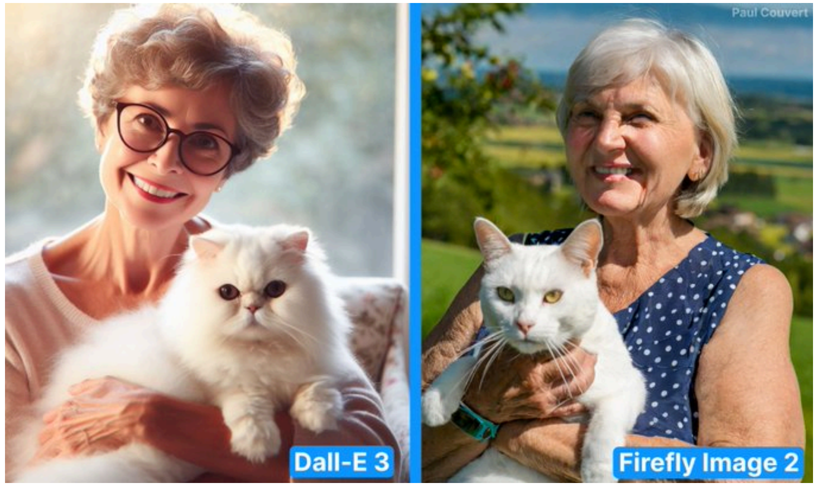
TEXAS
The University of Texas at Austin

InstaFlow: One-Step Stable Diffusion from Straight Probability Flows

Xingchao Liu

UT Austin

AI Generated Contents



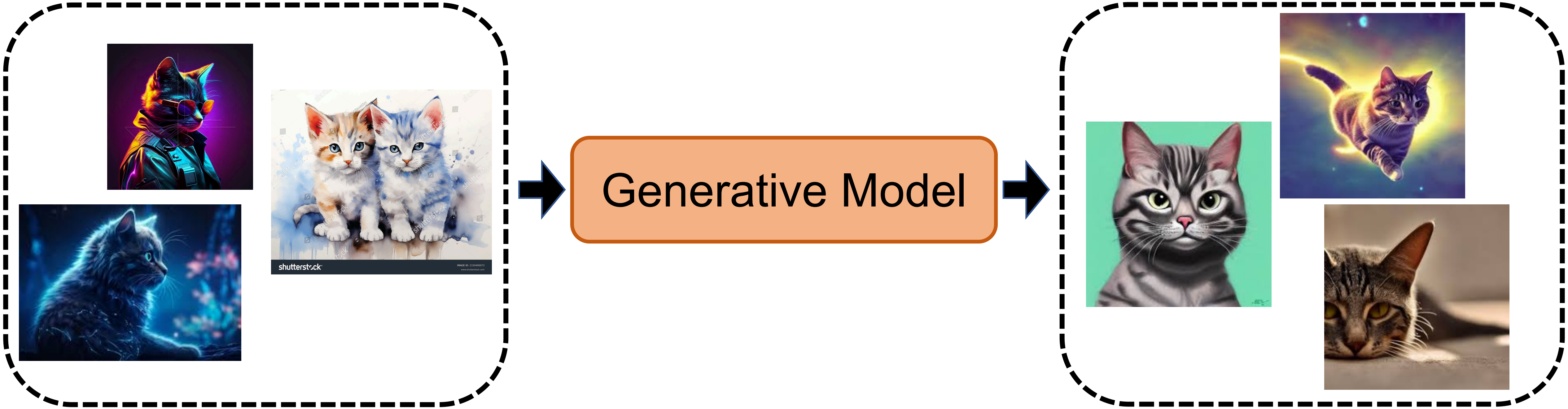
Images

Videos

Texts & Codes

Policies

AIGC Pipeline



Real-World Data

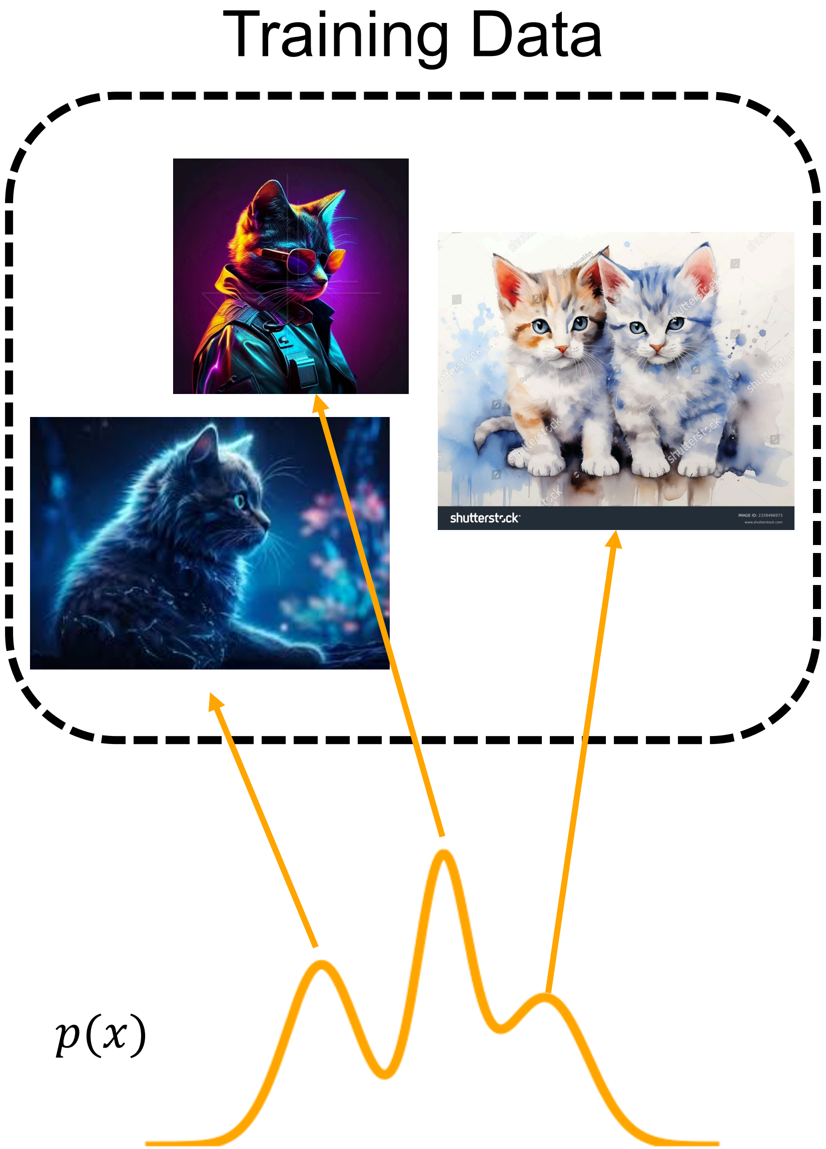
Training

Inference

Generative Models

Given: observed data points $\{x_i\}_{i=1}^{n}$

Unknown: the groundtruth data distribution $p(x)$



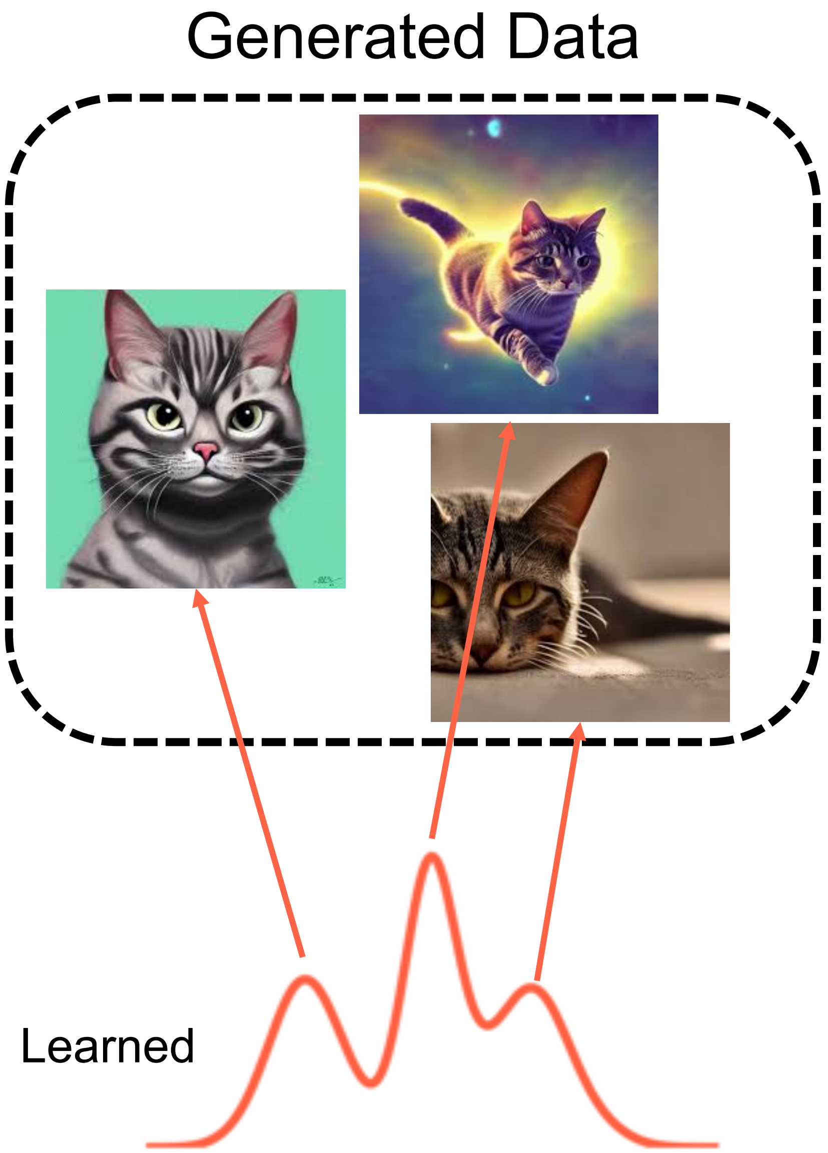
Generative Models

Given: observed data points $\{x_i\}_{i=1}^{n}$

Unknown: the groundtruth data distribution $p(x)$

Training: to learn a **model** to capture $p(x)$

Sampling: generate from the learned distribution



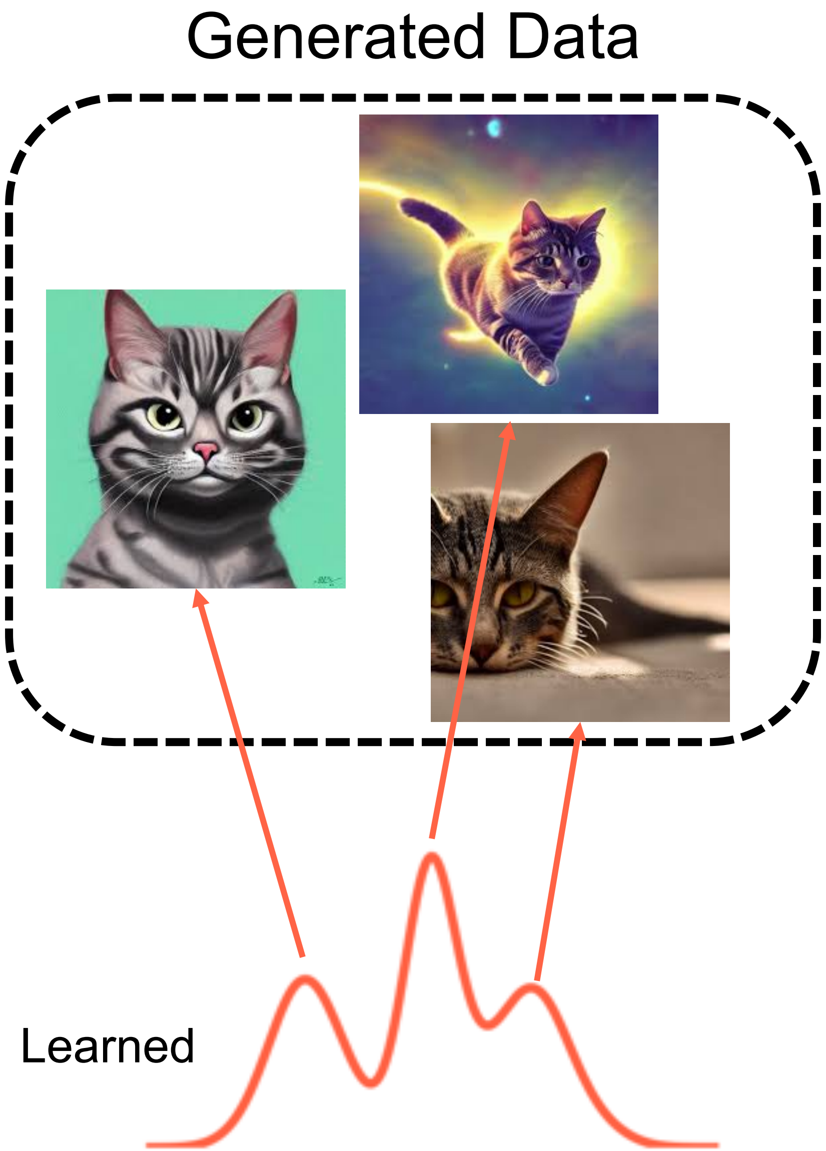
Generative Models

Given: observed data points $\{x_i\}_{i=1}^{n}$

Unknown: the groundtruth data distribution $p(x)$

Training: to learn a **model** to capture $p(x)$

Sampling: generate from the learned distribution



What do we expect for good generative model frameworks?

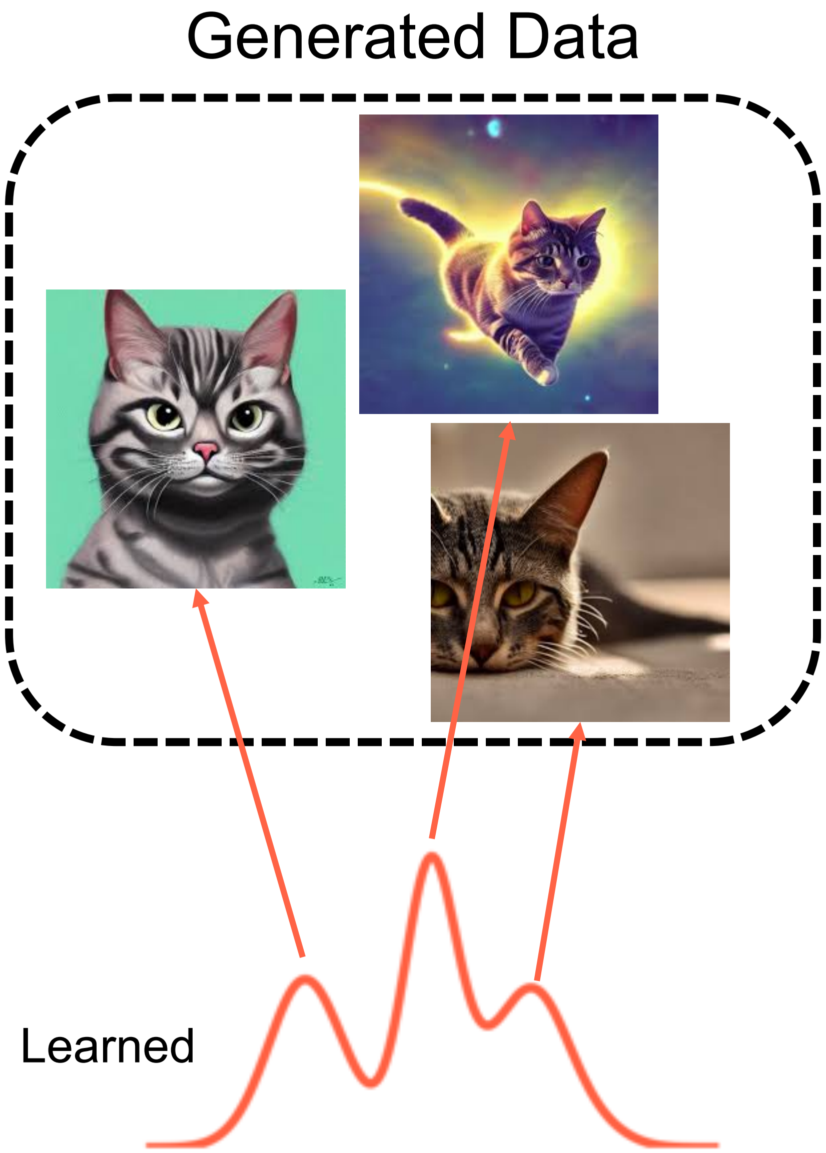
Generative Models

Given: observed data points $\{x_i\}_{i=1}^{n}$

Unknown: the groundtruth data distribution $p(x)$

Training: to learn a **model** to capture $p(x)$

Sampling: generate from the learned distribution



Efficient Training ← Many objectives are hard to optimize / of high variance

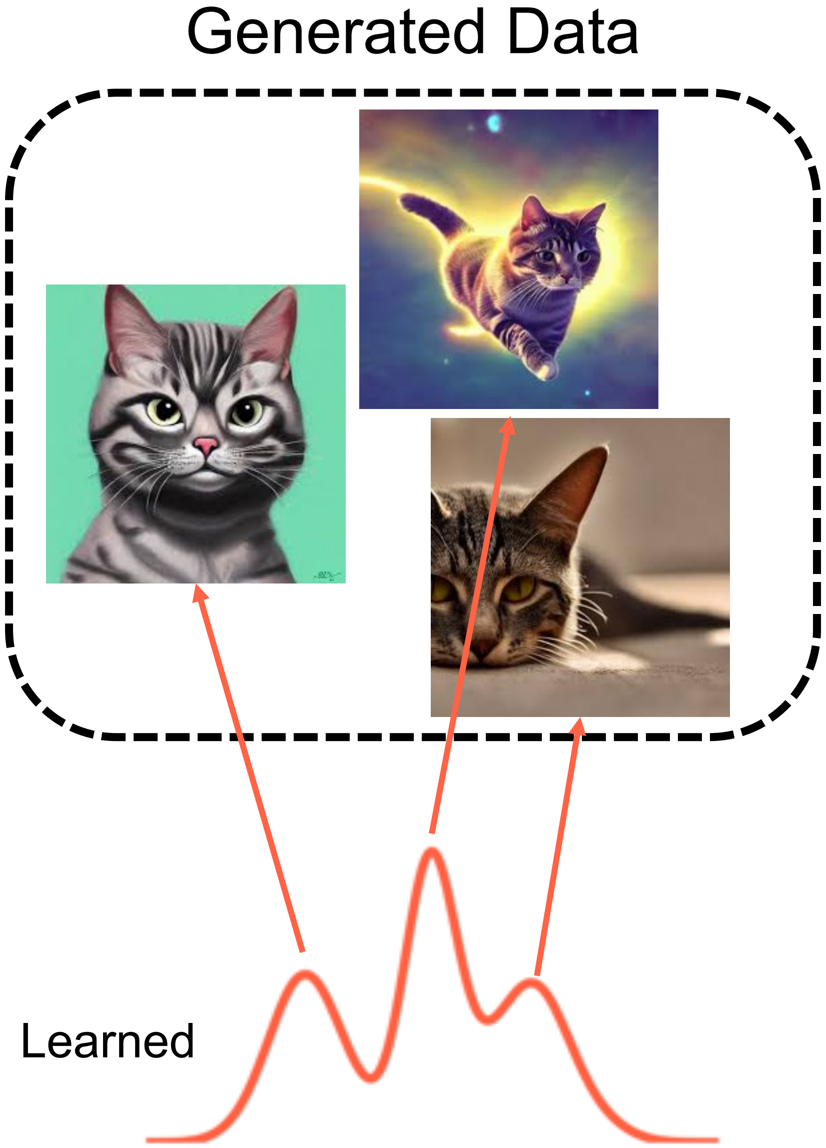
Generative Models

Given: observed data points $\{x_i\}_{i=1}^{n}$

Unknown: the groundtruth data distribution $p(x)$

Training: to learn a **model** to capture $p(x)$

Sampling: generate from the learned distribution



Efficient Sampling ← Sampling from general distributions is slow

[**Liu** et al., NeuIPS 2021 **spotlight**]
[Zhang, **Liu** et al., ICML 2022]

Frameworks of Generative Models

	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]	×	√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Frameworks of Generative Models

	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]	×	√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Frameworks of Generative Models

	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]	×	√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Frameworks of Generative Models

	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]	×	√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Frameworks of Generative Models

	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]	×	√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Frameworks of Generative Models

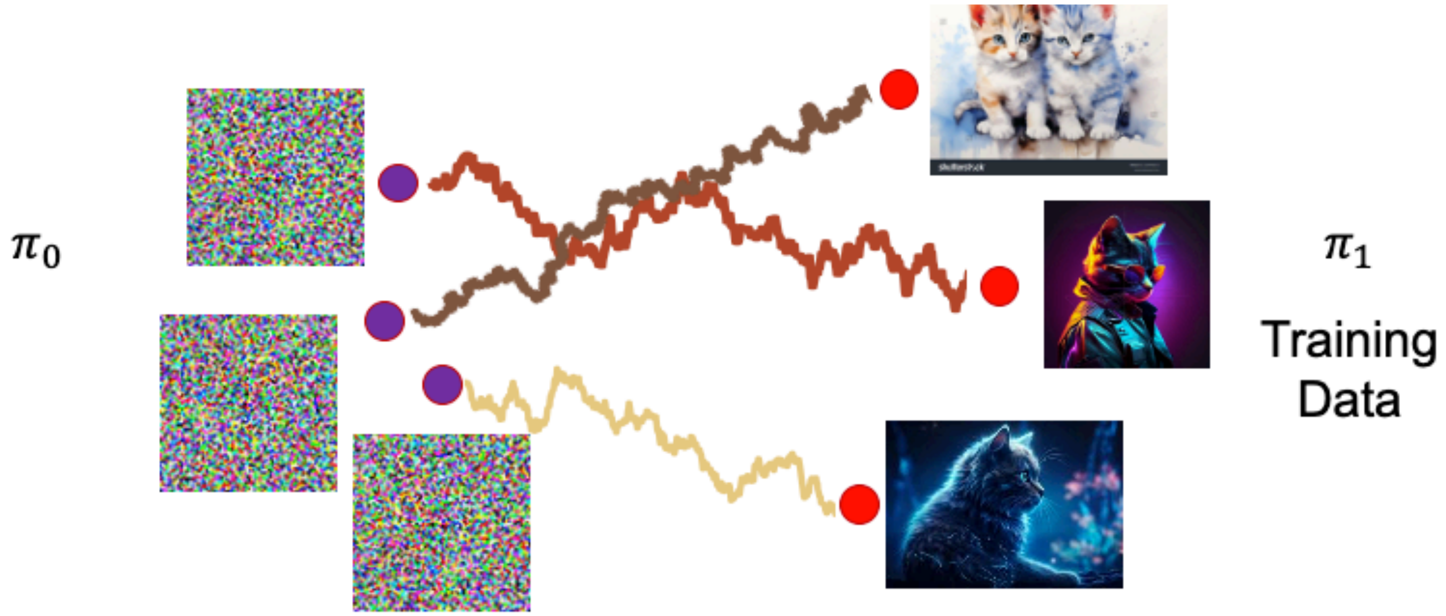
	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]	×	√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Frameworks of Generative Models

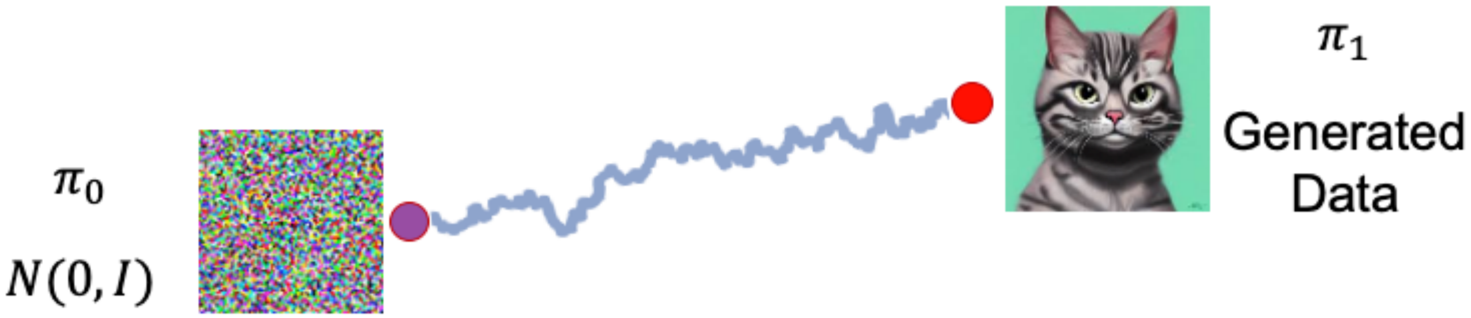
	Efficient Training	Efficient Sampling
Energy-Based Model [Hinton 1999, 2002]	×	×
Autoregressive Model [Frey 1998, Bengio & Bengio 2000]	√	×
GAN [Goodfellow et al. 2014]		√
VAE [Kingma & Welling 2014]	×	√
Normalizing Flow [Rezende & Mohamed 2015]	×	√
Diffusion Model [Sohl-Dickstein et al. 2015, Ho et al. 2020, Song et al. 2021]	√	×

Can we get both?

Diffusion Models



Training

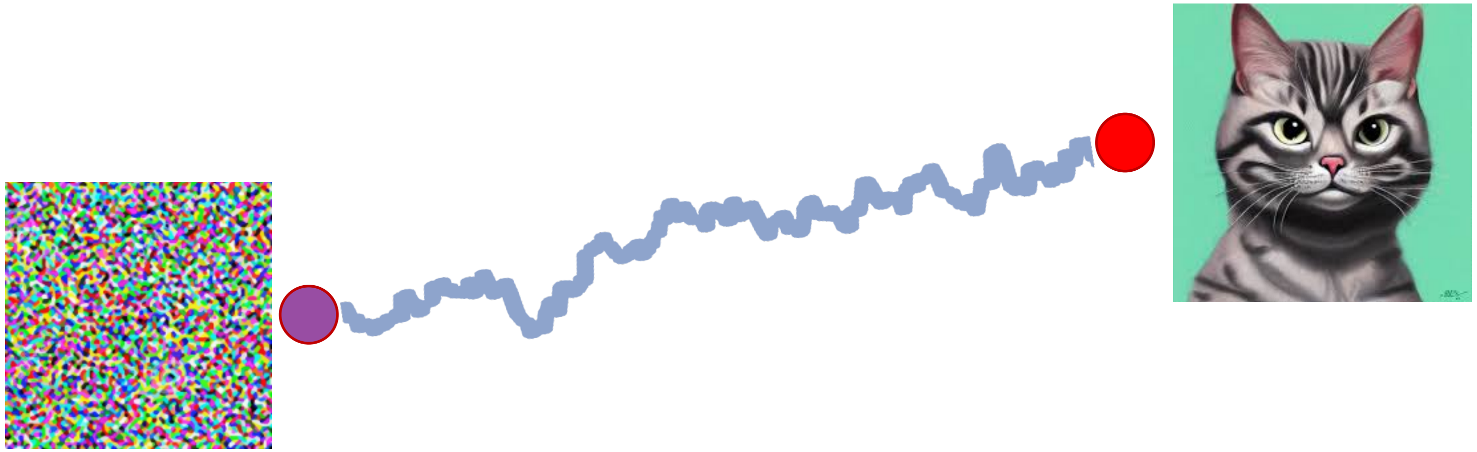


Sampling

Why are they slow?

π_0

$N(0, I)$



π_1

Generated Data

Problem: Noise in the diffusion process [**Liu** et al., ICLR2023 **spotlight**]

Solution: Marginal-preserving ordinary differential equation (ODE)

DDIM [Song et al. 2021], Heun [Karras et al. 2022], DPM-Solver [Lu et al. 2022], etc.

$$\mathrm{d}X = [f(X,t) - g^2(t)\nabla_X \log p_t(X)]\mathrm{d}t + \boxed{g(t)\mathrm{dW}t} \text{ Noise}$$

Reverse Stochastic Differential Equation (SDE)

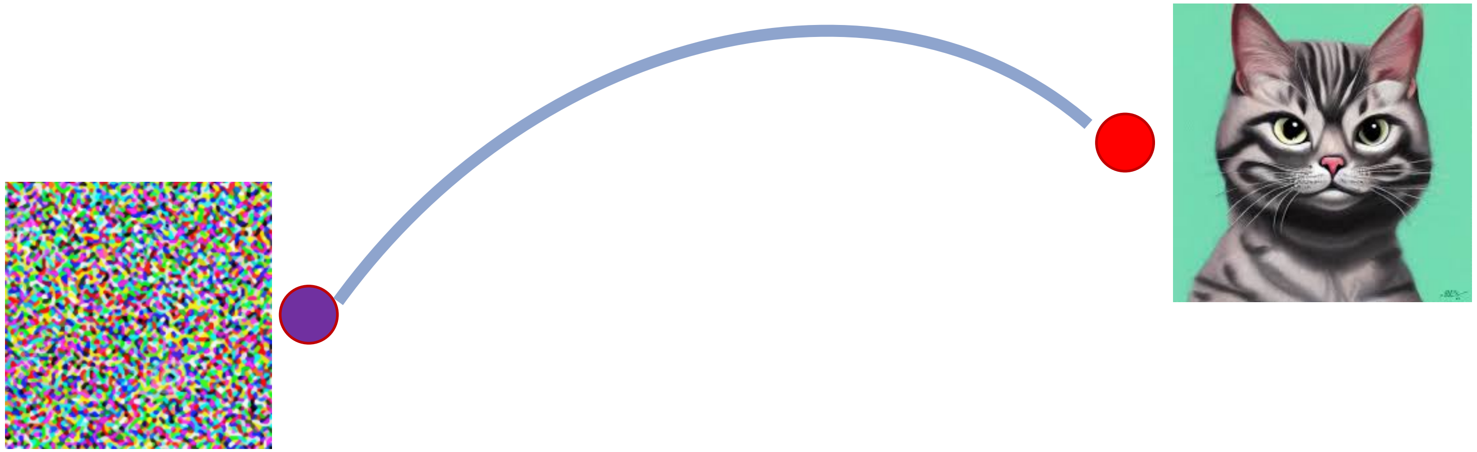
Why are they slow?

π_1

Generated Data

π_0

$N(0, I)$



Problem: Noise in the diffusion process [**Liu** et al., ICLR2023 **spotlight**]

Solution: Marginal-preserving ordinary differential equation (ODE)

DDIM [Song et al. 2021], Heun [Karras et al. 2022], DPM-Solver [Lu et al. 2022], etc.

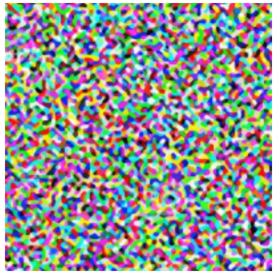
$$\mathrm{d}X = [f(X, t) - \frac{1}{2} g^2(t) \nabla_X \log p_t(X)]\mathrm{d}t$$

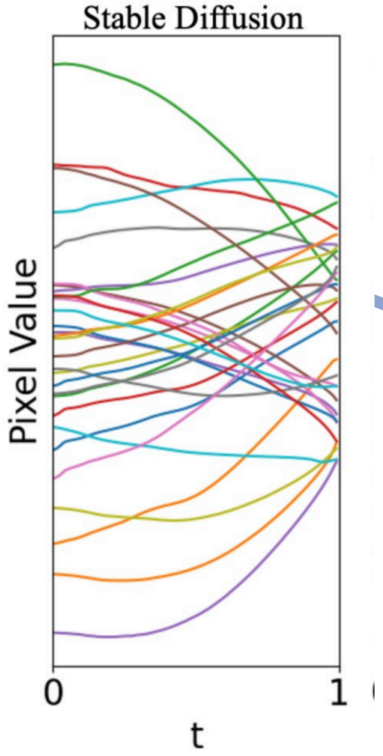
Probability Flow Ordinary Differential Equation

Why are they slow?

π_0

$N(0, I)$





π_1

Generated Data

New Problem: Curved ODE trajectory

Velocity $v(X,t)$

$$\mathrm{d}X = [\boxed{f(X,t) - \frac{1}{2} g^2(t) \nabla_X \log p_t(X)}]\mathrm{d}t$$

Probability Flow Ordinary Differential Equation

Discretization of ODE

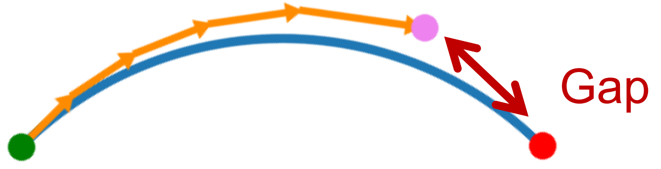
- In computer, we solve ODEs by Euler discretization

$$X_{t+\epsilon} = X_t + \epsilon\, v(X_t, t)$$

ϵ: step size

Large ϵ: Fast, inaccurate ; Small ϵ: Accurate, slow

Curved Trajectory



$N = 5, \epsilon = 1/N$

Straight Trajectory

$N = 1, \epsilon = 1/N$

$$\mathrm{d}X = v(X, t)\mathrm{d}t$$

Probability Flow Ordinary Differential Equation

Research Question

How do we learn straight generative ODEs?

Diffusion models connect two distribution with diffusion processes

Idea: Connect with straight lines!

Rectified Flow

- Learn from straight-line teachers
- Purely ODE-based; no more conversion from SDE to ODE
- A unified framework for both generative modeling and transfer learning
- Bridge the gap between one-step and continuous-time models **Reflow**

[**Liu** et al., ICLR2023 **spotlight**]

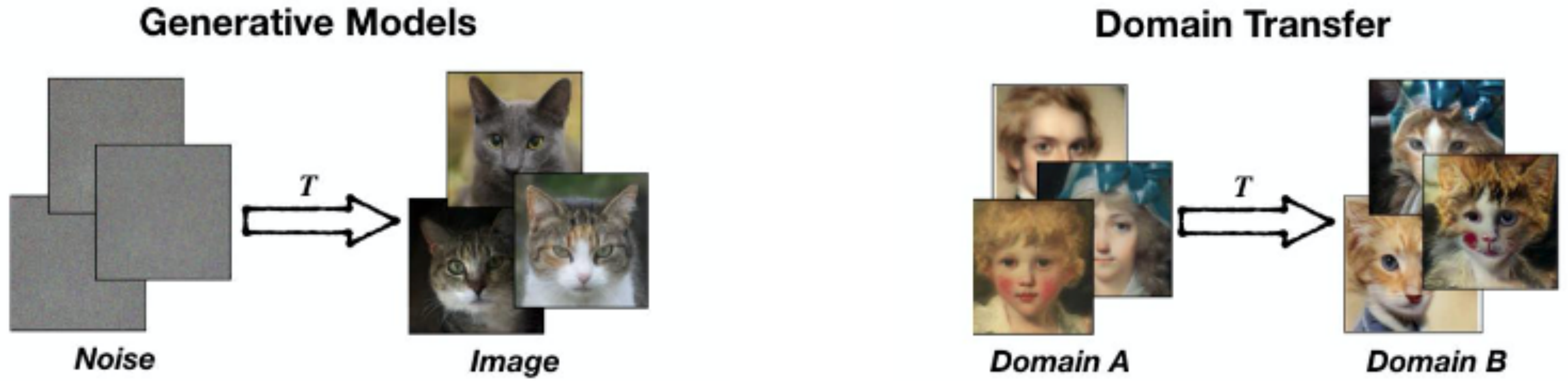
Rectified Flow: Problem of Interest

Given: observed data points from two distributions

$$\{x_i^0\}_{i=1}^n \sim \pi_0, \quad \{x_i^1\}_{i=1}^n \sim \pi_1$$

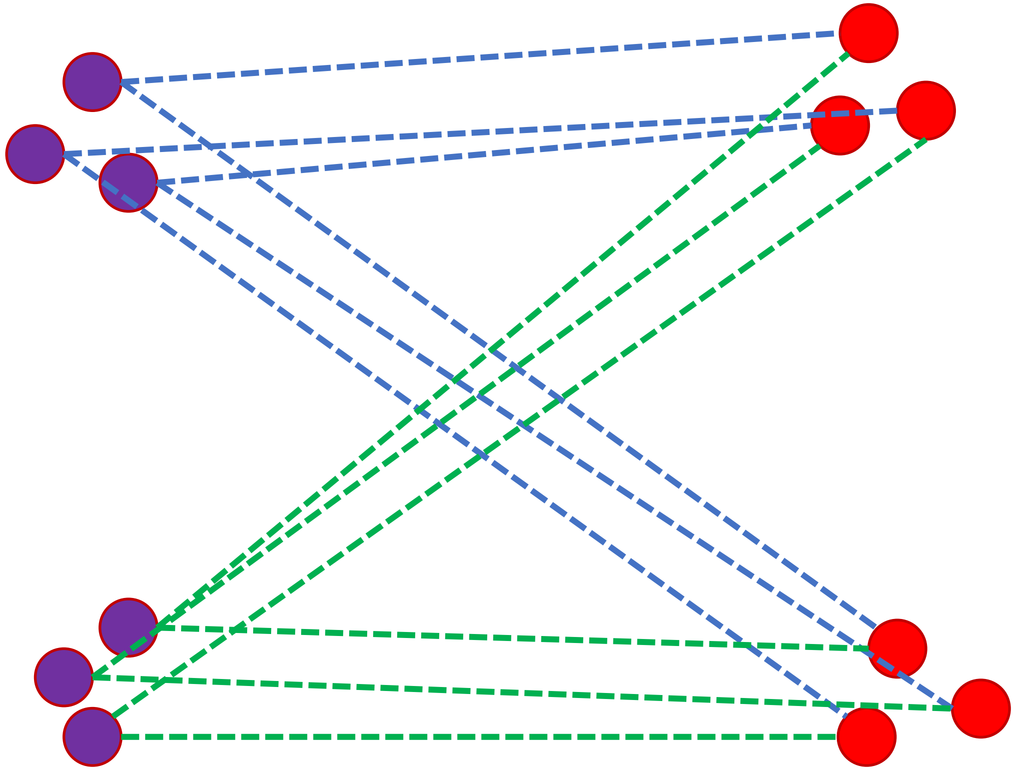
Goal: find a transport map T such that,

$$Z_1 \coloneqq T(Z_0) \sim \pi_1 \ \text{ when } \ Z_0 \sim \pi_0$$



Step 1: Construct Straight-Line Teachers

$\{x_i^0\}_{i=1}^n \sim \pi_0$

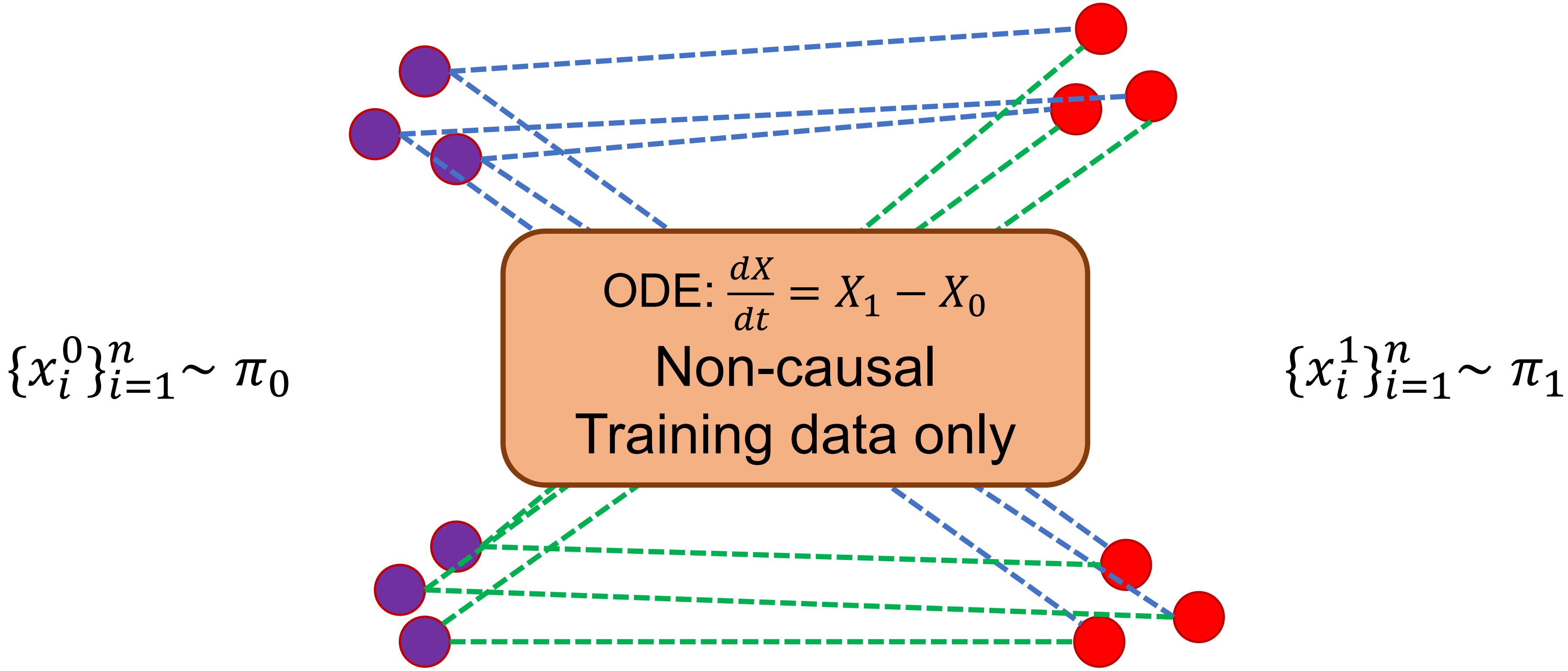


$\{x_i^1\}_{i=1}^n \sim \pi_1$

Linear Interpolation: $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = X_1 - X_0$

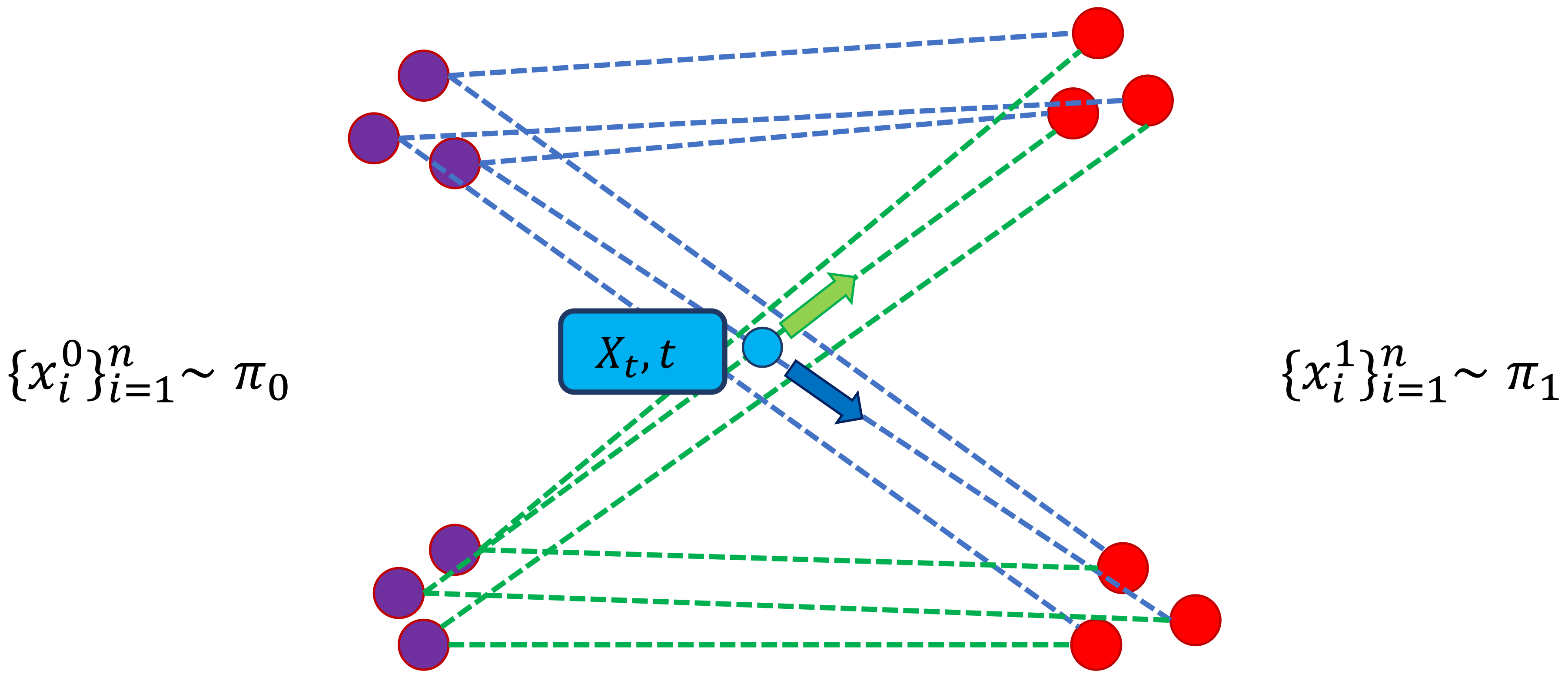
Step 1: Construct Straight-Line Teachers



Linear Interpolation: $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = X_1 - X_0$

Step 1: Construct Straight-Line Teachers



Linear Interpolation: $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = X_1 - X_0$

Step 1: Construct Straight-Line Teachers

$\{x_i^0\}_{i=1}^{n} \sim \pi_0$

2-GMM

$\{x_i^1\}_{i=1}^{n} \sim \pi_1$

2-GMM

Linear Interpolation: $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = X_1 - X_0$

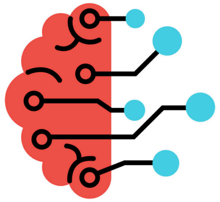
Step 2: Project to Causal Students

Teacher ODE (Non-causal)

$$\frac{dX}{dt} = X_1 - X_0$$

Student ODE (Causal)

$$\frac{dX}{dt} = v_\theta(X, t)$$



NEURAL NETWORK

Projection Loss

$$\min_\theta \int_0^1 \mathrm{E}_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left[\left|\left| \boxed{(X_1 - X_0)} - \boxed{v_\theta(X_t, t)} \right|\right|^2 \right] \mathrm{d}t$$

Teacher velocity

Student velocity

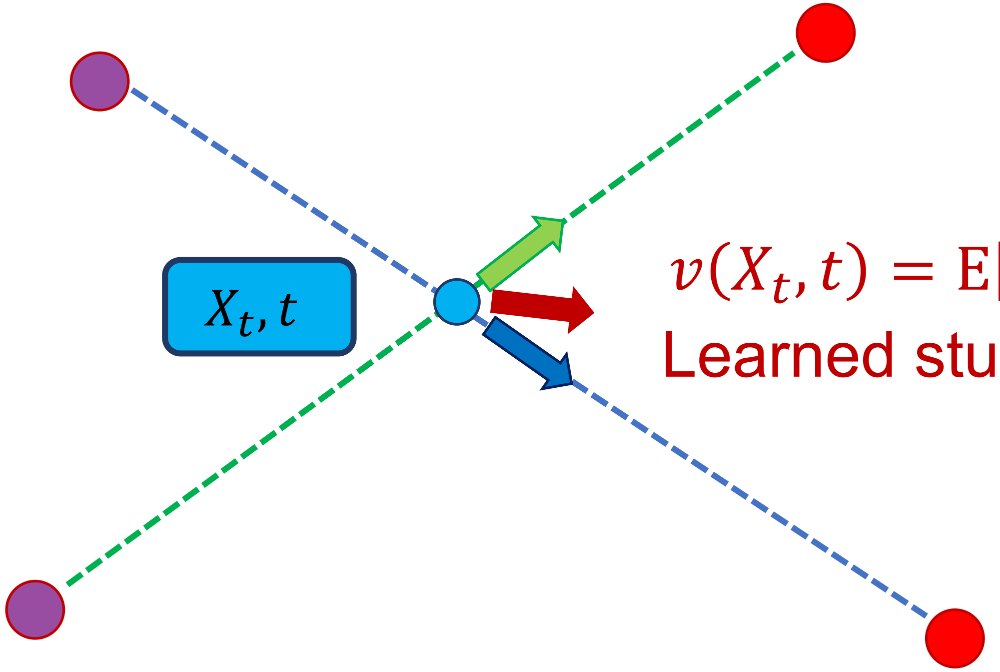
Step 2: Project to Causal Students

Projection Loss

$$\min_{\theta} \int_0^1 \mathrm{E}_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left[\left|\left| \boxed{(X_1 - X_0)} - \boxed{v_\theta(X_t, t)} \right|\right|^2 \right] \mathrm{d}t$$

Teacher velocity

Student velocity

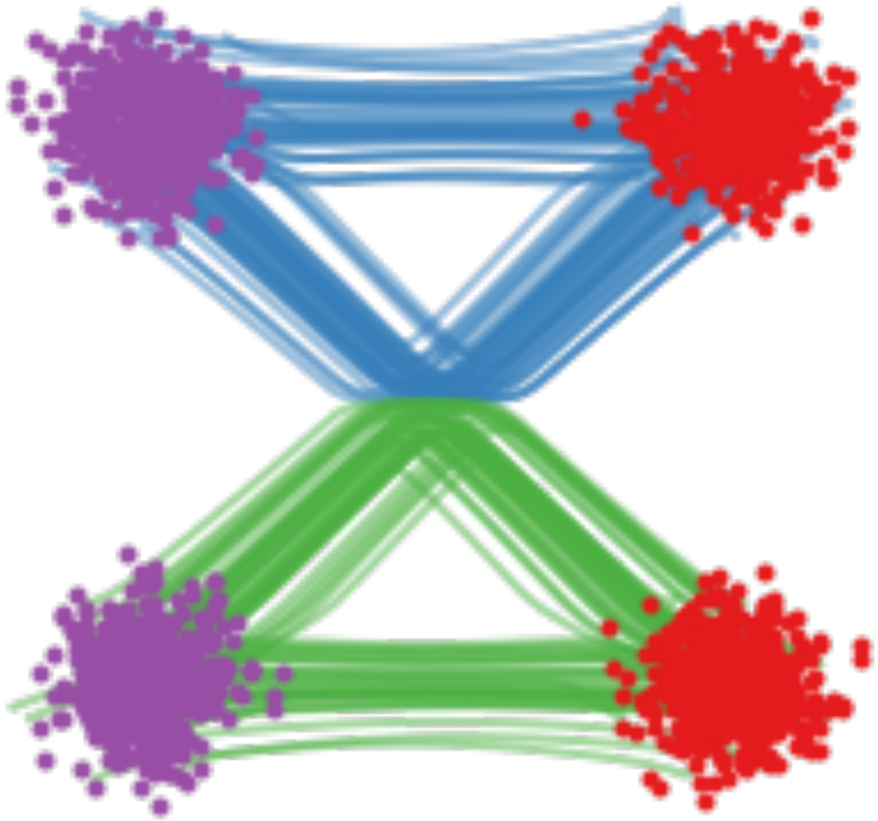


$v(X_t, t) = \mathrm{E}[X_1 - X_0 | X_t]$

Learned student velocity

Step 3: Generation with ODE solver

Randomly sample $X_0 \sim \pi_0$



Generated distribution $X_1 \sim \pi_1$

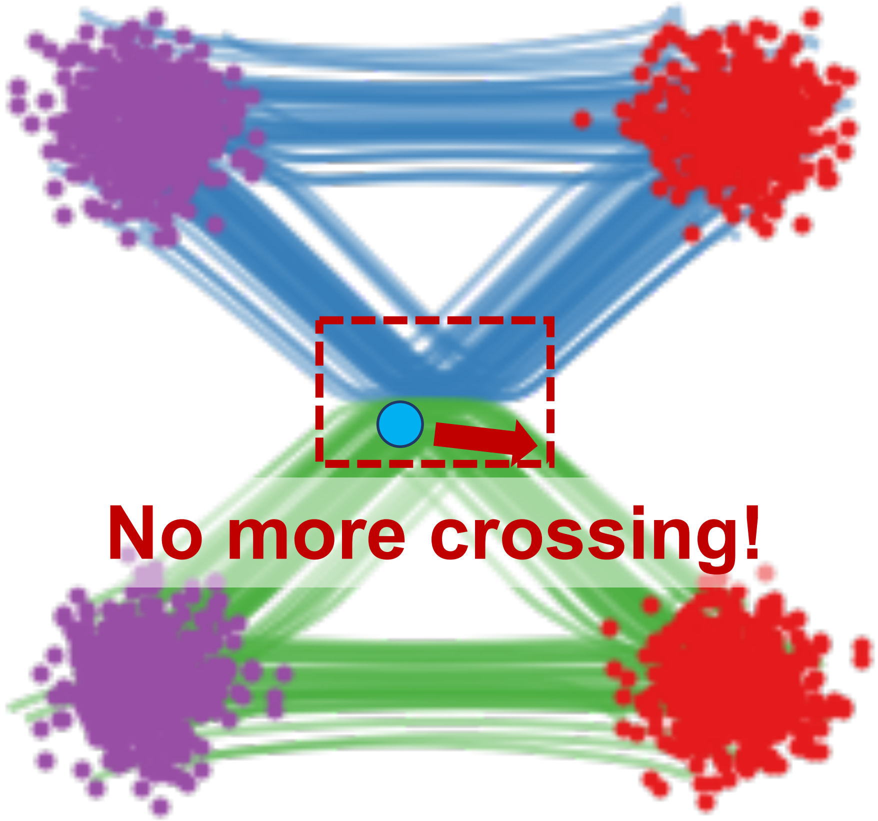
Guaranteed by math

Simulate with ODE solver, e.g., Euler

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Step 3: Generation with ODE solver

Randomly sample $X_0 \sim \pi_0$



Generated distribution $X_1 \sim \pi_1$

Guaranteed by math

Simulate with ODE solver, e.g., Euler

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Algorithm: Rectified Flow

- **Given:** $\{x_i^0\}_{i=1}^n \sim \pi_0$, $\{x_i^1\}_{i=1}^n \sim \pi_1$
- **Training Iteration (Batch size = 1)**:
  - Step 1: Randomly sample $X_0 \in \{x_i^0\}_{i=1}^n$ and $X_1 \in \{x_i^1\}_{i=1}^n$
  - Step 2: Randomly sample $t \in [0,1]$
  - Step 3: Compute gradient with loss

$$L(\theta) := \left|\left|X_1 - X_0 - v_\theta(X_t, t)\right|\right|^2,$$

where $\quad X_t = tX_1 + (1-t)X_0$

Empirical Results

CIFAR10

Method	NFE (↓)	IS (↑)	FID (↓)
VP SDE	2000	9.58	2.55
subVP SDE	2000	9.56	2.61
VP ODE	140	9.37	3.93
subVP ODE	146	9.46	3.16
Rectified Flow	**127**	**9.60**	**2.58**

Fast sampling + high-quality

(A) LSUN Church (B) CelebA HQ (C) LSUN Bedroom (D) AFHQ Cat

256 Resolution

Not There Yet

Randomly sample $X_0 \sim \pi_0$

ODE is still curved!

Generated distribution $X_1 \sim \pi_1$

Guaranteed by theory

Simulate with ODE solver, e.g., Euler

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Prior Attempts

Learning straight probability flow ODEs is investigated in the Neural ODE works

When continuous normalizing flows were hot

1. Jacobian and Kinetic Regularization [Finlay et al. 2020]

$$\sum_{i=1}^{N} \log p_\theta(x_i)$$

Likelihood of the training data

$$\int_0^1 \left|\left|v_\theta(X_t, t)\right|\right|^2 \mathrm{d}t$$

Kinetic energy

$$\int_0^1 \left|\left|\nabla_{X_t} v_\theta(X_t, t)\right|\right|_F^2 \mathrm{d}t$$

Integral of Frobenius norm of Jacobian

$$-\int_0^1 \mathrm{div}(v_\theta)(X_t, t)\,\mathrm{d}t$$

Log-determinant of Jacobian

2. Optimal Transport-Flow [Onken et al. 2021]

$$\sum_{i=1}^{N} \log p_\theta(x_i)$$

Likelihood of the training data

$$\int_0^1 \left|\left|v_\theta(X_t, t)\right|\right|^2 \mathrm{d}t$$

Transport Cost

$$\int_0^1 \left\|\partial_t \Phi(X_t, t) - \frac{1}{2}\left\|\nabla_{X_t}\Phi(X_t, t)\right\|\right\|^2 \mathrm{d}t$$
$$s.t. \quad v(X_t, t) = -\nabla_{X_t}\Phi(X_t, t)$$

Hamilton–Jacobi–Bellman Regularization

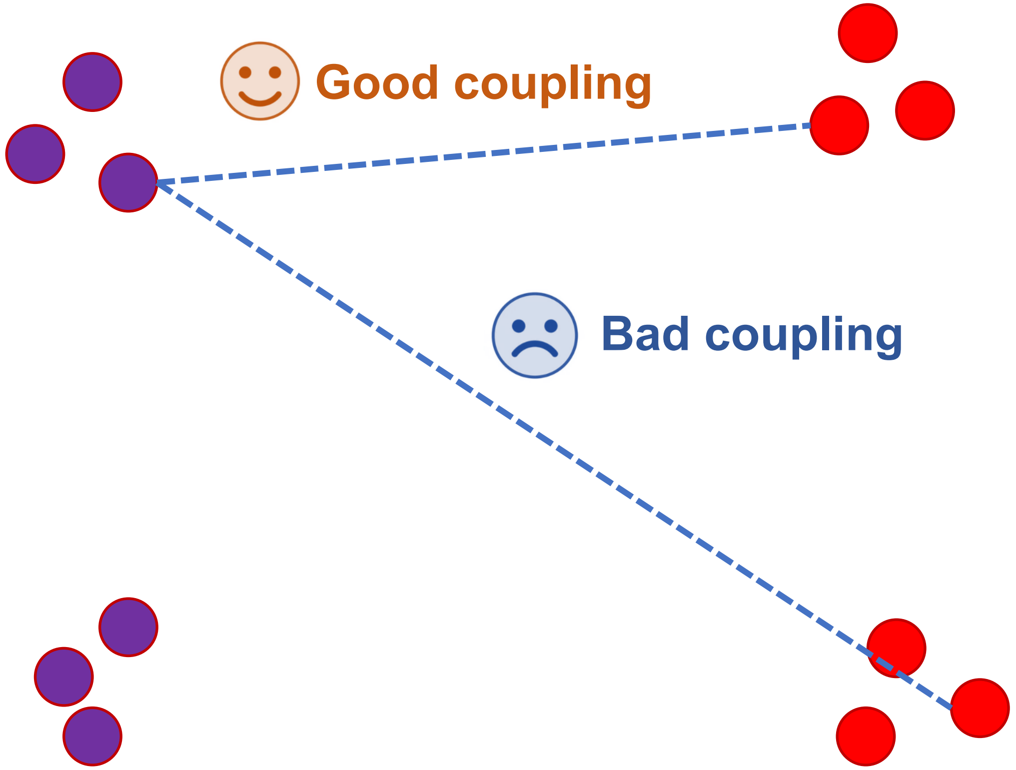
☹ Hard to Optimize | Limited Capacity | Fail to Scale up

Our Solution: **Reflow!**

Idea: Re-connect with straight lines!

Our Solution: **Reflow!**

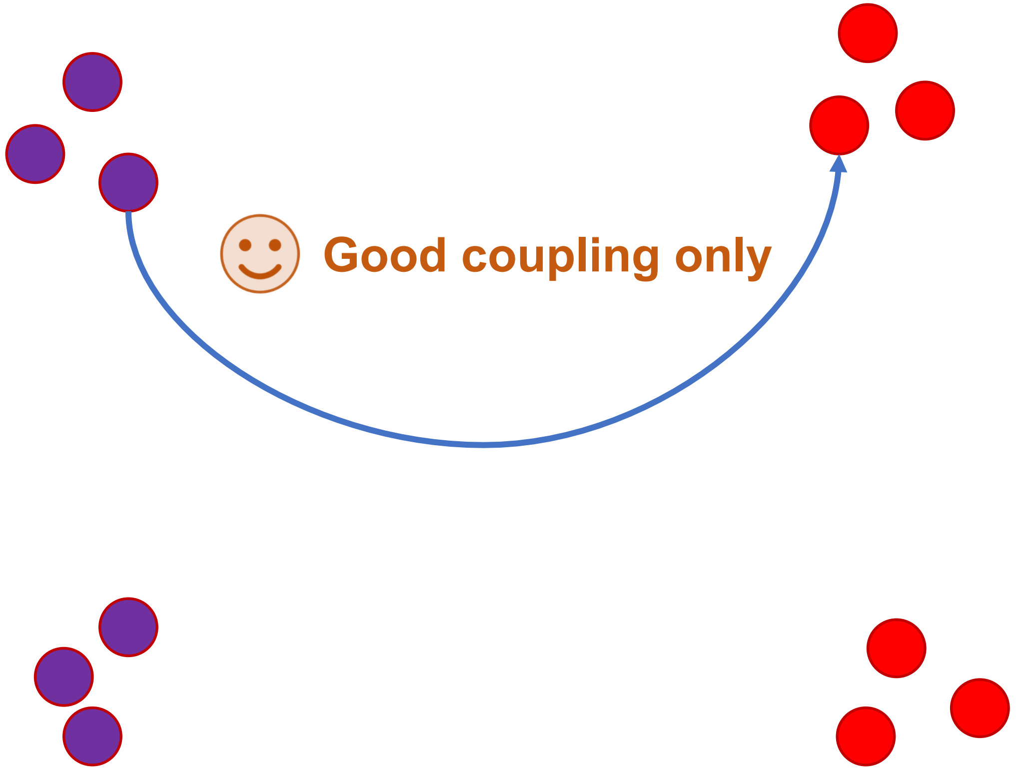
Curved student comes from crossing in training



We have no better coupling than random

Our Solution: **Reflow!**

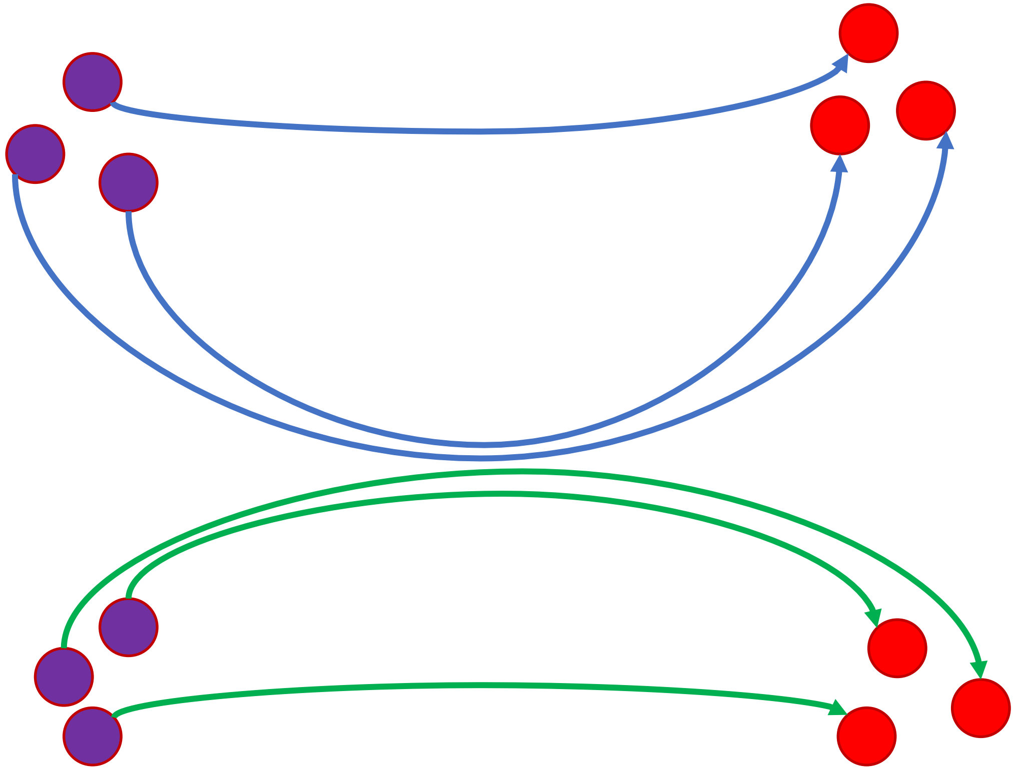
But the new student eliminates crossing!



It is a better teacher than random

Moreover, it keeps the target distribution π_1

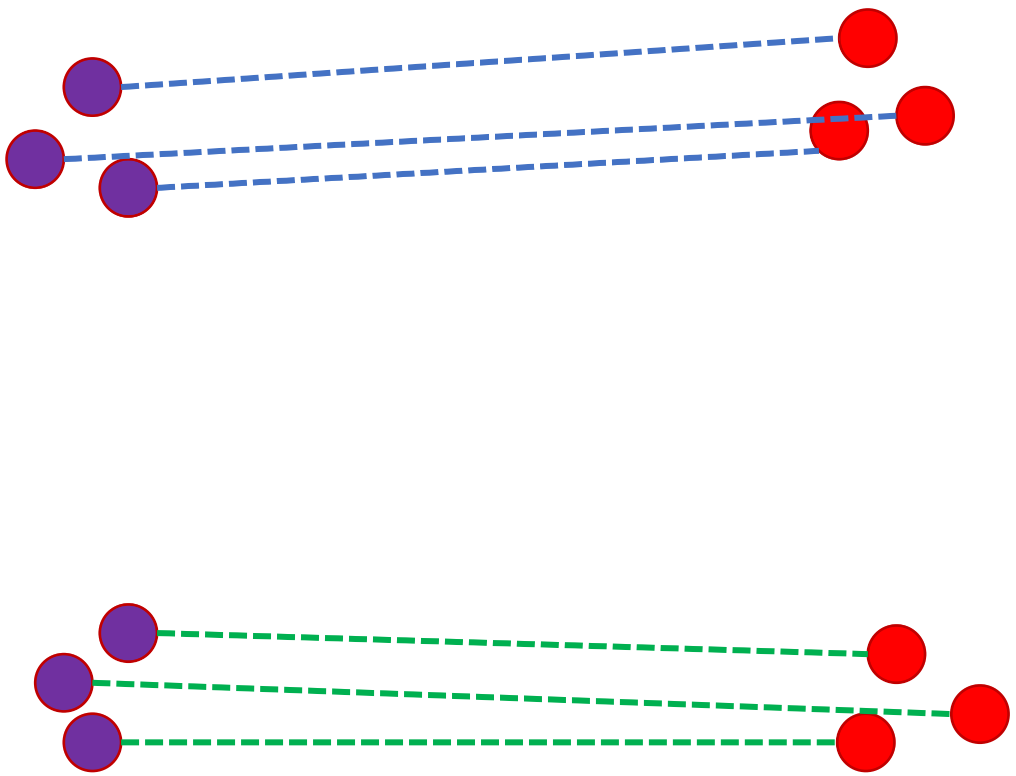
Reflow Step-1: Construct Straight-Line Teachers



Get the coupling by simulating with ODE solver, e.g., Euler

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

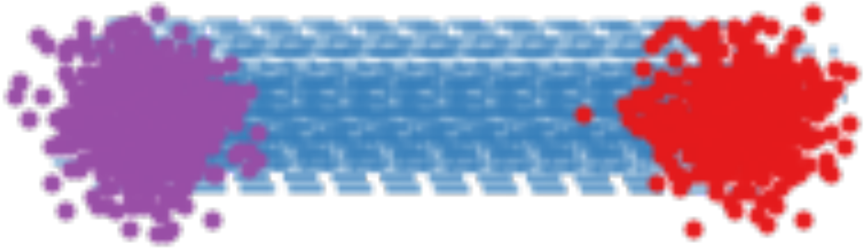
Reflow Step-1: Construct Straight-Line Teachers



Linear Interpolation (again): $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Reflow Step-1: Construct Straight-Line Teachers

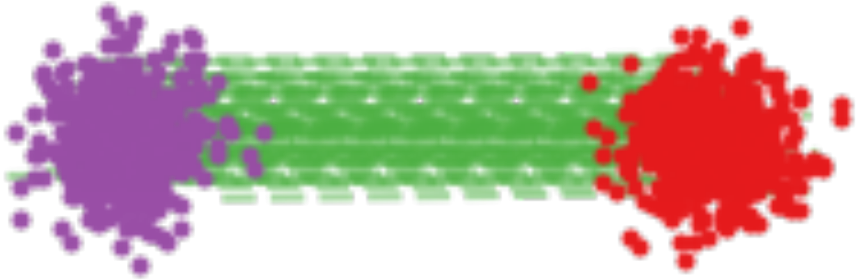


$\{x_i^0\}_{i=1}^n \sim \pi_0$

2-GMM

$\{x_i^1\}_{i=1}^n \sim \pi_1$

2-GMM



Linear Interpolation (again): $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Reflow Step-2: Project to Causal Students

Projection Loss (previous)

$$\min_{\theta} \int_0^1 \mathrm{E}_{\boxed{X_0 \sim \pi_0, X_1 \sim \pi_1}} \left[\left|\left| (X_1 - X_0) - v_\theta(X_t, t) \right|\right|^2 \right] \mathrm{d}t$$

Independent

Projection Loss (now)

$$\min_{\theta} \int_0^1 \mathrm{E}_{\boxed{X_0 \sim \pi_0, X_1 = ODE_{v_{old}}(X_0)}} \left[\left|\left| (X_1 - X_0) - v_\theta(X_t, t) \right|\right|^2 \right] \mathrm{d}t$$

Generated by ODE

Reflow Step-3: Generation with ODE solver

Randomly sample $X_0 \sim \pi_0$

ODE is straightened!

Generated distribution $X_1 \sim \pi_1$

Guaranteed by math

Simulate with ODE solver, e.g., Euler

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Algorithm: Reflow

- **Given:** $\{x_i^0\}_{i=1}^n \sim \pi_0$, $\{x_i^1\}_{i=1}^n \sim \pi_1$, old flow v_{old}

- **Training Iteration (Batch size = 1)**:

    - Step 1: Randomly sample $X_0 \in \{x_i^0\}_{i=1}^n$
    - $\boxed{\text{Step 2: Generate } X_1 = ODE_{v_{old}}(X_0)}$
    - Step 3: Randomly sample $t \in [0,1]$
    - Step 4: Compute gradient with loss

$$L(\theta) := \left|\left|X_1 - X_0 - v_\theta(X_t, t)\right|\right|^2,$$

where $\quad X_t = tX_1 + (1-t)X_0$

Reflow: Theoretical Properties

Guarantee straight ODE trajectories after infinite reflow

In practice, one reflow already has magic

k-Rectified Flow (v_k)

Reflow: Theoretical Properties

Reflow is a multi-objective OT solver

Every reflow monotonically decrease the transport cost for all convex cost functions c:

$$\mathrm{E}_{(X_0,X_1)\sim p_{v_k}(X_0,X_1)}[c(X_1 - X_0)] \leq \mathrm{E}_{(X_0,X_1)\sim p_{v_{k+1}}(X_0,X_1)}[c(X_1 - X_0)]$$

Distillation

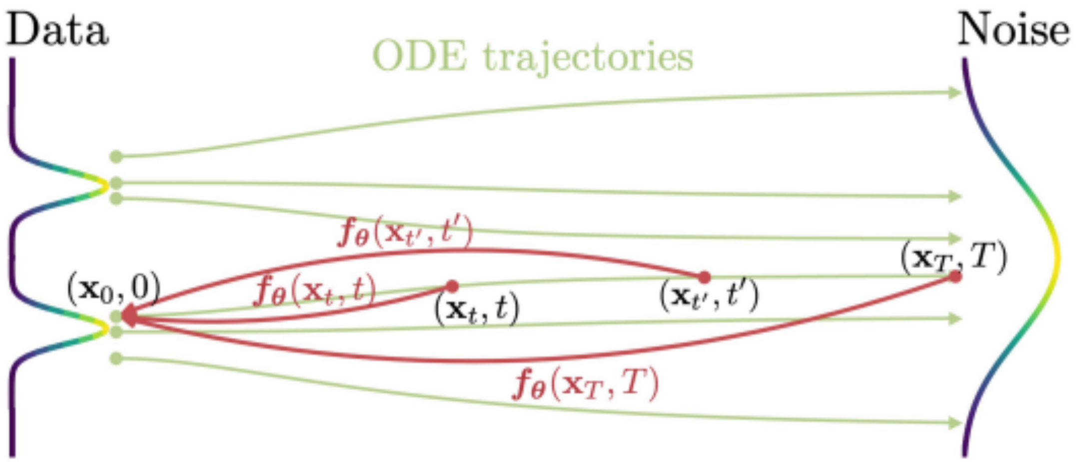
- **Distillation**

$$\min_{\phi} \mathrm{E}_{X_0\sim\pi_0, X_1=ODE_v(X_0)} \left\| f_\phi(X_0) - X_1 \right\|^2$$

- **Data-free Distillation**

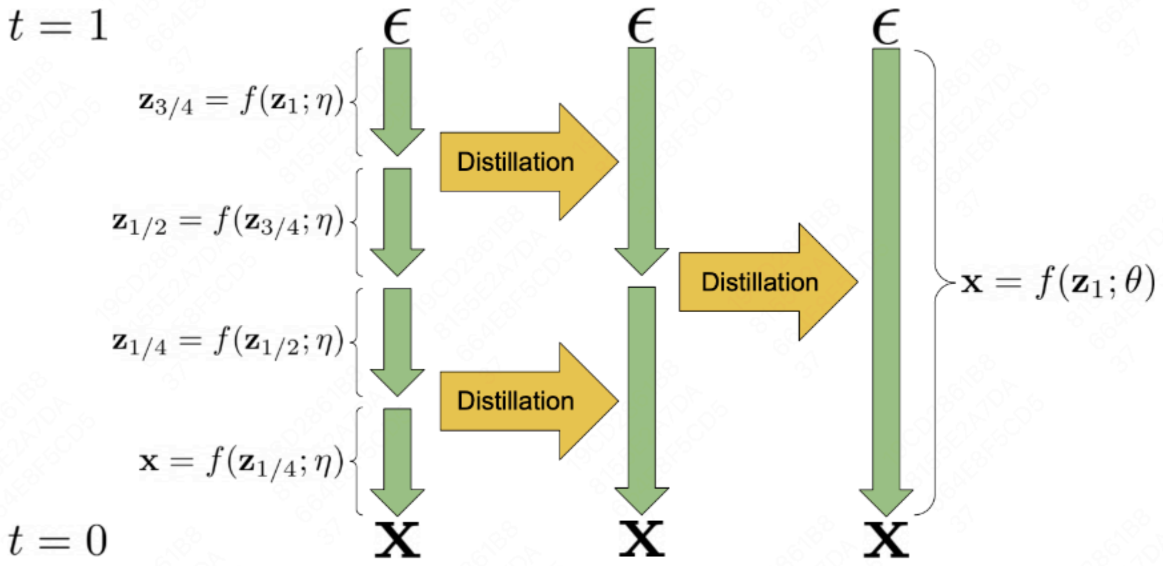
Consistency Distillation

[Song et al. 2023]



Progressive Distillation

[Salimans et al. 2022]



Reflow is Orthogonal to Distillation

Reflow is a multi-objective OT solver

It changes coupling, while distillation imitates

Reflow: Create better probability flow teacher

Distillation: Train one-step student from teacher

Rectified Flow

L2 loss
+
Supervised Learning

↓

Efficient Training

Reflow straightens the flow iteratively

↓

Efficient Sampling

	Efficient Training	Efficient Sampling
Rectified Flow	√	√

Reflow: Empirical Results

CIFAR10

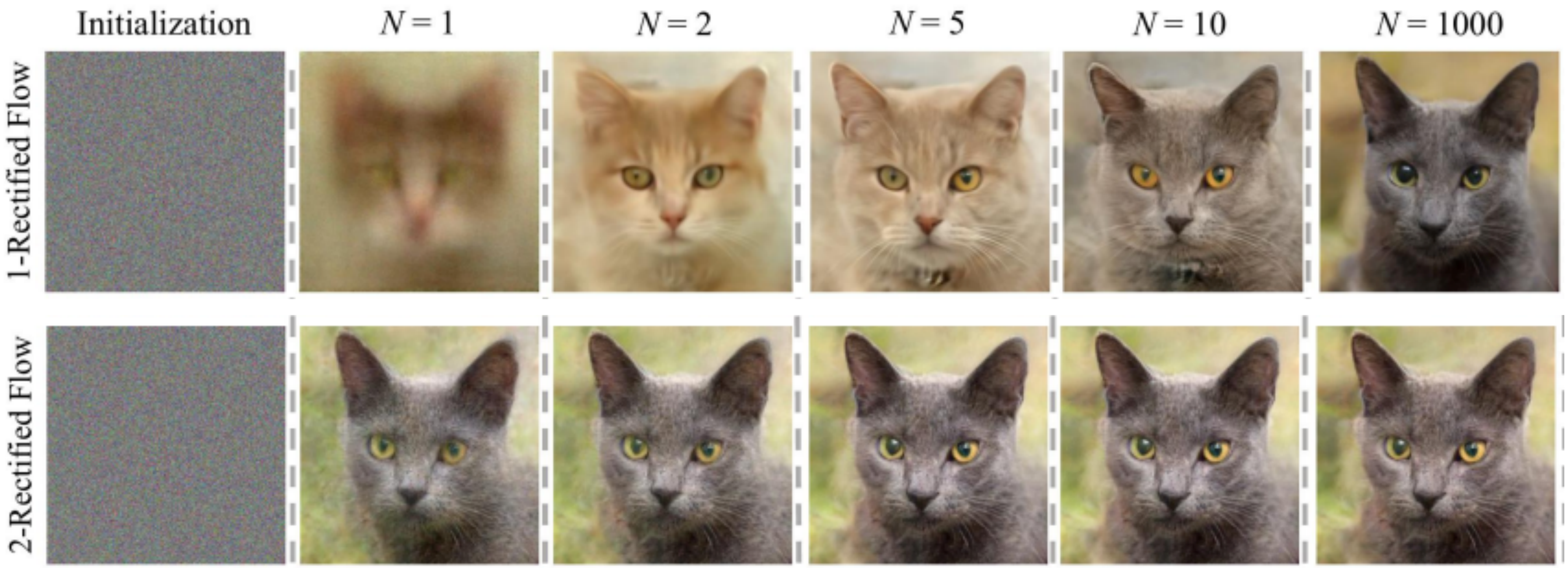
Method	**NFE (↓)**	**IS (↑)**	**FID (↓)**
1-Rectified Flow	127	9.60	2.58
2-Rectified Flow	110	9.24	3.36
3-Rectified Flow	104	9.01	3.96

Method	**NFE (↓)**	**IS (↑)**	**FID (↓)**
1-Rectified Flow	1	1.13	378
2-Rectified Flow	1	8.08	12.21
3-Rectified Flow	1	8.47	8.15

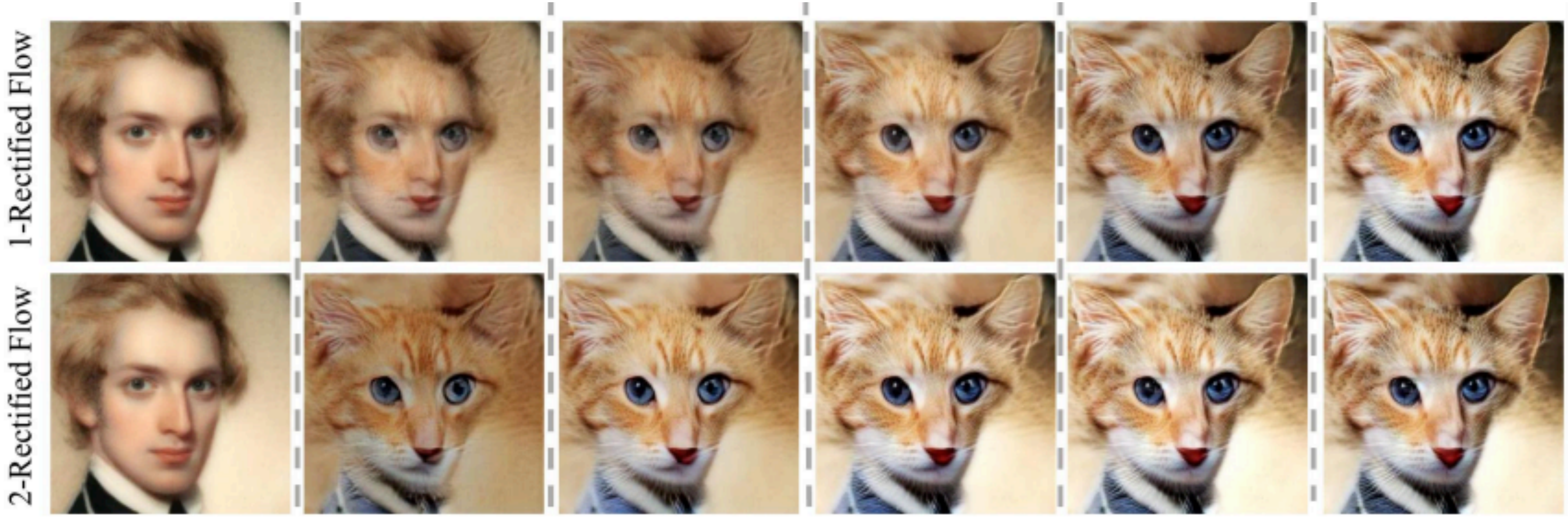
Method	**NFE (↓)**	**IS (↑)**	**FID (↓)**
1-Rectified Flow+Distill	1	9.08	6.18
2-Rectified Flow+Distill	1	9.01	$\boxed{4.85}$
3-Rectified Flow+Distill	1	8.79	5.21

SOTA
(when arXiv)

Reflow: Generative Modeling



Reflow: Domain Transfer



InstaFlow: Scale Up Rectified Flow

- Today's common sense: scaling-up makes things different!
- Will the rectified flow pipeline (reflow+distill) still work in Stable Diffusion level?

[**Liu** et al., in submission]

InstaFlow: Scale Up Rectified Flow

One-step InstaFlow-0.9B (0.09s per image, 512 × 512)

One-step InstaFlow-1.7B
(0.12s per image, 512 × 512)

InstaFlow: Scale Up Rectified Flow

- **Text-Conditioned Reflow**:

Random text from text dataset

Text-conditioned model

$$v_{k+1} = \arg\min_{v} \mathbb{E}_{X_0 \sim \pi_0, \boxed{\mathcal{T} \sim D_{\mathcal{T}}}} \left[\int_0^1 || (X_1 - X_0) - \boxed{v(X_t, t \mid \mathcal{T})} ||^2 \, \mathrm{d}t \right],$$
$$\text{with } \boxed{X_1 = \mathtt{ODE}[v_k](X_0 \mid \mathcal{T})} \quad \text{and} \quad X_t = tX_1 + (1-t)X_0,$$

Text-conditioned generation

- **Text Dataset**: 1.6M data points from LAION-2B (aesthetics score 6.0+)
- **Model**: Stable Diffusion (as 1-Rectified Flow)
- **Training cost**: 199 A100 GPU days (InstaFlow 0.9B)

Reflow Makes a Difference

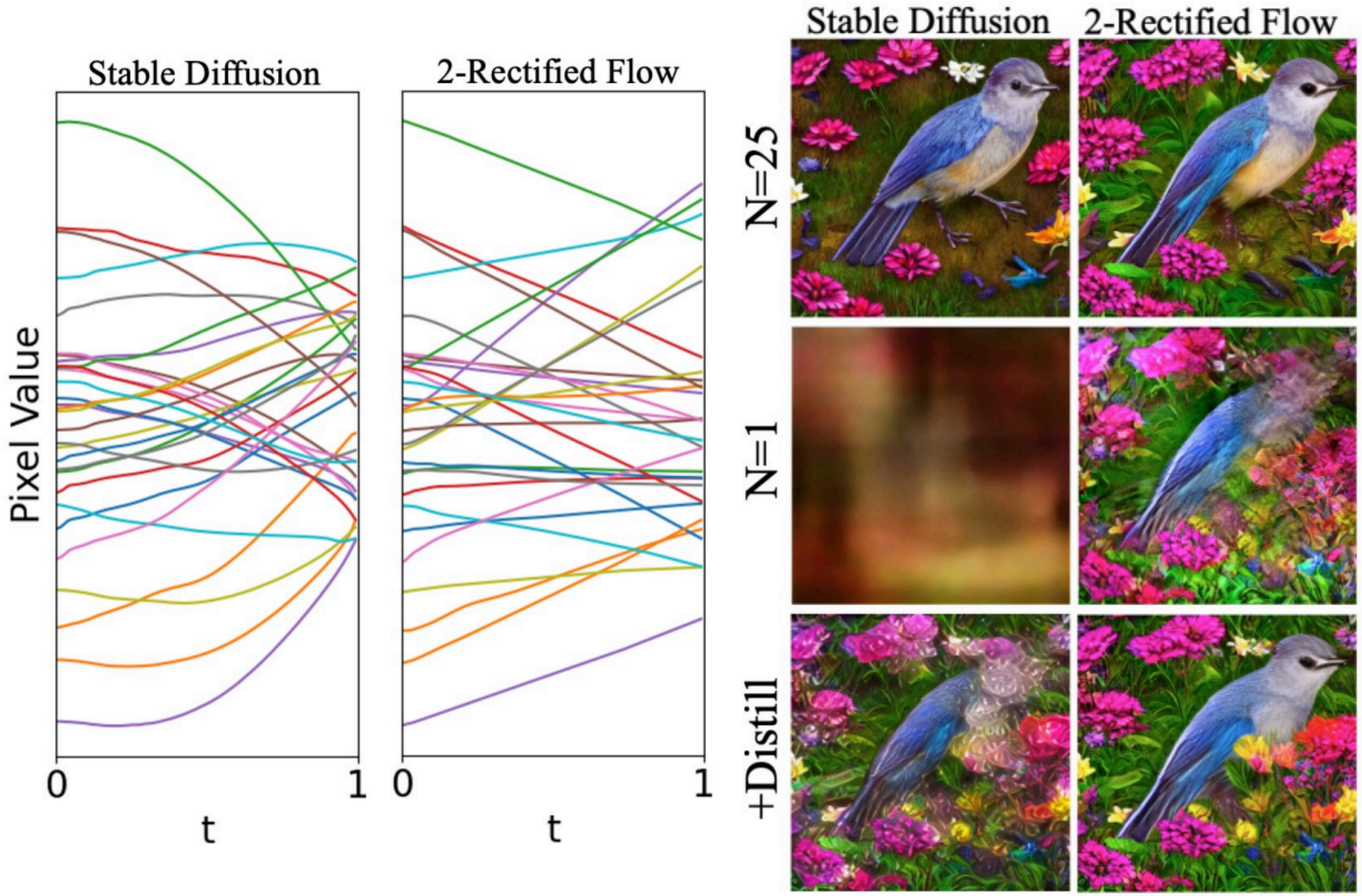
- **Direct Distillation**: 100k training steps
- **Reflow + Distillation**: 50k training steps + 50k training steps

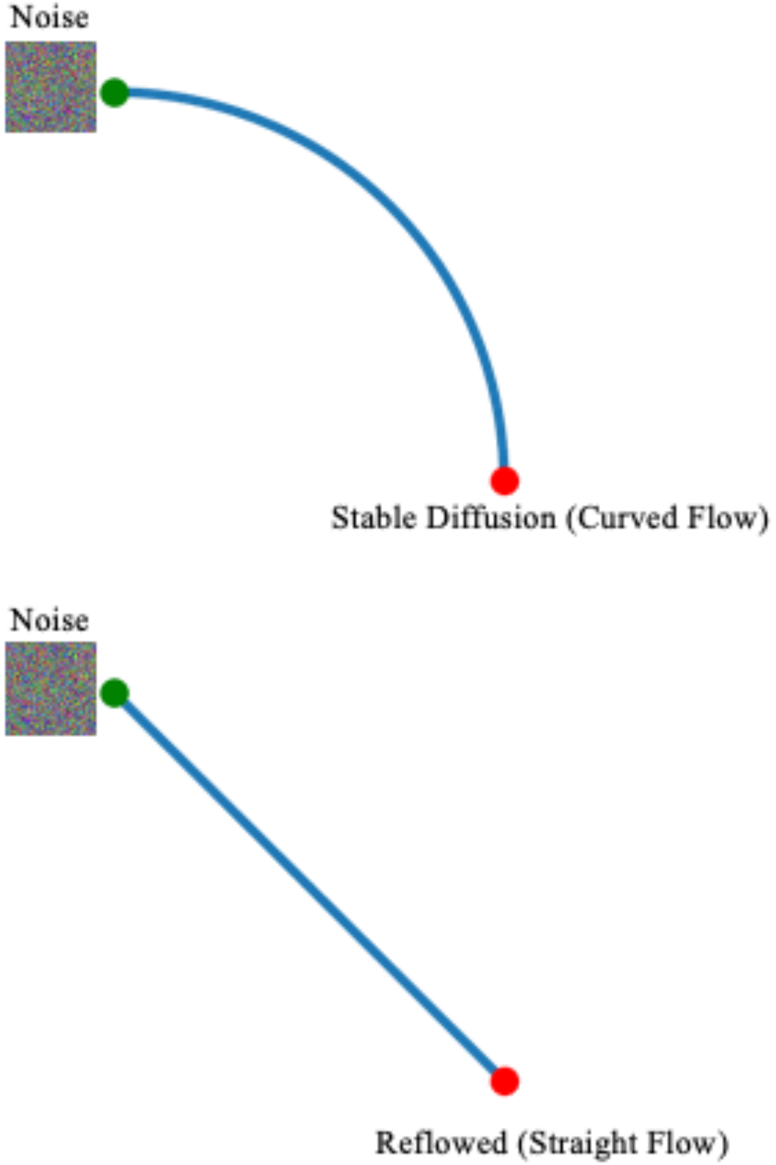
MS COCO 2017 – 5k images

Method	**Inf t (↓)**	**FID (↑)**	**CLIP (↑)**
SD 1.4	0.88s	22.8	0.315
2-Rectified Flow	0.88s	22.1	0.313

Method	**Inf t (↓)**	**FID (↑)**	**CLIP (↑)**
SD 1.4+Distill	0.09s	40.9	0.255
Progressive Distill	0.09s	37.2	0.275
2-Rectified Flow +Distill	0.09s	**31.0**	**0.285**

Reflow Makes a Difference





InstaFlow: Further Scaling Up

- The preliminary experiments only spends 24.65 A100 GPU days in training
- **Reflow + Distillation**: 24.65 A100 GPU days → 199 A100 GPU days

InstaFlow-0.9B

- **Expand Network**: 0.9B → 1.7B

InstaFlow-1.7B

InstaFlow: Empirical Results

MS COCO 2017 – 5k images

Method	Inf t (↓)	FID (↑)	CLIP (↑)
SD 1.4+Distill	0.09s	40.9	0.255
Progressive Distill (1-step)	0.09s	37.2	0.275
2-Rectified Flow+Distill (24.65 A100 GPU days)	0.09s	31.0	0.285
InstaFlow-0.9B (199 A100 GPU days)	**0.09s**	**23.4**	**0.304**
InstaFlow-1.7B	**0.12s**	**22.4**	**0.309**

MS COCO 2014 – 30k images

Method	Inf t (↓)	FID (↑)
Stable Diffusion	2.9s	9.62
StyleGAN-T	0.1s	13.90
GigaGAN	0.13s	9.09
InstaFlow-0.9B	**0.09s**	**13.10**
InstaFlow-1.7B	**0.12s**	**11.83**

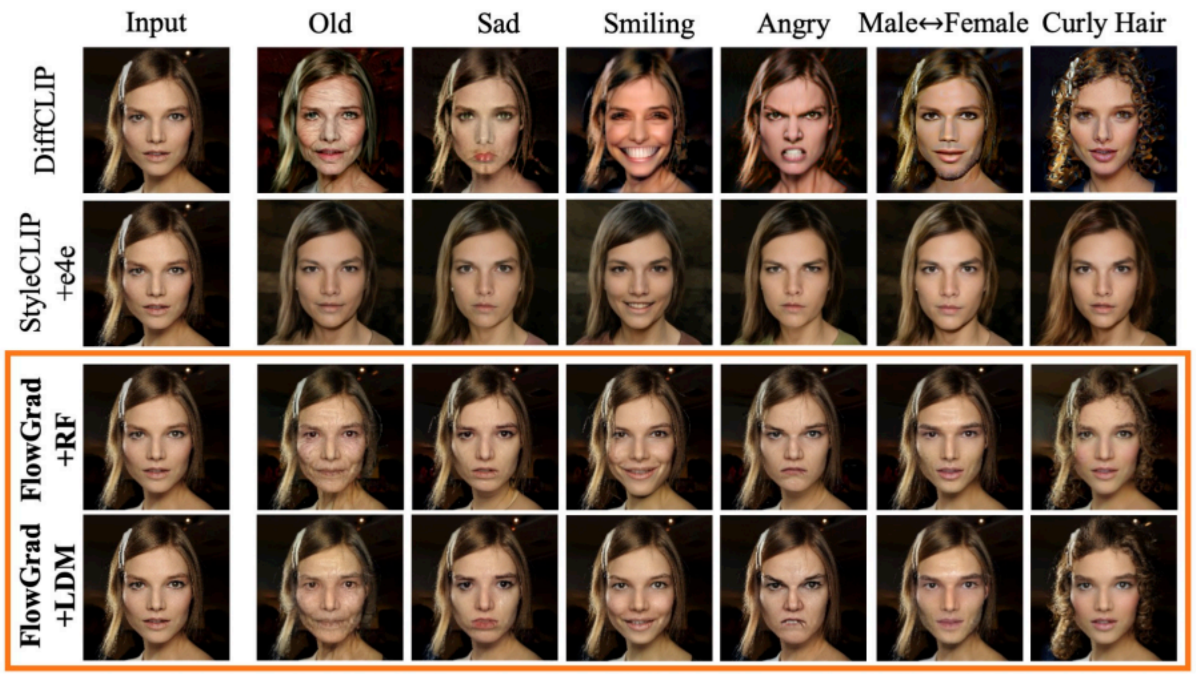
InstaFlow as Fast Previewer

One-Step

+SDXL Refiner

Fast preview + Slow Refiner

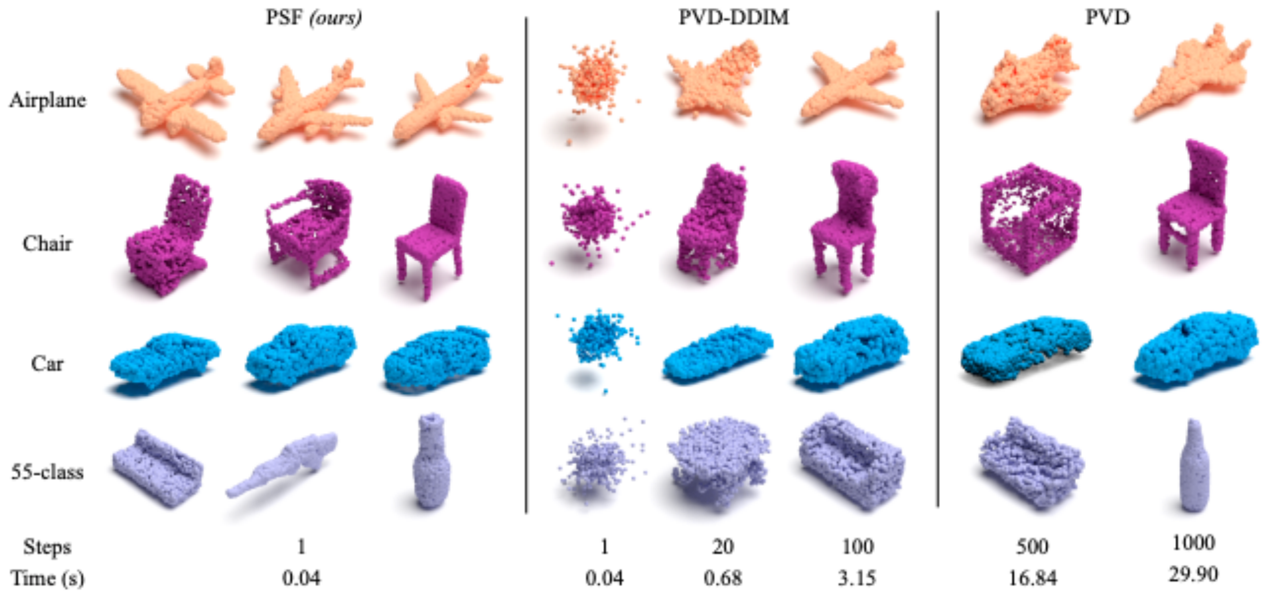
Other Works from Our Group



FlowGrad

Fast gradient-based editing with probability flows

[**Liu** et al., CVPR 2023]

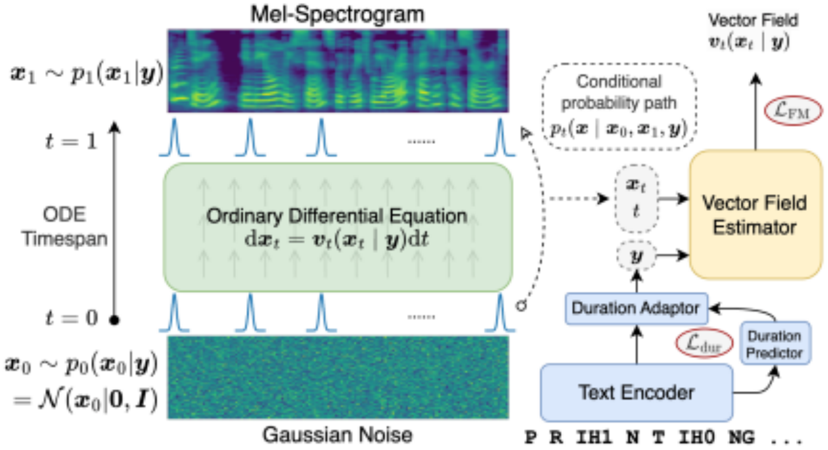


Point Straight Flow

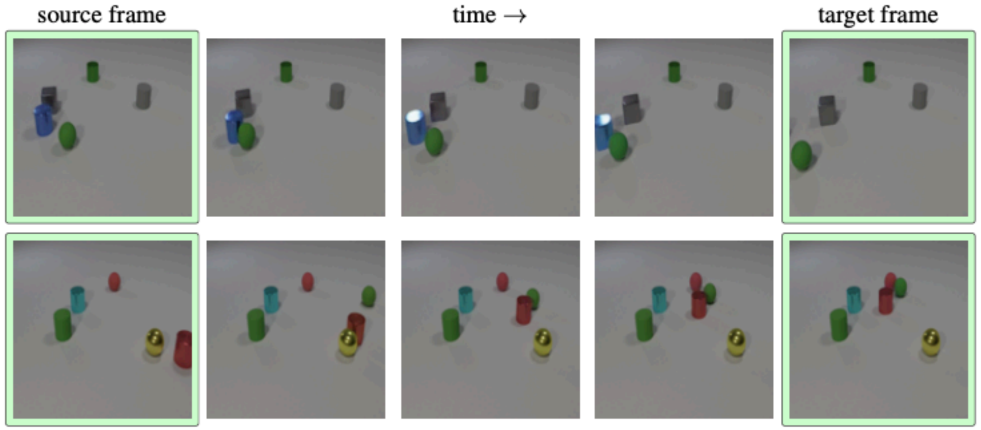
One-step point cloud generation (100× faster)

[Wu et al., CVPR 2023]

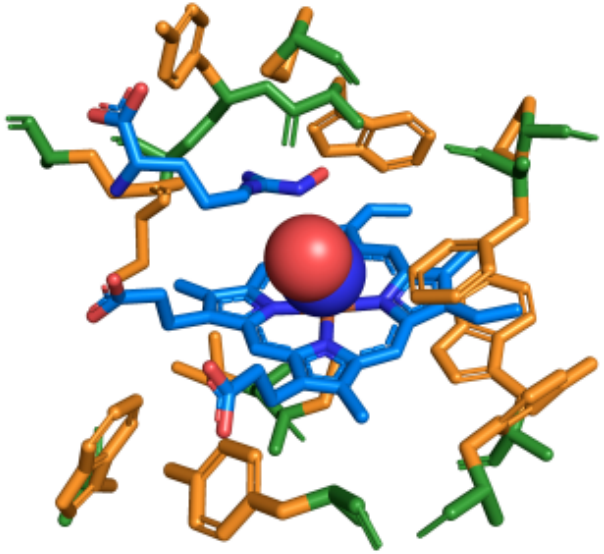
Applications From Other Labs



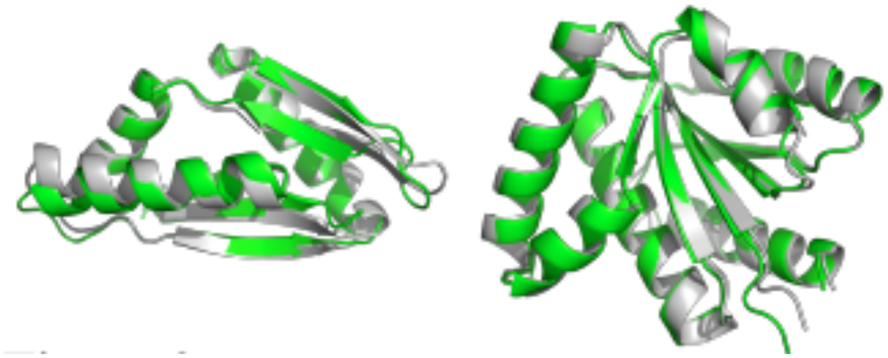
VoiceFlow (text-to-speech) [Guo et al. 2023]



RIVER (video prediction) [Davtyan et al. 2023]



FlowSite (binding site design) [Stark et al. 2023]



FoldFlow (protein structure design) [Yim et al. 2023]

Take-Aways

- Straight = Fast !

- Made possible by Rectified Flow !

- Scale up perfectly in large models !

TEXAS
The University of Texas at Austin

Thank you!

Questions?

🤗 Demo: https://huggingface.co/spaces/XCLiu/InstaFlow

Many thanks to my collaborators: Chengyue Gong, Qiang Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng

Concurrent works

There were concurrent works with the same idea, different names:

- Flow matching [Lipman et al. 2023]
- Stochastic Interpolants [Albergo et al. 2023]
- α-(de)blending [Heitz et al. 2023]
- Action matching [Neklyudov et al. 2023]