Understanding Recent **One-step** Flow Matching Approaches

Ruizhe Shi

University of Washington

March 11, 2026

Three Key Questions of Generative Modeling

- How to learn a good description of the data distribution from finite data?
- Can we approximately sample from the data distribution by leveraging the learned description of the data distribution?
- How can we efficiently sample?

Three Key Questions of Generative Modeling

- How to learn a good description of the data distribution from finite data?
  Sliced Score Matching. [UAI 2019])
  Flow Matching for Scalable Generative Modeling. [ICLR 2023]
- Can we approximately sample from the data distribution by leveraging the learned description of the data distribution?
  Sampling is as easy as learning the score. [ICLR 2023]
- How can we efficiently sample?
  Parallel sampling, one-step sampling, . . .

Preliminary of Flow Matching

Push-forward

$p_0 \in \Delta(\mathbb{R}^d)$ is the data distribution, $p_1 = \mathcal{N}(0, I_d)$ is the noise distribution. We want to have a deterministic mapping $f : \mathbb{R}^d \to \mathbb{R}^d$, s.t.

$$p_0 = f_{\#} p_1 \ ,$$

which means

$$\text{Law}(f(x_1)) = p_0 \ , \ x_1 \sim p_1 \ .$$

Training: Learn f.
Inference: Sample $x_1 \sim p_1$, get $x_0 = f(x_1)$, which satisfies $x_0 \sim p_0$.

Continuous push-forward: velocity field

Suppose we have a velocity field $v(x_t, t)$, then we can define $f_t(x_t) = x_0$ by solving the ODE:

$$\dot{x}_t = v(x_t, t) \ .$$

We want the mapping function $f_t : \mathbb{R}^d \to \mathbb{R}^d$ to satisfy:

$$p_0 = f_{1\#} p_1 \ .$$

Training: Learn v.

Inference: Sample $x_1 \sim p_1$, get x_0 by solving the ODE:

$$\dot{x}_t = v(x_t, t) \ ,$$

which satisfies $x_0 \sim p_0$.

Euler method: $\tilde{x}_{(n-1)k} = \tilde{x}_{nk} - v(\tilde{x}_{nk}, nk) \cdot k$, $n \in \mathbb{Z}$. Get $\tilde{x}_0$ to approximate x_0.

A simple construction of velocity field

Given $a \sim p_0$ and $b \sim p_1$, we construct a linear path

$$\tau(a, b) = \{((1-t)a + tb, t) | t \in [0, 1]\} \ .$$

A simple construction of velocity field

Given $a \sim p_0$ and $b \sim p_1$, we construct a linear path

$$\tau(a, b) = \{((1-t)a + tb, t) | t \in [0, 1]\} \; .$$

On this specific path, we have a constant velocity:

$$v_{\tau(a,b)} = \frac{d((1-t)a + tb)}{dt} = b - a \; .$$

A simple construction of velocity field

Given $a \sim p_0$ and $b \sim p_1$, we construct a linear path

$$\tau(a, b) = \{((1-t)a + tb, t) | t \in [0, 1]\} .$$

On this specific path, we have a constant velocity:

$$v_{\tau(a,b)} = \frac{d((1-t)a + tb)}{dt} = b - a .$$

Then we merge all these paths and velocities, and get

$$\begin{aligned} v(x, t) &= \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} v_{\tau(a,b)} \\ &= \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} [b - a] . \end{aligned}$$

A simple construction of velocity field

Given $a \sim p_0$ and $b \sim p_1$, we construct a linear path

$$\tau(a,b) = \{((1-t)a + tb, t) | t \in [0,1]\} .$$

On this specific path, we have a constant velocity:

$$v_{\tau(a,b)} = \frac{d((1-t)a + tb)}{dt} = b - a .$$

Then we merge all these paths and velocities, and get

$$\begin{aligned} v(x,t) &= \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} v_{\tau(a,b)} \\ &= \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} [b - a] . \end{aligned}$$

And we have $p_t = (1-t)p_0 + tp_1$.

A simple construction of velocity field

Given $a \sim p_0$ and $b \sim p_1$, we construct a linear path

$$\tau(a, b) = \{((1-t)a + tb, t) | t \in [0, 1]\} .$$

On this specific path, we have a constant velocity:

$$v_{\tau(a,b)} = \frac{d((1-t)a + tb)}{dt} = b - a .$$

Then we merge all these paths and velocities, and get

$$\begin{aligned} v(x, t) &= \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} v_{\tau(a,b)} \\ &= \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} [b - a] . \end{aligned}$$

And we have $p_t = (1-t)p_0 + tp_1$.

Each specific path is transporting an $a \sim p_0$ to a $b \sim p_1$, so v is transporting p_0 to p_1.

Preliminary of Flow Matching

Training velocity model

Recall we want to learn

$$v(x, t) = \mathop{\mathbb{E}}_{a \sim p_0, b \sim p_1, (1-t)a + tb = x} [b - a] .$$

Preliminary of Flow Matching

Training velocity model

Recall we want to learn

$$v(x,t) = \mathop{\mathbb{E}}_{a\sim p_0, b\sim p_1, (1-t)a+tb=x} [b-a] \, .$$

We have a neural network v_θ, a dataset $\mathcal{D} = \{a^{(i)}\}_{i=1}^N$, $p_0 = \text{Law}(a^{(i)})$. And we set $p_1 = N(0, I_d)$. Then we sample $t \in [0,1]$, and train

$$\mathop{\mathbb{E}}_{a\sim p_0, b\sim p_1} \left\| v_\theta((1-t)a + tb, t) - (b-a) \right\|^2 \, ,$$

Preliminary of Flow Matching

Training velocity model

Recall we want to learn

$$v(x,t) = \mathop{\mathbb{E}}_{a\sim p_0, b\sim p_1, (1-t)a+tb=x} [b-a] \, .$$

We have a neural network v_θ, a dataset $\mathcal{D} = \{a^{(i)}\}_{i=1}^N$, $p_0 = \mathrm{Law}(a^{(i)})$. And we set $p_1 = N(0, I_d)$. Then we sample $t \in [0,1]$, and train

$$\mathop{\mathbb{E}}_{a\sim p_0, b\sim p_1} \left\| v_\theta((1-t)a + tb, t) - (b-a) \right\|^2 ,$$

which is equivalent to

$$\mathop{\mathbb{E}}_{x\sim p_t} \left\| v_\theta(x,t) - \mathop{\mathbb{E}}_{a\sim p_0, b\sim p_1, (1-t)a+tb=x} [b-a] \right\|^2 = \mathop{\mathbb{E}}_{x\sim p_t} \left\| v_\theta(x,t) - v(x,t) \right\|^2 .$$

One-step Sampling: Back to Push-forward

Continuous push-forward: velocity field

Suppose we have a velocity field $v(x_t, t)$, then we can define $f_t(x_t) = x_0$ by solving the ODE:

$$\dot{x}_t = v(x_t, t) \ .$$

We can learn v, can we further learn f_t?

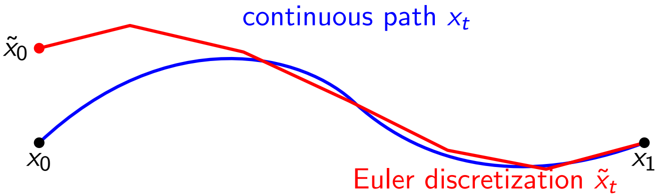
A Distillation Perspective

Recall $f_{t_1}(x_{t_1}) = f_{t_2}(x_{t_2}) = x_0$. We can train model $g_\theta(x, t)$ to distill f_t (Euler approximate).

Consistency model [ICML 2023]

We want the data on the Euler path pointing to the same x_0, *i.e.* $g_\theta(x_t, t) = g_\theta(x_t - vk, t - k)$:

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{n, x_{nk} \sim p_{nk}} \left\| g_\theta(x_{nk}, nk) - \mathrm{sg}\left(g_\theta(x_{nk} - v(x_{nk}, nk) \cdot k, (n-1)k)\right) \right\|^2 + \mathop{\mathbb{E}}_{x_0 \sim p_0} \left\| g_\theta(x_0, 0) - x_0 \right\|^2 .$$



Can we distill the continuous path?

Continuous consistency model [ICLR 2025]

We want the data on the continuous path $\{x_t\}$ pointing to the same x_0, *i.e.* $dg_\theta(x_t, t)/dt = 0$:

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{t, x_t \sim p_t} \|dg_\theta(x_t, t)/dt\|^2 + \mathop{\mathbb{E}}_{x_0 \sim p_0} \|g_\theta(x_0, 0) - x_0\|^2 ,$$

where

$$\frac{dg_\theta(x_t, t)}{dt} = \nabla_x g_\theta(x_t, t) \cdot \dot{x}_t + \partial_t g_\theta(x_t, t)/\partial t = v(x_t, t)\nabla_x g_\theta(x_t, t) + \partial_t g_\theta(x_t, t)/\partial t .$$

Quick Error Analysis

Consider a continuous path $\{x_t\}$.

And an Euler path $\{\tilde{x}_{nk}\}$, $\tilde{x}_{nk} = \tilde{x}_{(n+1)k} - v(\tilde{x}_{(n+1)k}, (n+1)k)k$, $\tilde{x}_1 = x_1$.

Quick Error Analysis

Consider a continuous path $\{x_t\}$.

And an Euler path $\{\tilde{x}_{nk}\}$, $\tilde{x}_{nk} = \tilde{x}_{(n+1)k} - v(\tilde{x}_{(n+1)k}, (n+1)k)k$, $\tilde{x}_1 = x_1$.

Discrete: Suppose $\|g_\theta(\tilde{x}_{nk}, nk) - g_\theta(\tilde{x}_{(n-1)k}, (n-1)k)\|_2 \leqslant \epsilon_1 k$,

Quick Error Analysis

Consider a continuous path $\{x_t\}$.
And an Euler path $\{\tilde{x}_{nk}\}$, $\tilde{x}_{nk} = \tilde{x}_{(n+1)k} - v(\tilde{x}_{(n+1)k}, (n+1)k)k$, $\tilde{x}_1 = x_1$.
Discrete: Suppose $\|g_\theta(\tilde{x}_{nk}, nk) - g_\theta(\tilde{x}_{(n-1)k}, (n-1)k)\|_2 \leqslant \epsilon_1 k$,then we have

$$\|g_\theta(x_1, 1) - x_0\|_2 \leqslant \|\sum_{n=1}^{N} g_\theta(\tilde{x}_{nk}, nk) - g_\theta(\tilde{x}_{(n-1)k}), (n-1)k)\|_2 + \|\tilde{x}_0 - x_0\|_2$$

$$\leqslant \epsilon_1 + \text{discretized error} .$$

Continuous: Suppose $\|g_\theta(x_0, 0) - x_0\|_2 \leqslant \epsilon_2$, and $\underbrace{|dg_\theta(x_{nk}, nk)/dt| \leqslant \epsilon_3}_{\text{finite training data}}$,

Quick Error Analysis

Consider a continuous path $\{x_t\}$.
And an Euler path $\{\tilde{x}_{nk}\}$, $\tilde{x}_{nk} = \tilde{x}_{(n+1)k} - v(\tilde{x}_{(n+1)k}, (n+1)k)k$, $\tilde{x}_1 = x_1$.
Discrete: Suppose $\|g_\theta(\tilde{x}_{nk}, nk) - g_\theta(\tilde{x}_{(n-1)k}, (n-1)k)\|_2 \leqslant \epsilon_1 k$,then we have

$$\|g_\theta(x_1, 1) - x_0\|_2 \leqslant \|\sum_{n=1}^{N} g_\theta(\tilde{x}_{nk}, nk) - g_\theta(\tilde{x}_{(n-1)k}), (n-1)k)\|_2 + \|\tilde{x}_0 - x_0\|_2$$
$$\leqslant \epsilon_1 + \text{discretized error} .$$

Continuous: Suppose $\|g_\theta(x_0, 0) - x_0\|_2 \leqslant \epsilon_2$, and $\underbrace{|dg_\theta(x_{nk}, nk)/dt| \leqslant \epsilon_3}_{\text{finite training data}}$,then we have

$$\|g_\theta(x_1, 1) - x_0\|_2 = \|g_\theta(x_1, 1) - f(x_1, 1)\|_2$$
$$= \|g_\theta(x_0, 0) - f(x_0, 0)\|_2 + \left\| \int [dg_\theta(x_t, t)/dt] dt \right\|_2$$
$$\leqslant \epsilon_2 + \epsilon_3 + \text{smoothness of the network } g_\theta .$$

Extended Version

We extend $g_\theta(x_t, t)$ to $g_\theta(x_t, t, r)$, which means transporting x_t to x_r.

Meanflow model

We still have $g_\theta(x_{t_1}, t_1, r) = g_\theta(x_{t_2}, t_2, r) = x_r$, for $t_1, t_2 > r$.

Extended Version

We extend $g_\theta(x_t, t)$ to $g_\theta(x_t, t, r)$, which means transporting x_t to x_r.

Meanflow model

We still have $g_\theta(x_{t_1}, t_1, r) = g_\theta(x_{t_2}, t_2, r) = x_r$, for $t_1, t_2 > r$.The general form is:

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{r<t, x_t \sim p_t} \left\| d g_\theta(x_t, t, r)/dt \right\|^2 + \mathop{\mathbb{E}}_{t, x_t \sim p_t} \left\| g_\theta(x_t, t, t) - x_t \right\|^2 ,$$

Extended Version

We extend $g_\theta(x_t, t)$ to $g_\theta(x_t, t, r)$, which means transporting x_t to x_r.

Meanflow model

We still have $g_\theta(x_{t_1}, t_1, r) = g_\theta(x_{t_2}, t_2, r) = x_r$, for $t_1, t_2 > r$. The general form is:

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{r<t, x_t \sim p_t} \left\| dg_\theta(x_t, t, r)/dt \right\|^2 + \mathop{\mathbb{E}}_{t, x_t \sim p_t} \left\| g_\theta(x_t, t, t) - x_t \right\|^2 ,$$

Recent works differ in parameterization:

- Meanflow [NeurIPS 2025]: $g_\theta(x_t, t, r) = x_t - (t - r) \underbrace{u_\theta(x_t, t, r)}_{\text{mean velocity}}$;

Extended Version

We extend $g_\theta(x_t, t)$ to $g_\theta(x_t, t, r)$, which means transporting x_t to x_r.

Meanflow model

We still have $g_\theta(x_{t_1}, t_1, r) = g_\theta(x_{t_2}, t_2, r) = x_r$, for $t_1, t_2 > r$.The general form is:

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{r<t, x_t \sim p_t} \|dg_\theta(x_t, t, r)/dt\|^2 + \mathop{\mathbb{E}}_{t, x_t \sim p_t} \|g_\theta(x_t, t, t) - x_t\|^2 ,$$

Recent works differ in parameterization:

- Meanflow [NeurIPS 2025]: $g_\theta(x_t, t, r) = x_t - (t - r) \underbrace{u_\theta(x_t, t, r)}_{\text{mean velocity}}$;
- Pixel Meanflow [2026]: $g_\theta(x_t, t, r) = \frac{(t-r)}{t} \underbrace{X_\theta(x_t, t, r)}_{\text{pixel prediction}} + \frac{r}{t} x_t$. (Manifold hypothesis)

Drifting Model [2026]

Directly learn the push-forward

Let's look into the desired push-forward function again:

$$p_0 = f_{\#} p_1 \ ,$$

Can we directly use a network f_θ to learn it?

Drifting Model [2026]

Directly learn the push-forward

Let's look into the desired push-forward function again:

$$p_0 = f_{\#} p_1 \ ,$$

Can we directly use a network f_θ to learn it?

Drifting: $p = p_0$ is a fixed point of $p' = T(p_0, p)_{\#} p$,

Drifting Model [2026]

Directly learn the push-forward

Let's look into the desired push-forward function again:

$$p_0 = f_{\#} p_1 \ ,$$

Can we directly use a network f_θ to learn it?

Drifting: $p = p_0$ is a fixed point of $p' = T(p_0, p)_{\#} p$,where we set

$$T(p,q)(x) = x + \mathop{\mathbb{E}}_{y^+ \sim p}\left[w_x^+(y^+) \cdot y^+\right] - \mathop{\mathbb{E}}_{y^- \sim q}\left[w_x^-(y^-) \cdot y^-\right] \ .$$

Drifting Model [2026]

Directly learn the push-forward

Let's look into the desired push-forward function again:

$$p_0 = f_{\#} p_1 \ ,$$

Can we directly use a network f_θ to learn it?

Drifting: $p = p_0$ is a fixed point of $p' = T(p_0, p)_{\#} p$,where we set

$$T(p, q)(x) = x + \mathop{\mathbb{E}}_{y^+ \sim p}\left[w_x^+(y^+) \cdot y^+\right] - \mathop{\mathbb{E}}_{y^- \sim q}\left[w_x^-(y^-) \cdot y^-\right] \ .$$

So $f_\theta = f$ is a fixed point of $f'_{\theta\#} p_1 = T(p_0, f_{\theta\#} p_1)_{\#}(f_{\theta\#} p_1)$.

Drifting Model [2026]

Directly learn the push-forward

Let's look into the desired push-forward function again:

$$p_0 = f_{\#} p_1 \ ,$$

Can we directly use a network f_θ to learn it?

Drifting: $p = p_0$ is a fixed point of $p' = T(p_0, p)_{\#} p$,where we set

$$T(p,q)(x) = x + \mathop{\mathbb{E}}_{y^+ \sim p}\left[w_x^+(y^+) \cdot y^+\right] - \mathop{\mathbb{E}}_{y^- \sim q}\left[w_x^-(y^-) \cdot y^-\right] \ .$$

So $f_\theta = f$ is a fixed point of $f'_{\theta\#} p_1 = T(p_0, f_{\theta\#} p_1)_{\#}(f_{\theta\#} p_1)$.

Training:

$$\mathcal{L}_\theta = \mathop{\mathbb{E}}_{x_1 \sim p_1}\left[\| f_\theta(x_1) - \text{sg}\left(T(p_0, f_{\theta\#}(p_1))(f_\theta(x_1))\right) \|^2\right] \ .$$