Toward Understanding the Generalization of Flow Matching

Quentin Bertrand

Joint work with A. Gagneux, S. Martin, M. Massias, and R. Emonet

Outline

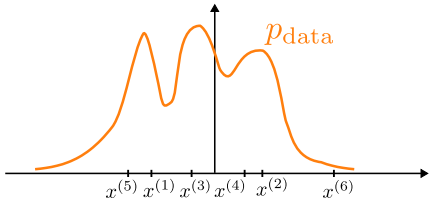
Short Intro to Generative Modelling & Neural ODEs

Flow Matching

Toward Generalization

Generative Modelling

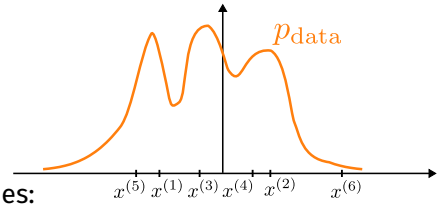
Given $x^{(1)}, \ldots, x^{(n)}$ sampled from p_{data}, learn to sample from p_{data}



Generative Modelling

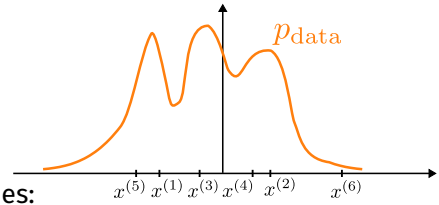
Given $x^{(1)}, \ldots, x^{(n)}$ sampled from p_{data}, learn to sample from p_{data}

Sampler, Desired Properties:



Generative Modelling

Given $x^{(1)}, \ldots, x^{(n)}$ sampled from p_{data}, learn to sample from p_{data}



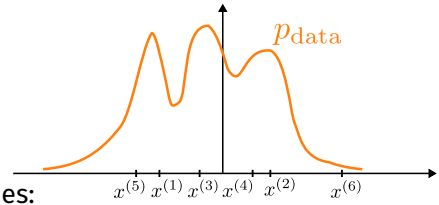
Sampler, Desired Properties:

- Easy to train

A Visual Dive into Conditional Flow Matching, Martin, Gagneux, Emonet, Bertrand & Massias, ICLR Blogpost 2025, `https://dl.heeere.com/cfm/`

Generative Modelling

Given $x^{(1)}, \ldots, x^{(n)}$ sampled from p_{data}, learn to sample from p_{data}



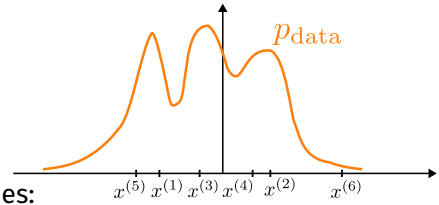
Sampler, Desired Properties:

- Easy to train
- Enforce fast sampling

A Visual Dive into Conditional Flow Matching, Martin, Gagneux, Emonet, Bertrand & Massias, ICLR Blogpost 2025, `https://dl.heeere.com/cfm/`

Generative Modelling

Given $x^{(1)}, \ldots, x^{(n)}$ sampled from p_{data}, learn to sample from p_{data}



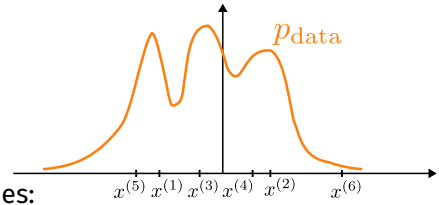
Sampler, Desired Properties:

- Easy to train
- Enforce fast sampling
- Generate high quality samples

A Visual Dive into Conditional Flow Matching, Martin, Gagneux, Emonet, Bertrand & Massias, ICLR Blogpost 2025, `https://dl.heeere.com/cfm/`

Generative Modelling

Given $x^{(1)}, \dots, x^{(n)}$ sampled from p_{data}, learn to sample from p_{data}



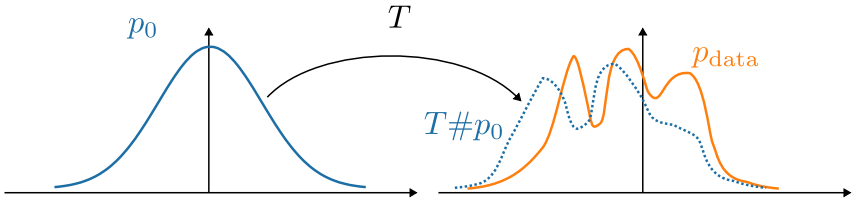
Sampler, Desired Properties:

- Easy to train
- Enforce fast sampling
- Generate high quality samples
- Properly cover the diversity of p_{data}

A Visual Dive into Conditional Flow Matching, Martin, Gagneux, Emonet, Bertrand & Massias, ICLR Blogpost 2025, `https://dl.heeere.com/cfm/`

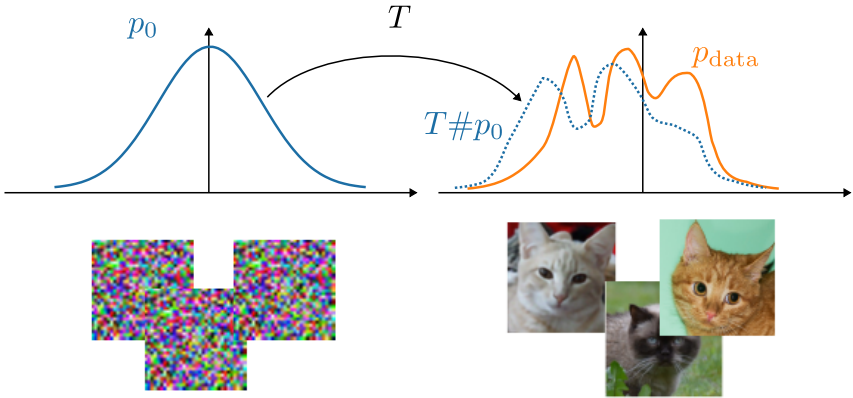
"Implicit" Generative Modelling

Map simple *base distribution*, p_0, to p_{data} through a map T



"Implicit" Generative Modelling

Map simple *base distribution*, p_0, to p_{data} through a map T



Technical wording *pushforward*: $T\#p_0$ is the distribution of $T(x)$ when $x \sim p_0$

How to Find a Good T?

Want: $T\#p_0$ close to p_{data}

How to Find a Good T?

Want: $T\#p_0$ close to p_{data}

- Learn T: T_θ

How to Find a Good T?

Want: $T\#p_0$ close to p_{data}

- Learn T: T_θ
- Idea: minimize some distance between $T_\theta\#p_0$ and p_{data}

$$\theta^* = \operatorname*{argmin}_\theta \text{Dist}(T_\theta\#p_0, p_{\text{data}})$$

How to Find a Good T?

Want: $T\#p_0$ close to p_{data}

- Learn T: T_θ
- Idea: minimize some distance between $T_\theta\#p_0$ and p_{data}

$$\theta^* = \operatorname*{argmin}_\theta \text{Dist}(T_\theta\#p_0, p_{\text{data}})$$

- "Equivalent" to maximum log-likelihood:

$$\theta^* = \text{argmax}_\theta \sum_{i=1}^n \log\left(\underbrace{(T_\theta\#p_0)}_{:=p_1}(x^{(i)})\right)$$

How to Find a Good T?

Want: $T\#p_0$ close to p_{data}

- Learn T: T_θ
- Idea: minimize some distance between $T_\theta\#p_0$ and p_{data}

$$\theta^* = \operatorname*{argmin}_\theta \text{Dist}(T_\theta\#p_0, p_{\text{data}})$$

- "Equivalent" to maximum log-likelihood:

$$\theta^* = \text{argmax}_\theta \sum_{i=1}^n \log\left(\underbrace{(T_\theta\#p_0)}_{:=p_1}(x^{(i)})\right)$$

- Question: how to compute $\log(T_\theta\#p_0(x^{(i)}))$? and $\nabla_\theta \log(T_\theta\#p_0(x^{(i)}))$?

The Change of Variable Formula

$$\boxed{\log T_\theta \# p_0(x) = \log p_0(T_\theta^{-1}(x)) + \log |\det J_{T_\theta^{-1}}(x)|}$$

The Change of Variable Formula

$$\boxed{\log T_\theta \# p_0(x) = \log p_0(T_\theta^{-1}(x)) + \log |\det J_{T_\theta^{-1}}(x)|}$$

Practical Downsides:

- T_θ must be invertible

The Change of Variable Formula

$$\boxed{\log T_\theta \# p_0(x) = \log p_0(T_\theta^{-1}(x)) + \log |\det J_{T_\theta^{-1}}(x)|}$$

Practical Downsides:

- T_θ must be invertible
- T_θ^{-1} should be easy to compute in order to evaluate the first right-hand side term

The Change of Variable Formula

$$\boxed{\log T_\theta \# p_0(x) = \log p_0(T_\theta^{-1}(x)) + \log |\det J_{T_\theta^{-1}}(x)|}$$

Practical Downsides:

- T_θ must be invertible
- T_θ^{-1} should be easy to compute in order to evaluate the first right-hand side term
- T_θ^{-1} must be differentiable

The Change of Variable Formula

$$\boxed{\log T_\theta \# p_0(x) = \log p_0(T_\theta^{-1}(x)) + \log |\det J_{T_\theta^{-1}}(x)|}$$

Practical Downsides:

- T_θ must be invertible
- T_θ^{-1} should be easy to compute in order to evaluate the first right-hand side term
- T_θ^{-1} must be differentiable
- the (log) determinant of the Jacobian of T_θ^{-1} must not be too costly to compute

The Change of Variable Formula

$$\log T_\theta \# p_0(x) = \log p_0(T_\theta^{-1}(x)) + \log |\det J_{T_\theta^{-1}}(x)|$$

Practical Downsides:

- T_θ must be invertible
- T_θ^{-1} should be easy to compute in order to evaluate the first right-hand side term
- T_θ^{-1} must be differentiable
- the (log) determinant of the Jacobian of T_θ^{-1} must not be too costly to compute

Normalizing Flows = neural architectures satisfying these requirements

How to Ensure That T is Invertible?

Idea:

- Choose T as the solution of an Ordinary Differential Equation
- Learn the velocity field

$$\begin{cases} x(0) = x_0 \sim p_0 \\ \partial_t x(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{cases}$$

The ODEmapping $T : x_0 \mapsto x(1)$
is invertible under mild assumptions

Outline

Short Intro to Generative Modelling & Neural ODEs

Flow Matching

Toward Generalization

Recap

We have:

- Source distribution $p_0 = \mathcal{N}(0, \mathrm{Id})$
- Target distribution p_{data} (*e.g.*, images)

Recap

We have:

- Source distribution $p_0 = \mathcal{N}(0, \mathrm{Id})$
- Target distribution p_{data} (*e.g.*, images)

Goal:

- Generate new samples from p_{data}

Recap

We have:

- Source distribution $p_0 = \mathcal{N}(0, \mathrm{Id})$
- Target distribution p_{data} (*e.g.*, images)

Goal:

- Generate new samples from p_{data}

How?

- Solving

$$\begin{cases} x(0) = x_0 \sim p_0 \\ \dot{x}(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{cases}$$

- Such that solution $x(1) \sim p_{\mathrm{data}}$

Recap

We have:

- Source distribution $p_0 = \mathcal{N}(0, \mathrm{Id})$
- Target distribution p_{data} (*e.g.*, images)

Goal:

- Generate new samples from p_{data}

How?

- Solving

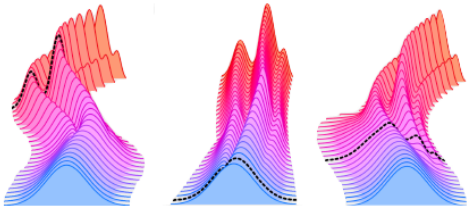
$$\begin{cases} x(0) = x_0 \sim p_0 \\ \dot{x}(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{cases}$$

- Such that solution $x(1) \sim p_{\mathrm{data}}$

How to learn a "good" velocity field u?

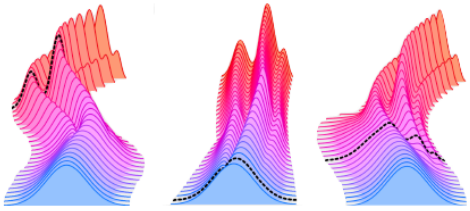
In Search for a Good u, $1/3$

$$\begin{cases} x(0) = x_0 \\ \dot{x}(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{cases}$$



In Search for a Good u, $1/3$

$$\begin{cases} x(0) = x_0 \\ \dot{x}(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{cases}$$

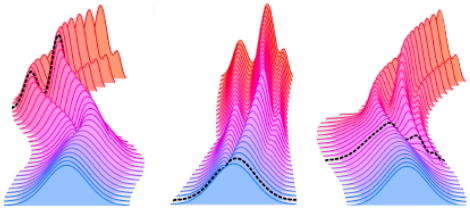


- ODE defines *probability path* $(p_t)_{t\in[0,1]}$ = laws of the solution $x(t)$ when $x(0) \sim p_0$
- Requirements on p_t

  $\hookrightarrow$ $p_0 = p_0$ and $p_1 = p_{\text{data}}$

In Search for a Good u, $1/3$

$$\begin{cases} x(0) = x_0 \\ \dot{x}(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{cases}$$



- ODE defines *probability path* $(p_t)_{t\in[0,1]}$ = laws of the solution $x(t)$ when $x(0) \sim p_0$
- Requirements on p_t

  $\hookrightarrow$ $p_0 = p_0$ and $p_1 = p_{\text{data}}$

u must drive a progressive transformation of p_0 into p_{data}

In Search for a Good u, $2/3$

$$
\begin{aligned}
&x(0) = x_0 \\
&\partial_t x(t) = u(x(t), t) \quad \forall t \in [0, 1]
\end{aligned}
$$

In Search for a Good u, $2/3$

$$\boxed{\begin{aligned} &x(0) = x_0 \\ &\partial_t x(t) = u(x(t), t) \quad \forall t \in [0, 1] \end{aligned}}$$

Key objects:

- the **velocity field** $u : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$
- the **probability path** $(p_t)_{t \in [0,1]}$ = the distributions of $x_0 + \int_0^t u(x(s), s) ds$ when $x \sim p_0$

$$p_t = \left(x_0 + \int_0^t u(x(s), s) ds \right)_{x_0 \sim p_0} \tag{1}$$

In Search for a Good u, $2/3$

$$\boxed{\begin{aligned} &x(0) = x_0 \\ &\partial_t x(t) = u(x(t), t) \quad \forall t \in [0,1] \end{aligned}}$$

Key objects:

- the **velocity field** $u : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$
- the **probability path** $(p_t)_{t\in[0,1]}$ = the distributions of $x_0 + \int_0^t u(x(s), s)ds$ when $x \sim p_0$

$$p_t = \left(x_0 + \int_0^t u(x(s), s)ds \right)_{x_0 \sim p_0} \tag{1}$$

Linked through the continuity equation

$$\partial_t p_t + \operatorname{div}(u_t p_t) = 0$$

In Search for a Good u, $2/3$

$$\boxed{\begin{aligned} &x(0) = x_0 \\ &\partial_t x(t) = u(x(t), t) \quad \forall t \in [0,1] \end{aligned}}$$

Key objects:

- the **velocity field** $u : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$
- the **probability path** $(p_t)_{t\in[0,1]}$ = the distributions of $x_0 + \int_0^t u(x(s), s)ds$ when $x \sim p_0$

$$p_t = \left(x_0 + \int_0^t u(x(s), s)ds \right)_{x_0 \sim p_0} \tag{1}$$

Linked through the continuity equation

$$\partial_t p_t + \operatorname{div}(u_t p_t) = 0$$

Finding a good velocity u " $\equiv$ " Finding a good proba. path p_t

In Search for a Good u, $3/3$

- Can you find a good velocity field u?

In Search for a Good u, $3/3$

- Can you find a good velocity field u?

  $\hookrightarrow$ Too hard

In Search for a Good u, $3/3$

- Can you find a good velocity field u?

  $\hookrightarrow$ Too hard

- Can you find a good probability path p_t?

In Search for a Good u, $3/3$

- Can you find a good velocity field u?
  - $\hookrightarrow$ Too hard
- Can you find a good probability path p_t?
  - $\hookrightarrow$ Also too hard (possible, but tricky)

In Search for a Good u, $3/3$

- Can you find a good velocity field u?

  $\hookrightarrow$ Too hard

- Can you find a good probability path p_t?

  $\hookrightarrow$ Also too hard (possible, but tricky)

- **Idea**: can you find a good *conditional* probability path?

In Search for a Good u, $3/3$

- Can you find a good velocity field u?

  $\hookrightarrow$ Too hard

- Can you find a good probability path p_t?

  $\hookrightarrow$ Also too hard (possible, but tricky)

- **Idea**: can you find a good *conditional* probability path?

  $\hookrightarrow$ Yes!

In Search for a Good u, $3/3$

- Can you find a good velocity field u?

  $\hookrightarrow$ Too hard

- Can you find a good probability path p_t?

  $\hookrightarrow$ Also too hard (possible, but tricky)

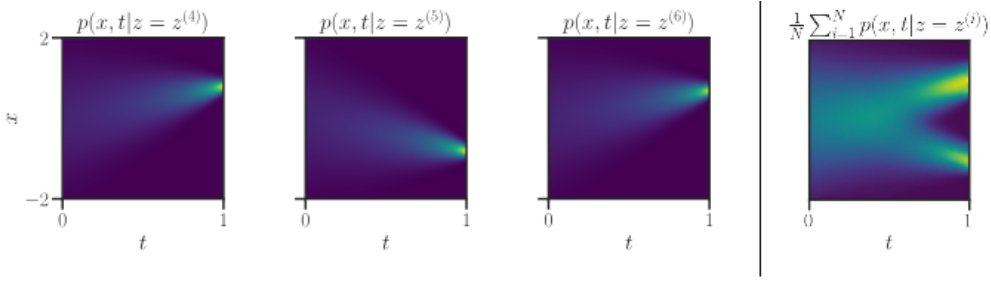
- **Idea**: can you find a good *conditional* probability path?

  $\hookrightarrow$ Yes!

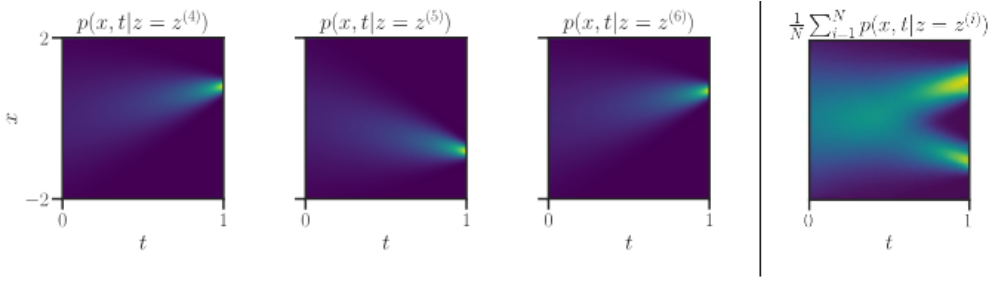
e.g.,

$$p(x|x_1, t) = \mathcal{N}(tx_1, (1 - t)^2 \mathrm{Id})(x)$$

Link Between $p(\cdot|z = x_1, t)$ and $p(\cdot|t)$



Link Between $p(\cdot|z = x_1, t)$ and $p(\cdot|t)$



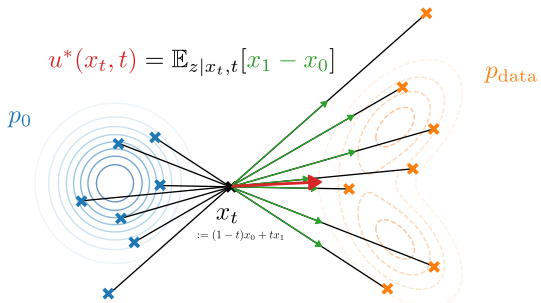
One can check that :

- $p(\cdot|t = 0) = p_0$
- $p(\cdot|t = 1) = p_{\text{data}}$

Link between u^{cond} and u

Notation:

- *Conditioning* variable $z = x_1 \sim p_{\text{data}}$
- *Conditional* probability path $p(\cdot|z = x_1, t) = \mathcal{N}(tx_1, (1-t)^2 \text{Id})$
- Associated *conditional* velocity: $u^{\text{cond}}(x, z = x_1, t) = \frac{x_1 - x}{1-t} = x_1 - x_0$



The Flow Matching Loss

We have our target, valid velocity:

$$u^\star(x,t) = \mathbb{E}_{z|x,t}[u^{\text{cond}}(x,z,t)]$$

The Flow Matching Loss

We have our target, valid velocity:

$$u^\star(x,t) = \mathbb{E}_{z|x,t}[u^{\text{cond}}(x,z,t)]$$

We just need to approximate it with a neural net $u_\theta : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$:

$$\min_\theta \left\{ \mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{\substack{t\sim\mathcal{U}([0,1]) \\ x_t\sim p(\cdot|t)}} \|u_\theta(x_t,t) - u^\star(x_t,t)\|^2 \right\}$$

The Flow Matching Loss

We have our target, valid velocity:

$$u^\star(x,t) = \mathbb{E}_{z|x,t}[u^{\text{cond}}(x,z,t)]$$

We just need to approximate it with a neural net $u_\theta : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$:

$$\min_\theta \left\{ \mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{\substack{t\sim\mathcal{U}([0,1]) \\ x_t\sim p(\cdot|t)}} \|u_\theta(x_t,t) - u^\star(x_t,t)\|^2 \right\}$$

We are almost there

The Conditional Flow Matching Loss

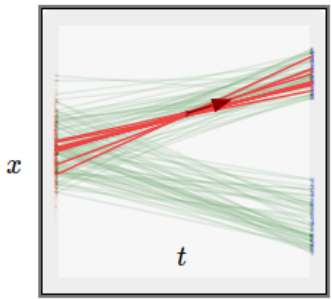
Ideal loss:

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{\substack{t\sim\mathcal{U}([0,1]) \\ x_t\sim p(\cdot|t)}} \|u_\theta(x_t,t) - u^\star(x_t,t)\|^2$$

Theorem 2: (Lipman, Liu, Albergo 2023) Up to a constant, $\mathcal{L}_{\mathrm{FM}}$ is equal to

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{\substack{x_0\sim p_0 \\ x_1\sim p_{\mathrm{data}} \\ t\sim\mathcal{U}([0,1])}} \|u_\theta(x_t,t) - \underbrace{u^{\mathrm{cond}}(x_t, z=x_1, t)}_{=x_1-x_0}\|^2$$

where $x_t := (1-t)x_0 + tx_1$



Minimizing $\mathcal{L}_{\text{CFM}}$

To minimize

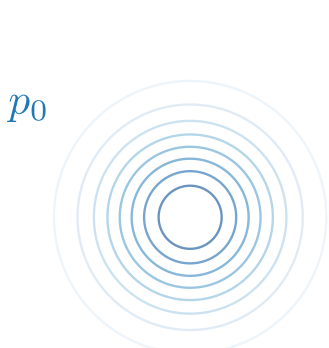
$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \|u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t)\|^2$$

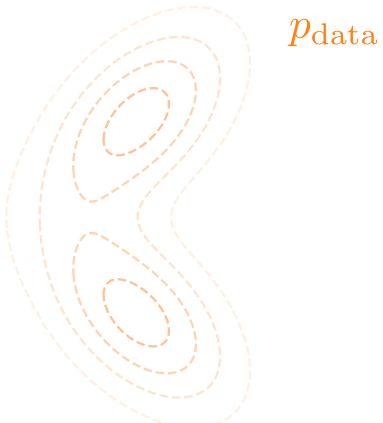
$$(x_t := (1-t)x_0 + tx_1)$$

- sample $x_0 \sim p_0$: easy!
- sample $t \sim \mathcal{U}([0,1])$! easy!
- sample $x_1 \sim p_{\text{data}}$? easy if we replace by $x_1 \sim \hat{p}_{\text{data}} := \frac{1}{n} \sum_{i=1}^{n} \delta_{x^{(i)}}$

Training Flow Matching

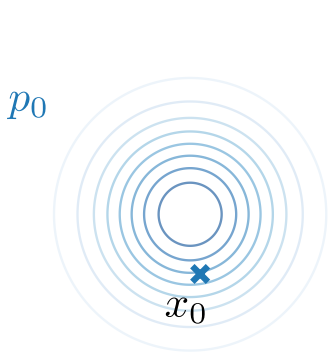
$$\min_{\theta} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \left[\| u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t) \|^2 \right] \qquad (x_t := (1-t)x_0 + t x_1)$$

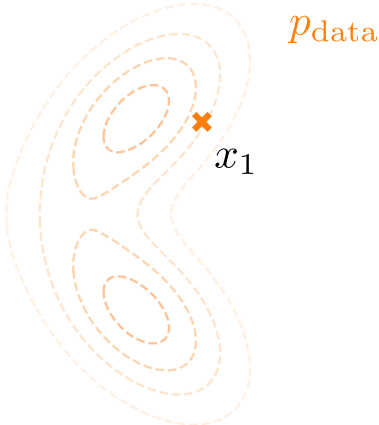




Training Flow Matching

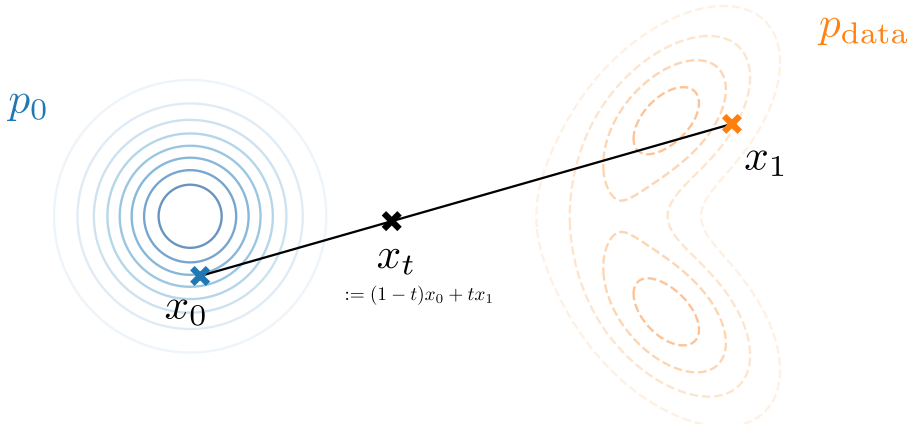
$$\min_{\theta} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \left[\| u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t) \|^2 \right] \qquad (x_t := (1-t)x_0 + t x_1)$$





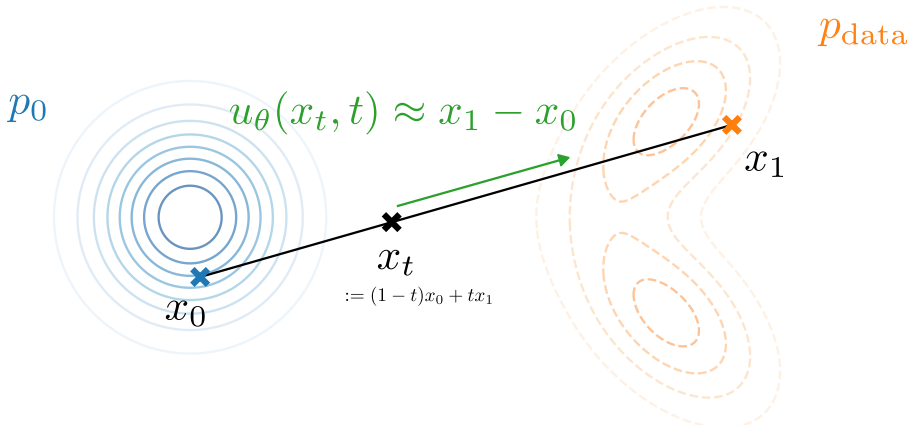
Training Flow Matching

$$\min_{\theta} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \left[\| u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t) \|^2 \right] \qquad (x_t := (1-t)x_0 + t x_1)$$



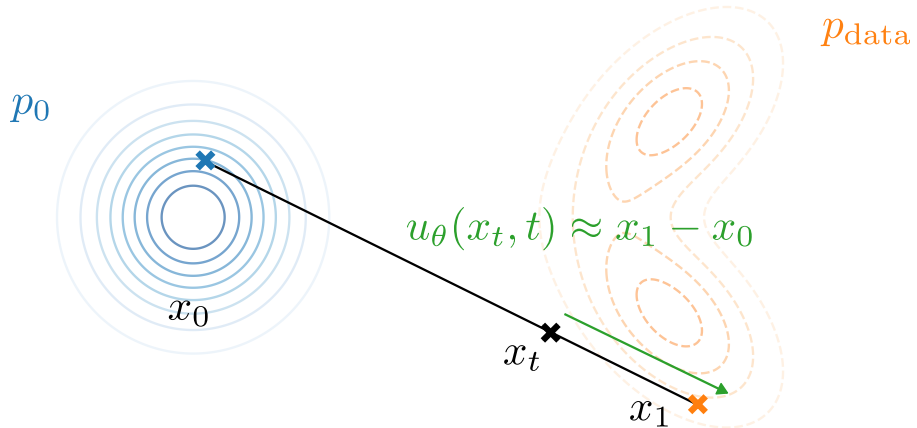
Training Flow Matching

$$\min_{\theta} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \left[\| u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t) \|^2 \right] \qquad (x_t := (1-t)x_0 + t x_1)$$



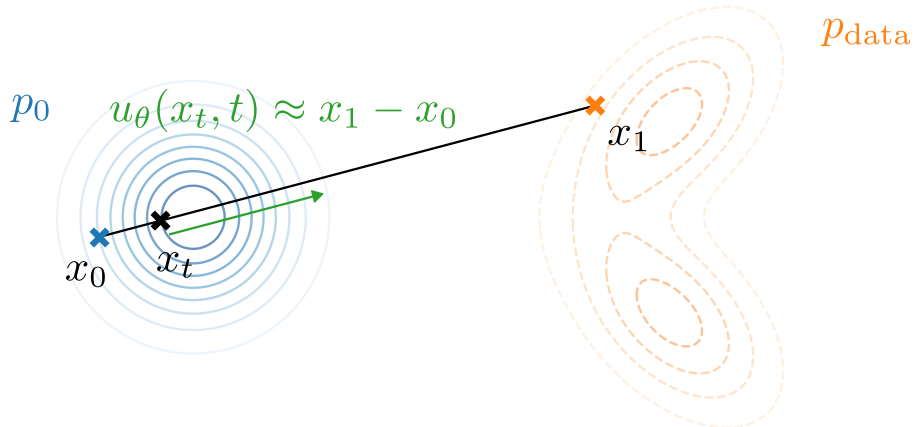
Training Flow Matching

$$\min_{\theta} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \left[\| u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t) \|^2 \right] \qquad (x_t := (1-t)x_0 + t x_1)$$



Training Flow Matching

$$\min_{\theta} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \left[\| u_\theta(x_t, t) - u^{\text{cond}}(x_t, z = x_1, t) \|^2 \right] \qquad (x_t := (1-t)x_0 + t x_1)$$



Outline

Short Intro to Generative Modelling & Neural ODEs

Flow Matching

Toward Generalization

Closed-Form of the Optimal Velocity

$$u^\star(x,t) = \mathbb{E}_{z|x,t}\, u^{\mathrm{cond}}(x,z,t)$$

Prop: If p_{data} is replaced by $\hat{p}_{\mathrm{data}} := \frac{1}{n}\sum_{i=1}^n \delta_{x^{(i)}}$, the optimal velocity has a closed-form:

$$\hat{u}^\star(x,t) = \sum_{i=1}^n \lambda_i(x,t)\frac{x^{(i)} - x}{1-t}$$

with $\lambda(x,t) = \mathrm{softmax}((-\frac{1}{2(1-t)^2}\|x - tx^{(i')}\|^2)_{i'=1,\dots,n}) \in \mathbb{R}^n$

$\hat{u}^\star$ is now a finite sum!

What can we observe for $\hat{u}^\star$ as $t \to 1$?

Flow Matching Generalization?

Minimizing

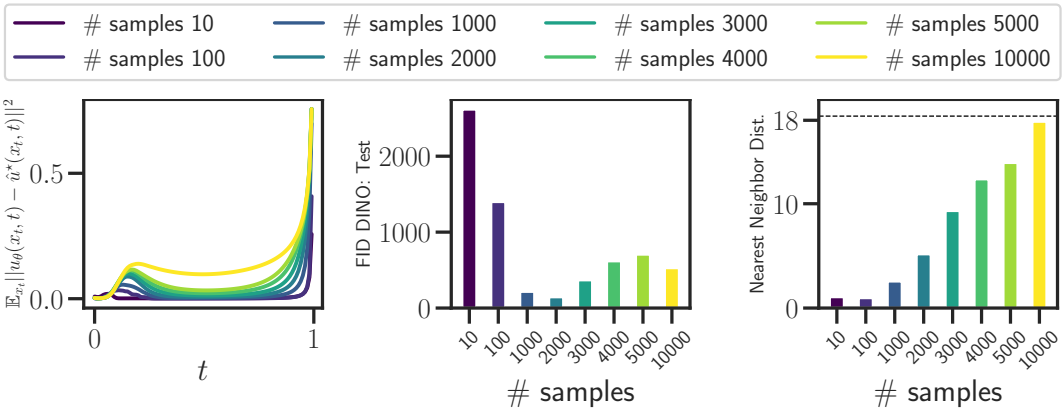
$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim \hat{p}_{\mathrm{data}} \\ t \sim \mathcal{U}([0,1])}} \left\| u_\theta(x_t, t) - u^{\mathrm{cond}}(x_t, z = x_1, t) \right\|^2$$

$$(x_t := (1-t)x_0 + t x_1)$$

- the minimizer of $\mathcal{L}_{\mathrm{CFM}}$ is available in closed-form
- this closed-form $\hat{u}^\star(x,t)$ blows up for $t \to 1$ if $x \notin \{x^{(1)}, \ldots, x^{(n)}\}$
- it can only generate training points!

So why does flow matching generalize?

There is No Benign Overfitting



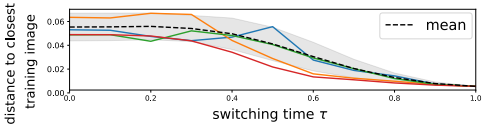
- Generalization when failure to approximate the ¨optimal¨ velocity
- u_θ fails to learn $\hat{u}^\star$ for both $t \approx 0.2$ and $t \approx 0.9$

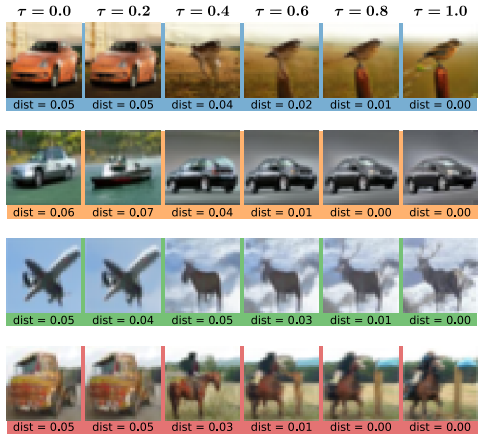
Which t Matters Most?

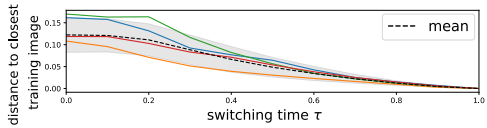
From a good trained u_θ, we build a *hybrid* model (fixed $\tau \in [0, 1]$):

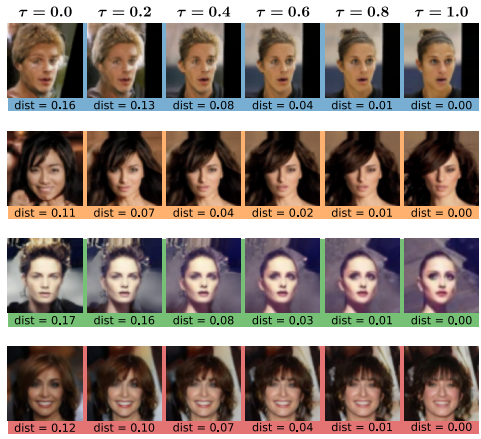
- on $[0, \tau]$: follow $\hat{u}^\star$
- on $[\tau, 1]$: follow u_θ

- $\tau = 1$ means completely following $\hat{u}^\star$ (no generalization)
- $\tau = 0$ means completely following u_θ (good generalization)







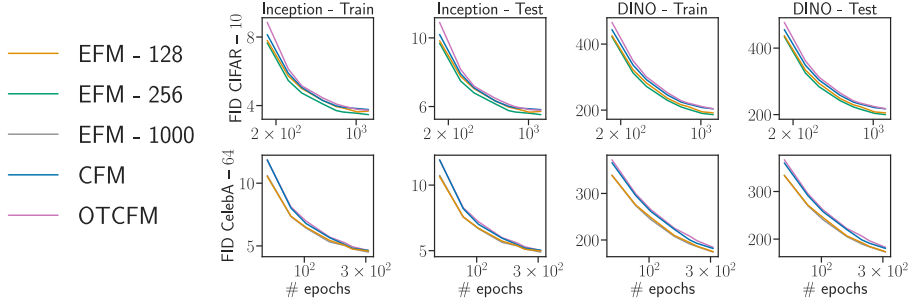


generalization arises early!

Refuting the Stochasticity Argument: Regressing Against $\hat{u}^\star$

From $$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim \hat{p}_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \|u_\theta(x_t, t) - (x_1 - x_0)\|^2$$

to $$\mathcal{L}_{\text{EFM}}(\theta) = \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim \hat{p}_{\text{data}} \\ t \sim \mathcal{U}([0,1])}} \|u_\theta(x_t, t) - \hat{u}^\star(x_t, t)\|^2$$



Learning with a non-stochastic target *does not* degrade performance

Summary

- by design, the true velocity in flow matching is available in closed-form
- flow matching should not create new images, yet it does
- stochasticity is definitely not the reason for it
- small and large times appear to matter most
- failure of u_θ to learn $\hat{u}^\star$ for small t is critical

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity, Bertrand, Gagneux, Massias & Emonet, NeurIPS 2025

Detour: How to Measure Generalization

Fréchet Inception Distance (FID) to compare generated images to true (train or test) images:

- compute embeddings for both groups (Inception network)
- approximate each distrib of embedding by Gaussians
- use closed-form OT formula for Gaussians $\|\mu_1 - \mu_2\|^2 + \mathrm{tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{1/2})$

Detour: How to Measure Generalization

Fréchet Inception Distance (FID) to compare generated images to true (train or test) images:

- compute embeddings for both groups (Inception network)
- approximate each distrib of embedding by Gaussians
- use closed-form OT formula for Gaussians $\|\mu_1 - \mu_2\|^2 + \mathrm{tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{1/2})$
- it's a wonder that people use it:
  - it has (hidden) dependence on number of samples used
  - empirically, a model that generates only train images has SOTA *test* FID
- as complement, we use min distance of generated image to training data