Flow Through Generative Modeling: A Tutorial

Qiang Liu

UT Austin

July 14 2025

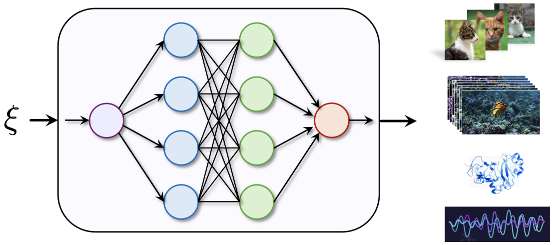
With help from: Runlong Liao, Xixi Hu, Bo Liu, Baiyu Su, Yuanzhi Zhu, Lizhang Chen

Generative Models: Noise to Data

Input: Data $\mathcal{D} = \{X_i\}_{i=1}^n$ from an unknown distribution P^*.

Goal: Learn a generative model that can sample from P^* via

$$X = T^\theta(Z), \quad Z \sim P_{\texttt{noise}}.$$



One-Step vs. Process Models

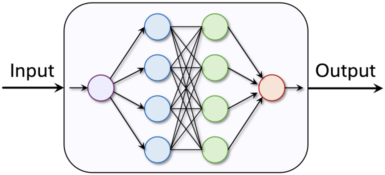
$$\texttt{Data} = T^{\theta}(\texttt{Noise}).$$

One-Step Models

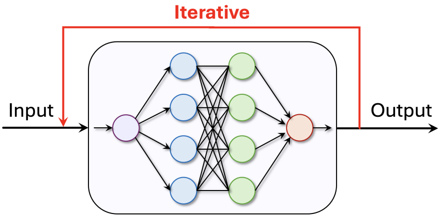
- Learn T^{θ} as a black box.
  - GANs
  - Autoencoders
  - Invertible models

Iterative Process Models

- Learn T^{θ} as an iterative process.
  - Diffusion models: SDE
  - Flow models: ODE
  - GPT: Auto-regressive



Learn a function



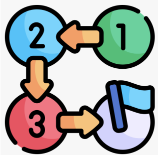
Learn an algorithm

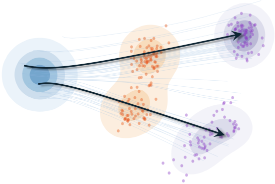
All Successful Models Today Are *Process Models*

Divide and Conquer

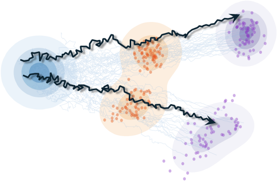
Break complex generation into simpler steps.

- Improves expressivity
- Simplifies training

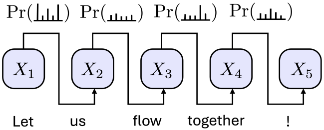




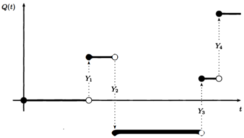
ODE (Flow)



SDE (Diffusion)



GPT (Autoregressive)



Jump

Process Models: Decomposition + Imitation

- Process Decomposition: Break complex data generation into simpler steps along a latent trajectory:

$$\text{Data}\to\text{Latent} \qquad Q^*(X^{\texttt{data}})\, Q^{\text{aug}}(X^{\texttt{latent}} \mid X^{\texttt{data}})$$

- Process Imitation: Learn a generative model that mimics the stepwise process:

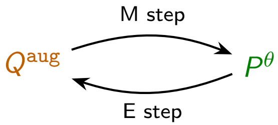
$$\text{Latent}\to\text{Data} \qquad P^\theta(X_0, \ldots, X_T) = \prod_i P(X_i \mid X_{<i})$$

such that the marginal distributions are matched:

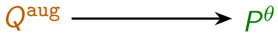
$$P^\theta(X^{\texttt{data}}) = Q^*(X^{\texttt{data}}).$$

How is this different from classical latent variable models and VAE?

Classical (full EM): Q^{aug} and P^{θ} are updated iteratively to fit each other.



New (lazy EM): Q^{aug} is fixed (pre-defined); only P^{θ} is updated to fit Q^{aug}.



- Why is this okay and preferred?
  - Large neural nets P^{θ} are universal approximators; can fit any given Q^{aug}.
  - MLE solutions are not unique anyway.
  - Computationally easier to use fixed Q^{aug}.
  - Can inject priors to encourage simplicity and efficiency.

How is this different from classical latent variable models and VAE?

- Classical (exact matching): Try to fit P^{θ} exactly with Q^{aug} on joint distribution:

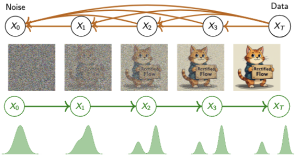
$$Q^{\texttt{aug}}(X^{\texttt{latent}}, X^{\texttt{data}}) = P^{\theta}(X^{\texttt{latent}}, X^{\texttt{data}}).$$

- New (marginal matching): P^{θ} only match Q^{aug} on marginals acoss steps:

$$P^{\theta}(X_t) = Q^{\texttt{aug}}(X_t), \qquad \forall t.$$

  - In fact, we will see that P^{θ} "simplifies and improves" Q^{aug} while preserving marginals.

$$Q^{\texttt{aug}} \xrightarrow[\textbf{Causalizes}\ \textbf{Derandomizes}]{\textbf{Rectifies}\ \textbf{Markovizes}} P^{\theta}$$



Challenges

- **Slow inference**
- **Conceptual understanding**
- **Optimal algorithm design**

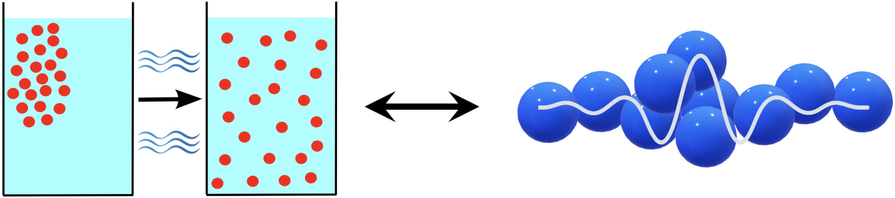
Key question: Can we combine the best of both worlds?

Model	Training	Inference	Performance
One-Step	**Hard**	**Fast**	**Limited**
Process	**Easy**	**Slow**	**Strong**

Chaos and Beauty: Intriguing math + Powerful Applications

Denoising diffusion probabilistic models (DDPM), Denoising diffusion implicit models (DDIM), Annealed Langevin dynamics, Nonequilibrium Thermodynamics, Score-based Generative Models, Energy Models, Score matching, Time-reversed diffusion processes, Probability flow ODEs, Schrödinger Bridge, Brownian bridges, Diffusion Bridges, Doob's h-transform, Föllmer Process, EDM, Rectified Flow, Stochastic Interpolantss Flow Matching, Reflow, Bridge Matching, Markovization, Gyöngy projection, Hierarchical VAE, Optimal Transport, Straight Transport, Consistent Models, Score Distillation, Distribution Matching Distillation, Discrete Diffusion, Discrete Flow, Optimal Control

Diffusion Models in ML Are Like Quantum Mechanics in Physics



Flow / Diffusion

Particle-wave / Nelson / Bohmian Mechanics

This Tutorial: Starts From Rectified Flow

$$\texttt{Generation} = \texttt{Rectify}(\texttt{Interpolation})$$

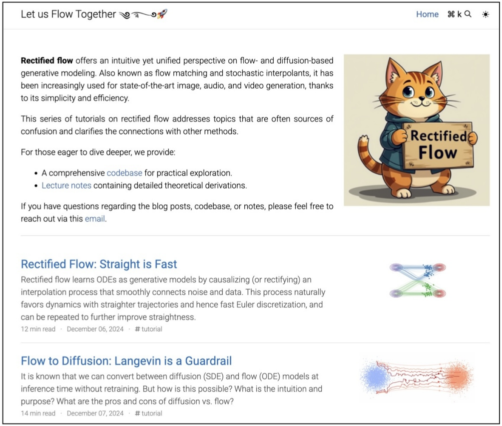
$$P^{\theta} = \texttt{Rectify}(Q^{\text{aug}}).$$

Theme: Understanding and using $\texttt{Rectify}()$ operator.

- Topics:
  - The Rewiring demon: Rectified Flow
  - Bless of Continuity: Marginal Preservation
  - Bless of Straightness: Transport Cost
  - Bless of Gaussian: Score and KL
  - Bless of Noise: Diffusion
  - Bless of Consistency: Distillation
  - Bless of Reward: Tilting
  - Bless of Singularity: Constrained and Discrete

More Information in blog and notes [Liu24]

Email: qiang.liu.research@gmail.com



Let us Flow Together

Home ⌘ k

Rectified flow offers an intuitive yet unified perspective on flow- and diffusion-based generative modeling. Also known as flow matching and stochastic interpolants, it has been increasingly used for state-of-the-art image, audio, and video generation, thanks to its simplicity and efficiency.

This series of tutorials on rectified flow addresses topics that are often sources of confusion and clarifies the connections with other methods.

For those eager to dive deeper, we provide:

- A comprehensive codebase for practical exploration.
- Lecture notes containing detailed theoretical derivations.

If you have questions regarding the blog posts, codebase, or notes, please feel free to reach out via this email.

Rectified Flow: Straight is Fast

Rectified flow learns ODEs as generative models by causalizing (or rectifying) an interpolation process that smoothly connects noise and data. This process naturally favors dynamics with straighter trajectories and hence fast Euler discretization, and can be repeated to further improve straightness.

12 min read · December 06, 2024 · # tutorial

Flow to Diffusion: Langevin is a Guardrail

It is known that we can convert between diffusion (SDE) and flow (ODE) models at inference time without retraining. But how is this possible? What is the intuition and purpose? What are the pros and cons of diffusion vs. flow?

14 min read · December 07, 2024 · # tutorial

Blogs

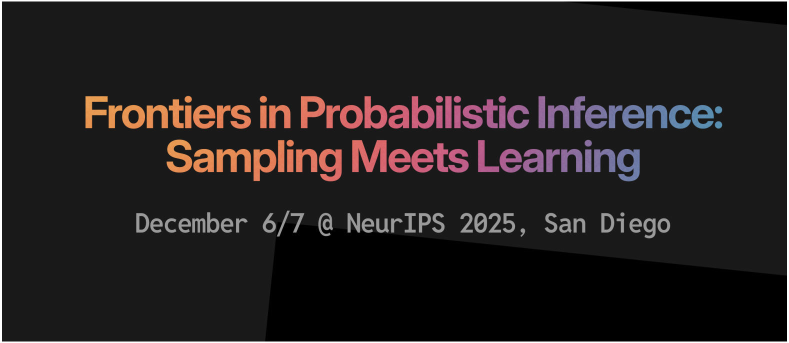
Lecture Notes

To be updated...

Funding supports from NSF, ONR, IFML, Google, Meta.

Frontiers in Probabilistic Inference: Learning Meets Sampling (FPI 2025)

- FPI Neurips 2025 Workshop
- **Call for Papers + Open Questions**
- https://fpineurips.framer.website/

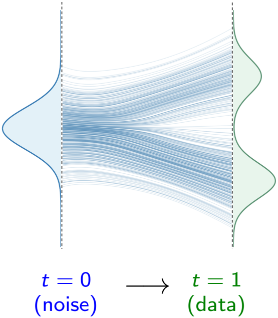


Problem: Flow Transport

- **Given:** Data from source P_0 and target P_1.
- **Goal:** Learn an ODE velocity field $v(z,t)$:

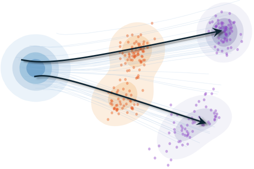
$$\frac{d}{dt}Z_t = v(Z_t, t), \quad Z_0 \sim P_0, \quad t \in [0,1].$$

Transport $Z_0 \sim P_0$ (noise) to $Z_1 \sim P_1$ (data).



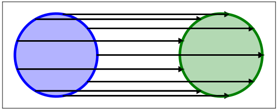
Assume $Z_0 \sim P_0$, $Z_1 \sim P_1$:

- **The Transport Process** is the stochastic process $\{Z_t : t \in [0,1]\}$ connecting Z_0 and Z_1.
- **The Transport Plan (Coupling)** is the joint distribution of the start-end pair (Z_0, Z_1).
- **The Transport Map** is a mapping $Z_1 = T(Z_0)$ that pushes P_0 to P_1.

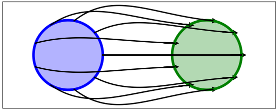


Transport Maps are not Unique

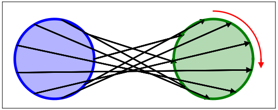
- There can be **infinite many possible maps** between P_0 and P_1.
  - The flow can go **different trajectories**.
  - The flow can yield **different couplings**.



Optimal transport



Different trajectories



Different couplings

Optimal Transport Problem

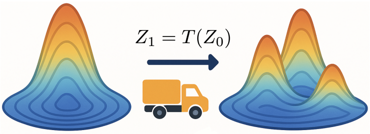
- Optimal transport: special transports minimizing transport costs induced by a convex function $c(\cdot)$.

c-Optimal Transport (Static)

$$\min_{(Z_0,Z_1)} \mathbb{E}\left[c(Z_1 - Z_0)\right]$$
$$\text{s.t. } Z_0 \sim P_0,\ Z_1 \sim P_1$$

c-Optimal Transport (Dynamic)

$$\min_{\{Z_t\}} \mathbb{E}\left[\int_0^1 c(\dot{Z}_t)\, dt\right]$$
$$\text{s.t. } Z_0 \sim P_0,\ Z_1 \sim P_1$$



Optimal Transport Problem

- Optimal transport: special transports minimizing transport costs induced by a convex function $c(\cdot)$.

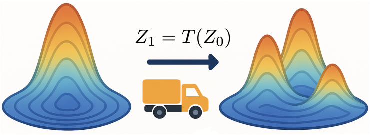
c-Optimal Transport (Static)

$$\min_{(Z_0,Z_1)} \mathbb{E}\left[c(Z_1 - Z_0)\right]$$
$$\text{s.t. } Z_0 \sim P_0,\ Z_1 \sim P_1$$

c-Optimal Transport (Dynamic)

$$\min_{\{Z_t\}} \mathbb{E}\left[\int_0^1 c(\dot{Z}_t)\, dt\right]$$
$$\text{s.t. } Z_0 \sim P_0,\ Z_1 \sim P_1$$

- However, solving OT is
  - **computationally challenging**.
  - **unnecessary** for the purpose of generative modeling.

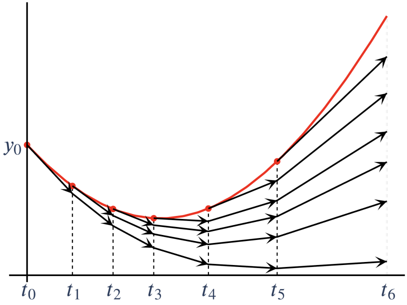


Approximation Error

- In practice, the ODE is solved with numerical methods, such as Euler method:

$$\hat{Z}_{t+\epsilon} = \hat{Z}_t + \epsilon v^\theta(\hat{Z}_t, t), \qquad \text{for} \quad t \in \{0, \epsilon, 2\epsilon, ..., 1\},$$

where $\epsilon = 1/N$ is step size.

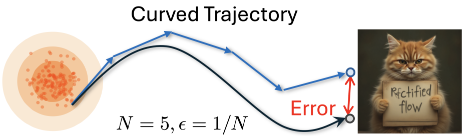


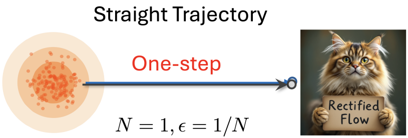
Straightness = Fast

- Euler discretization error depends on the **trajectory curvature**:

$$\left\| \hat{Z}_t - Z_t \right\| = O(\epsilon M), \qquad M = \sup_t \left\| \ddot{Z}_t \right\|.$$

- **Perfectly straight trajectories = one-step generation**



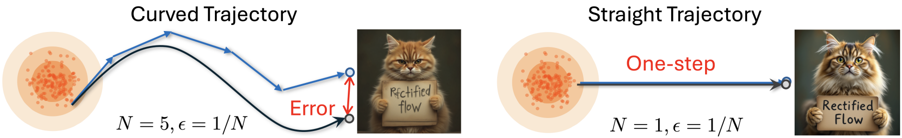


Straightness = Fast

- Euler discretization error depends on the **trajectory curvature**:

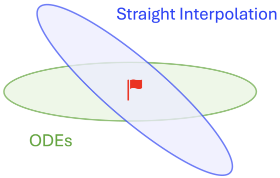
$$\left\|\hat{Z}_t - Z_t\right\| = O(\epsilon M), \qquad M = \sup_t \left\|\ddot{Z}_t\right\|.$$

- **Perfectly straight trajectories = one-step generation**



Idea Goal: find **Straight ODE transports** from P_0 to P_1 that follow straight trajectories.

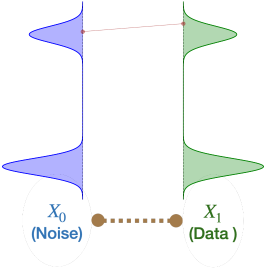
$$\{\texttt{ODE}\} \cap \{\texttt{Straight Trajectories}\}$$



Rectified Flow in a Nutshell

- **Coupling:** Sample from a noise-data pair (X_0, X_1).
- **Interpolation:** Construct interpolation:

$$X_t = tX_1 + (1-t)X_0.$$



Rectified Flow in a Nutshell

- **Coupling:** Sample from a noise-data pair (X_0, X_1).
- **Interpolation:** Construct interpolation:

$$X_t = tX_1 + (1-t)X_0.$$

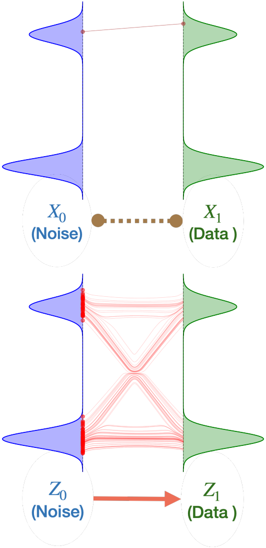
- **Causalization:** Convert interpolation to a causal process:

$$\dot{Z}_t = v_t(Z_t)$$

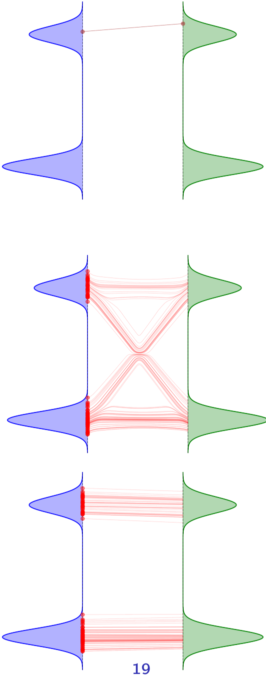
by minimizing:

$$\min_v \int_0^1 \mathbb{E}_{(X_0,X_1)} \left[\| \dot{X}_t - v_t(X_t) \|^2 \right] dt,$$

where $\dot{X}_t = X_1 - X_0$ are the line directions.



Rectified Flow in a Nutshell



- **Coupling:** Sample from a noise-data pair (X_0, X_1).
- **Interpolation:** Construct interpolation:

$$X_t = tX_1 + (1-t)X_0.$$

- **Causalization:** Convert interpolation to a causal process:

$$\dot{Z}_t = v_t(Z_t)$$

by minimizing:

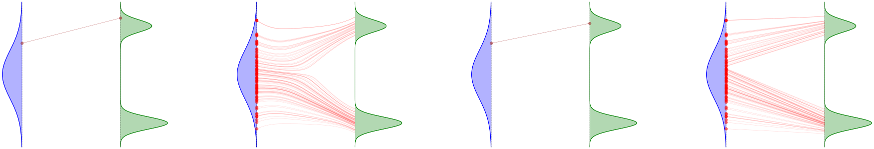
$$\min_v \int_0^1 \mathbb{E}_{(X_0,X_1)} \left[\|\dot{X}_t - v_t(X_t)\|^2 \right] dt,$$

where $\dot{X}_t = X_1 - X_0$ are the line directions.

- **Reflow:** Simulate ODE $\dot{Z}_t = v_t(Z_t)$ to obtain new couplings (Z_0, Z_1). **Repeat**.

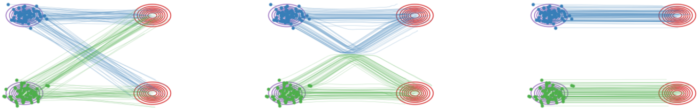
Rectified Flow

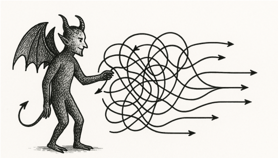
- Interpolation $\rightarrow$ Generation $\rightarrow$ Faster Generation



Rewiring Trajectories

- Interpolation paths can intersect and cross
- But trajectories of ODEs can never cross each other.
- Rectified Flow rewires the crossings of interpolation.





The Rewiring Demon

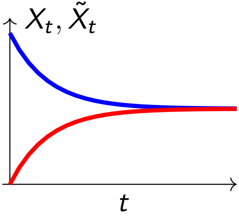
ODEs Trajectories Can Not Cross Each Other

$$\dot{X}_t = v_t(X_t).$$

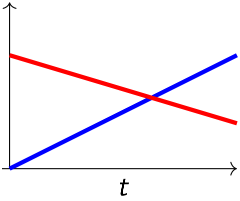
- The update direction $\dot{X}_t$ is uniquely determined by X_t.

Let $\{X_t\}$ and $\{\tilde{X}_t\}$ be solutions of the same ODE. Then

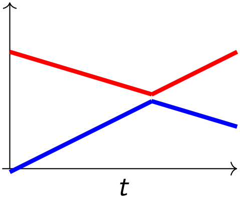
$$X_0 = \tilde{X}_0 \implies X_t = \tilde{X}_t \quad \text{for all } t \text{ in the existence interval.}$$



Possible for ODE



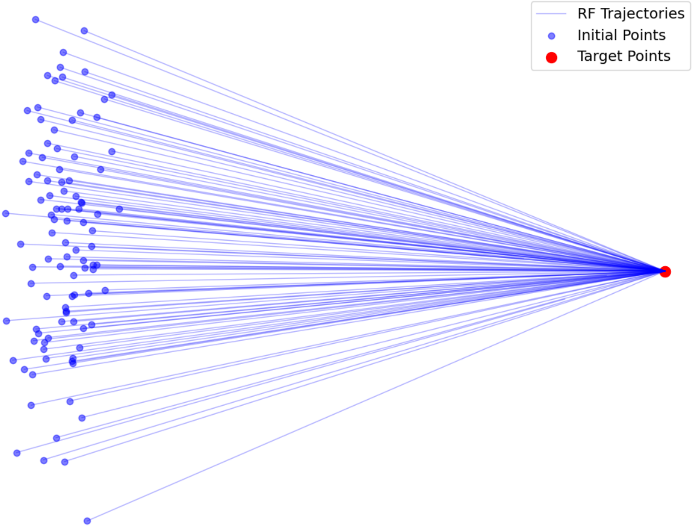
Impossible for ODE



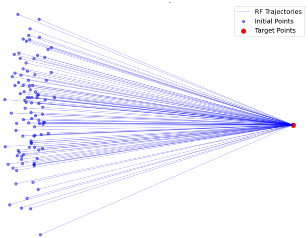
Rewired at crossing

Rectified Flow: Single Data Case

- Consider the case of a single point $x^{\texttt{data}}$:



Rectified Flow: Single Data Case



- Interpolation:

$$X_t = tx^{\mathtt{data}} + (1-t)X_0.$$

- This interpolation also defines an ODE:

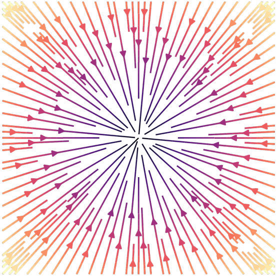
$$\frac{\mathrm{d}}{\mathrm{d}t}X_t = x^{\mathtt{data}} - X_0 = \frac{x^{\mathtt{data}} - X_t}{1-t}.$$

where X_0 is eliminated using the interpolation formula.

$$v^*(x,t) = \frac{x^{\mathtt{data}} - x}{1-t}$$ is the RF velocity field.

Single Point Rectified Flow

$$\frac{\mathrm{d}}{\mathrm{d}t} X_t = \frac{x^{\mathtt{data}} - X_t}{1 - t}, \quad t \in [0, 1]$$



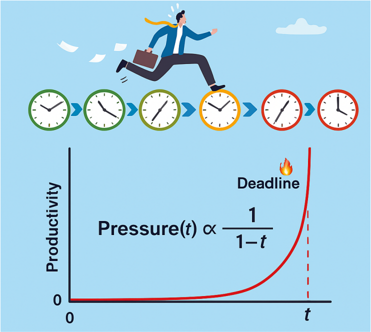
- Apparent singularity from the $1/(1-t)$ factor.
- Yet the solution is perfectly regular and stable:
  - Straight trajectories
  - Finite uniform speed
  - Always arrives at $X_t = x^{\mathtt{data}}$ when $t = 1$

- Also perfectly numerically stable: Euler's method yields exact solution in one step.

Single Point Rectified Flow

$$\frac{\mathrm{d}}{\mathrm{d}t}X_t = \frac{x^{\mathtt{data}} - X_t}{1-t}, \qquad t \in [0,1]$$

- Intuitively, $1/(1-t)$ is a "deadline pressure".
- Carefully calculated to land $x^{\mathtt{data}}$ precisely at $t=1$.



Time-Scaled Gradient Flow

- Reparameterize time:

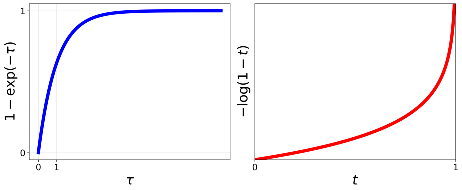
$$\tau = -\log(1-t) \qquad \iff \qquad t = 1 - e^{-\tau}.$$

- Define new variable: $Y_\tau := X_{t(\tau)}$
- Then, the dynamics become:

$$\dot{Y}_\tau = x^{\mathtt{data}} - Y_\tau$$

- This is the standard gradient flow of the quadratic potential:

$$f(y) = \frac{1}{2}\left\|x^{\mathtt{data}} - y\right\|^2$$



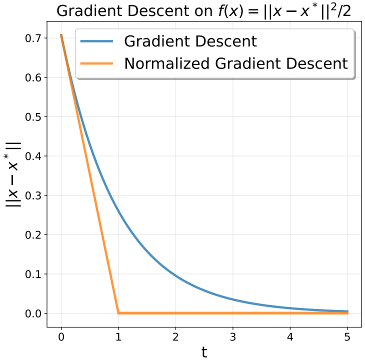
Normalized Gradient Flow

The straight-line ODE $\dot{X}_t = \frac{x^* - X_t}{1-t}$ is also equivalent to

$$\dot{X}_t = -\eta \frac{\nabla f(x)}{\|\nabla f(x)\|},$$

with

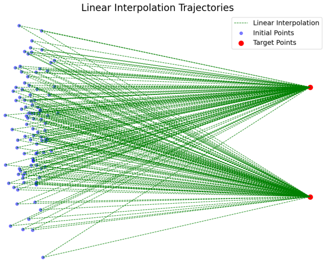
$$f(x) = \frac{1}{2}\|x - x^*\|^2, \quad \eta = \|x_0\|.$$

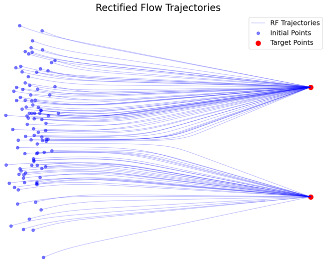


In general, normalized gradient flow on strongly convex functions [RB20]:

- Normalize the update norm across updates.
- Squeeze gradient flow into finite time.

Rectified Flow: More Data Points





Interpolation Paths

- The interpolated paths have crossings, hence "non-causal"

Rectified Flow

- Learns a causal ODE that best approximates the interpolation path.
- Unentangles the path into a forward generative process.
- It de-randomizes, causalizes, and Markovizes the interpolation.

From Interpolation to Generation

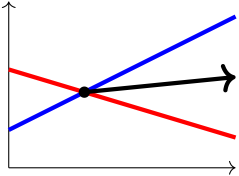
- Projecting the Interpolation Process to the ODE :

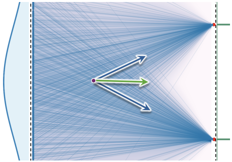
$$\min_{v} \mathbb{E}_{(X_0, X_1, t)} \left[\| \dot{X}_t - v_t(X_t) \|^2 \right].$$

- The Explicit solution is

$$v^*(x, t) = \mathbb{E}\left[\dot{X}_t \mid X_t = x \right].$$

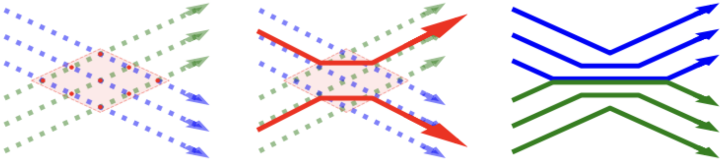
- The "mean field" velocity: Take the average direction whenever intersection happens.

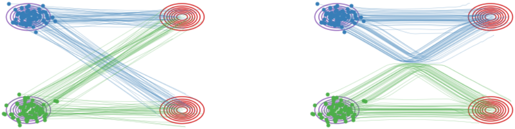




How Does Rewiring Actually Happen by Velocity Averaging?

- How Does Averaging Velocity Lead to Trajectory Rewiring?





Bias-variance Decomposition:

$$L(v) = \mathbb{E}\left[\|\dot{X}_t - v_t(X_t)\|^2\right]$$
$$= \underbrace{\mathbb{E}\left[\|\dot{X}_t - \mathbb{E}[\dot{X}_t \mid X_t]\|^2\right]}_{\substack{\text{Conditional variance} \\ = \mathbb{E}[\mathrm{Var}(\dot{X}_t|X_t)]}} + \underbrace{\mathbb{E}\left[\|v_t(X_t) - \mathbb{E}[\dot{X}_t \mid X_t]\|^2\right]}_{\text{Estimation bias}}$$

- Hence, the optimal solution should achieve zero bias:

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

Bias-variance Decomposition:

$$L(v) = \mathbb{E}\left[\|\dot{X}_t - v_t(X_t)\|^2\right]$$
$$= \underbrace{\mathbb{E}\left[\|\dot{X}_t - \mathbb{E}[\dot{X}_t \mid X_t]\|^2\right]}_{\substack{\text{Conditional variance} \\ = \mathbb{E}[\mathrm{Var}(\dot{X}_t | X_t)]}} + \underbrace{\mathbb{E}\left[\|v_t(X_t) - \mathbb{E}[\dot{X}_t \mid X_t]\|^2\right]}_{\text{Estimation bias}}$$

- Hence, the optimal solution should achieve zero bias:

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

- The minimum loss value is

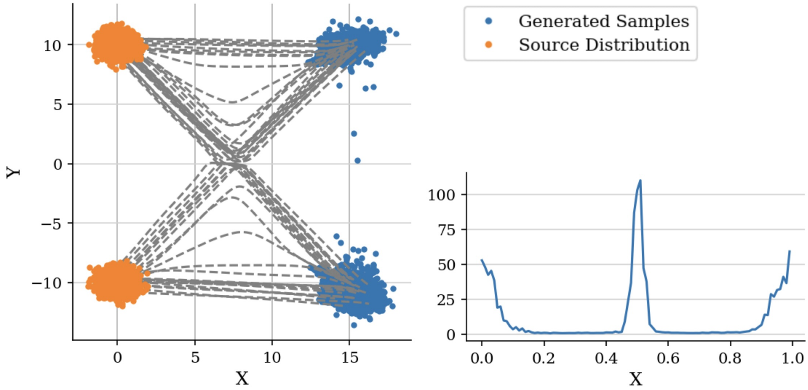
$$L(v^*) = \mathbb{E}\left[\mathrm{Var}(\dot{X}_t \mid X_t)\right].$$

It reflects:

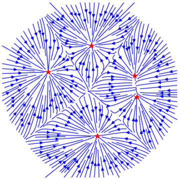
  - The **degree of intersection** of interpolation process $\{X_t\}$.
  - The **trajectory straightness** of the rectified flow $\{Z_t\}$.

Loss as Straightness

The lower the loss, the **straighter** the ODE path from noise to data.



Singular Velocity on Finite Data Points



On a finite number of data points $\{x^{(i)}\}_{i=1}^{n}$:

$$v^*(x,t) = \sum_{i=1}^{n} \omega_t^{(i)}(x) \left(\frac{x^{(i)} - x}{1 - t} \right),$$

with posterior weights $\omega_t^{(i)}(x) = \frac{\rho_0\left(\hat{x}_0^{(i)} \mid x^{(i)}\right)}{\sum_j \rho_0\left(\hat{x}_0^{(j)} \mid x^{(j)}\right)}$, $\hat{x}_0^{(i)} = \frac{x - tx^{(i)}}{1-t}$.

Finite Mixture of $\frac{x^{(i)}-x}{1-t}$

- Singular velocity due to $1/(1-t)$.
- Dynamics exactly achieves the training data.
- Minimum training loss, but large evaluation loss.
- Neural network **must** provide smoothing as it **can not fit** the $1/(1-t)$ singularity.

Analytic Velocity on Smooth Densities

With smooth densities, we get

$$v_t^*(x) = \mathbb{E}_{X_1 \sim \pi_1}\left[\omega_t(X_1 \mid x)\frac{X_1 - x}{1-t}\right],$$

where $\omega_t(x_1 \mid x)$ is the posterior probability:

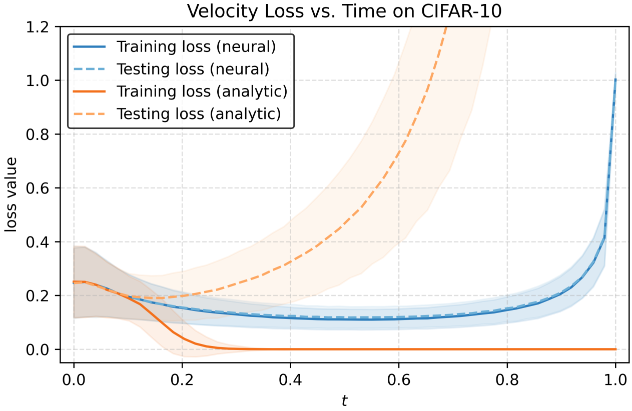
$$\omega_t(x_1 \mid x) := \mathbb{P}(X_1 = x_1 \mid X_t = x) = \frac{\rho_0\left(\hat{x}_0 \mid x_1\right)}{\mathbb{E}_{X_1}\left[\rho_0\left(\hat{X}_0 \mid X_1\right)\right]}, \quad \hat{x}_0 := \frac{x - tx_1}{1-t}$$

where $\rho_0(x_0 \mid x_1)$ is the density of X_0 given X_1.

- **Infinite mixture** of the one-point velocity $\dfrac{x^{\mathtt{data}} - x}{1-t}$.
- Singularity may be smoothed out.

Bless of Neural Fitting Error

- The singular analytic velocity on training data fails to generalize.
- But the neural net training refuses the singular solution.
- Avoiding singularity ensures data outside of training set can be sampled, leading to generalization.



Analytic model yields very small training loss yet exploding testing loss.

Open Question:

- Why does neural network generalizes in a way that matches human perception?
- Related: mechanistic explanation of diffusion generalization [NZMW24, SZT17, NBMS17].