathbb{E}_{G_\theta^t|G_\theta,z} \left[\delta_{\psi,\phi}(G_\theta^t)^T (x_\psi(G_\theta^t, t) - G_\theta)\right] \\
= \mathcal{L}_\theta + \frac{1}{\sigma_t^2} \mathbb{E}_{G_\theta^t|G_\theta,z} \left[\Delta_{\phi^*(\theta),\phi}(G_\theta^t)^T (x_\psi(G_\theta^t, t) - G_\theta)\right]
\end{gathered} \tag{82}$$

As we can see, there is no $\delta_{\phi^*(\theta),\psi}$.
Final loss

$$\boxed{\begin{gathered}
\mathcal{L}_\theta = \frac{w(t)}{\sigma_t^2} \mathbb{E}_{G_\theta^t|G_\theta,z} \left[(x_\phi(G_\theta^t, t) - x_\psi(G_\theta^t, t))^T (x_\psi(G_\theta^t, t) - G_\theta)\right] \\
-\alpha w(t) \mathbb{E}_z \|x_\phi(G_\theta^t, t) - x_\psi(G_\theta^t, t)\|^2
\end{gathered}} \tag{83}$$