Distillation

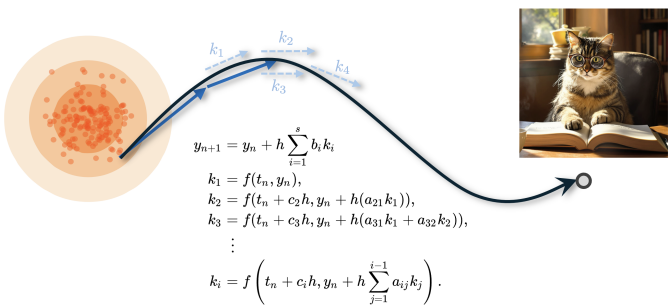
$$\{Z_t\} = \texttt{Rectify}(\{X_t\})$$

Are you doing $\texttt{Distill}(\{Z_t\})$ or $\texttt{Distill}(\{X_t\})$?

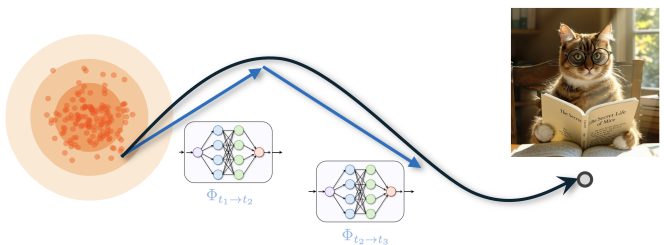
Inference Methods

- Once the ODE is learned, the next step is efficient inference.
- **Goal**: Speed up the inference process.
- **Numerical methods**: Use off-the-shelf ODE/SDE solvers (e.g., Runge-Kutta, Euler-Maruyama).
- **Neural methods**: Train neural networks to approximate the ODE/SDE trajectories directly.

Numerical Methods



Neural Methods



Neural Distillation of ODEs

- Given $\mathrm{d}Z_t = v_t(Z_t)\mathrm{d}t$. Let $\Phi_{t\to s}$ be the transport (or flow) map from Z_t to Z_s:

$$\Phi_{t\to s}(Z_t) = Z_s, \qquad \forall s, t.$$

- **Goal:** Learn neural approximation of $\Phi_{t\to s}$:

$$\Phi_{t\to s}(x) = f^{\theta}(x, t, s).$$

- **Examples:** Consistency Models [SDCS23], Consistent Trajectory Models [KLL$^+$23], Flow Map Matching [BAVE25b], Shortcut Models [FHLA24], Mean-flow [GDB$^+$25], etc.
- **Key question:** what loss function should we use?

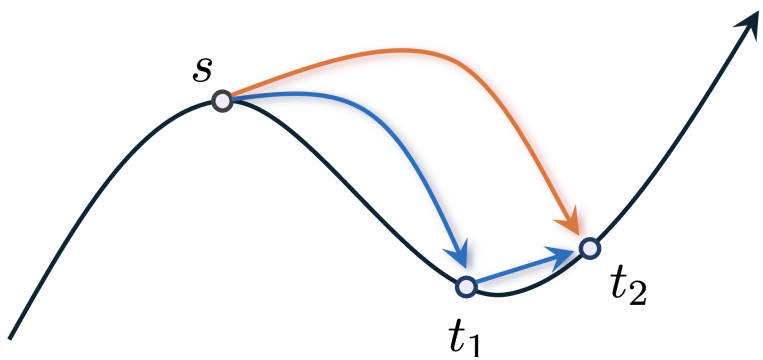
ODE Distillation: Forward Losses

- Obviously, $Z_t = \Phi_{s\to t}(x)$ should follow the ODE for every s and x:

  Forward equation: $\dfrac{\mathrm{d}}{\mathrm{d}t}\Phi_{s\to t}(x) = v_t(\Phi_{s\to t}(x)), \qquad \forall s \in [0,1], x \in \mathbb{R}^d.$

- Local forward loss:

$$L_{fwd}(\Phi) := \mathbb{E}\left[\left\|\frac{\mathrm{d}}{\mathrm{d}t}\Phi_{s\to t}(x_s, s) - v_t(\Phi_{s\to t}(x_s, s))\right\|^2\right]$$



ODE Distillation: Forward Losses

- Obviously, $Z_t = \Phi_{s\to t}(x)$ should follow the ODE for every s and x:

  Forward equation: $\dfrac{\mathrm{d}}{\mathrm{d}t}\Phi_{s\to t}(x) = v_t(\Phi_{s\to t}(x)), \qquad \forall s \in [0,1], x \in \mathbb{R}^d.$

- Local forward loss:

$$L_{fwd}(\Phi) := \mathbb{E}\left[\left\|\frac{\mathrm{d}}{\mathrm{d}t}\Phi_{s\to t}(x_s, s) - v_t(\Phi_{s\to t}(x_s, s))\right\|^2\right]$$

- Global forward loss:

$$L_{fwdSolver}(\Phi) := \mathbb{E}\left[\left\|\Phi_{s\to t_2}(x_s, s) - \mathrm{ODESolver}_{t_1\to t_2}(\Phi_{s\to t_1}(x_s, s))\right\|^2\right],$$

  where $\mathrm{ODESolver}_{t_1\to t_2}$ is any numerical solver that integrates t_1 to t_2.

- Examples: Consistent Trajectory Models [KLL+23], Flow Map Matching [BAVE25b], etc.

ODE Distillation: Backward Losses

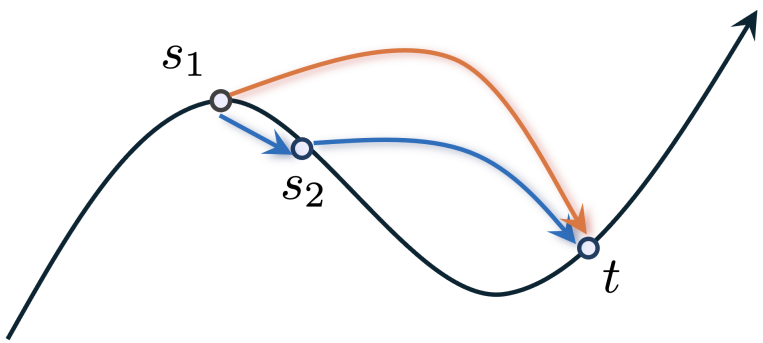
- Along the trajectory the ODE trajectory $\{Z_t\}$, we have

$$Z_t = \Phi_{s\to t}(Z_s).$$

  Left side is independent of s.

- Backward equation: Taking derivative w.r.t. s on both sides:

$$\frac{\mathrm{d}}{\mathrm{d}s}\Phi_{s\to t}(x_s, s) = 0 \quad \Longrightarrow \quad \partial_s \Phi_{s\to t}(x_s, s) + \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot v_s(x_s) = 0.$$



ODE Distillation: Backward Losses

- Along the trajectory the ODE trajectory $\{Z_t\}$, we have

$$Z_t = \Phi_{s\to t}(Z_s).$$

  Left side is independent of s.

- Backward equation: Taking derivative w.r.t. s on both sides:

$$\frac{\mathrm{d}}{\mathrm{d}s}\Phi_{s\to t}(x_s, s) = 0 \quad \Longrightarrow \quad \partial_s \Phi_{s\to t}(x_s, s) + \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot v_s(x_s) = 0.$$

- Backward loss:

$$L_{bwd}(\hat{X}) := \mathbb{E}\left[\left\|\partial_s \hat{X}_t(x_s, s) + \nabla_{x_s}\hat{X}_t(x_s, s)\cdot v_s(x_s)\right\|^2\right]$$

  Essentially the same as Bellman backward loss in Q learning.

- Examples: Consistency Models [SDCS23], Mean-flow [GDB+25], etc.
- Helpful: finite different approximation + stop gradient.

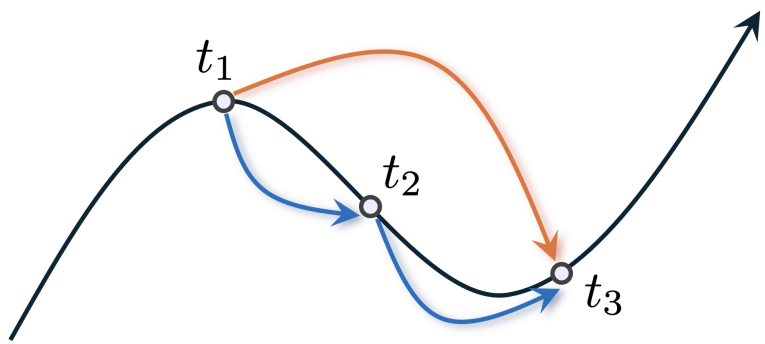
ODE Distillation: Three-Point Consistency Loss

- Tri-consistency loss (semigroup property):

$$\Phi_{t_1 \to t_2}(\Phi_{t_0 \to t_1}(x)) = \Phi_{t_0 \to t_2}(x), \qquad \forall t_0, t_1, t_2 \in [0,1], \quad \forall x \in \mathbb{R}^d.$$

- Tri-consistency loss:

$$L_{tri}(\Phi) = \mathbb{E}\left[\left\|\Phi_{t_1 \to t_2}(\Phi_{t_0 \to t_1}(x_{t_0})) - \Phi_{t_0 \to t_2}(x_{t_0})\right\|^2\right].$$



ODE Distillation: Three-Point Consistency Loss

- Tri-consistency loss (semigroup property):

$$\Phi_{t_1 \to t_2}(\Phi_{t_0 \to t_1}(x)) = \Phi_{t_0 \to t_2}(x), \qquad \forall t_0, t_1, t_2 \in [0,1], \quad \forall x \in \mathbb{R}^d.$$

- Tri-consistency loss:

$$L_{tri}(\Phi) = \mathbb{E}\left[\left\|\Phi_{t_1 \to t_2}(\Phi_{t_0 \to t_1}(x_{t_0})) - \Phi_{t_0 \to t_2}(x_{t_0})\right\|^2\right].$$

- Examples: Shortcut Models [FHLA24], TraFLow [WFWC25], etc.

- Can recover other losses as special cases:
  - $L_{tri}(\Phi) \approx L_{fwd}(\Phi)$ when $t_1 \approx t_2$.
  - $L_{tri}(\Phi) \approx L_{bwd}(\Phi)$ when $t_0 \approx t_1$.
  - Middle point: $t_1 = (t_0 + t_2)/2$, e.g., Shortcut model [FHLA24].

Forward vs. Backward

- Forward loss:

$$L_{\text{fwd}}(\hat{X}) := \mathbb{E}\left[\|\underbrace{-\partial_t \Phi_{s\to t}(x_s) + v_t(\Phi_{s\to t}(x_s))}_{\Delta_{\text{fwd}}(x_s,s;t)}\|^2\right]$$

- Backward loss:

$$L_{\text{bwd}}(\hat{X}) := \mathbb{E}\left[\|\underbrace{\partial_s \Phi_{s\to t}(x_s) + \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot v_s(x_s)}_{\Delta_{\text{bwd}}(x_s,s;t)}\|^2\right]$$

- Connection: For $x_t = \Phi_{s\to t}(x_s)$, we have:

$$\Delta_{\text{bwd}}(x_s, s; t) = \underbrace{\nabla_{x_s}\Phi_{s\to t}(x_s)}_{\text{Jacobian}} \cdot \Delta_{\text{fwd}}(x_t, t; s).$$

Backward = Jacobian × Forward

Proof.

We have the identity:

$$\Phi_{s\to t}(\Phi_{t\to s}(x_t)) = x_t.$$

Differentiating both sides with respect to s gives:

$$\nabla_{x_s}\Phi_{s\to t}(x_s)\cdot\partial_s\Phi_{t\to s}(x_t) + \partial_s\Phi_{s\to t}(x_s) = 0.$$

Rewriting the backward error term:

$$\begin{aligned}
\Delta_{\mathrm{bwd}}(x_s, s; t) &= \partial_s\Phi_{s\to t}(x_s) + \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot v_s(x_s) \\
&= -\nabla_{x_s}\Phi_{s\to t}(x_s)\cdot\partial_s\Phi_{t\to s}(x_t) + \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot v_s(x_s) \\
&= \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot(-\partial_s\Phi_{t\to s}(x_t) + v_s(x_s)) \\
&= \nabla_{x_s}\Phi_{s\to t}(x_s)\cdot\Delta_{\mathrm{fwd}}(x_t, t; s).
\end{aligned}$$

□

Boundary Conditions of $\Phi_{s\to t}$

- We can examine boundary conditions to guide the design of parameterized networks.
- The map $\Phi_{s\to t}$ reduces to the identity when $s = t$:

$$\Phi_{t\to t}(x) = x.$$

This motivates the following first-order parameterization:

$$\Phi^{\theta}_{s\to t}(x) = x + (t - s) \cdot \texttt{neuralnet}^{\theta}(x; t, s).$$

Boundary Conditions of $\Phi_{s\to t}$

- We can examine boundary conditions to guide the design of parameterized networks.
- The map $\Phi_{s\to t}$ reduces to the identity when $s = t$:

$$\Phi_{t\to t}(x) = x.$$

This motivates the following first-order parameterization:

$$\Phi^{\theta}_{s\to t}(x) = x + (t - s) \cdot \texttt{neuralnet}^{\theta}(x; t, s).$$

- Moreover, $\Phi_{s\to t}$ should locally match the pretrained velocity v_t via

$$\partial_t \Phi_{s\to t}(x)\big|_{s=t} = v_t(x).$$

This motivates a second-order form [Liu24]:

$$\Phi_{s\to t}(x) = x + (t - s)v_t(x) + (t - s)^2 \cdot \texttt{neuralnet}^{\theta}(x; t, s).$$

Distillation Losses: Summary

Paper	Forward	Backward	Tri-consistency
MeanFlow [GDB+25]		✓	
Flow Map Matching [BAVE25b]	✓	✓	
sCM [LS24]		✓	
Align Your Flow [SFK25]		✓	
[BAVE25a]	✓	✓	✓
PD [SH22]			✓
CTM [KLL+23]	✓		✓
ShortCut Model [FHLA24]			✓
Traflow [WFWC25]			✓
[LY25]			✓

Open Questions:

- Best choices / combinations of losses?
  - Optimization difficulty vs. performance
- Optimal choices of time grids t_0, t_1, t_2?
- Role of stopping gradient.

Interpolation Distillation

- Goal: Directly train a one-step generative model

$$X^\theta = G^\theta(\xi, t)$$

to approximate the distribution of $X_1 \sim \rho_1$, using a pretrained rectified flow.

- Idea: Match the interpolation processes induced by X^θ and real data:

$$X_t^\theta = t\, G^\theta(\xi, t) + (1-t)\, X_0, \qquad X_t^{\texttt{data}} = t\, X^{\texttt{data}} + (1-t)\, X_0,$$

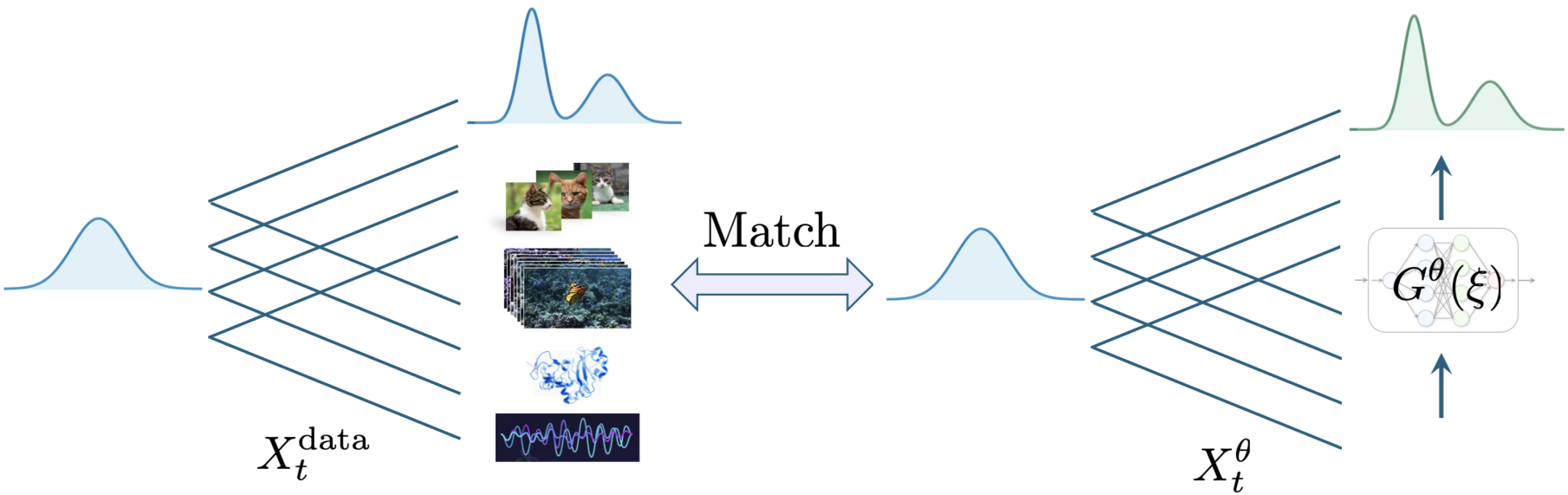
where $X_0 \sim \pi_0$ is independent noise.

- The goal is to match their induced rectified flows:

$$\texttt{Rectify}(\{X_t^\theta\}) \approx \texttt{Rectify}(\{X_t^{\texttt{data}}\}).$$

Interpolation Distillation

$$\texttt{Rectify}(\{X_t^\theta\}) \approx \texttt{Rectify}(\{X_t^{\texttt{data}}\}).$$



Velocity Matching Objective

- Minimize the velocity matching loss:

$$L(\theta) = \int_0^1 w_t \, \mathbb{E}\left[\| v_t^\theta(X_t^\theta) - v_t^{\texttt{pretrained}}(X_t^{\texttt{data}})\|^2\right] \mathrm{d}t.$$

Student velocity v_t^θ:

$$v_t^\theta(x) = \mathbb{E}\left[\dot{X}_t^\theta \mid X_t^\theta = x\right],$$

estimated dynamically as θ updates.

Teacher velocity $v_t^{\texttt{pretrained}}$:

$$v_t^{\texttt{data}}(x) = \mathbb{E}\left[\dot{X}_t^{\texttt{data}} \mid X_t^{\texttt{data}} = x\right],$$

from the pretrained rectified flow.

- In practice, alternate between estimating v_t^θ and minimizing $L(\theta)$, which can be cast as a *minimax* optimization problem.
- Since v_t^θ is not directly accessible from one-step model θ, an auxiliary model v_t^ψ can be introduced by learning $\texttt{Rectify}(X_t^\theta)$.

Loss Functions for Interpolation Distillation

- **Velocity matching loss**:

$$L_{\text{vel}}(\theta) = \int_0^1 w_t \, \mathbb{E}\left[\| v_t^\theta(X_t^\theta) - v_t^{\texttt{data}}(X_t^\theta)\|^2\right] \, \mathrm{d}t.$$

- **Integrated KL loss**:

$$L_{\text{KL}}(\theta) = \int_0^1 w_t \, \mathrm{KL}\left(\rho_t^\theta \parallel \rho_t^{\texttt{data}}\right) \mathrm{d}t.$$

- **Score matching (Fisher divergence) loss**:

$$L_{\text{Fisher}}(\theta) = \int_0^1 w_t \mathbb{E}\left[\left\|\nabla \log \rho_t^\theta(X_t^\theta) - \nabla \log \rho_t^{\texttt{data}}(X_t^\theta)\right\|^2\right] \mathrm{d}t.$$

Loss Functions for Interpolation Distillation

- **Velocity matching loss**:

$$L_{\mathsf{vel}}(\theta) = \int_0^1 w_t \, \mathbb{E}\left[\|v_t^\theta(X_t^\theta) - v_t^{\mathtt{data}}(X_t^\theta)\|^2\right] \mathrm{d}t.$$

- **Integrated KL loss**:

$$L_{\mathsf{KL}}(\theta) = \int_0^1 w_t \, \mathrm{KL}\left(\rho_t^\theta \parallel \rho_t^{\mathtt{data}}\right) \mathrm{d}t.$$

- *Examples: Variational Score Distillation [WLW+23], Diff-instruct [LHZ+23], Distribution Matching Distillation [YGZ+24, YGP+24], Swift Brush [NT24], f-distill [XNV25]...*

- **Score matching (Fisher divergence) loss**:

$$L_{\mathsf{Fisher}}(\theta) = \int_0^1 w_t \mathbb{E}\left[\left\|\nabla \log \rho_t^\theta(X_t^\theta) - \nabla \log \rho_t^{\mathtt{data}}(X_t^\theta)\right\|^2\right] \mathrm{d}t.$$

- *Examples: Score Identity Distillation [ZZW+24], Score Implicit Matching [LHG+24], Continuous Semi-Implicit Models [YZY+25], Uni-Instruct [WBZ+25]...*

Relationship between Losses

- Of course, these losses are not disconnected.
- In the case of independent Gaussian noise, they are equivalent up to a change of time weighting. For example,

$$\int_0^1 \mathrm{KL}(\rho_t^\theta \,\|\, \rho_t^{\texttt{data}})\,\mathrm{d}t = \int_0^1 t\,\mathbb{E}\left[\|v_t^\theta(X_t^\theta) - v_t^{\texttt{data}}(X_t^\theta)\|^2\right]\,\mathrm{d}t$$

$$= \int_0^1 \frac{(1-t)^2}{t}\,\mathbb{E}\left[\|\nabla \log \rho_t^\theta(X_t^\theta) - \nabla \log \rho_t^{\texttt{data}}(X_t^\theta)\|^2\right]\,\mathrm{d}t$$

- Thus, minimizing velocity or score matching indirectly reduces KL divergence.