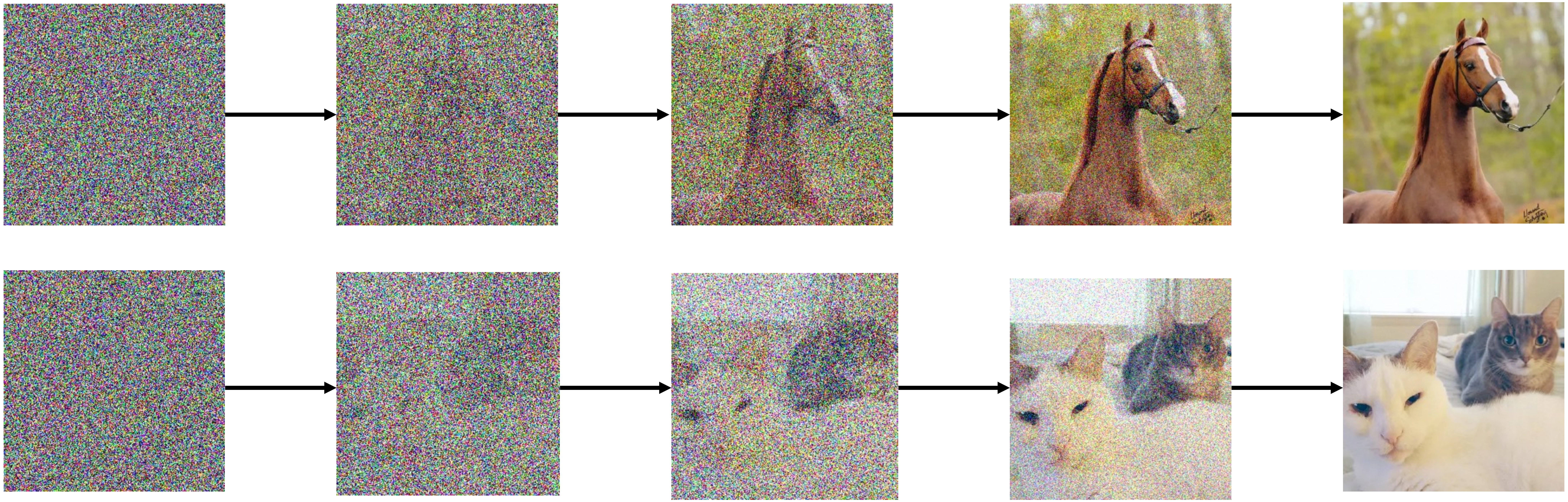
Research

Diffusion Distillation Approaches

Dmitry Baranchuk

Diffusion Probabilistic Models

Gradually denoise until a clear image sample appears

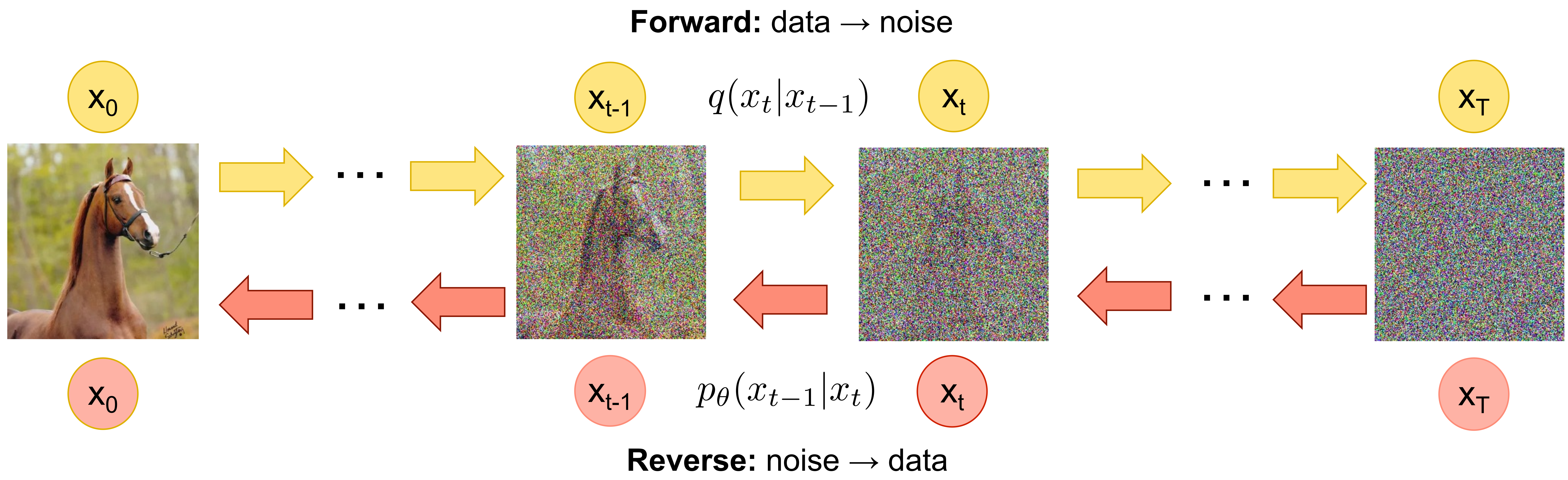


Classical Formulation

Given:

- Data: $x_0 \sim q(x)$
- Diffusion process: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I), \quad x_T \sim N(0, I)$

Goal: reverse it! $q(x_{t-1}|x_t) \approx p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$



Stochastic Differential Equations

Forward SDE (data → noise)

$$\mathbf{x}(0) \longrightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \longrightarrow \mathbf{x}(T)$$

score function

$$\mathbf{x}(0) \longleftarrow \mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \longleftarrow \mathbf{x}(T)$$

Reverse SDE (noise → data)

Y. Song et al, Score-Based Generative Modeling through Stochastic Differential Equations, ICLR2021

Probabilistic Flow ODE

Forward SDE

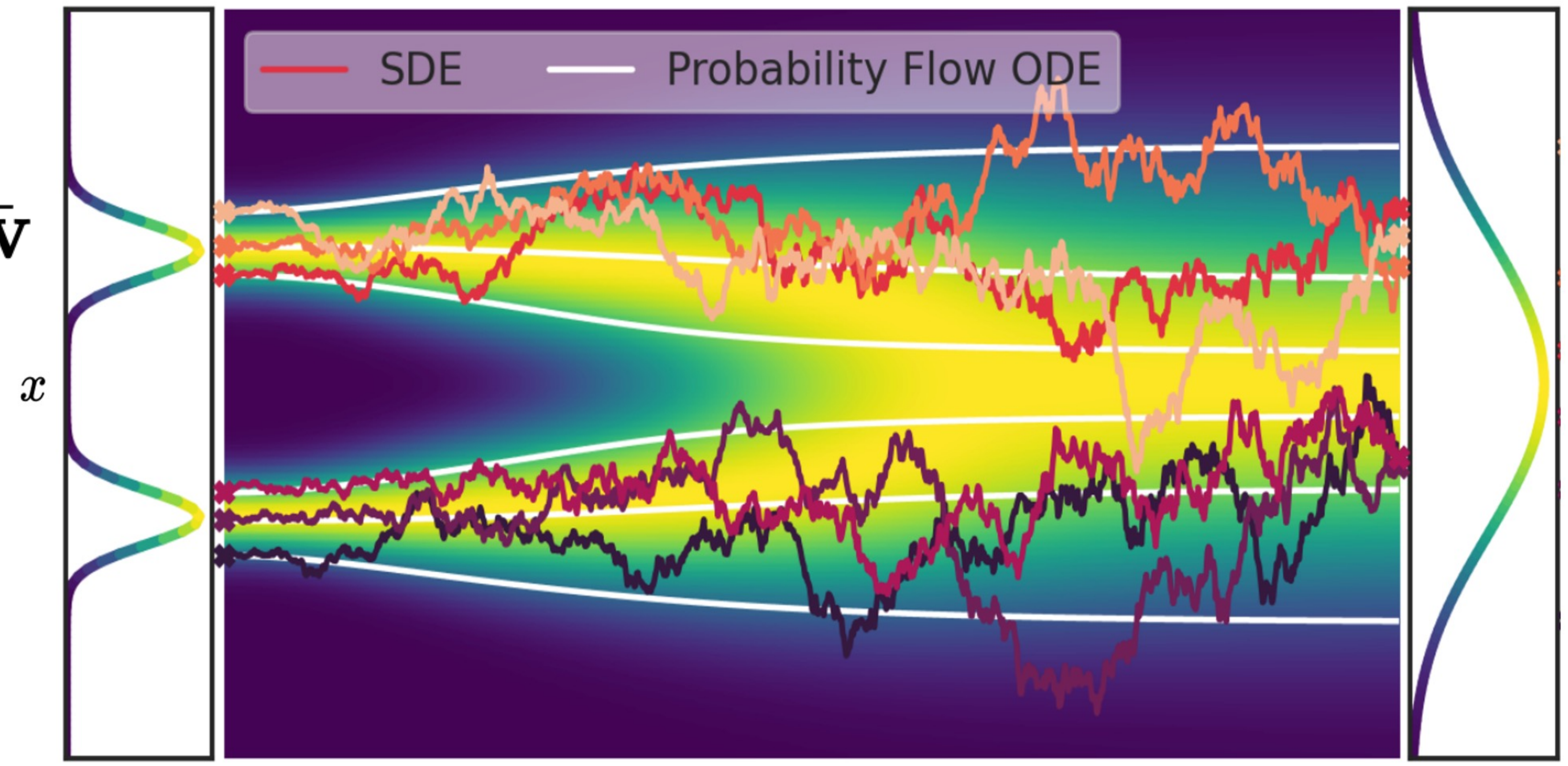
$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w}$$

Backward SDE

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})]dt + g(t)d\bar{\mathbf{w}}$$

Probabilistic Flow ODE

$$d\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]dt$$



Why?

- ODE solvers have lower error than SDE ones → **less sampling steps**
- Higher order ODE solvers perform reasonably in ~10-15 steps.

What about sampling in 1-4 steps?

Classes of Distillation Methods

Classical distillation techniques

- Non-DPM specific distillation methods;
- Often combined with other distillation methods.

Distribution matching distillation

- Minimizing reverse KL using pretrained DPMs.

Rectified flows

- Directly rectifying the DPM trajectories and then distilling using classical approaches.

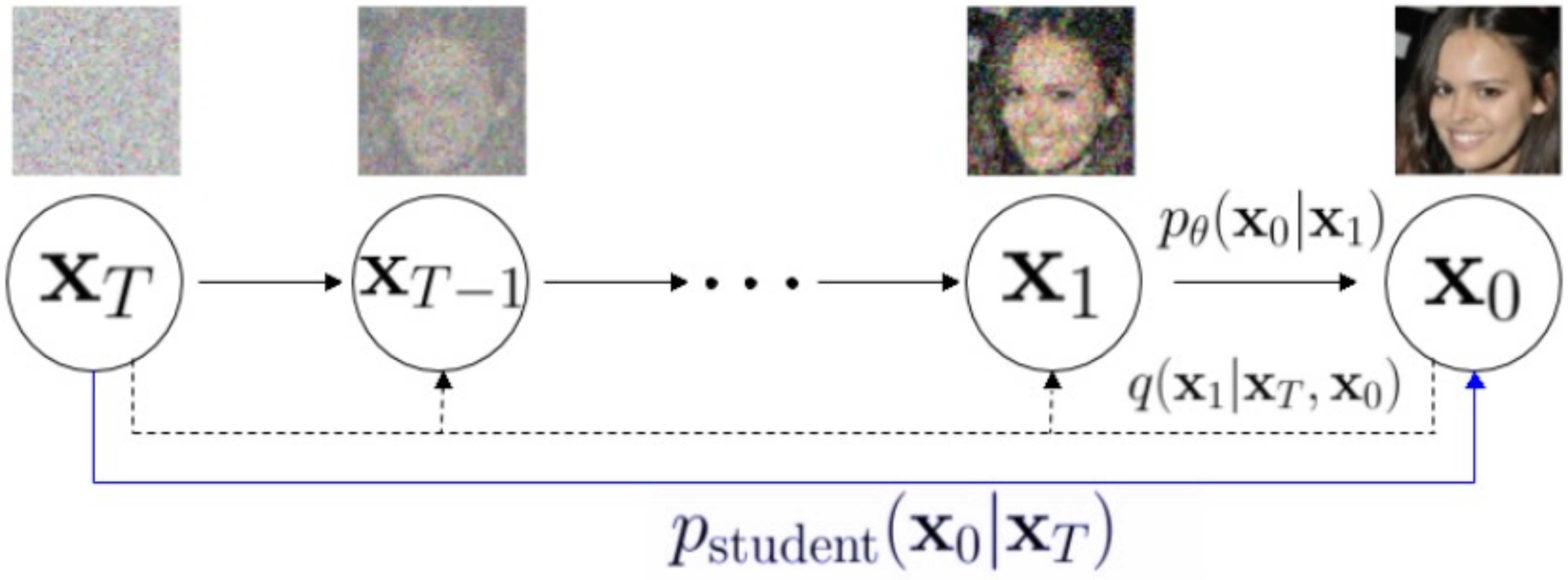
Consistency models

- Learning a consistency function on top of the teacher trajectories.

Naive Knowledge Distillation

1. Prepare pairs: $\{x_0^i,\ x_T^i\}$, where $x_T^i \sim N(0, I)$ and $x_T^i \ \rightarrow x_0^i$ using a teacher DPM ϵ_θ
2. Train a student model f_φ: $\left\| f_\varphi\left(x_T^i\right) - x_0^i \right\| \rightarrow \min_\varphi$

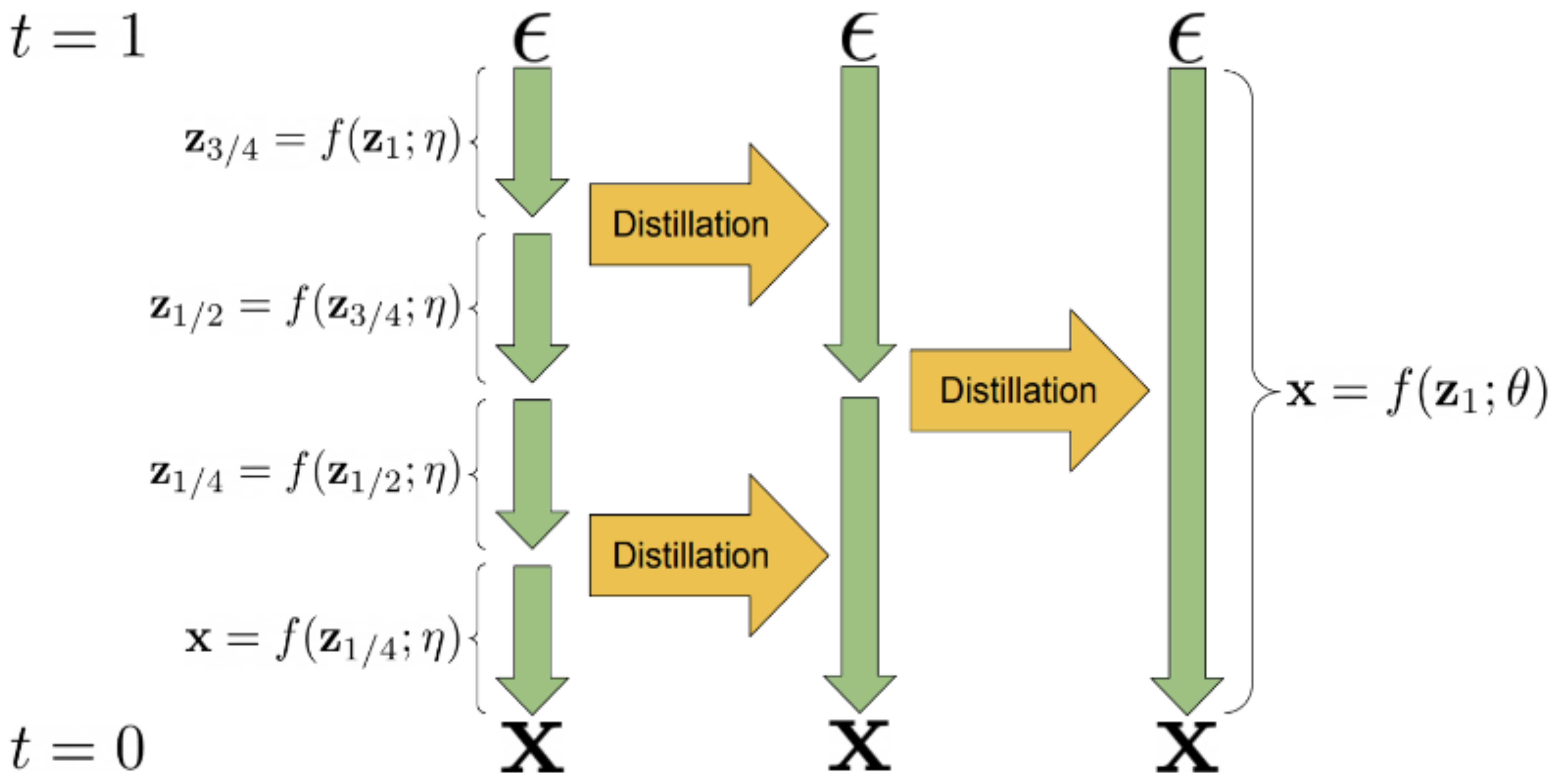
› Takeaways: single-stage, data-free, expensive training, low performance, single step.



Progressive Distillation

1. Train a student to approximate two subsequent teacher steps;
2. Student → teacher;
3. Go to 1 until desired # steps.

› Takeaways
1. Better quality than KD;
2. Multi-stage training;
3. Allows multiple steps at inference.



T. Salimans and J.Ho Progressive Distillation for Fast Sampling of Diffusion Models, ICLR2022

Distribution Matching Distillation

- Student $G_\theta(z)$, $z \sim N(0, I)$ represents $p_{fake}(x)$;

$$D_{KL}\left(p_{\text{fake}} \parallel p_{\text{real}}\right) = \underset{x \sim p_{\text{fake}}}{\mathbb{E}} \left(\log \left(\frac{p_{\text{fake}}(x)}{p_{\text{real}}(x)} \right) \right)$$

$$= \underset{\substack{z \sim \mathcal{N}(0;\mathbf{I}) \\ x = G_\theta(z)}}{\mathbb{E}} -\big(\log\ p_{\text{real}}(x) - \log\ p_{\text{fake}}(x) \big)$$

Distribution Matching Distillation

- Student $G_\theta(z),\ z \sim N(0, I)$ represents $p_{fake}(x)$;

$$D_{KL}\left(p_{\text{fake}} \parallel p_{\text{real}}\right) = \underset{x \sim p_{\text{fake}}}{\mathbb{E}} \left(\log \left(\frac{p_{\text{fake}}(x)}{p_{\text{real}}(x)} \right) \right)$$
$$= \underset{\substack{z \sim \mathcal{N}(0;\mathbf{I}) \\ x = G_\theta(z)}}{\mathbb{E}} -\big(\log\ p_{\text{real}}(x) - \log\ p_{\text{fake}}(x) \big)$$

$$\nabla_\theta D_{KL} = \underset{\substack{z \sim \mathcal{N}(0;\mathbf{I}) \\ x = G_\theta(z)}}{\mathbb{E}} \left[-\big(s_{\text{real}}(x) - s_{\text{fake}}(x) \big) \nabla_\theta G_\theta(z) \right]$$

$$s_{\text{real}}(x) = \nabla_x \log p_{\text{real}}(x),\ s_{\text{fake}}(x) = \nabla_x \log p_{\text{fake}}(x)$$

T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation

Key idea: approximate $\nabla_x D_{KL}$ using DPM forward passes

- Pretrained DPM teacher approximates $s_{real}(x_t, t) = \nabla_{x_t} log\, p_{real}(x_t)$;
- Auxiliary DPM is trained to approximate $s_{fake}(x_t, t) = \nabla_{x_t} log\, p_{fake}(x_t)$ during distillation.

$$\nabla_\theta D_{KL} \simeq \mathop{\mathbb{E}}_{z,t,x,x_t} \left[w_t \alpha_t \left(s_{\text{fake}}(x_t, t) - s_{\text{real}}(x_t, t) \right) \nabla_\theta G_\theta(z) \right],$$

$$z \sim \mathcal{N}(0; \mathbf{I}),\; x = G_\theta(z),\; t \sim \mathcal{U}(T_{\min}, T_{\max}),\; x_t \sim q_t(x_t | x)$$

Questions

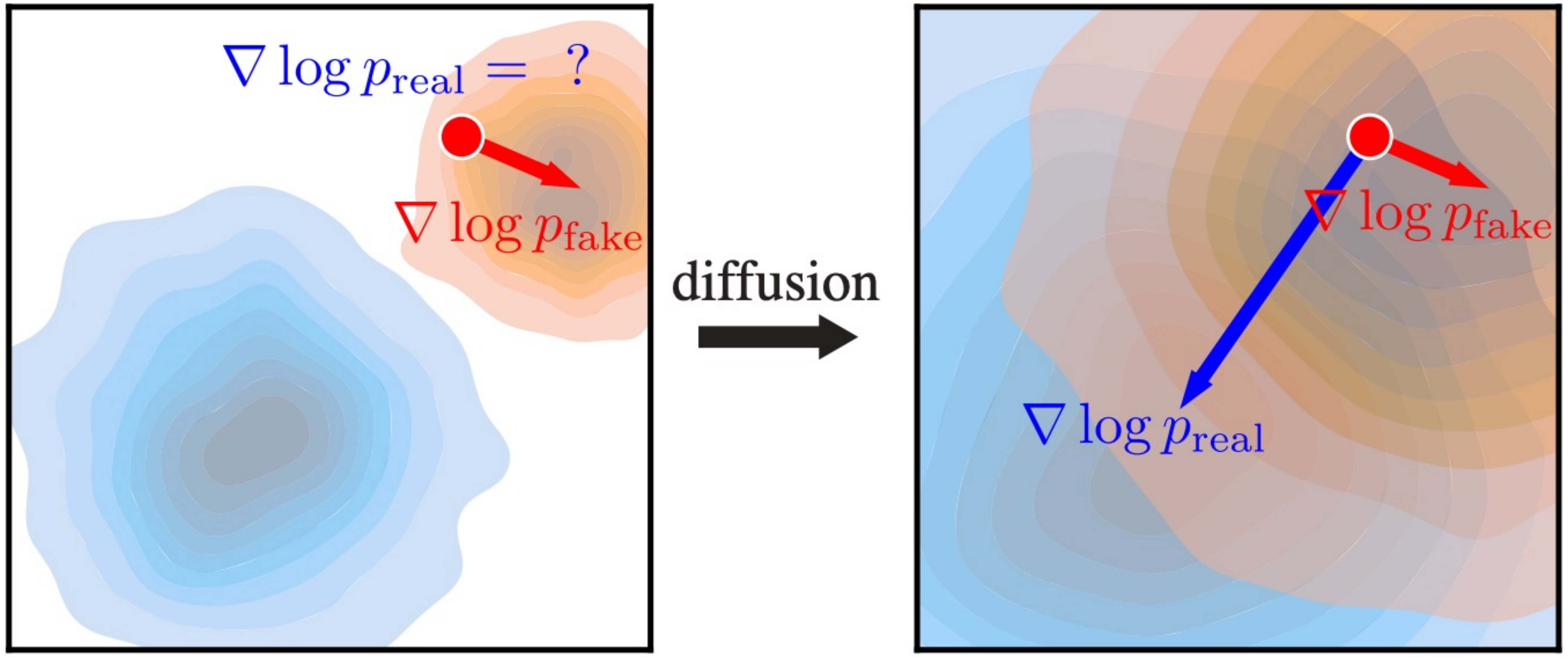
1. The effect of scores for $t \gg 0$;
2. Does reverse KL cause problems?
3. Can we avoid training the auxiliary DPM for $s_{fake}(x_t, t)$?

T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation

The effect of scores for $t \gg 0$

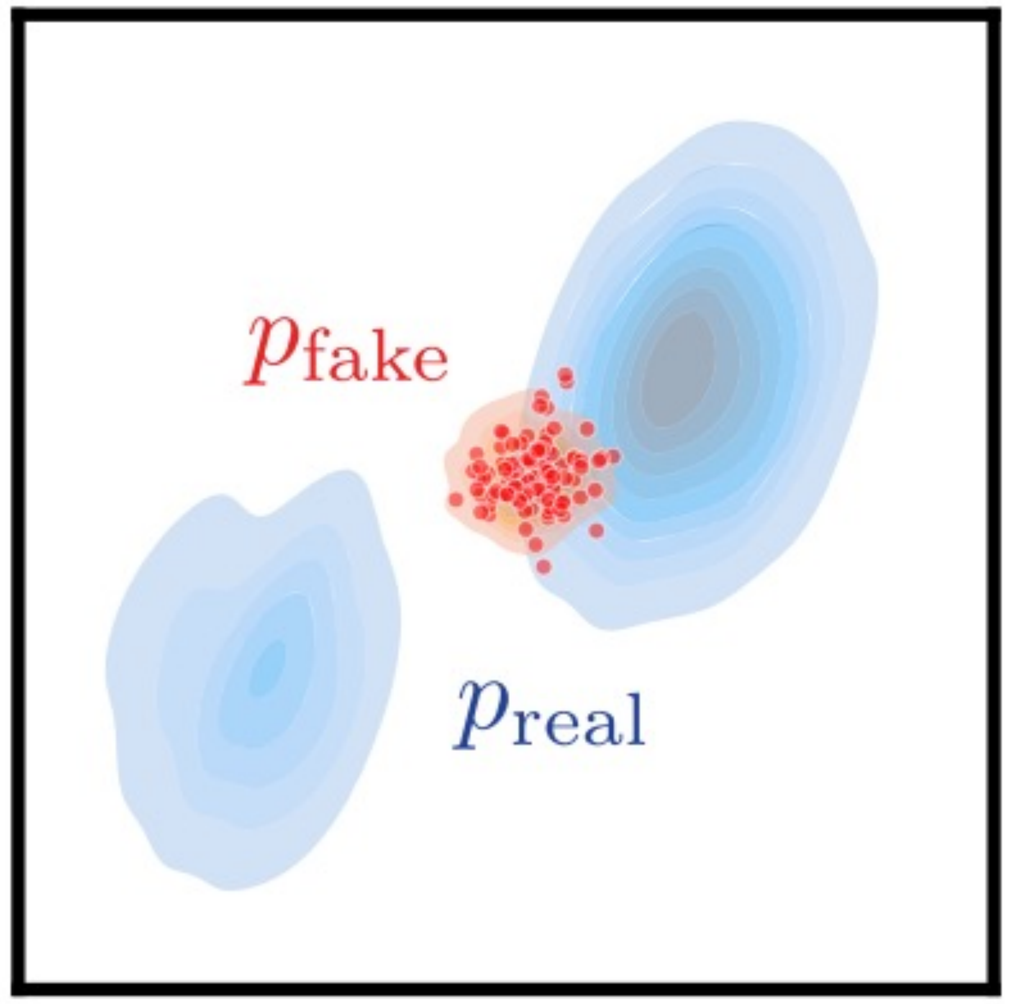
- $\nabla_{x_t} log\ p_{real}(x_t)$ and $\nabla_{x_t} log\ p_{fake}(x_t)$ for $t \gg 0$ help to avoid unreliable scores;



T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation

Reverse KL problem

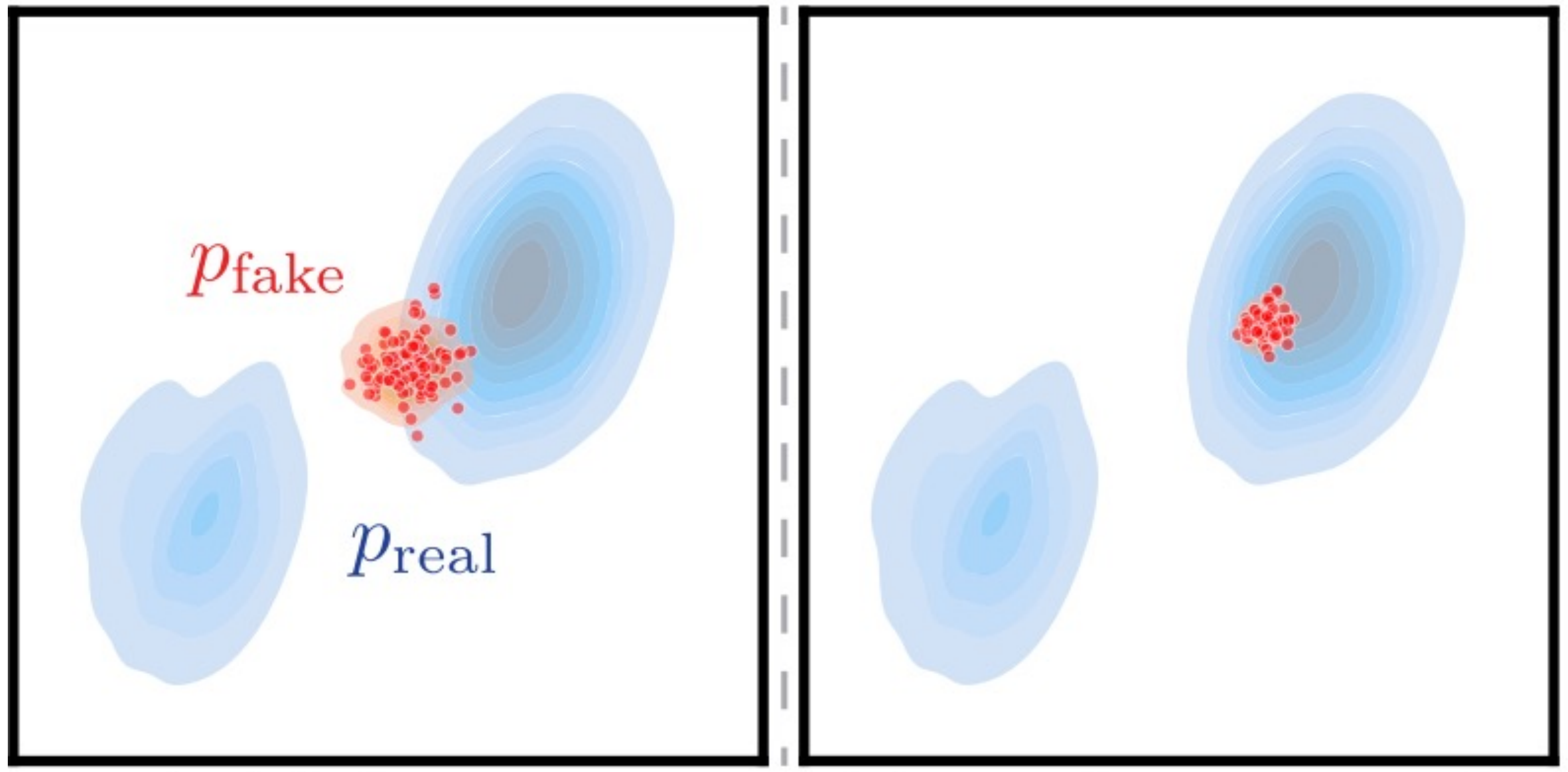


initial state

Distribution Matching Distillation

Reverse KL problem

$$\mathbb{E}_{\substack{z\sim\mathcal{N}(0;\mathbf{I})\\ x=G_\theta(z)}} -\left(\log\, p_{\text{real}}(x) - \cancel{\log\, p_{\text{fake}}(x)}\right)$$



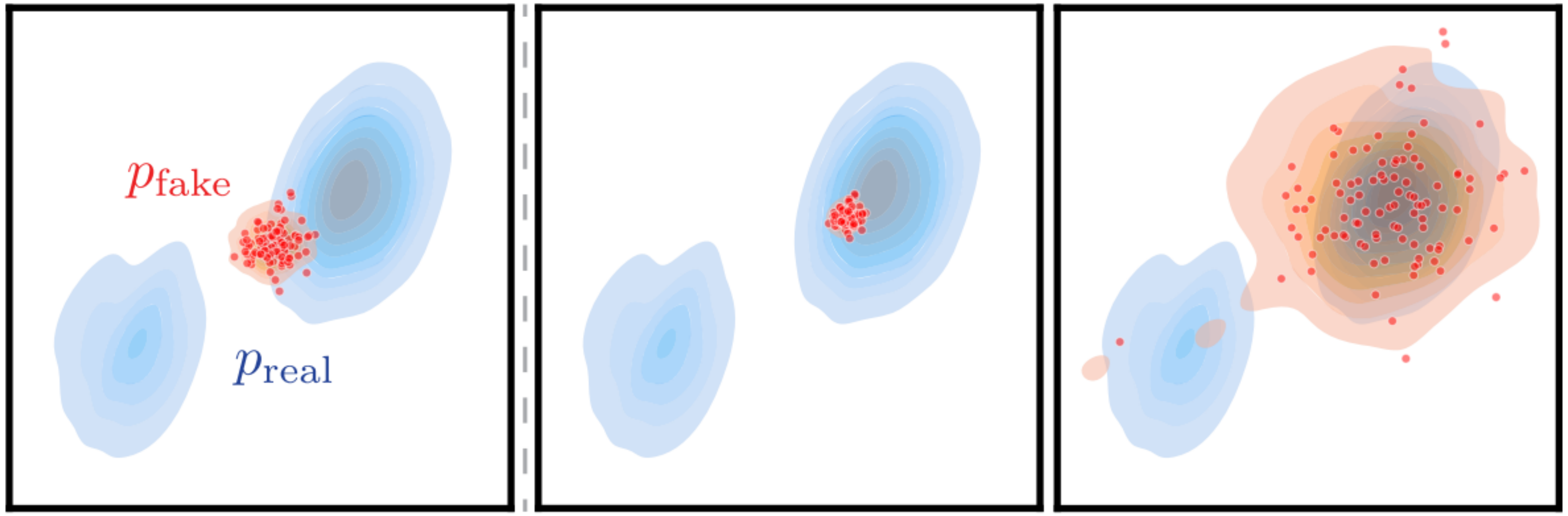
initial state (a) real score only

T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation

Reverse KL problem

$$\mathbb{E}_{\substack{z\sim\mathcal{N}(0;\mathbf{I})\\ x=G_\theta(z)}} -\left(\log\ p_{\text{real}}(x) - \log\ p_{\text{fake}}(x)\right)$$

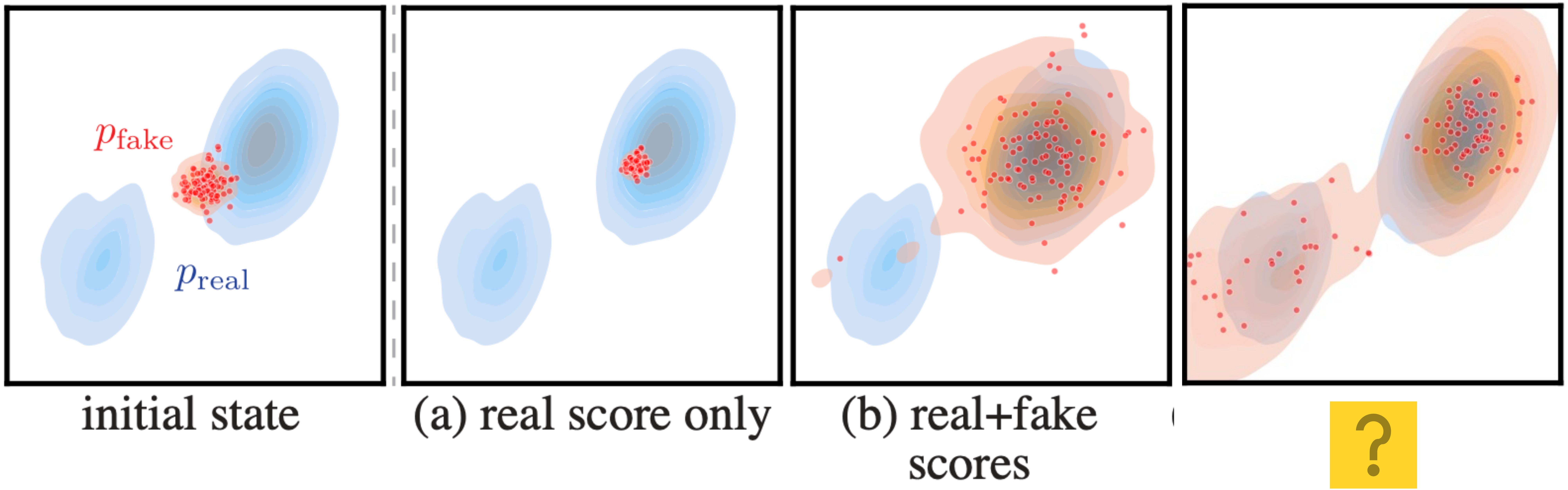


initial state | (a) real score only | (b) real+fake scores

T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation

How to mitigate mode collapse?

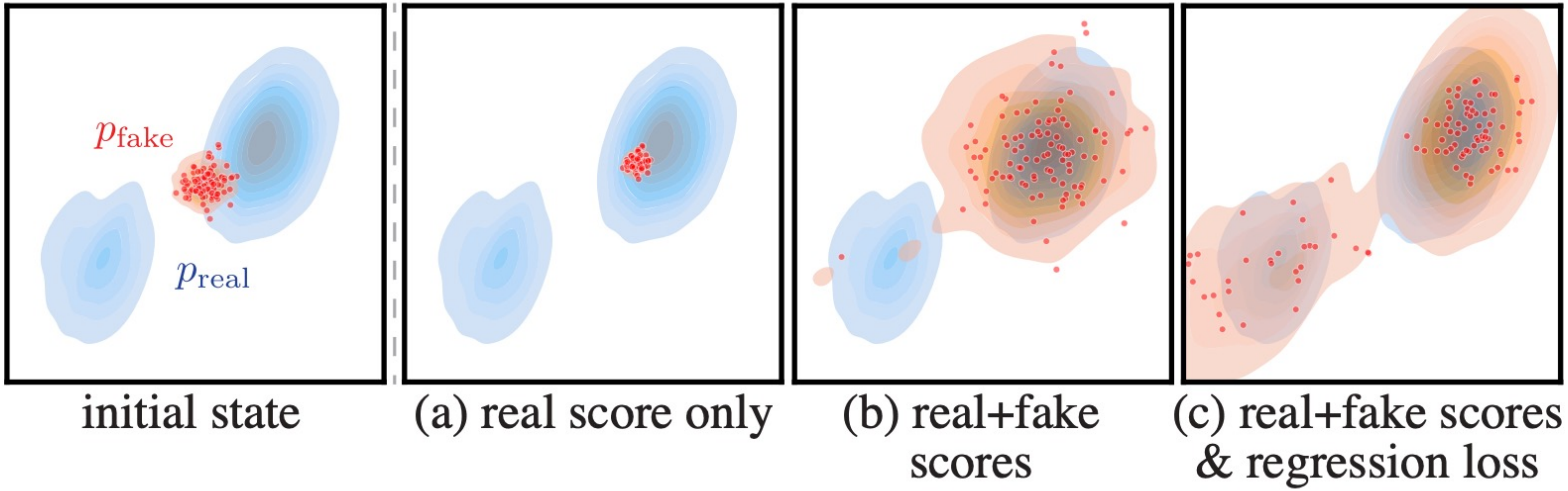


T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation

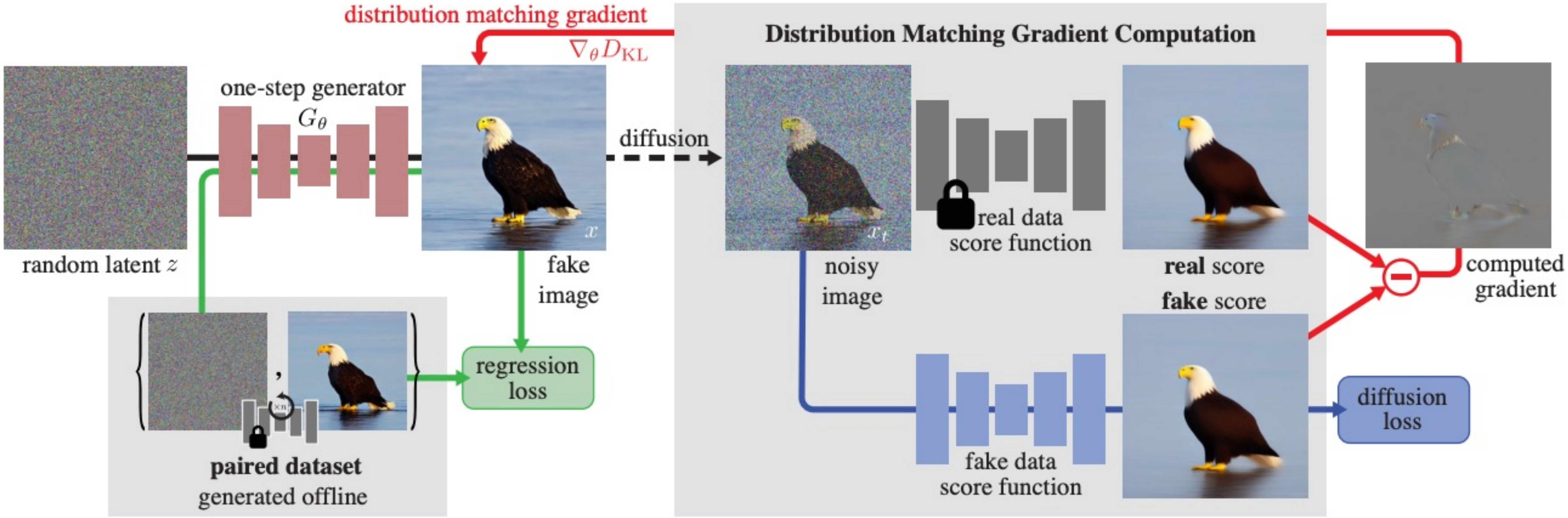
How to mitigate mode collapse?

- Combine with mode preserving distillation methods, e.g., naive distillation approach.



T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation



T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Distribution Matching Distillation | Summary

- Minimizing reverse KL using pretrained DPMs;
- Requires training an additional DPM during distillation;
- Requires additional distillation approaches to avoid mode collapse;
- The model is unavailable → unclear if it provides superior quality / diversity trade-off;
- Data-free.

T. Yin, et al, One-step Diffusion with Distribution Matching Distillation, CVPR2024

Score Distillation Sampling

How to avoid training the auxiliary DPM during distillation?

- Define $p_{fake}(x)$ as a set of delta distributions: $x^i = G_{\theta_i}$;
- $q(x_t|x^i) = N(x_t|\alpha_t G_{\theta_i}, \sigma_t^2 I) \rightarrow s_{fake}(x_t, t) = \nabla_{x_t} log\ q(x_t|x^i) \approx -(x_t - \alpha_t G_{\theta_i}) / \sigma_t^2 = \boxed{-\epsilon / \sigma_t}$

$$\nabla_\theta D_{KL} \simeq \mathbb{E}_{x_t \sim q(x_t|x^i)} \left[w_t \alpha_t \left(\boxed{s_{\text{fake}}(x_t, t)} - s_{\text{real}}(x_t, t) \right) \nabla_\theta G_\theta(z) \right]$$

Questions

- What are the key differences with DMD?
- What is the role of $s_{fake}(x_t, t)$ in SDS? Does it regularize the training?
- How DMD stands against SDS in terms of text-to-image performance?

B. Poole, et al., DreamFusion: Text-to-3D using 2D Diffusion, ICLR2023

Unbiased Score Estimator

How to avoid training the auxiliary DPM during distillation?

- **Idea:** estimate $\nabla_{x_t} log\, p_{fake}(x_t)$ using unbiased score estimate from synthetic data:

$$\nabla_{x_t} \log p_{fake}(x_t) = -\mathbb{E}\left[\frac{x - \alpha_t x_t}{\sigma_t^2} | x_t\right]$$

- where $x \sim p_{fake}$, $x_t \sim \mathrm{q}(x_t|\mathrm{x})$
- Single point estimate: $\nabla_{x_t} log\, p_{fake}(x_t) \approx -(x_t - \alpha_t \mathrm{x})/\sigma_t^2$
- The estimate may be more accurate with more samples from p_{fake}, see [1].

Questions

- How many samples are needed for the reasonable estimate?
- Would it be more efficient sampling these samples using $G_\theta(z)$ at each training iteration?

[1] M. Neidoba, et al, Nearest Neighbour Score Estimators for Diffusion Generative Models, arXiv2024

Adversarial Diffusion Distillation

ADD combines SDS and GAN objectives

1. Impressive image quality and text-alignment for 1-4 steps;
2. Both objectives lead to mode collapse → very poor image diversity for a given prompt.

A. Sauer et al., Adversarial Diffusion Distillation, 2023

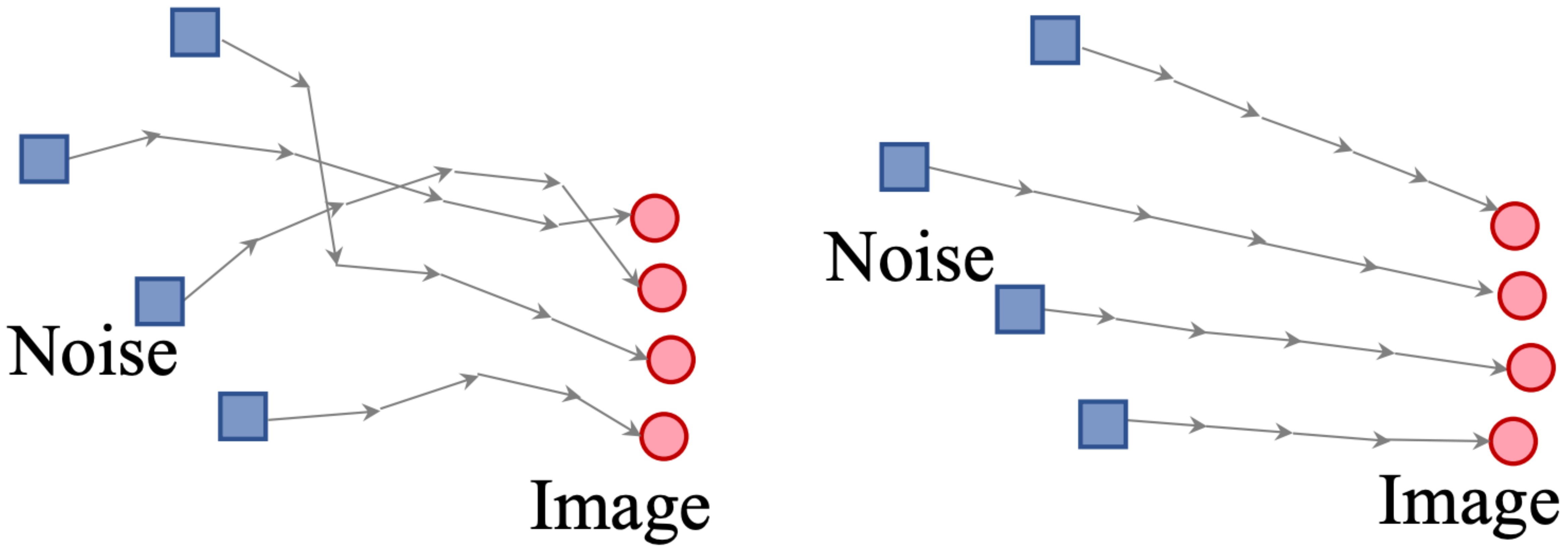
Rectified Flows

General idea

- Perform k stages of rectifying the trajectories of the pretrained DPM;
- Distill the resultant ”flow” model using one of the classical distillation methods;

Motivation

- DPM trajectory rectification simplifies the learned mapping;
- Straighter trajectories lead to lower error of ODE solvers and are easier to distill.



X. Liu et al, Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, ICLR2023
X. Liu et al, InstaFlow: One Step is Enough for High-Quality Diffusion-Based Text-to-Image Generation, 2023

Rectified Flows | Formulation

Notation

- π_0 – real distribution, π_1 – standard Gaussian distribution;
- $v(Z_t, t)$ – velocity function, $t \in [0, 1]$.

Rectified flow

$$\frac{\mathrm{d}Z_t}{\mathrm{d}t} = v(Z_t, t), \quad \text{initialized from } Z_0 \sim \pi_0, \text{ such that } Z_1 \sim \pi_1$$

Objective

$$\min_v \int_0^1 \mathbb{E}\left[\left\|(X_1 - X_0) - v\big(X_t,\ t\big)\right\|^2\right] \mathrm{d}t, \quad \text{with} \quad X_t = tX_1 + (1-t)X_0$$

X. Liu et al., Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, ICLR2023
X. Liu et al., InstaFlow: One Step is Enough for High-Quality Diffusion-Based Text-to-Image Generation, 2023

Rectified Flows | Algorithm

Preparation

- Initialize $v_\theta(Z_t, t)$ with the teacher parameters θ;
- Construct initial pairs of $(X_{0,}\ X_1) \sim \pi_0 \times \pi_1$ using PF-ODE of the pretrained DPM.

Procedure: $\boldsymbol{Z} = \texttt{RectFlow}((X_0, X_1))$:

Inputs: Draws from a coupling (X_0, X_1) of π_0 and π_1; velocity model $v_\theta : \mathbb{R}^d \to \mathbb{R}^d$ with parameter θ.

Training: $\hat{\theta} = \arg\min_{\theta} \mathbb{E}\left[\|X_1 - X_0 - v(tX_1 + (1-t)X_0,\ t)\|^2\right]$, with $t \sim \text{Uniform}([0,1])$.

Sampling: Draw (Z_0, Z_1) following $\mathrm{d}Z_t = v_{\hat{\theta}}(Z_t, t)\mathrm{d}t$ starting from $Z_0 \sim \pi_0$ (or backwardly $Z_1 \sim \pi_1$).

Return: $\boldsymbol{Z} = \{Z_t : t \in [0,1]\}$.

Reflow (optional): $\boldsymbol{Z}^{k+1} = \texttt{RectFlow}((Z_0^k, Z_1^k))$, starting from $(Z_0^0, Z_1^0) = (X_0, X_1)$.

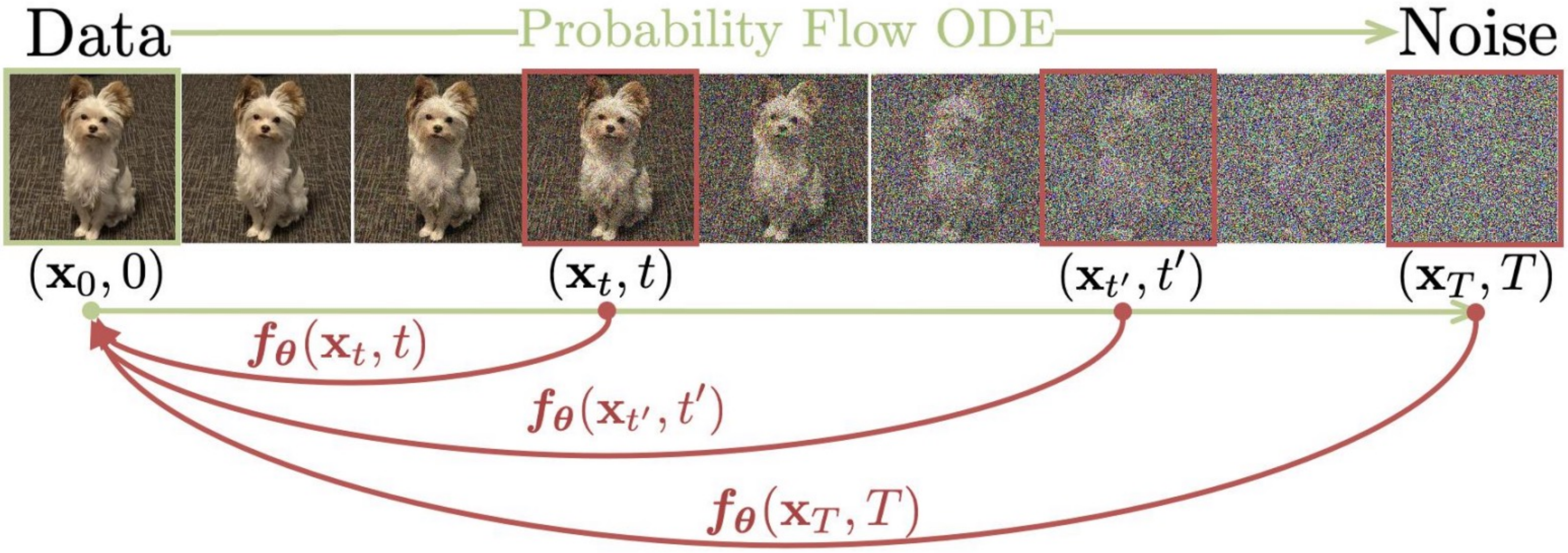
Distill (optional): Learn a neural network $\hat{T}$ to distill the k-rectified flow, such that $Z_1^k \approx \hat{T}(Z_0^k)$.

X. Liu et al., Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, ICLR2023
X. Liu et al, InstaFlow: One Step is Enough for High-Quality Diffusion-Based Text-to-Image Generation, 2023

Rectified Flows | Summary

- Clear and reasonable intuition behind the trajectory rectification;
- Multiple rectification stages (typically 2 stages) + distillation procedure;
- Not-established in practice (looking forward to the SD3 public release);

Consistency Distillation



Y. Song et al., Consistency Distillation, ICML2023

Consistency Distillation | Formulation

Consistency function

- Given a solution trajectory $\{x_t\}_{t\in[\epsilon,T]}$ define $\boldsymbol{f} : (\mathbf{x}_t, t) \mapsto \mathbf{x}_\epsilon$
- Self-consistency property: $\boldsymbol{f}(\mathbf{x}_t, t) = \boldsymbol{f}(\mathbf{x}_{t'}, t')$ for all $t, t' \in [\epsilon, T]$
- Boundary condition: $\boldsymbol{f}(\mathbf{x}_\epsilon, \epsilon) = \mathbf{x}_\epsilon$

Y. Song et al., Consistency Distillation, ICML2023

Consistency Distillation | Training

1. Parameterize f_θ with the teacher DPM;

2. Sample $x \sim p_{data}(x),\ x_{t_n} \sim q(x_{t_n}|x)$;

3. Obtain $x^{\phi}_{t_{n-1}}$ using ODE solver with the teacher model ϕ:

$$x^{\phi}_{t_{n-1}} \leftarrow x_{t_n} - (t_{n-1} - t_n) t_n \boldsymbol{s}_\phi(x_{t_n}, t_n)$$ (Euler step)

Objective

$$\mathcal{L}_{\mathcal{CD}} = \mathop{\mathbb{E}}_{x \sim p_{data}} \lambda(t_n) d(f_\theta(x_{t_n}, t_n), f_\theta(x^{\phi}_{t_{n-1}}, t_{n-1}))$$

- $\lambda = 1$ in practice. d(·) – arbitrary distance function, e.g., L1, L2, lpips.

Question: what is intuition behind this objective?

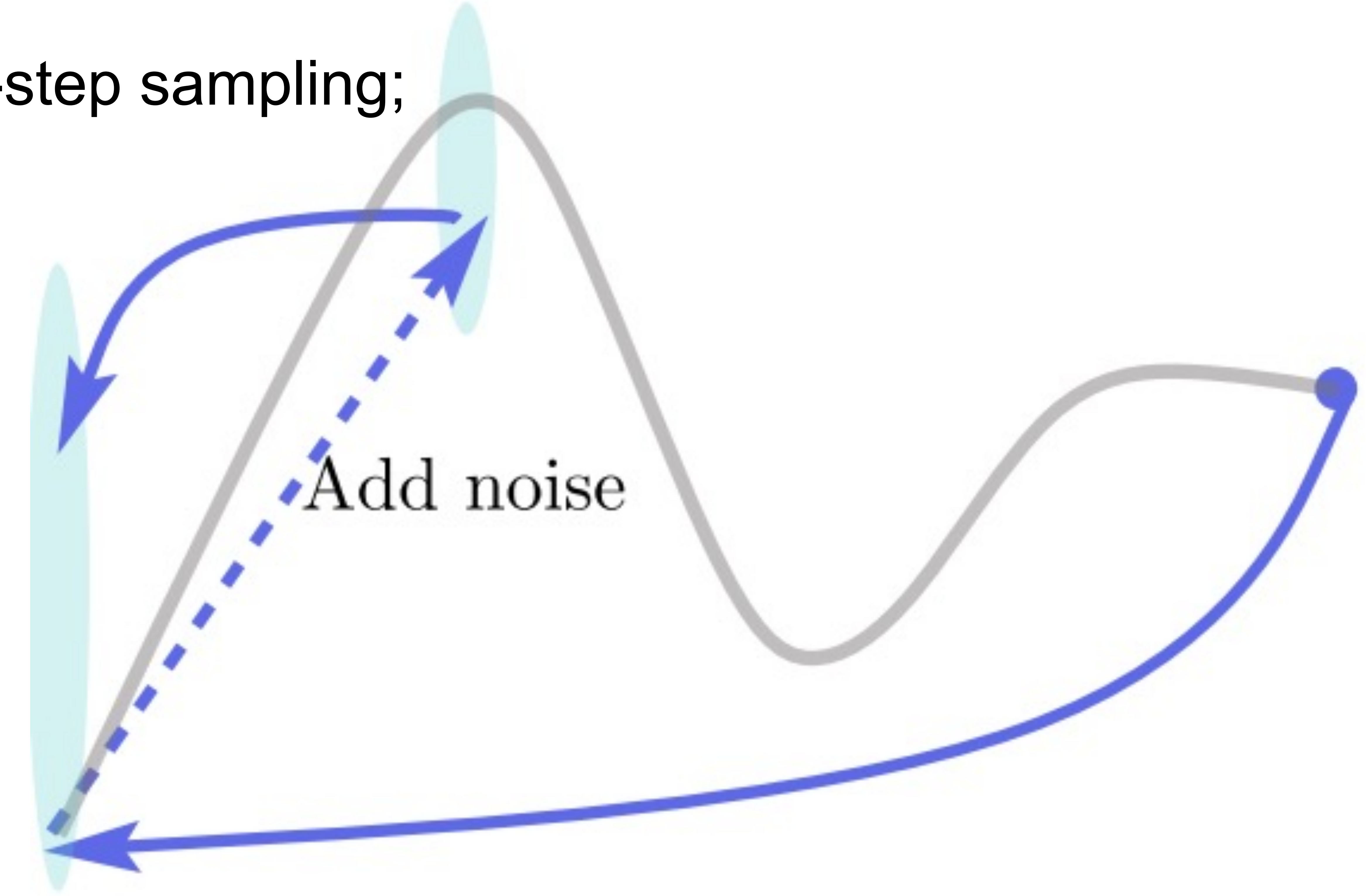
Consistency Distillation | Sampling

Initialization

- Sample $x_{t_N} \sim N(0, I)$;
- Select intermediate time steps $\{t_n\}$ for multi-step sampling;

Algorithm

1. Estimate $x_{t_0}^{\theta}$ for one step using $f_\theta\,(x_{t_n}, t_n)$;
2. Add noise $x_{t_n} \sim\, q\left(x_{t_n} |\, x_{t_0}^{\theta}\right)$;
3. Go to 1.



Consistency Distillation | Summary

1. Single-stage integrator-learning method that may have some interesting interpretations;
2. Uses real data during distillation;
3. Does not support deterministic multi-step sampling;
4. Lower fidelity and much higher diversity compared to the collapsed alternatives.

Questions

- Is it important how to approximate $\nabla_{x_{t_n}} log\, p_{t_n}(x_{t_n})$ to get $x^{\phi}_{t_{n-1}}$?
- Can we come up with the deterministic sampling?
- Do we need real data?

Y. Song et al., Consistency Distillation, ICML2023

Consistency Trajectory Models

Consistency models

- Left endpoint is always fixed → does not support integrating arbitrary trajectory intervals;
- Performance degrades with more sampling steps;
- Does not support deterministic multi-step sampling.

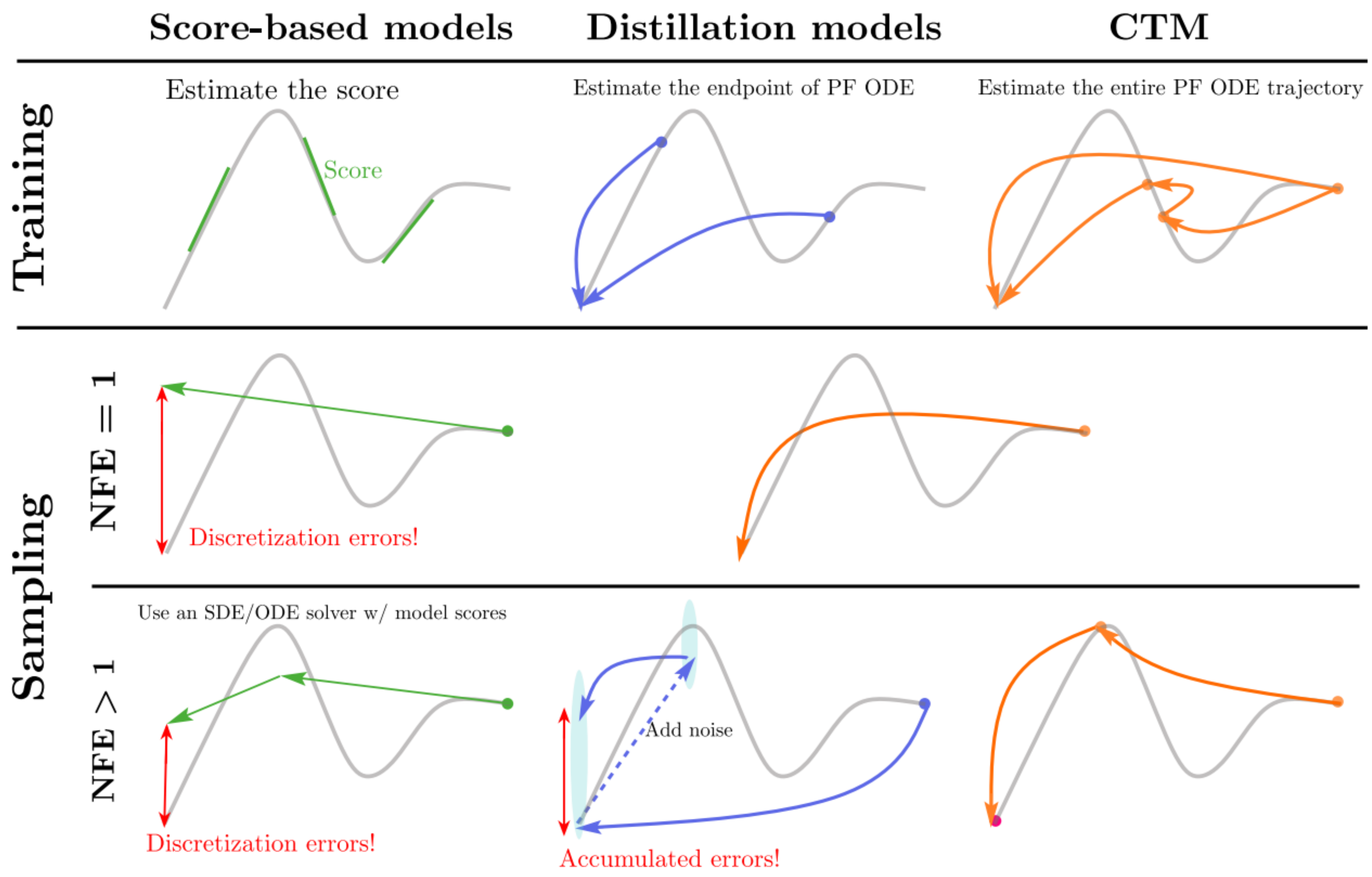
Consistency Trajectory Models

- Generalize consistency models by learning a more versatile integrator:

1. Integrate arbitrary sub-trajectories;
2. Recover the score function when $\Delta t \to 0$ → allow using ODE solvers;
3. Allows deterministic multi-step sampling.

D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Consistency Trajectory Models



D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Consistency Trajectory Models | Training

- Define an integrator of the trajectory interval $\{x_u\}_{u \in [s,t]}$

$$G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s) \approx \texttt{Solver}(\mathbf{x}_t, t, s; \boldsymbol{\phi}) \approx G(\mathbf{x}_t, t, s)$$

$$G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s) = \frac{s}{t}\mathbf{x}_t + \left(1 - \frac{s}{t}\right) g_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s)$$

Consistency Trajectory Models | Training

- Define an integrator of the trajectory interval $\{x_u\}_{u\in[s,t]}$

$$G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s) \approx \texttt{Solver}(\mathbf{x}_t, t, s; \boldsymbol{\phi}) \approx G(\mathbf{x}_t, t, s)$$

$$G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s) = \frac{s}{t}\mathbf{x}_t + \left(1 - \frac{s}{t}\right) g_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s)$$

Soft consistency matching

$$G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s) \approx G_{\texttt{sg}(\boldsymbol{\theta})}\big(\texttt{Solver}(\mathbf{x}_t, t, u; \boldsymbol{\phi}), u, s\big)$$

- *Local consistency:* $\mathrm{u} = \mathrm{t} - \Delta\mathrm{t}$ (Similar to CD);
- *Global consistency:* $\mathrm{u} = \mathrm{s}$ apply teacher for the entire interval;

D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Consistency Trajectory Models | Training

CTM objective

$$\mathbf{x}_{\text{est}}(\mathbf{x}_t, t, s) := G_{\texttt{sg}(\boldsymbol{\theta})}\big(G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s), s, 0\big)$$

$$\mathbf{x}_{\text{target}}(\mathbf{x}_t, t, u, s) := G_{\texttt{sg}(\boldsymbol{\theta})}\Big(G_{\texttt{sg}(\boldsymbol{\theta})}\big(\texttt{Solver}(\mathbf{x}_t, t, u; \boldsymbol{\phi}), u, s\big), s, 0\Big)$$

$$\mathcal{L}_{\text{CTM}}(\boldsymbol{\theta}; \boldsymbol{\phi}) := \mathbb{E}_{t\in[0,T]}\mathbb{E}_{s\in[0,t]}\mathbb{E}_{u\in[s,t)}\mathbb{E}_{\mathbf{x}_0}\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\Big[d\big(\mathbf{x}_{\text{target}}(\mathbf{x}_t, t, u, s), \mathbf{x}_{\text{est}}(\mathbf{x}_t, t, s)\big)\Big]$$

D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Consistency Trajectory Models | Training

CTM objective

$$\mathbf{x}_{\text{est}}(\mathbf{x}_t, t, s) := G_{\text{sg}(\boldsymbol{\theta})}(G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s), s, 0)$$

$$\mathbf{x}_{\text{target}}(\mathbf{x}_t, t, u, s) := G_{\text{sg}(\boldsymbol{\theta})}(G_{\text{sg}(\boldsymbol{\theta})}\big(\texttt{Solver}(\mathbf{x}_t, t, u; \boldsymbol{\phi}), u, s\big), s, 0)$$

$$\mathcal{L}_{\text{CTM}}(\boldsymbol{\theta}; \boldsymbol{\phi}) := \mathbb{E}_{t\in[0,T]}\mathbb{E}_{s\in[0,t]}\mathbb{E}_{u\in[s,t)}\mathbb{E}_{\mathbf{x}_0}\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\Big[d\big(\mathbf{x}_{\text{target}}(\mathbf{x}_t, t, u, s), \mathbf{x}_{\text{est}}(\mathbf{x}_t, t, s)\big)\Big]$$

DSM objective

- For $s == t$, directly minimize the DPM objective → learning the score function.

$$\boxed{\mathcal{L}_{\text{DSM}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\mathbf{x}_0}\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}[\|\mathbf{x}_0 - g_{\boldsymbol{\theta}}(\mathbf{x}_t, t, t)\|_2^2]}$$

Consistency Trajectory Models | Training

CTM objective

$$\mathbf{x}_{\text{est}}(\mathbf{x}_t, t, s) := G_{\texttt{sg}(\boldsymbol{\theta})}(G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s), s, 0)$$

$$\mathbf{x}_{\text{target}}(\mathbf{x}_t, t, u, s) := G_{\texttt{sg}(\boldsymbol{\theta})}(G_{\texttt{sg}(\boldsymbol{\theta})}\big(\texttt{Solver}(\mathbf{x}_t, t, u; \boldsymbol{\phi}), u, s\big), s, 0)$$

$$\mathcal{L}_{\text{CTM}}(\boldsymbol{\theta}; \boldsymbol{\phi}) := \mathbb{E}_{t\in[0,T]}\mathbb{E}_{s\in[0,t]}\mathbb{E}_{u\in[s,t)}\mathbb{E}_{\mathbf{x}_0}\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\Big[d\big(\mathbf{x}_{\text{target}}(\mathbf{x}_t, t, u, s), \mathbf{x}_{\text{est}}(\mathbf{x}_t, t, s)\big)\Big]$$

DSM objective

› For $s == t$, directly minimize the DPM objective → learning the score function.

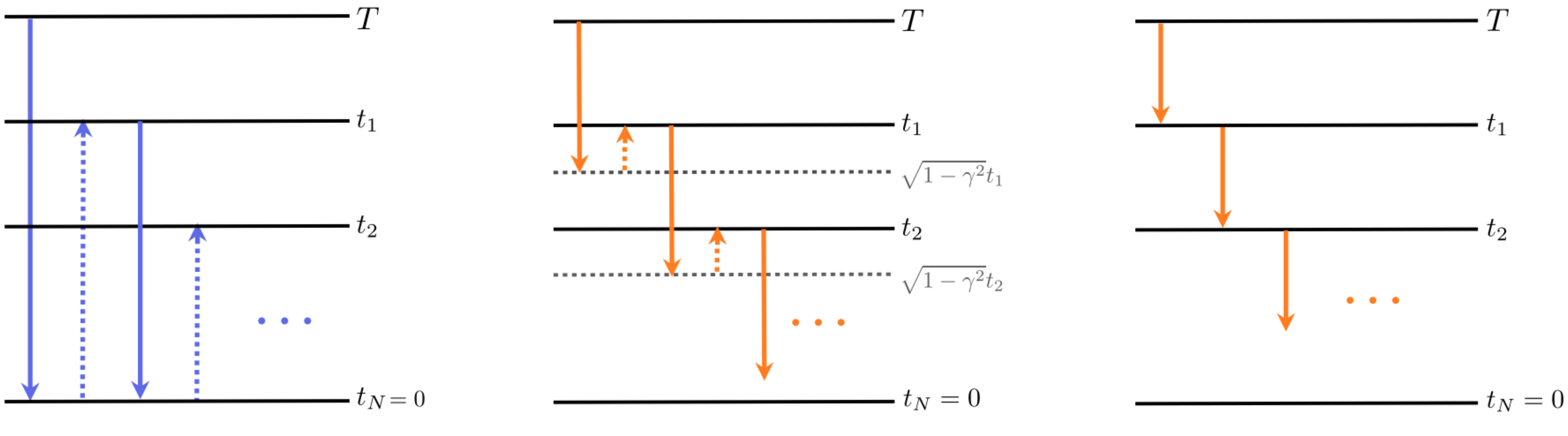
$$\mathcal{L}_{\text{DSM}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\mathbf{x}_0}\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}[\|\mathbf{x}_0 - g_{\boldsymbol{\theta}}(\mathbf{x}_t, t, t)\|_2^2]$$

Final objective

$$\boxed{\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}) := \mathcal{L}_{\text{CTM}}(\boldsymbol{\theta}; \boldsymbol{\phi}) + \lambda_{\text{DSM}}\mathcal{L}_{\text{DSM}}(\boldsymbol{\theta}) + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}}(\boldsymbol{\theta}, \boldsymbol{\eta})}$$

D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Consistency Trajectory Models | Sampling



(a) $\gamma = 1$ (Fully stochastic) (b) $1 > \gamma > 0$ (c) $\gamma = 0$ (Deterministic)

$\gamma = 0$ – provides the best performance in the single and multi-step settings.

D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Consistency Trajectory Models | Summary

- Generalize consistency models by learning a more versatile integrator;
- Unlocks deterministic sampling;
- Unlocks high-quality multi-step sampling;
- Training procedure is overloaded. Many details seem redundant. In our experiments, doing CD on individual intervals works fine.
- Expensive training due to global consistency that requires many teacher steps;

Questions

- Which of the proposed modifications are critical?

D. Kim et al., Consistency Trajectory Models: Learning Probability Flow ODE Trajectory of Diffusion, ICLR2024

Research Directions in Consistency Models

Bi-directional integrator

- Would like to learn a bi-directional integrator that allows both accurate and efficient inversion.

Intuition: ”integration” might be a localized and disentangled task

- CD requires relatively few steps for convergence → observes small amount of data;
- Small portion of weights are important during distillation;
- Distilled models can be readily pluged-and-played into different DPMs and editing methods;
- Can we distill faster and better if we focus on trajectory properties, e.g., curvature?

How to distill effectively for high CFG scales?

- Higher CFG scales lead to more curved trajectories → more difficult and unstable distillation.

Combine CD and RF ideas

- Generalize CMs to arbitrary vector fields and apply it to RF.