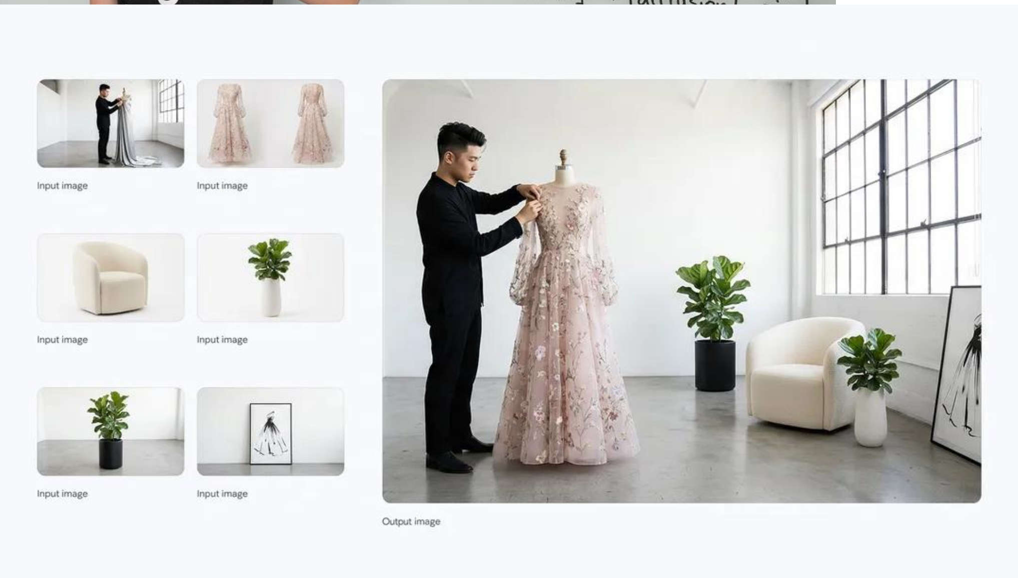
Towards Efficient Inference-Time Scaling without Distillation

Linqi (Alex) Zhou

Luma

Rise of Diffusion Models

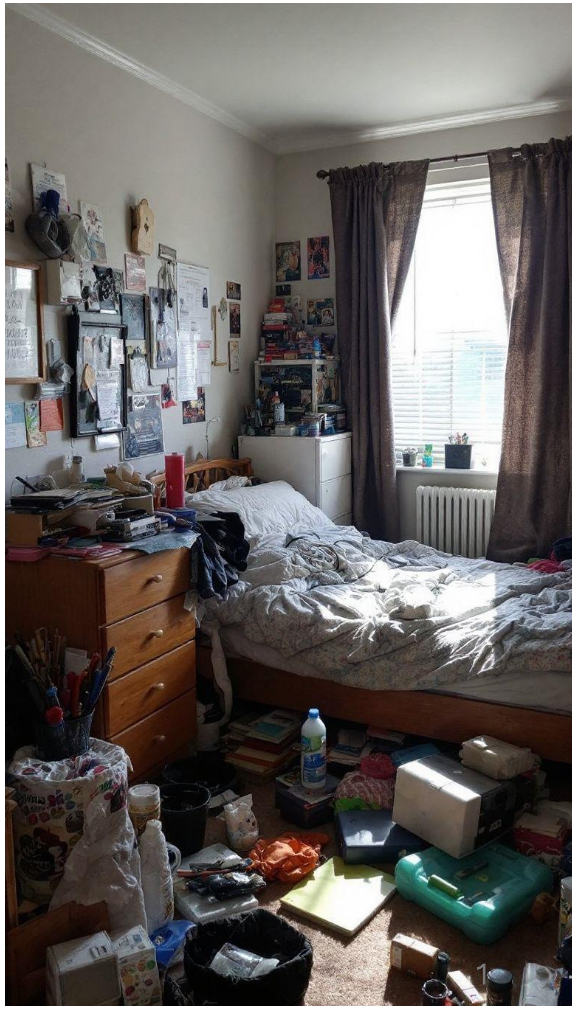
GPT-4o Image



Nano Banana

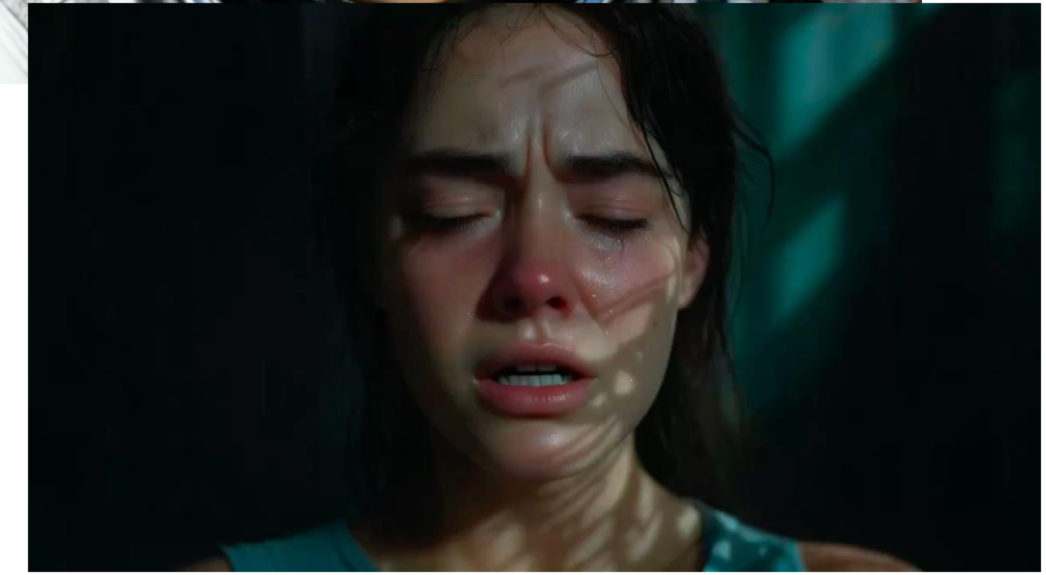
Midjourney v7

FLUX



Text-to-Video Models at Luma

Luma
Ray 3



Why are Diffusion and Flow Matching so Good?

- Simple L2 loss → stable and scalable!

Problems of Diffusion and Flow Matching

$$\mathrm{d}\mathbf{x}_t = \mathbf{u}_t \mathrm{d}t$$ (probability-flow ODE)

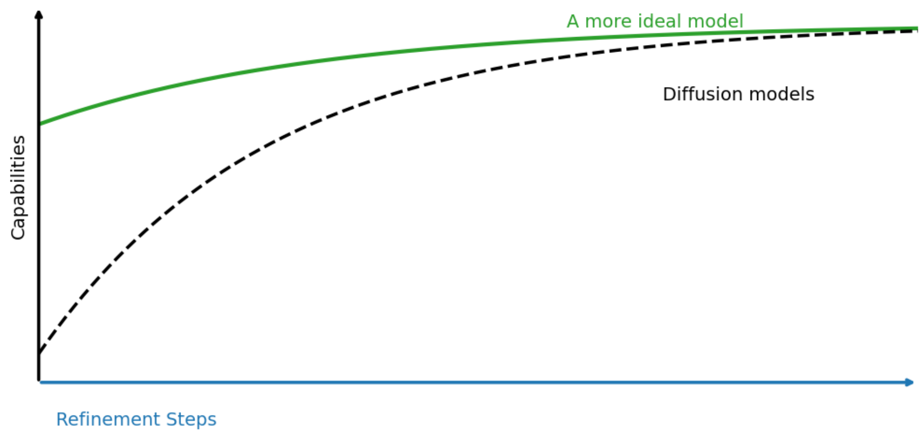
NOT optimal in utilizing network capacity.

ODE simulation error

Slow inference

Ideal case: one- or few-step mapping from prior to data.

(**efficient** inference-time scaling)



Towards Efficient Inference-Time Scaling

Diffusion Pretrain
+ Step Distillation

Problems:

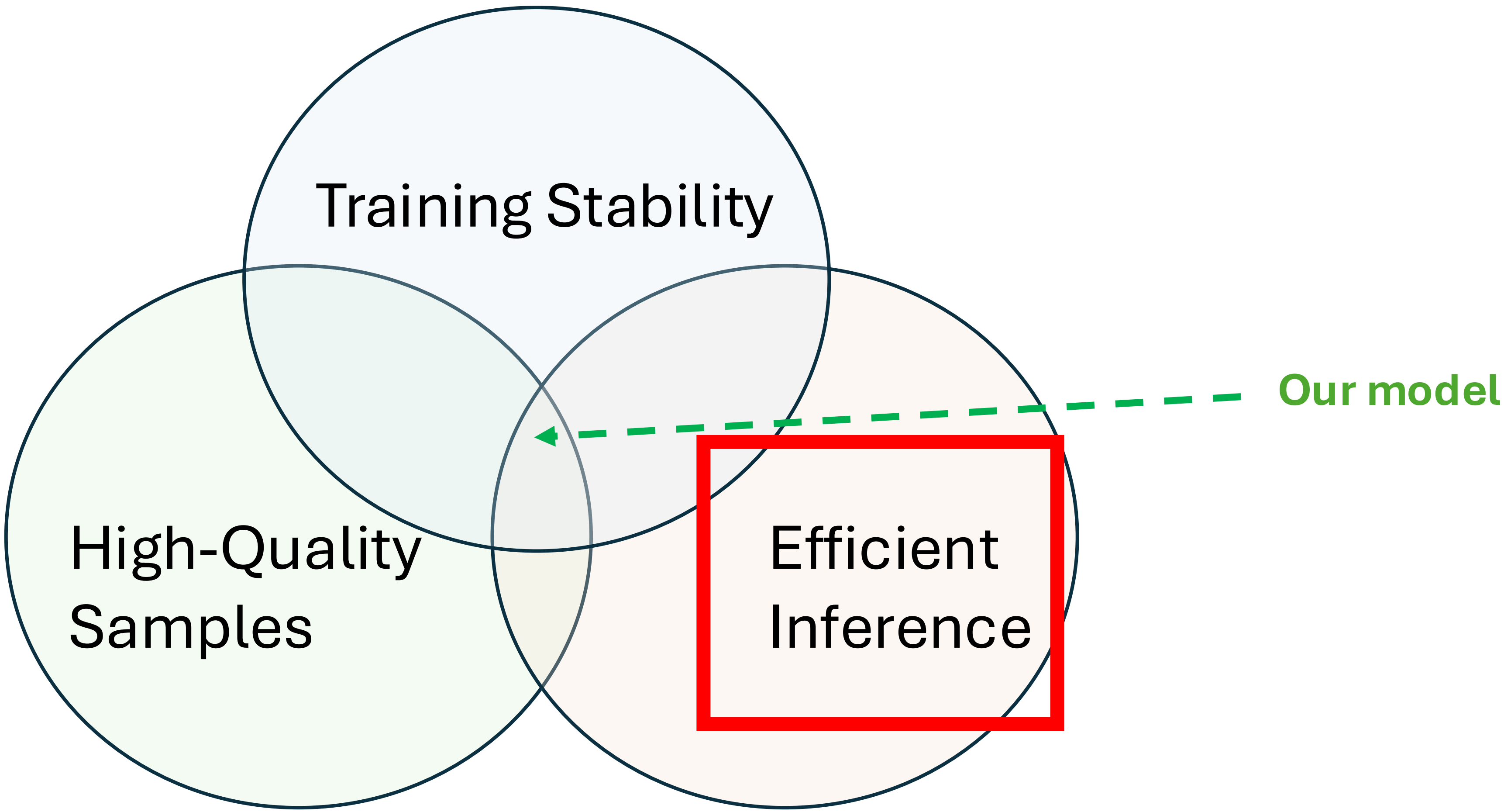
- Can be unstable
- Too many models and additional engineering complexities

Single-Stage Pretraining

- Consistency Training
- MeanFlow
- Inductive Moment Matching
- Terminal Velocity Matching

This talk

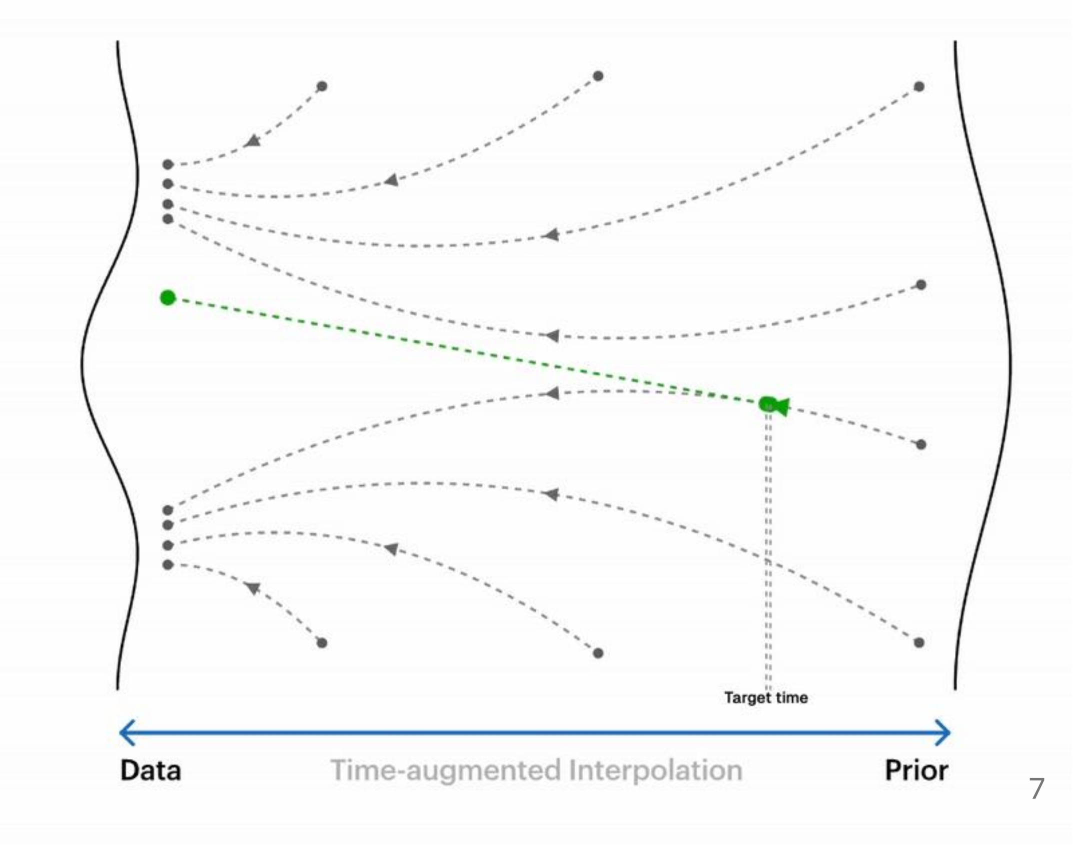
Desiderata of Efficient Inference-Time Scaling



Problems with Diffusion Inference

- Want: **Large** jump in timesteps (**NOT** infinitesimal jump ODE)
  $d\mathbf{x}_t = \mathbf{u}_t dt$ (probability flow)
- Denoising Diffusion Implicit Models (DDIM)
  - Euler under FM schedule

$$\mathbf{x}_s = \mathbf{x}_t + \hat{\mathbf{u}} \cdot (\boxed{s} - t)$$
$$\hat{\mathbf{u}} = \hat{\mathbf{u}}(\mathbf{x}_t; t)$$



- Linear w.r.t. *s*

Fixing the Capacity Issue

- Inject s into the network

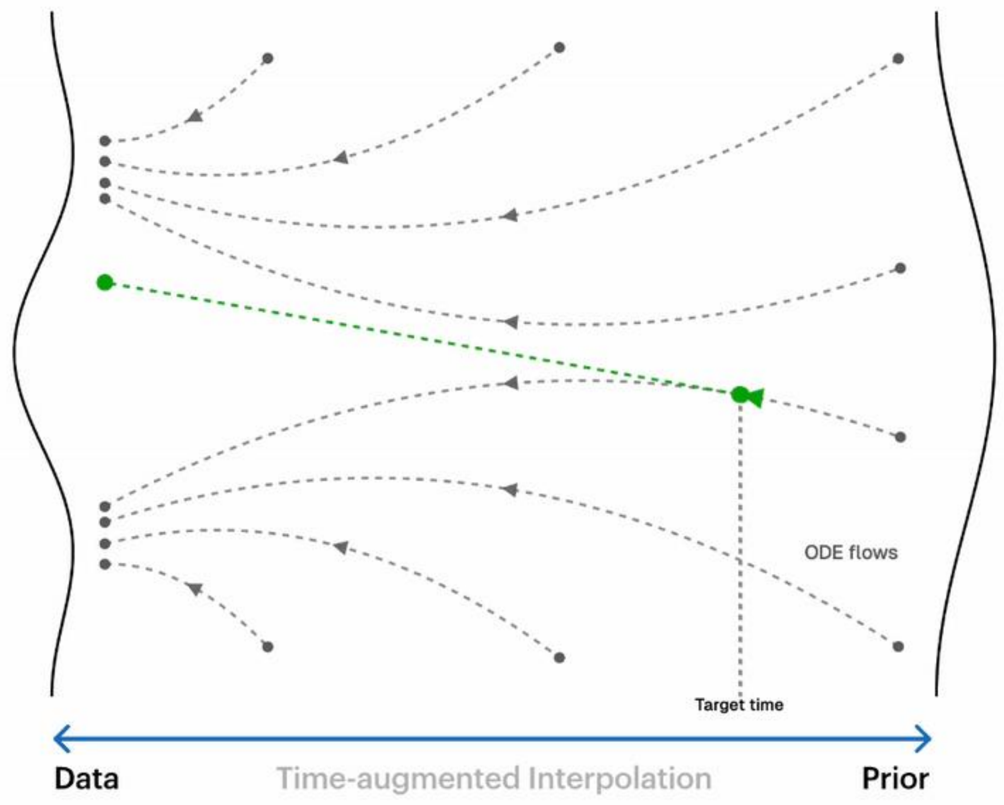
$$\mathbf{x}_s = \mathbf{x}_t + \hat{\mathbf{u}} \cdot (s - t)$$

Before: $\hat{\mathbf{u}} = \hat{\mathbf{u}}(\hat{\mathbf{x}}_t; t)$

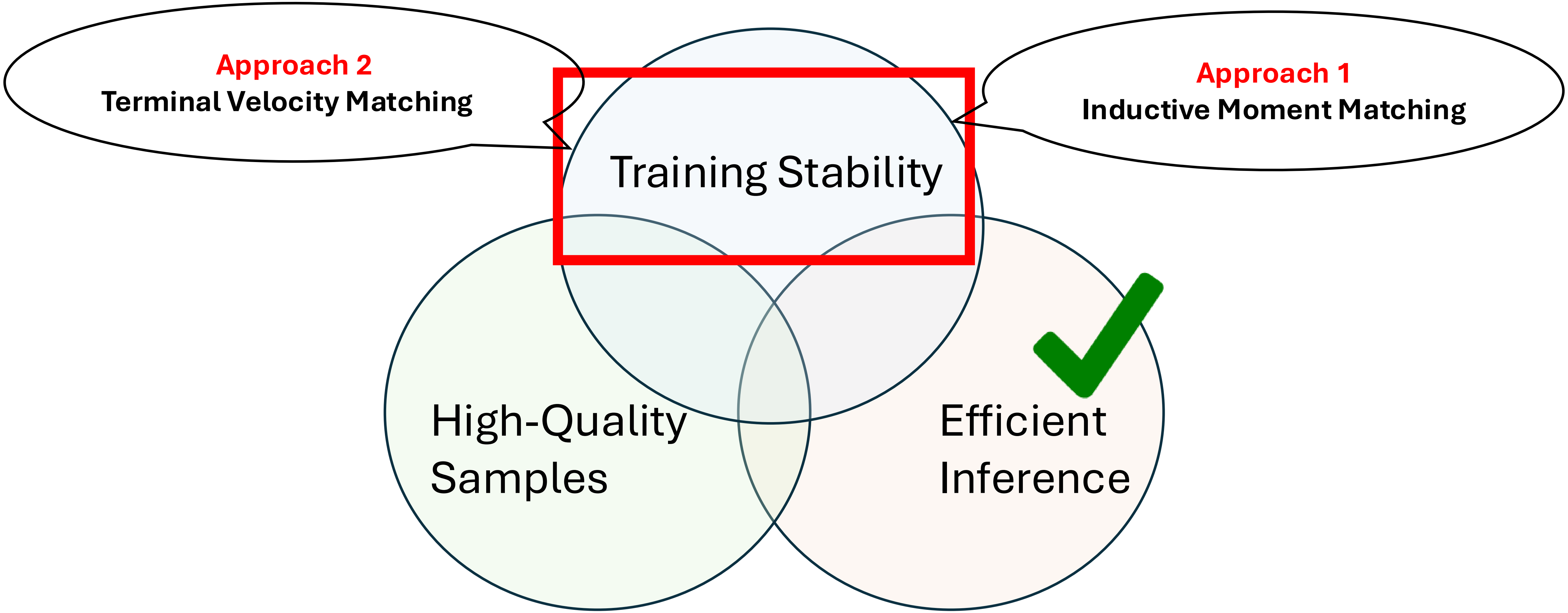
After: $\hat{\mathbf{u}} = \hat{\mathbf{u}}(\hat{\mathbf{x}}_t; t, s)$

- Covers complex solutions
  - ODE integration

- Can perform large time jump (speeds up sampling)



Desiderata of Efficient Inference-Time Scaling



Inductive Moment Matching

A Distribution-Matching Few-Step Method

Inductive Moment Matching

- Key components

Training Objective

Sample-based Distribution Matching.

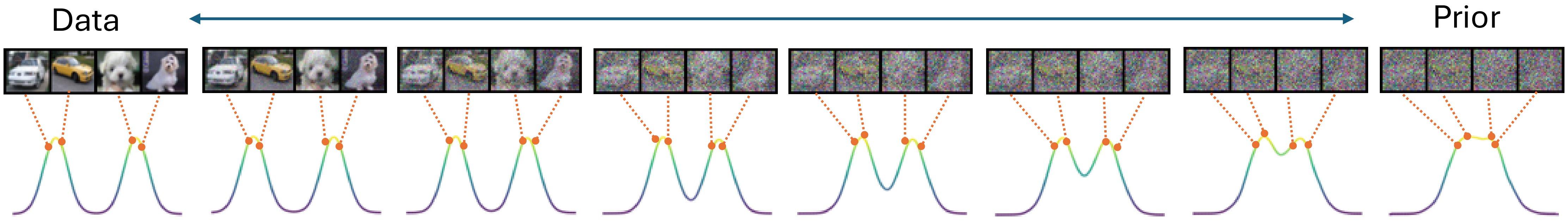
- Maximum Mean Discrepancy (MMD)

Training Target

Inductive Learning.

- Mathematical induction to match the model's own distribution

1. Sample-based Distribution Matching

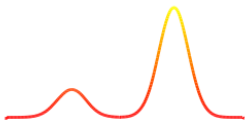


$q_s(\mathbf{x}_s)$

$q_t(\mathbf{x}_t)$

A naïve approach:

Match in distribution



$p^{\theta}_{s,t}(\mathbf{x}_s)$

DDIM (w/ 2 timesteps)

$$\mathbf{x}_s = \mathbf{x}_t + \hat{\mathbf{u}}_\theta(s - t)$$

We learn this DDIM mapping

- Use MMD due to its stability

Maximum Mean Discrepancy

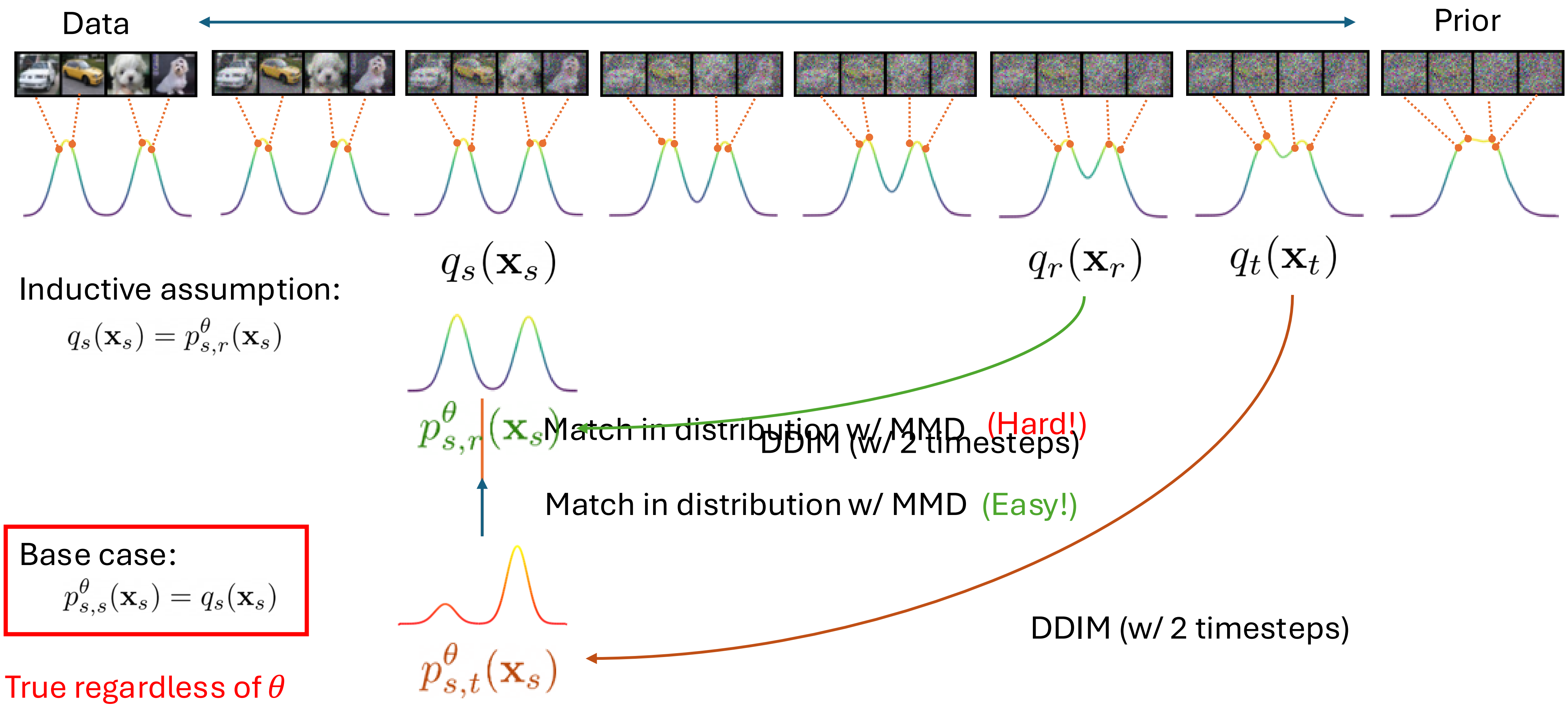
Advantages:

- No adversarial training:
  - GAN-like, **optimal discriminator** chosen in RKHS
- Standard kernel functions:
  - RBF, Laplace, etc. match **all moments**

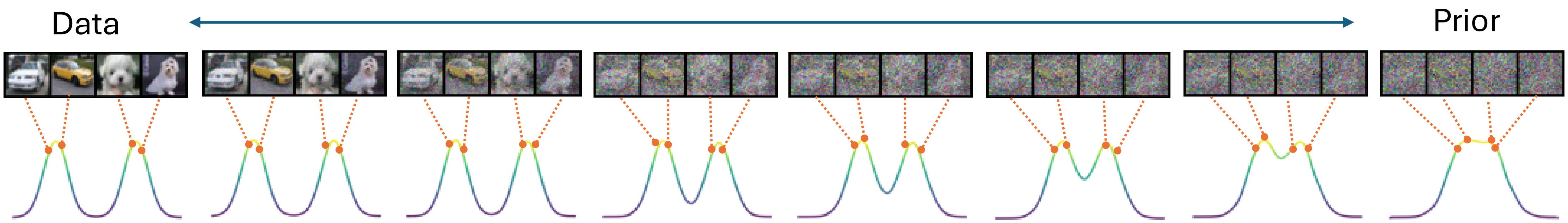
Empirical implementation:

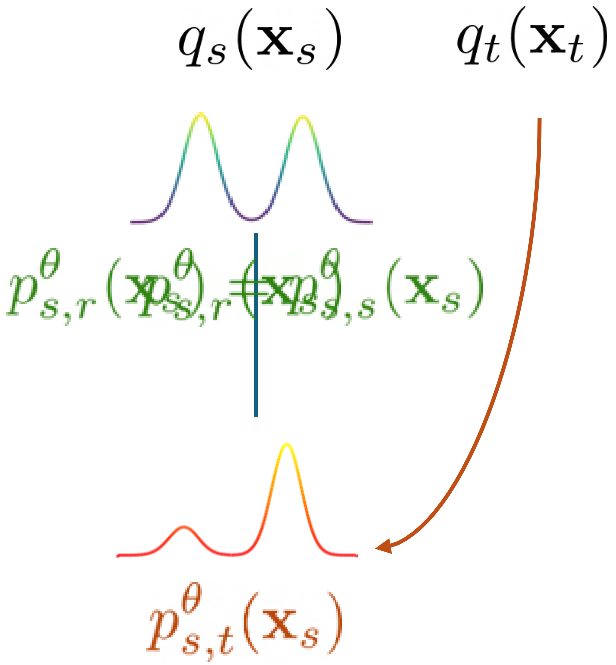
- Multiple particles to estimate expectation

2. Inductive Learning



An Intuition on Inductive Learning



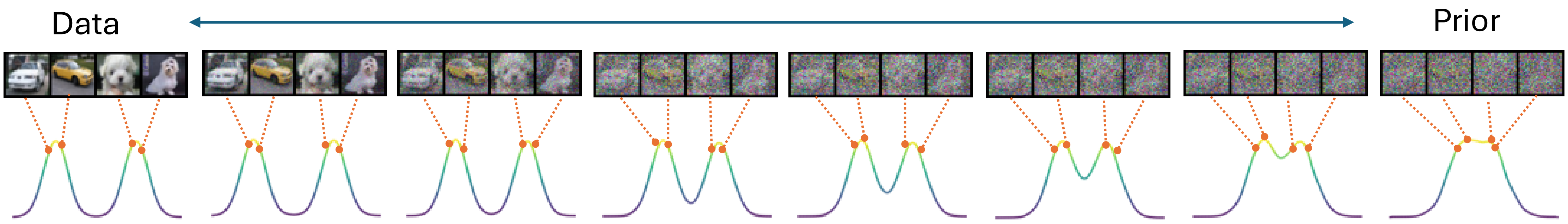


$s \leq r < t$,
$s = r$ when t is close to s

Base case:
$p^\theta_{s,s}(\mathbf{x}_s) = q_s(\mathbf{x}_s)$

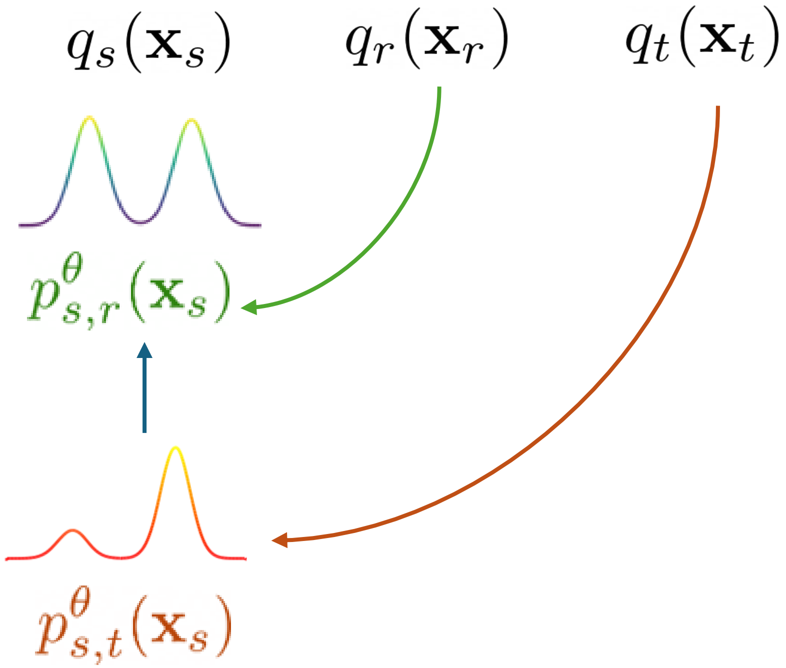
True regardless of θ

An Intuition on Inductive Learning



Inductive assumption:

$q_s(\mathbf{x}_s) = p^{\theta}_{s,r}(\mathbf{x}_s)$

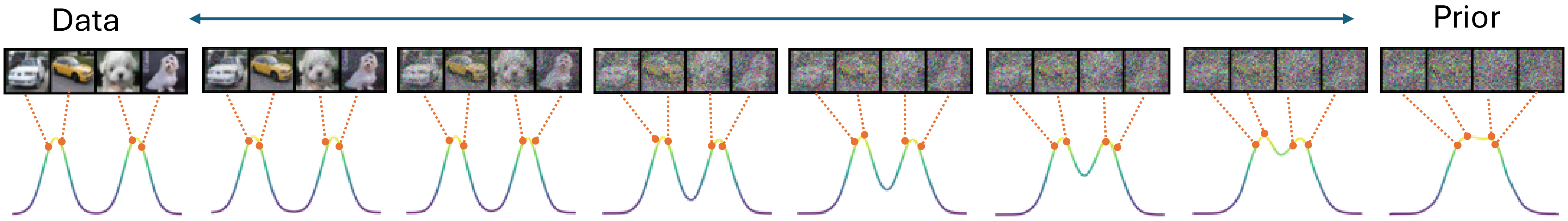


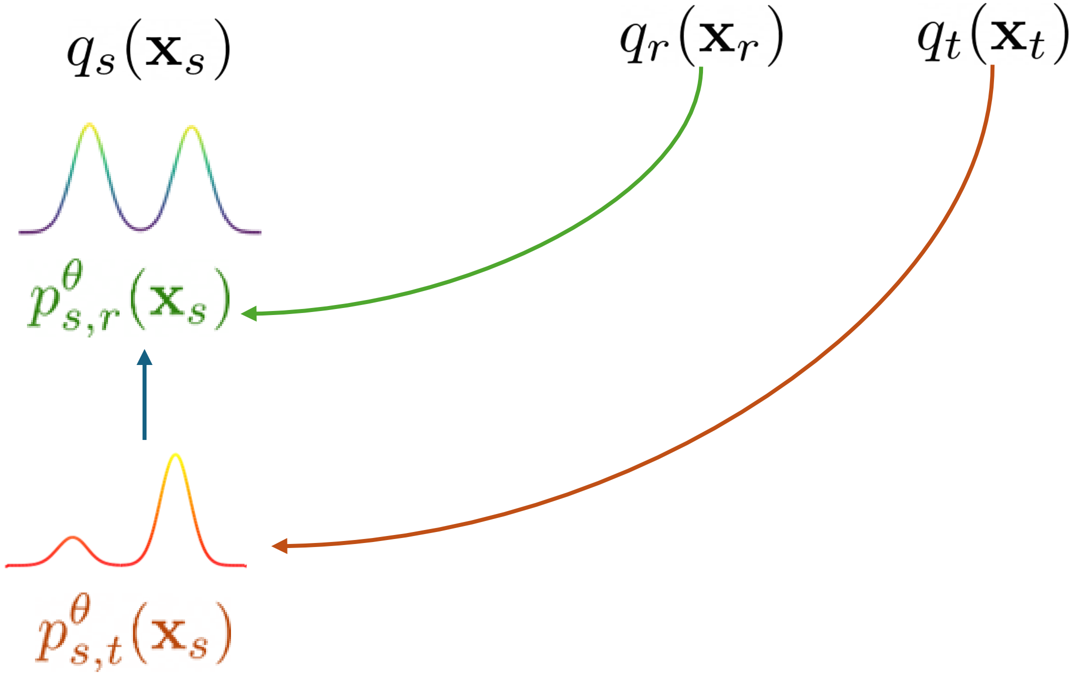
Base case:

$p^{\theta}_{s,s}(\mathbf{x}_s) = q_s(\mathbf{x}_s)$

True regardless of θ

An Intuition on Inductive Learning





Inductive assumption:

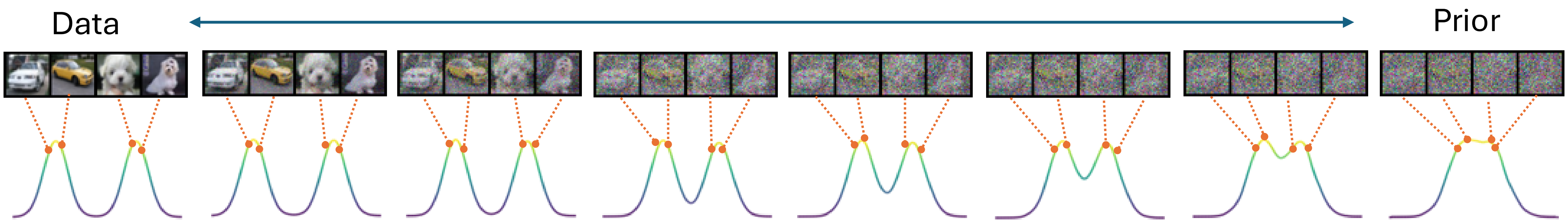
$$q_s(\mathbf{x}_s) = p^{\theta}_{s,r}(\mathbf{x}_s)$$

Base case:

$$p^{\theta}_{s,s}(\mathbf{x}_s) = q_s(\mathbf{x}_s)$$

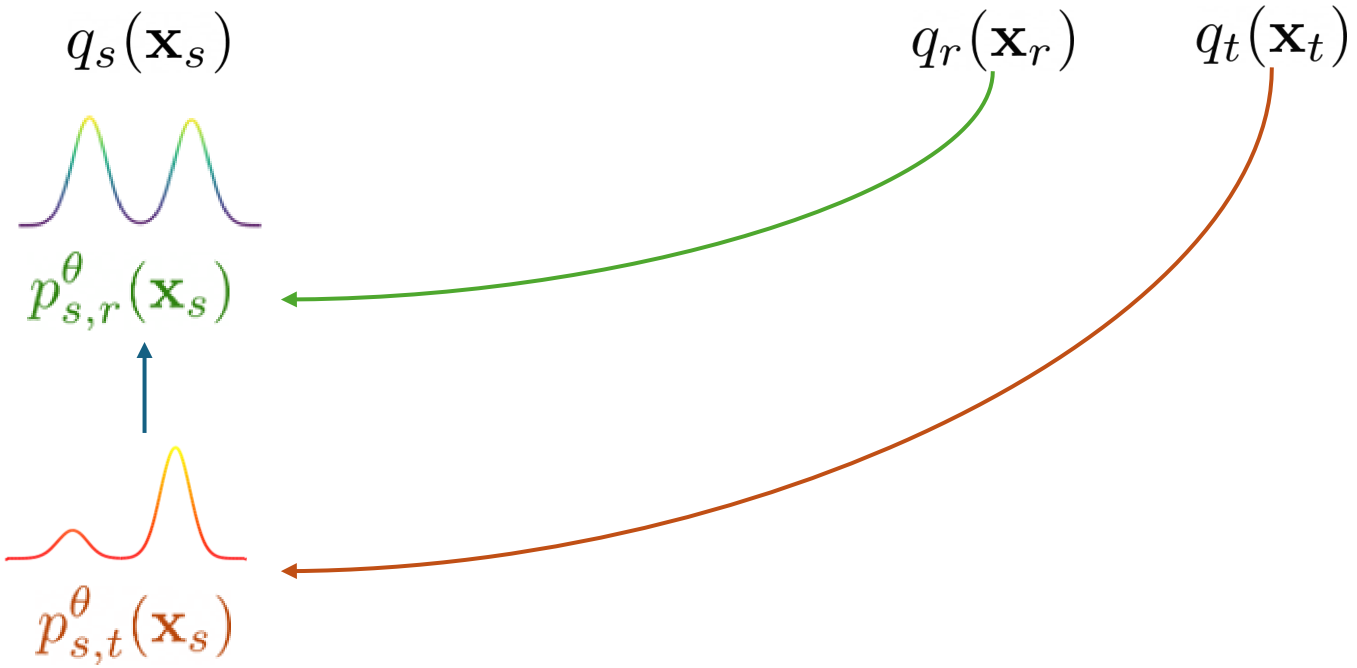
True regardless of θ

An Intuition on Inductive Learning



Inductive assumption:

$q_s(\mathbf{x}_s) = p^{\theta}_{s,r}(\mathbf{x}_s)$



Base case:

$p^{\theta}_{s,s}(\mathbf{x}_s) = q_s(\mathbf{x}_s)$

True regardless of θ

Stable Training

- Stable training as long as >=4 particles

- Consistency training is a 1-particle special case

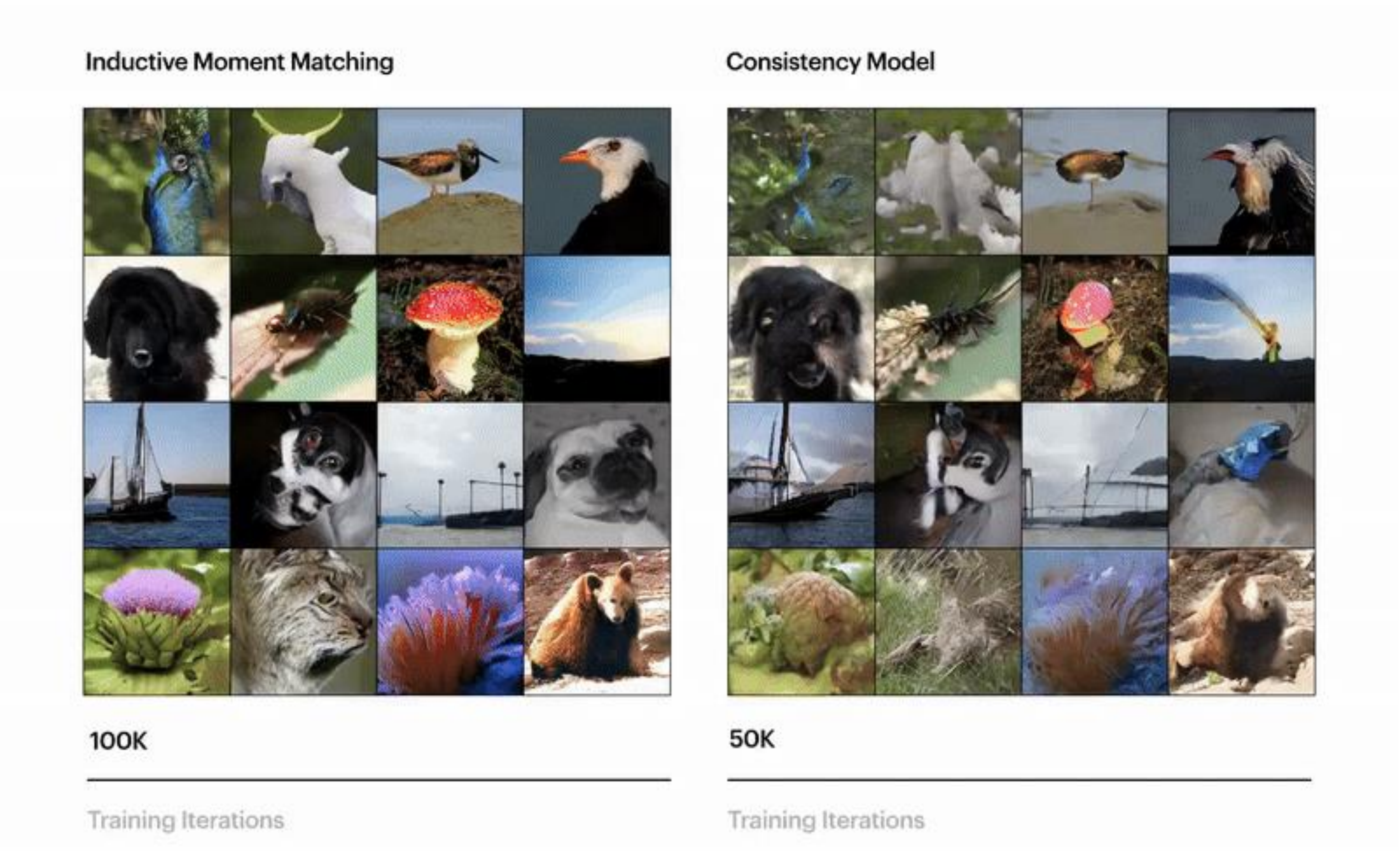
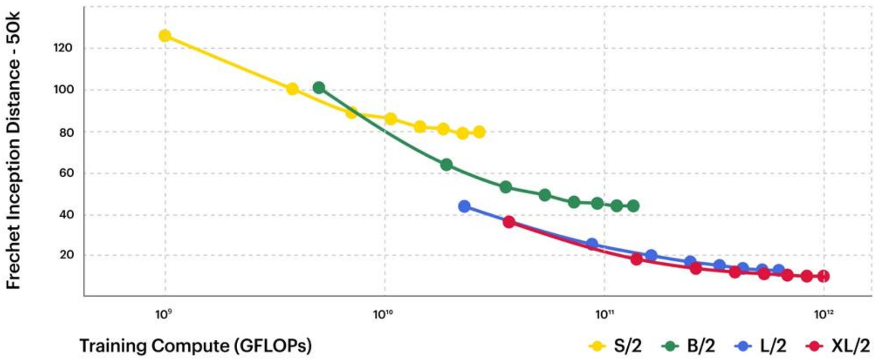


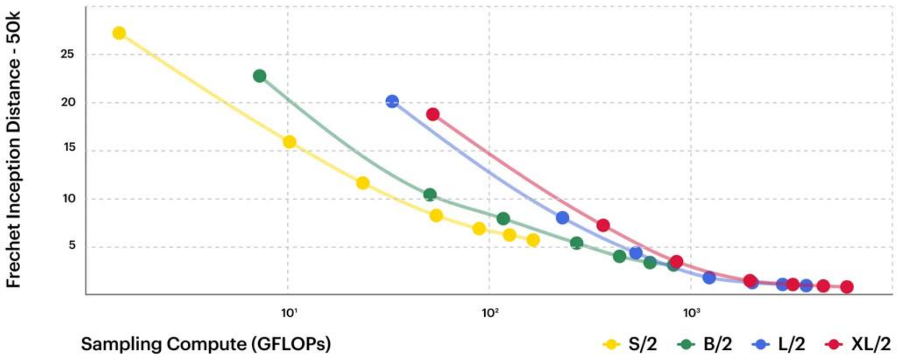
Image Generation

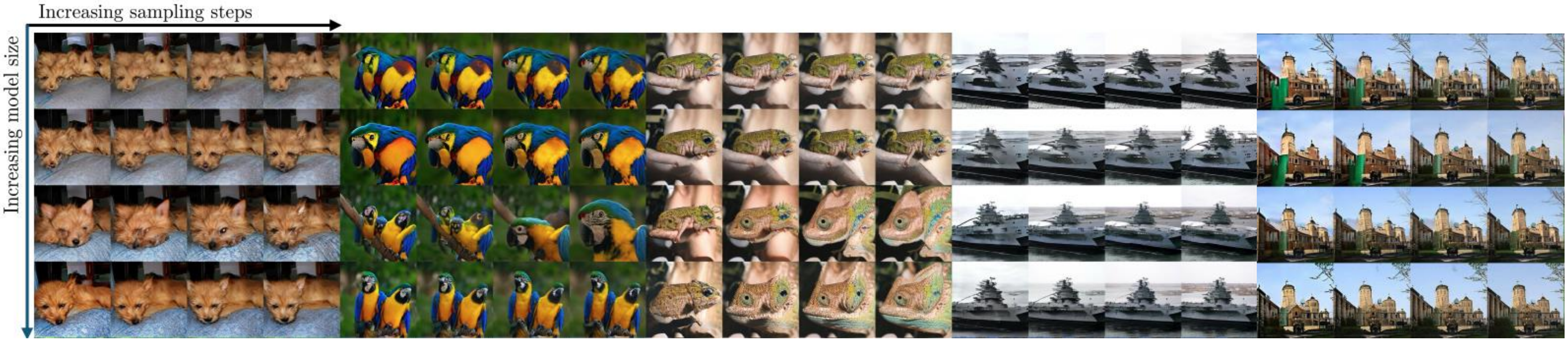
- ImageNet-256x256

	FID
DiT (250-step)	2.27
SiT (250-step)	2.15
VAR-d20	2.57
VAR-d30	1.92
IMM (8-step)	**1.99**
(16-step)	**1.90**

Scaling Property







Relation to Other Works

- Consistency Training (2023)
  - Particle number = 1, L2 kernel
- Generative Moment Matching Networks (2015)
  - t = 1, s=r=0
- Generative Modeling via Drifting (2026)
  - Drift field parameterized via MMD attraction + repulsion

Limitations of IMM

- Multi-sample objective
  - => difficult to scale to high-dimension

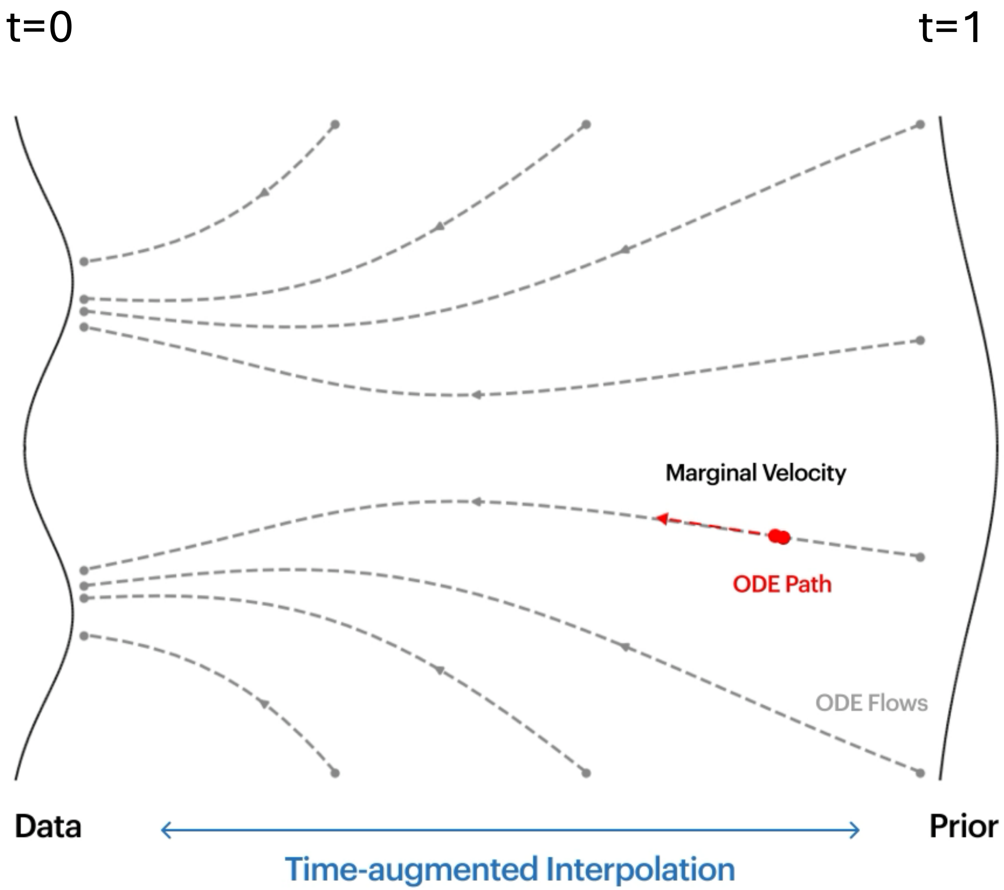
- Mapping function $r(s,t)$ requires high precision (e.g. FP16)
  - => Large scale models require BF16 or lower

Terminal Velocity Matching

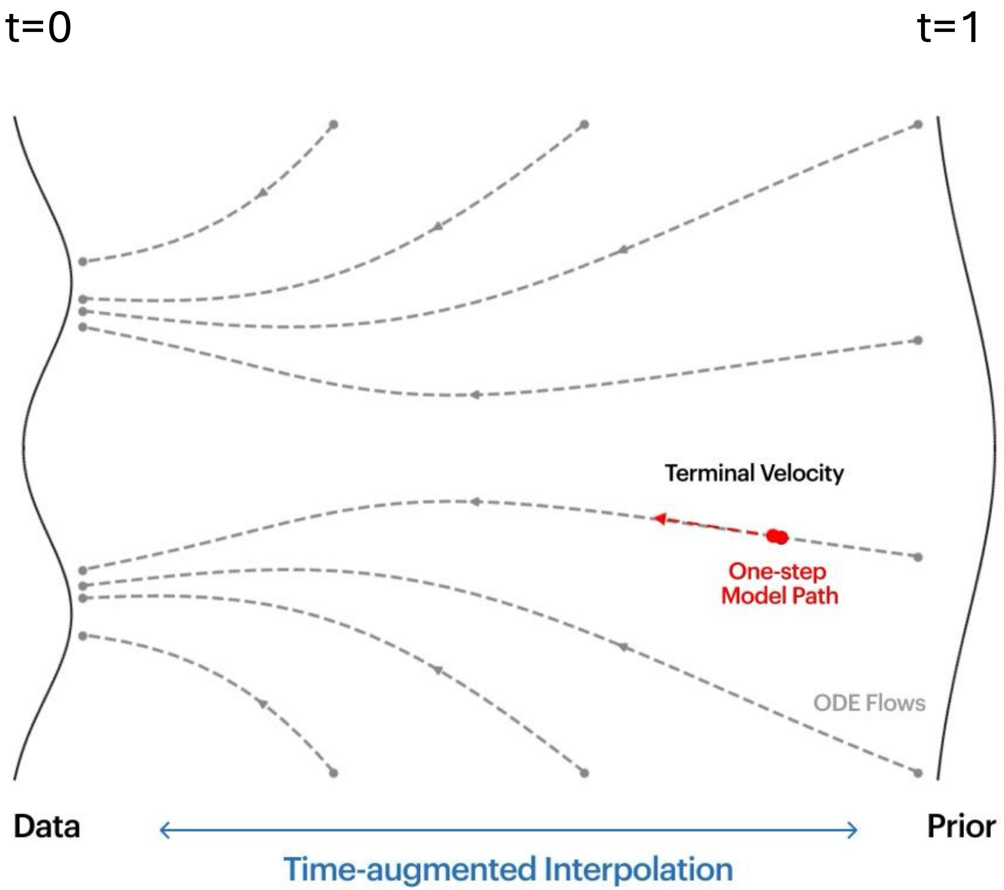
Going Back to Flows

Intuition for Terminal Velocity Matching

Diffusion



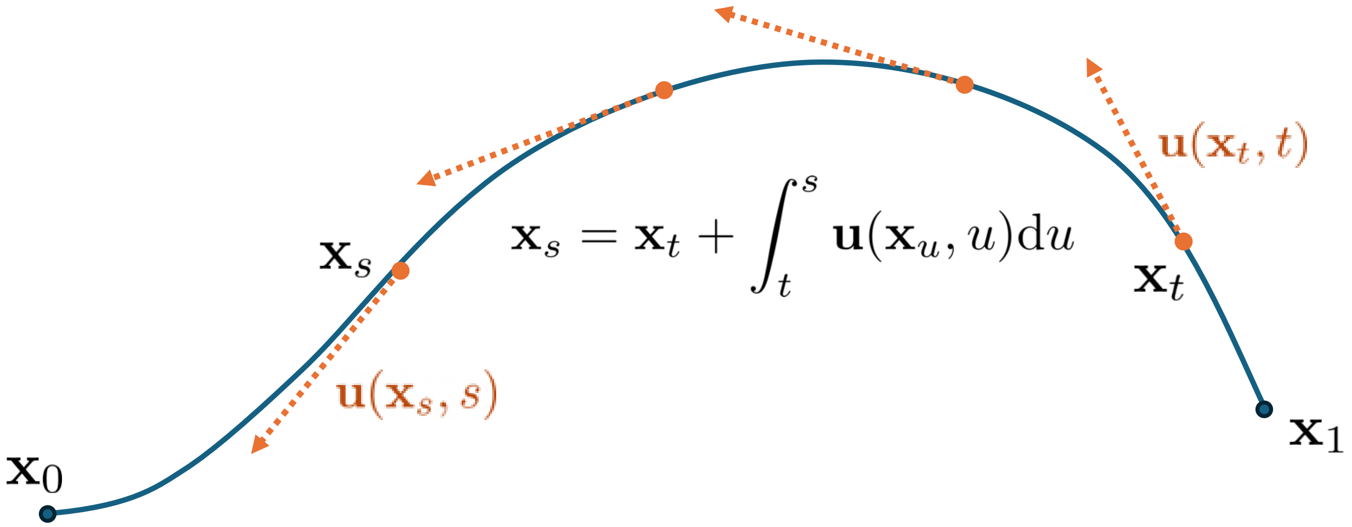
TVM



Learning One-Step Trajectory Mapping

$$\mathbf{f}(\mathbf{x}_t, t, s) = \int_t^s \mathbf{u}(\mathbf{x}_r, r)\mathrm{d}r$$

Diffusion/FM



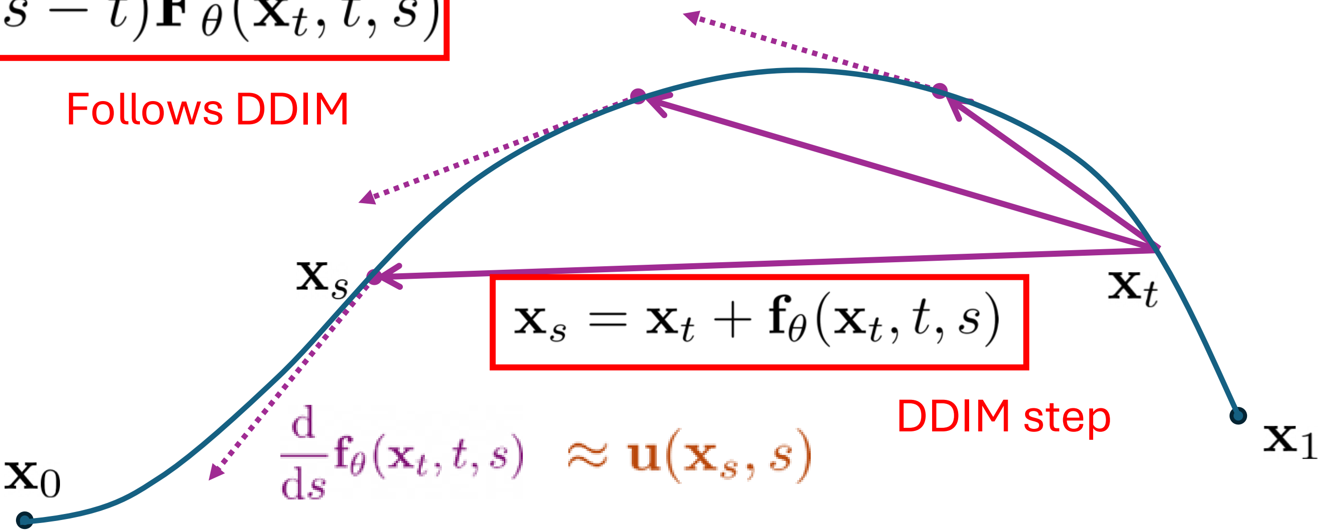
Learning One-Step Trajectory Mapping

$$\mathbf{f}(\mathbf{x}_t, t, s) = \int_t^s \mathbf{u}(\mathbf{x}_r, r)\mathrm{d}r$$

$$\mathbf{f}_\theta(\mathbf{x}_t, t, s) = \boxed{(s-t)\mathbf{F}_\theta(\mathbf{x}_t, t, s)}$$

Follows DDIM

TVM



Naively: Match Displacement!

$$\mathcal{L}^t_{\text{displ}} = \mathbb{E}_{\mathbf{x}_t}\left[\left\| \mathbf{f}_\theta(\mathbf{x}_t, t, 0) - \int_t^0 \mathbf{u}(\mathbf{x}_r, r)\mathrm{d}r \right\|_2^2\right]$$

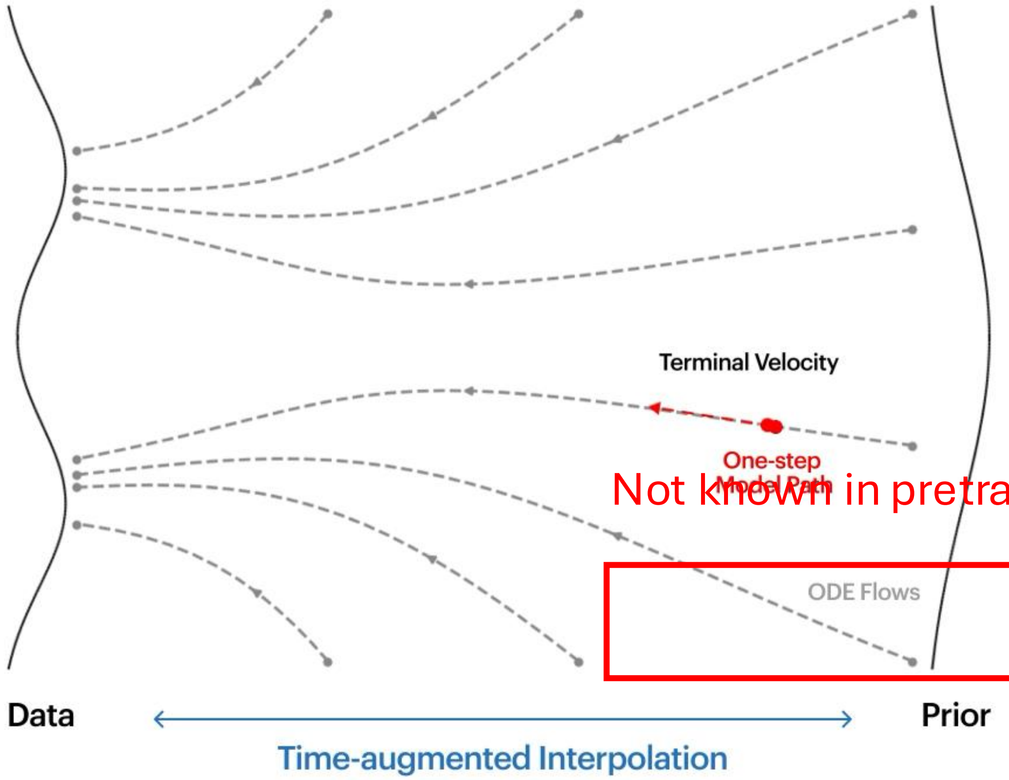
$$\boxed{\frac{\mathrm{d}}{\mathrm{d}s}\mathbf{f}(\mathbf{x}_t, t, s) = \mathbf{u}(\mathbf{f}(\mathbf{x}_t, t, s), s)}$$

Terminal Velocity Condition

Learning One-Step Trajectory Mapping

- Displacemer
- Terminal Velo

$(\mathbf{x}_r, r)\mathrm{d}r\Big\|_2^2\Big]$



Not known in pretraining!

$\mathcal{L}^t_{\mathrm{displ}}(\theta) \leq$

$\mathrm{d}s$

Minimize!

Proxy for Ground-Truth

- Just use network itself as approximation

$$\mathbf{u}_\theta(\mathbf{x}_t + \mathbf{f}_\theta(\mathbf{x}_t, t, s), s) \approx \mathbf{u}(\mathbf{x}_t + \mathbf{f}(\mathbf{x}_t, t, s), s)$$

Terrible approximation at start of training!

  - Solution: Train such that $\mathbf{u}_\theta(\mathbf{x}_t, t) \approx \mathbf{u}(\mathbf{x}_t, t)$

Flow Matching

- Final Objective:

$$\mathcal{L}_{\text{TVM}}^{t,s}(\theta) = \mathbb{E}_{\mathbf{x}_t, \mathbf{x}_s, \mathbf{v}_s}\left[\left\|\frac{\mathrm{d}}{\mathrm{d}s}\mathbf{f}_\theta(\mathbf{x}_t, t, s) - \mathbf{u}_\theta(\mathbf{x}_t + \mathbf{f}_\theta(\mathbf{x}_t, t, s), s)\right\|_2^2 + \left\|\mathbf{u}_\theta(\mathbf{x}_s, s) - \mathbf{v}_s\right\|_2^2\right]$$

Relation to Distribution Divergence

- TVM loss bounds Wasserstein distance up to a constant

Network Lipschitzness

$$W_2^2(\mathbf{f}^\theta_{t\to 0}\#p_t, p_0) \le \int_0^t \lambda[L](s)\mathcal{L}^{t,s}_{\mathrm{TVM}}(\theta)\mathrm{d}s + C,$$

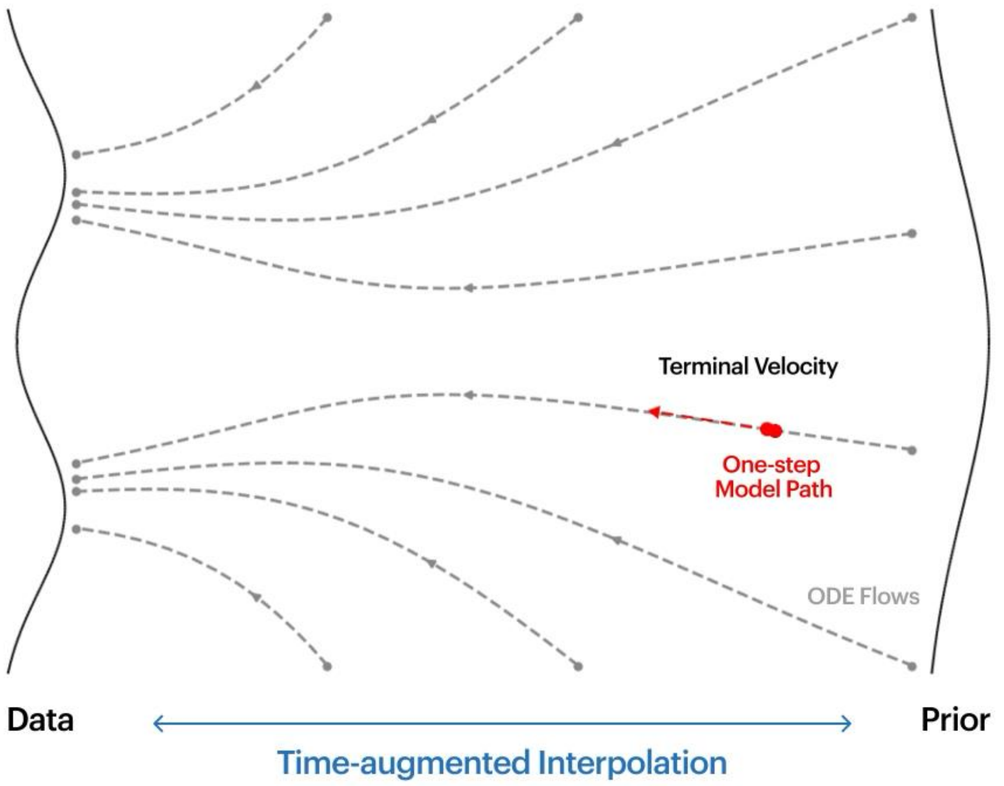
- **Implication:** diffusion transformers are **NOT Lipschitz-continuous**. Need ”semi-Lipschitz control”!

  - LayerNorm -> RMSNorm
  - QKNorm w/ RMSNorm
  - Add RMSNorm without parameters to time modulation

Calculating Terminal Velocity

- How to calc
- Recall: f

$$\frac{d}{ds}$$

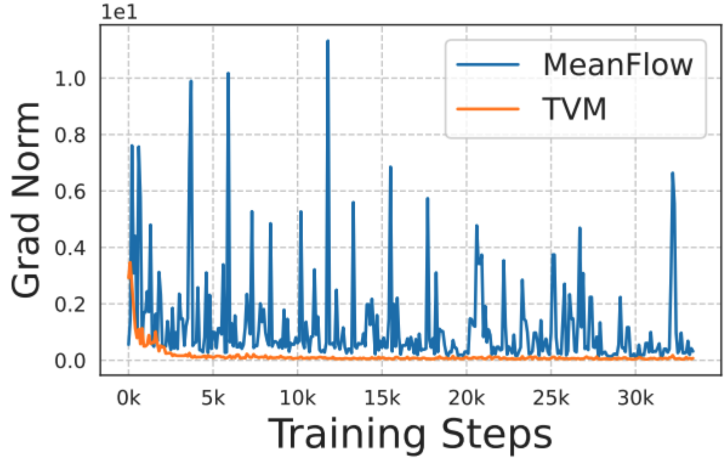


(JVP)

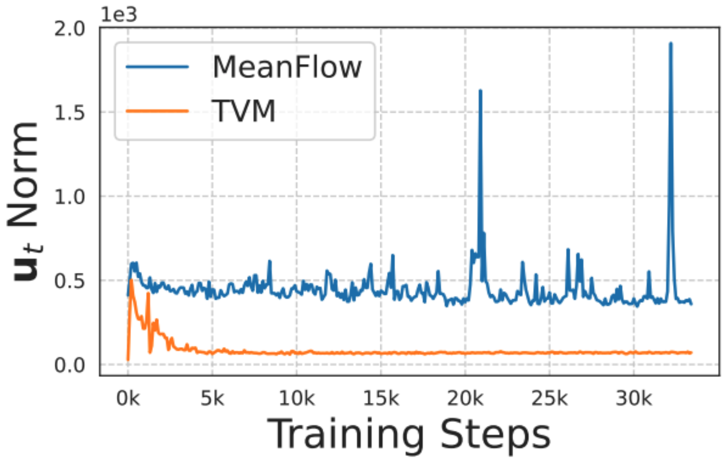
ckwards

Stable Training vs. MeanFlow

- Well-conditioned gradient profile



- Well-conditioned norm $\mathbf{u}_\theta(\mathbf{x}_t + \mathbf{f}_\theta(\mathbf{x}_t, t, s), s)$ w/ random CFG



ImageNet Generation

- SOTA 1-NFE
- Outperforms DiT, SiT in 4 NFE

One-step samples

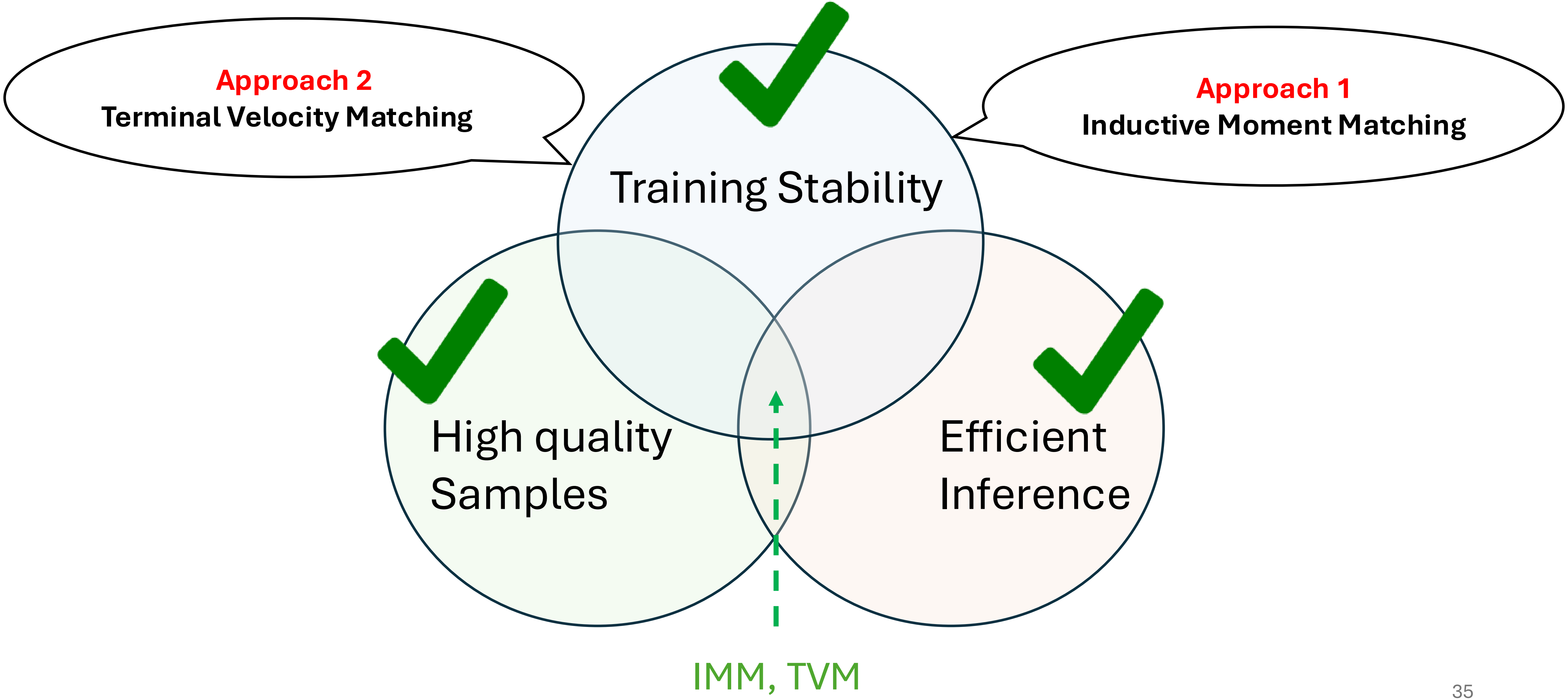
TVM at 10B+ Scale

https://lumalabs.ai/blog/engineering/tvm

4-NFE TVM on T2I

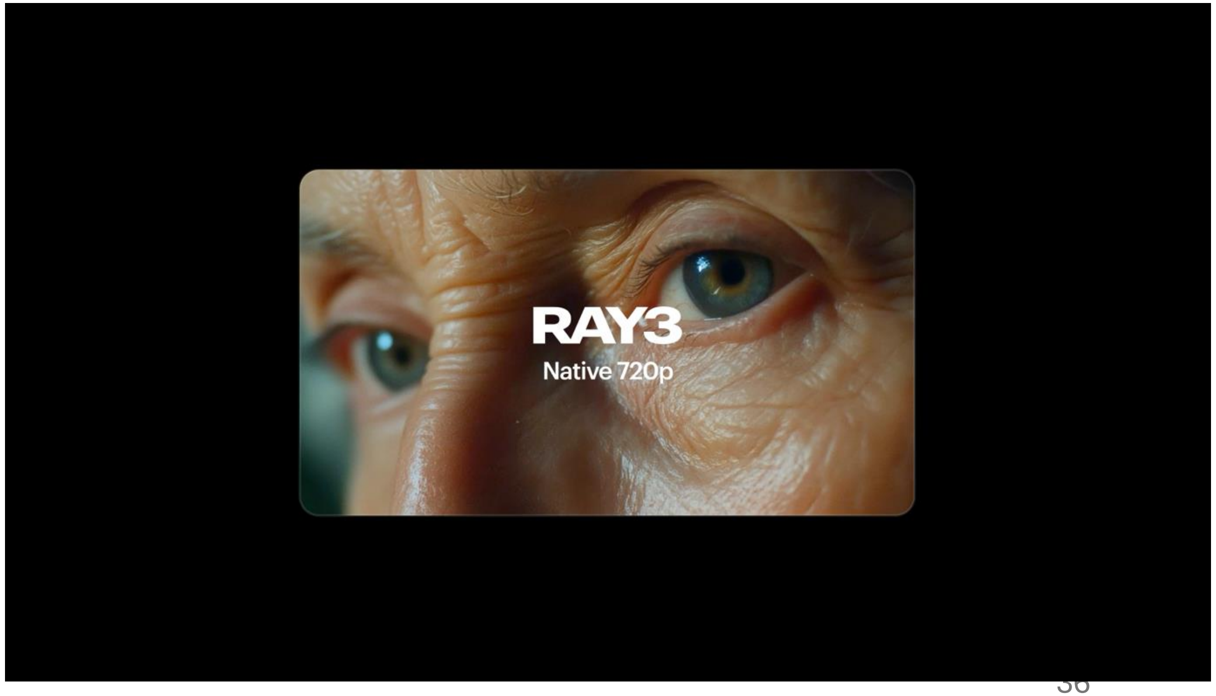
- Challenges at Scale:
  - FSDP with JVP – Solution: Wrap JVP inside each layer of FSDP.
  - Writing JVP kernel (with backwards support) for arbitrary sequence length

Desiderata of Efficient Inference-Time Scaling



Luma AI is a research and product lab aiming to build multimodal AGI.

- Recent Series-C: $900M.
- Average age in research team: ~27
- Inventors of **DDIM** & **NeRF** work here.



If you are interested in advancing multimodal generative AI, join us!

https://lumalabs.ai/join

Research team have multiple roles open around:

1. Omni models (unifying understanding + generation)
2. Video / Audio models
3. Voice agents
4. World models
5. Multimodal agents
6. AI Infra

(“Internships / Residency” also available, but ideally >= 6 months and not driven by publications)