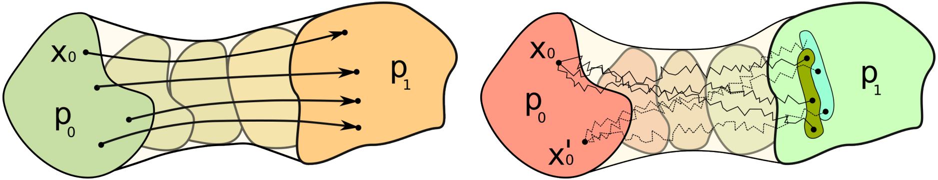


New Perspective Methods of Generative AI

[based on flows and diffusion bridges]

Alexander Korotin

Skoltech
Skolkovo Institute of Science and Technology
AIRI

Moscow, 2024

Modern Generative Models for Images

Text prompt: woman's transparent futuristic inspired sneakers, glitter, depth of field

KANDINSKY

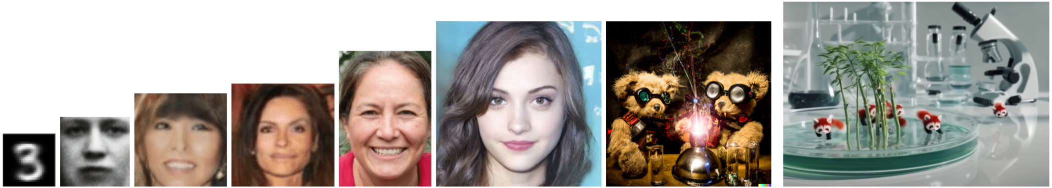
Text prompt: Chicken with potatoes baked in mayonnaise-sour cream sauce

SHEDEVRUM

Text prompt: 1967 Dodge Charger, moody lighting, side view, black, front view, lobby of the Louvre ...

MIDJOURNEY

Evolution of Generative Models



2013-2014
Initial attempts to learn generative models; VAE/GAN algorithms

2015-2017
Active development of (primarily GAN-based) algorithms, first convincingly scalable image generation models appear.

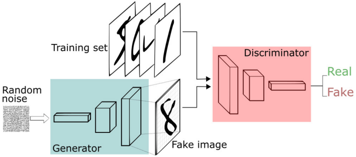
2019-2021
Diffusion-based models are introduced and actively developed.

2022
First inspiring large **text-to-image** models appear (based on diffusion models).

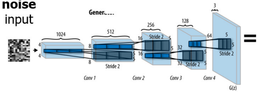
2024
Powerful **text-to-video** models appear based on diffusion models.

Principal Approaches to Generative Modeling[1,2]

Adversarial models (GANs, 2014)



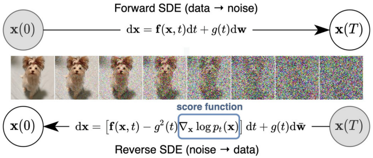
Random **noise** input



Deep **neural network** (generator)

Generated images
(these people do not exist!)

Diffusion Models (DM, 2019)



[1] Ian Goodfellow et al. (2014). "Generative adversarial nets". In: *Advances in neural information processing systems*, pp. 2672–2680.

[2] Jascha Sohl-Dickstein et al. (2015). "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International conference on machine learning*. PMLR, pp. 2256–2265.

Diffusion Models[3,4]

MAIN IDEA: reverse the data noising process.

Forward diffusion (noising SDE)

Take a data distribution $x_0 \sim p_0$ and gradually turn it to noise distribution $x_T \sim p_T = \mathcal{N}(0, \sigma^2 I)$.

$$dx_t = f(x_t, t)dt + g(t)dW_t$$

$$\left(\text{e.g., } dx_t = -\tfrac{1}{2}\beta_t dt + \sqrt{\beta_t} dW_t\right)$$

Reverse diffusion (denoising SDE)

Sample from noise distribution $x_T \sim p_T$ and reverse the diffusion to get $x_0 \sim p_0$:

$$dx_t = [f(x_t, t) - g^2(t)\nabla_x \log p(x_t, t)]dt + g(t)d\overline{W_t}$$

$$\left(\text{or } dx_t = [f(x_t, t) - \tfrac{1}{2}g^2(t)\nabla_x \log p(x_t, t)]dt\right)$$

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel (2020). "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33, pp. 6840–6851.

[4] Yang Song et al. (2020). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*.

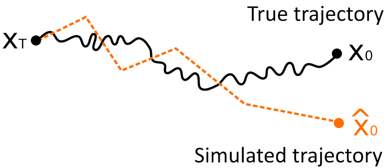
The Key Limitation of Diffusion Models: Time-Consuming Inference

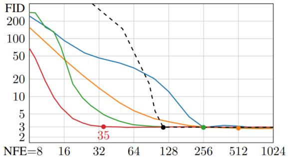
To simulate the denoising process:

$$\mathrm{d}x_t = \left[f(x_t, t) - g^2(t)\nabla_x \log p(x_t, t)\right]\mathrm{d}t + g(t)\mathrm{d}\overline{W}_t$$

one uses the discretization (e.g., Euler-Maruyama simulation):

$$x_{t-\Delta t} = x_t - \left[f(x_t, t) - g^2(t)\nabla_x \log p(x_t, t)\right]\Delta t + g(t)\sqrt{\Delta t}\xi_t \quad , \xi_t \sim \mathcal{N}(0, I).$$





Diff. models performance, CIFAR-10. FID w.r.t NFE.

Remark:

NFE (# function evaluations) ≡ (# discretization steps)

Straightening The Trajectories of Diffusion Models

What we have

Not straight (deterministic or stochastic) trajectories, which are HARD to simulate.

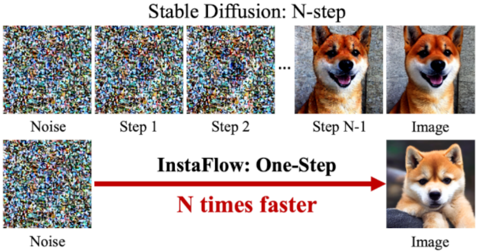
What we want

Straight (deterministic?) trajectories, which are EASY to simulate.

Part I: Flow Matching Framework and Rectified Flows

Teaser: Flow Matching Capabilities[5][6]

InstaFlow: 1 Step is Enough for HQ Diffusion-based Text-to-image Synthesis



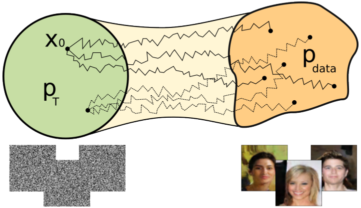
Scaling Rectified **Flow** Transformers for High-Resolution Image Synthesis

[5]Xingchao Liu, Xiwen Zhang, et al. (2023). "Instaflow: One step is enough for high-quality diffusion-based text-to-image generation". In: *The Twelfth International Conference on Learning Representations*.
[6]Patrick Esser et al. (2024). "Scaling rectified flow transformers for high-resolution image synthesis". In: *Forty-first International Conference on Machine Learning*.

Flow matching vs. Diffusion Models: Key Differences

Diffusion models framework (2019)

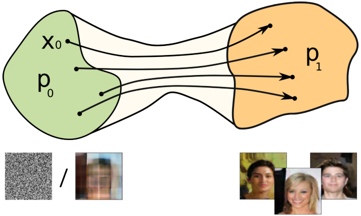
- maps given complex data distribution to the normal distribution.



- uses pre-defined noising process.
- (theoretically) requires infinite time horizon $[0, T]$.
- based on SDEs ($\Rightarrow$ complex stuff).

Flow matching framework (2023)

- maps arbitrary distribution p_0 to arbitrary distribution p_1.

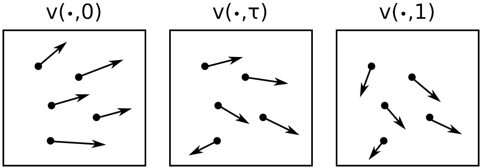


- no pre-defined process.
- finite time horizon $[0, \underset{\overset{\|}{T}}{1}]$.
- based on ODEs.

Main Flow Matching-related papers

1. **Flow Matching (FM):** Yaron Lipman et al. (2022). "Flow Matching for Generative Modeling". In: *The Eleventh International Conference on Learning Representations*
2. **Rectified Flows (RF):** Xingchao Liu, Chengyue Gong, et al. (2022). "Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow". In: *The Eleventh International Conference on Learning Representations*
3. **Conditional FM (OT-CFM):** Alexander Tong et al. (2024). "Improving and generalizing flow-based generative models with minibatch optimal transport". In: *Transactions on Machine Learning Research*. Expert Certification. ISSN: 2835-8856. URL: `https://openreview.net/forum?id=CD9Snc73AW`
4. **Straightening FM:** Aram-Alexandre Pooladian et al. (2023). "Multisample Flow Matching: Straightening Flows with Minibatch Couplings". In: *International Conference on Machine Learning*. PMLR, pp. 28100–28127
5. **Optimal FM (OFM):** Nikita Kornilov et al. (2024). "Optimal Flow Matching: Learning Straight Trajectories in Just One Step". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: `https://openreview.net/forum?id=kqmucDKVcU`

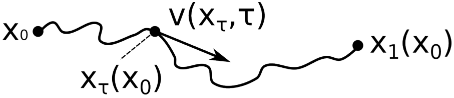
Vector field $v : \mathbb{R}^D \times [0,1] \to \mathbb{R}^D$.



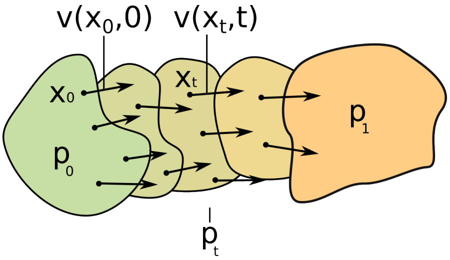
Movement of a point along the field.

Let $x_t(x_0)$ be the solution to $\mathrm{d}x_t = v(x_t, t)\mathrm{d}t$ with initial condition $x_{t=0} = x_0$, i.e.:

$$x_t(x_0) = x_0 + \int_0^t v(x_\tau(x_0), \tau)\mathrm{d}\tau.$$

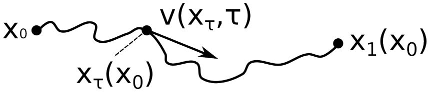


Flow Transport: Key Idea



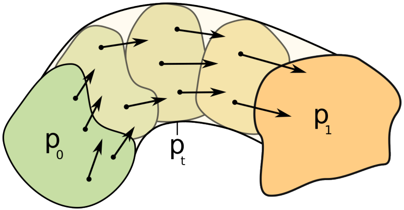
Find a (time-dependent) vector field $v(x_t, t)$ which transports the probability mass of distribution p_0 to distribution p_1, i.e.:

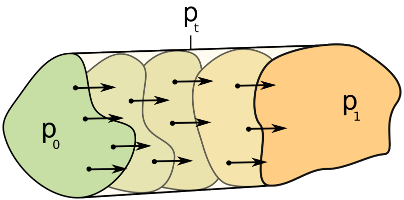
$$\text{If } x_0 \sim p_0 \text{ then } x_1(x_0) \sim p_1$$



Flow Transport: Non-uniqueness

There may exist multiple vector fields v transporting p_0 to p_1.





Let p_t denote the distribution of $x_t(x_0)$ (for $x_0 \sim p_0$) obtained from p_0 by transporting its mass along the vector field $v(x_t, t)$.

How to construct at least one sequence of distributions p_t which transports p_0 to p_1?

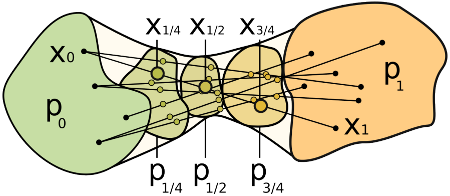
Remark: This should be much easier than constructing vector field v itself.

Flow Transport: Creating Interpolating Curve

Simple interpolation. Pick $x_0 \sim p_0$, $x_1 \sim p_1$ and set:

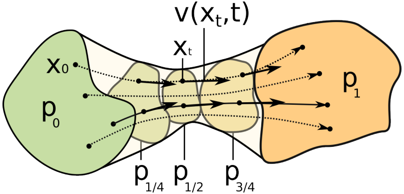
$$x_t = x_0 \cdot (1 - t) + x_1 \cdot t$$

Let p_t be the distribution of points x_t obtained with the procedure above.



How to find some vector field $v(x_t, t)$ which produces this sequence of distributions p_t by transporting p_0?

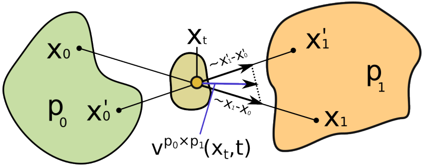
Flow Matching: Basic Algorithm[7]



$$\begin{cases} \mathrm{d}x_t = v(x_t, t)\mathrm{d}t \\ x_0 \sim p_0 \end{cases}$$

One of such vector fields could be found as a solution to flow matching (FM) objective:

$$\min_{v} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \left\{ \mathbb{E}_{t \sim [0,1]} \left\| v(\underset{x_0 \cdot (1-t) + x_1 \cdot t}{\underset{\parallel}{x_t}}, t) - (x_1 - x_0) \right\|^2 \right\}.$$



Let $v^{p_0 \times p_1}(x_t, t)$ denote the minimizer to this problem.

[7] Xingchao Liu, Chengyue Gong, et al. (2022). "Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow". In: *The Eleventh International Conference on Learning Representations*.

Flow Matching: Practical Aspects[8]

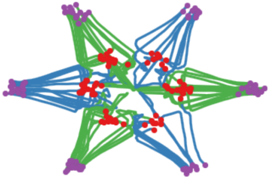
$$v_{\theta^*} = \arg\min_{v_\theta} \mathop{\mathbb{E}}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \left\{ \mathop{\mathbb{E}}_{t \sim [0,1]} \left\| v_\theta(\underbrace{x_t}_{\substack{\| \\ x_0 \cdot (1-t) + x_1 \cdot t}}, t) - (x_1 - x_0) \right\|^2 \right\}$$

- $\mathbb{E}_{x_0 \sim p_0, x_1 \sim p_1}(\cdot)$ is estimated using train samples $x_0 \sim p_0$ and $x_1 \sim p_1$ (datasets).
- t is sampled at random from Uniform$[0, 1]$.
- $v = v_\theta$ is a (large) Neural Network which takes data point x_t and time (time embedding) t as the input.

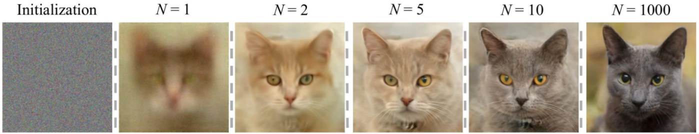
[8] Yaron Lipman et al. (2022). "Flow Matching for Generative Modeling". In: *The Eleventh International Conference on Learning Representations*.

Flow Matching: Preliminary Examples and Some Issues

Toy example:
Gaussians -> Gaussians



Example of image generation:
(different number of trajectory integration steps is shown)



Remark: if the trajectories were really straight, generation with different numbers of steps would give the same result, because there would be no integration errors.

Problem: trajectories are not straight enough.

Rectified Flows[9]

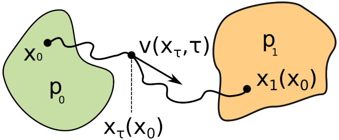
Key idea: iteratively repeat flow matching (FM) procedure to get straighter trajectories.

Step 1: pure flow matching.

$$v^1 = \arg\min_{v} \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \left\{ \mathbb{E}_{t \sim [0,1]} \left\| v(\underbrace{x_t}_{x_0 \cdot (1-t) + x_1 \cdot t}, t) - (x_1 - x_0) \right\|^2 \right\}.$$

Result: vector field v^1 moving p_0 to p_1:

$$x_0 \sim p_0 \wedge \mathrm{d}x_t = v^1(x_t, t)\mathrm{d}t \quad \Longrightarrow \quad x_1^{v^1}(x_0) \sim p_1.$$



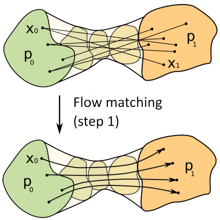
[9] Xingchao Liu, Chengyue Gong, et al. (2022). "Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow". In: *The Eleventh International Conference on Learning Representations*.

Rectified Flows

Step k+1: flow matching using samples $(x_0, x_1^{v^k}(x_0))$:

$$v^{k+1} = \arg\min_{v} \mathbb{E}_{x_0 \sim p_0} \left\{ \mathbb{E}_{t \sim [0,1]} \left\| v(\underbrace{x_t}_{x_0 \cdot (1-t) + x_1^{v^k}(x_0) \cdot t}, t) - (x_1^{v^k}(x_0) - x_0) \right\|^2 \right\}.$$

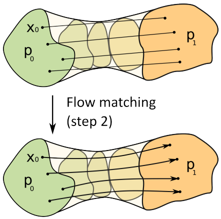
Step 1



$dx_t = v^1(x_t)dt$

Learn some vector field v^1 transporting p_0 to p_1.

Step 2

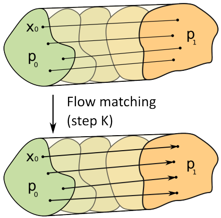


$dx_t = v^2(x_t)dt$

Use the previously obtained field v^1 to get samples $(x_0, x_1^{v^1}(x_0))$ to train the new field v^2.

. . .

Step k+1



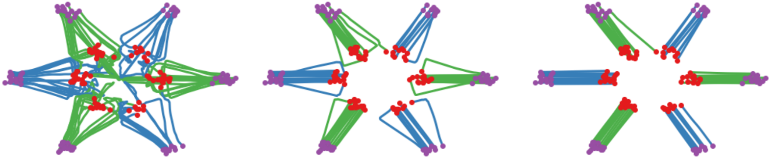
$dx_t = v^{k+1}(x_t)dt$

Use the previously obtained field v^k to get samples $(x_0, x_1^{v^k}(x_0))$ to train the new field v^{k+1}.

Rectified Flows: The Straightening Effect

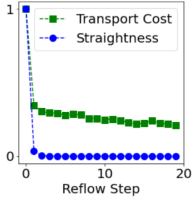
Theorem (*Informal*[10])

Vector field v^K produces more straight trajectories ($\mathrm{d}x_t = v^K(x_t, t)\mathrm{d}t \wedge x_0 \sim p_0$) as $K \to \infty$.



(a)The 1st rectified flow Z^1 (b) Reflow Z^2 (c) Reflow Z^3

Rectified Flows performance between source and target, $K = 1, 2, 3$.

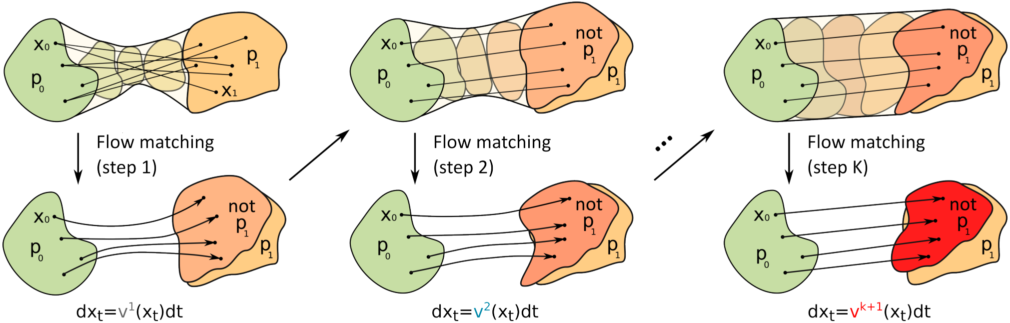


Transport cost, Straightness.

[10] Xingchao Liu, Chengyue Gong, et al. (2022). "Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow". In: *The Eleventh International Conference on Learning Representations*.

Rectified Flows: Practical Issues

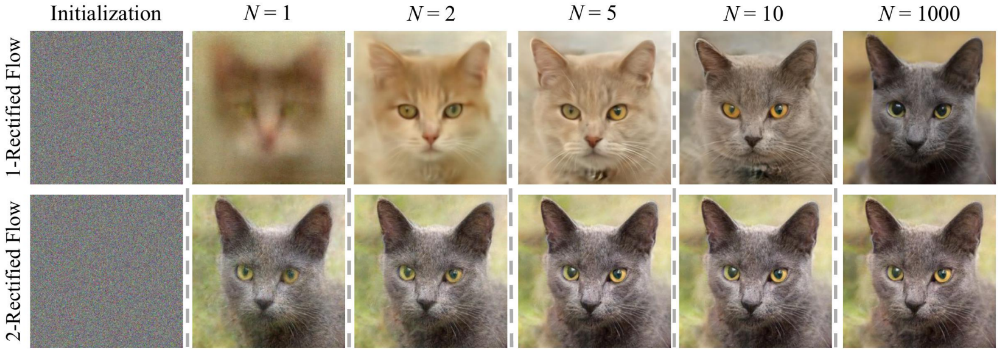
After the first flow matching step, the iterative approach **never** utilizes the true data $x_1 \sim p_1$ but uses learned samples $x_1^{v^1}\ p_1$, i.e., those obtained via $x_0 \sim p_0 \wedge \mathrm{d}x_t = v^1(x_t, t)\mathrm{d}t$.



This issue leads to the target matching error, i.e., the model never learns the true p_1 due to statistical, approximation, optimization errors, which accumulates with FM iterations.

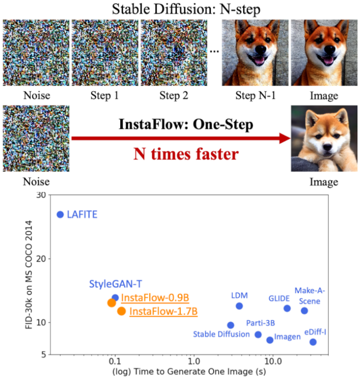
Rectified Flows: Image Generation Examples

Columns - different NFE, **rows** – different FM iterations.



For the 2-step FM the trajectories are rather straight which can be deduced from the fact that the model works reasonable even for $NFE = 1$ (2-Rectified flow, $N = 1$).

Instaflow: Rectified Flow as a Tool for Model Distillation[11]



[11]Xingchao Liu, Xiwen Zhang, et al. (2023). "Instaflow: One step is enough for high-quality diffusion-based text-to-image generation". In: *The Twelfth International Conference on Learning Representations*.

Stable Diffusion: Flow-based Large Text-to-Image Model[12]

a space elevator, cinematic scifi art | A cheeseburger with juicy beef patties and melted cheese sits on top of a toilet that looks like a throne and stands in the middle of the royal chamber. | a hole in the floor of my bathroom with small gremlins living in it | a small office made out of car parts | This dreamlike digital art captures a vibrant, kaleidoscopic bird in a lush rainforest. | human life depicted entirely out of fractals | an origami pig on fire in the middle of a dark room with a pentagram on the floor

an old rusted robot wearing pants and a jacket riding skis in a supermarket. | smiling cartoon dog sits at a table, coffee mug on hand, as a room goes up in flames. "This is fine," the dog assures himself.

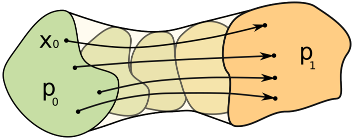
[12] Patrick Esser et al. (2024). "Scaling rectified flow transformers for high-resolution image synthesis". In: *Forty-first International Conference on Machine Learning*.

Part II: Bridge Matching Framework and Diffusion Schrodinger Bridge Matching

Flow Matching vs. Bridge Matching: a Comparison

Flow Matching

$$\mathrm{d}x_t = v(x_t, t)\mathrm{d}t$$



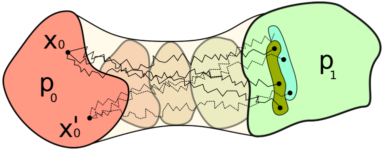
Define interpolation: $x_t \stackrel{\text{def}}{=} x_0 \cdot (1-t) + x_1 \cdot t.$

$$\min_{v} \mathop{\mathbb{E}}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \mathop{\mathbb{E}}_{t \sim [0,1]} \left\| v(x_t, t) - (x_1 - x_0) \right\|^2.$$

Can be iterated to straighten the flows.
Related to the **Optimal Transport** (OT).

Bridge Matching

$$\mathrm{d}x_t = v(x_t, t)\mathrm{d}t + \sqrt{\epsilon}\mathrm{d}W_t \quad (\epsilon > 0).$$



Define a distribution: $p_t^{\epsilon} \stackrel{\text{def}}{=} \mathcal{N}\left(x_t, \epsilon t(1-t)\right)$

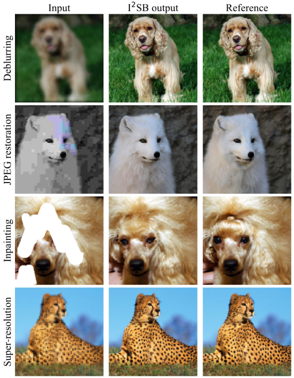
$$\min_{v} \mathop{\mathbb{E}}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \mathop{\mathbb{E}}_{t \sim [0,1]} \mathop{\mathbb{E}}_{\tilde{x}_t \sim p_t^{\epsilon}} \left\| v(\tilde{x}_t, t) - \frac{x_1 - \tilde{x}_t}{1 - t} \right\|^2.$$

Can be iterated and converges to the **Schrödinger bridge**.

Examples of Bridge Matching Models for Images[13][14]

Image-to-image Schrodinger Bridge

for various image restoration problems



Diffusion Schrodinger Bridge Matching

for unpaired image-to-image translation

cat $\rightarrow$ wild

wild $\rightarrow$ cat

[13] Guan-Horng Liu et al. (2023). "I2SB: image-to-image Schrödinger bridge". In: *Proceedings of the 40th International Conference on Machine Learning*, pp. 22042–22062.
[14] Yuyang Shi et al. (2024). "Diffusion Schrödinger bridge matching". In: *Advances in Neural Information Processing Systems* 36.

Main Bridge Matching-related papers

1. **Bridge Matching (BM):** Stefano Peluchetti (2022). *Non-Denoising Forward-Time Diffusions*. URL: https://openreview.net/forum?id=oVfIKuhqfC
2. **Bridge Matching (BM):** Xingchao Liu, Lemeng Wu, Mao Ye, et al. (n.d.). "Let us Build Bridges: Understanding and Extending Diffusion Generative Models". In: *NeurIPS 2022 Workshop on Score-Based Methods*
3. **Iterative Bridge Matching (IBM):** Stefano Peluchetti (2022). *Non-Denoising Forward-Time Diffusions*. URL: https://openreview.net/forum?id=oVfIKuhqfC
4. **Iterative Markovian Fitting (IMF):** Yuyang Shi et al. (2024). "Diffusion Schrödinger bridge matching". In: *Advances in Neural Information Processing Systems* 36
5. **Discrete Iterative Markovian Fitting (D-IMF):** Nikita Gushchin, Daniil Selikhanovych, et al. (2024). "Adversarial Schrödinger Bridge Matching". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=L3Knnigicu
6. **Optimal Schrödinger Bridge Matching (LightSB-M)** Nikita Gushchin, Sergei Kholkin, et al. (2024). "Light and Optimal Schrödinger Bridge Matching". In: *Forty-first International Conference on Machine Learning*

Part III: Our Results Related to Bridge/Flow Matching Models

Optimal Bridge/Flow Matching & Adversarial Bridge Matching

Our published papers (NeurIPS, ICML 2024):

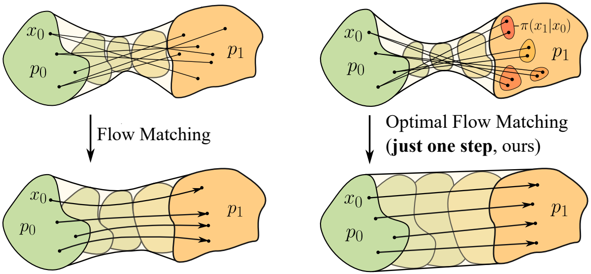
1. Nikita Kornilov et al. (2024). "Optimal Flow Matching: Learning Straight Trajectories in Just One Step". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=kqmucDKVcU
2. Nikita Gushchin, Sergei Kholkin, et al. (2024). "Light and Optimal Schrödinger Bridge Matching". In: *Forty-first International Conference on Machine Learning*
3. Nikita Gushchin, Daniil Selikhanovych, et al. (2024). "Adversarial Schrödinger Bridge Matching". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=L3Knnigicu

Our related pre-prints:

1. Sergei Kholkin et al. (2024). "Diffusion & Adversarial Schrodinger Bridges via Iterative Proportional Markovian Fitting". In: *arXiv preprint arXiv:2410.02601*

Optimal Flow Matching (OFM)[15]

Main idea: during FM minimization, consider *only specific vector fields* generating exactly straight trajectories. This optimization provably leads to *optimal transport* displacements.

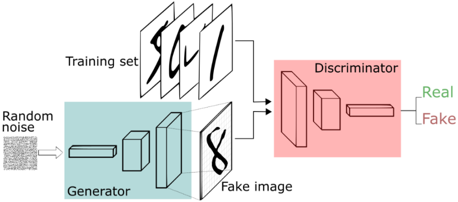


In **just one FM minimization** and **for any initial** π, we get straight trajectories + solve OT.

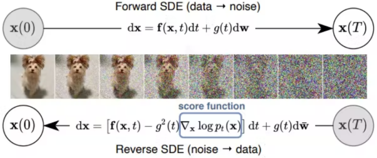
[15]Nikita Kornilov et al. (2024). "Optimal Flow Matching: Learning Straight Trajectories in Just One Step". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=kqmucDKVcU.

Conclusion

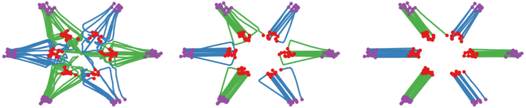
Generative Adversarial Networks (2014-...)



Diffusion models (2019-...)



Flow/bridge matching models (2022-...)



What is next?

?