Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow

Xingchao Liu[*]
University of Texas at Austin
xcliu@utexas.edu

Chengyue Gong[*]
University of Texas at Austin
cygong@cs.utexas.edu

Qiang Liu
University of Texas at Austin
lqiang@cs.utexas.edu

Abstract

We present rectified flow, a surprisingly simple approach to learning (neural) ordinary differential equation (ODE) models to transport between two empirically observed distributions π_0 and π_1, hence providing a unified solution to generative modeling and domain transfer, among various other tasks involving distribution transport. The idea of rectified flow is to learn the ODE to follow the straight paths connecting the points drawn from π_0 and π_1 as much as possible. This is achieved by solving a straightforward nonlinear least squares optimization problem, which can be easily scaled to large models without introducing extra parameters beyond standard supervised learning. The straight paths are special and preferred because they are the shortest paths between two points, and can be simulated exactly without time discretization and hence yield computationally efficient models. We show that the procedure of learning a rectified flow from data, called rectification, turns an arbitrary coupling of π_0 and π_1 to a new deterministic coupling with provably non-increasing convex transport costs. In addition, recursively applying rectification allows us to obtain a sequence of flows with increasingly straight paths, which can be simulated accurately with coarse time discretization in the inference phase. In empirical studies, we show that rectified flow performs superbly on image generation and image-to-image translation. In particular, on image generation and translation, our method yields nearly straight flows that give high quality results even with *a single Euler discretization step*. The full paper can be found at https://arxiv.org/abs/2209.03003.

1 Introduction

Compared with supervised learning, the shared difficulty of various forms of unsupervised learning is the lack of *paired* input/output data with which standard regression or classification tasks can be invoked. The gist of most unsupervised methods is to find, in one way or another, meaningful correspondences between points from two distributions. For example, generative models such as generative adversarial networks (GAN) and variational autoencoders (VAE) [e.g., 4, 8, 2] seek to map data points to latent codes following a simple elementary (Gaussian) distribution with which the data can be generated and manipulated. Representation learning rests on the idea that if a sufficiently smooth function can map a structured data distribution to an elementary distribution, it can (likely) be endowed with certain semantically meaningful interpretation and useful for various downstream learning tasks. On the other hand, domain transfer methods find mappings to transfer points from two

[*]XL and CG contributed equally to this work.

NeurIPS 2022 Workshop on Score-Based Methods.

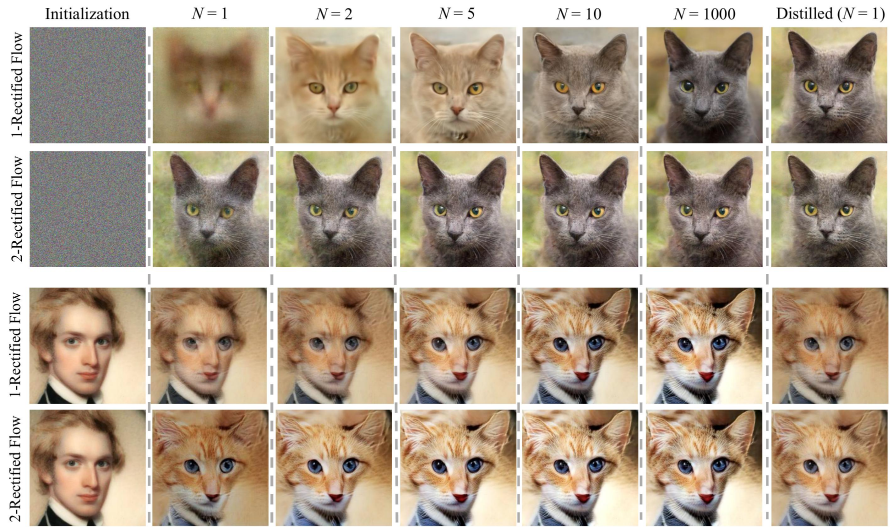


Figure 1: The trajectories of rectified flows for image generation (π_0: standard Gaussian noise, π_1: cat faces, top two rows), and image transfer between human and cat faces (π_0: human faces, π_1: cat faces, bottom two rows), when simulated using Euler method with step size $1/N$ for N steps. The first rectified flow induced from the training data (*1-rectified flow*) yields good results with a very small number (e.g., ≥ 2) of steps; the straightened reflow induced from *1-rectified flow* (denoted as *2-rectified flow*) has nearly straight line trajectories and yield good results even with one discretization step.

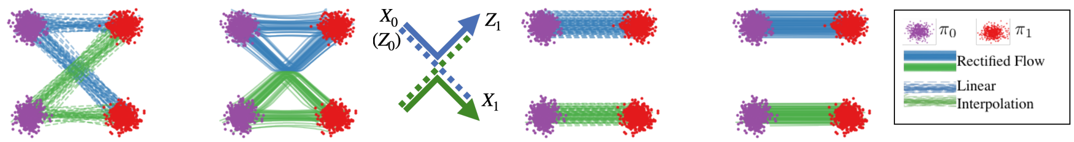
different data distributions, both observed empirically, for the purpose of image-to-image translation, style transfer, and domain adaption [e.g., 21, 3, 17, 13]. All these tasks can be framed unifiedly as finding a transport map between two distributions:

The Transport Mapping Problem *Given empirical observations of two distributions $X_0 \sim \pi_0$, $X_1 \sim \pi_1$ on $\mathbb{R}^d$, find a transport map $T\colon \mathbb{R}^d \to \mathbb{R}^d$ (hopefully nice or optimal in certain sense), such that $Z_1 := T(Z_0) \sim \pi_1$ when $Z_0 \sim \pi_0$, that is, (Z_0, Z_1) is a coupling (a.k.a transport plan) of π_0, π_1.*

We present a simple approach to this transport mapping problem. Our method, called *rectified flow*, learns an ODE model that transports distribution π_0 to π_1 by *following straight line paths as much as possible.* The straight paths are preferred both theoretically because it is the shortest path between two end points, and computationally because it can be exactly simulated without time discretization. Hence, flows with straight paths are simultaneously continuous-time (or infinite-step) models, and one-step models (as the case of GANs and VAEs).

Algorithmically, the rectified flow is trained with a simple and scalable unconstrained least squares optimization procedure, which avoids the instability issues of GANs, the intractable likelihood of MLE methods, and the subtle hyper-parameter decisions of denoising diffusion models. The procedure of obtaining the rectified flow from the training data has the attractive theoretical property of 1) yielding a coupling with non-increasing transport cost jointly for all convex cost c, and 2) making the paths of flow increasingly straight and hence incurring lower error with numerical solvers. Therefore, with a *reflow* procedure that iteratively trains new rectified flows with the data simulated from the previously obtained rectified flow, we obtain nearly straight flows that yield good results even with the coarsest time discretization, i.e., one Euler step. Our method is purely ODE-based, and is both conceptually simpler and practically faster in inference time than the SDE-based approaches of [5, 16, 15].

Empirically, rectified flow can yield high-quality results for image generation when simulated with a very few number of Euler steps (see Figure 1, top row). Moreover, with just one step of reflow, the flow becomes nearly straight and hence yield good results with a single Euler discretization step (Figure 1, the second row). This substantially improves over the standard denoising diffusion methods. Quantitatively, we claim a state-of-the-art result of FID (4.85) and recall (0.51) on CIFAR10 for one-step fast diffusion/flow models [1, 11, 18, 20, 10]. The same algorithm also achieves superb result on domain transfer tasks such as image-to-image translation (the bottom two rows of Figure 1).



(a) Linear interpolation $X_t = tX_1 + (1-t)X_0$ (b) Rectified flow Z_t induced by (X_0, X_1) (c) Linear interpolation $Z_t = tZ_1 + (1-t)Z_0$ (d) Rectified flow Z'_t induced by (Z_0, Z_1)

Figure 2: (a) Linear interpolation of data input $(X_0, X_1) \sim \pi_0 \times \pi_1$. (b) The rectified flow Z_t induced by (X_0, X_1); the trajectories are "rewired" at the intersection points to avoid the crossing. (c) The linear interpolation of (Z_0, Z_1) from flow Z_t. (d) The rectified flow induced by (Z_0, Z_1), which has straight paths.

2 Method

Rectified flow Given empirical observations of $X_0 \sim \pi_0, X_1 \sim \pi_1$, the rectified flow induced from (X_0, X_1) is an ordinary differentiable model (ODE) on time $t \in [0, 1]$,

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

which converts Z_0 from π_0 to a Z_1 following π_1. The drift force $v \colon \mathbb{R}^d \to \mathbb{R}^d$ is set to drive the flow to follow the direction $(X_1 - X_0)$ of the linear path pointing from X_0 to X_1 as much as possible, by solving a simple least squares regression problem:

$$\min_v \int_0^1 \mathbb{E}\left[\left\|(X_1 - X_0) - v\big(X_t,\ t\big)\right\|^2\right] \mathrm{d}t, \quad \text{with} \quad X_t = tX_1 + (1-t)X_0, \tag{1}$$

where X_t is the linear interpolation of X_0 and X_1. Naviely, X_t follows the ODE of $\mathrm{d}X_t = (X_1 - X_0)\mathrm{d}t$, which is non-causal (or anticipating) as the update of X_t requires the information of the final point X_1. By fitting the drift v with $X_1 - X_0$, the rectified flow *causalizes* the paths of linear interpolation X_t, yielding an ODE flow that can be simulated without seeing the future. In practice, we parameterize v with a neural network or other nonlinear models and solve (1) with any off-the-shelf stochastic optimizer, such as SGD, with empirical draws of (X_0, X_1). See Algorithm 1.

Flows avoid crossing A key to understanding the method is the non-crossing property of flows: the different paths following a well defined ODE $\mathrm{d}Z_t = v(Z_t, t)\mathrm{d}t$, whose solution exists and is unique, *cannot cross each other* at any time $t \in [0, 1)$. Specifically, there exists no location $z \in \mathbb{R}^d$ and time $t \in [0, 1)$, such that two paths go across z at time t along different directions, because otherwise the solution of the ODE would be non-unique. On the other hand, the paths of the interpolation process X_t may intersect with each other (Figure 2a), which makes it non-causal. Hence, as shown in Figure 2b, the rectified flow *rewires* the individual trajectories passing through the intersection points to avoid crossing, while tracing out the same density map as the linear interpolation paths due to the optimization of (1).

Rectified coupling reduces transport costs If (1) is solved exactly, the pair (Z_0, Z_1) of the rectified flow has the following two key properties:

1) (Z_0, Z_1) is guaranteed to be a valid coupling of π_0, π_1, that is, Z_1 follows π_1 if $Z_0 \sim \pi_0$.

2) (Z_0, Z_1) guarantees to yield no larger transport cost than the data pair (X_0, X_1) simultaneously for all *convex cost functions c, that is,*

$$\mathbb{E}[c(Z_1 - Z_0)] \le \mathbb{E}[c(X_1 - X_0)], \quad \forall \textit{ convex } c \colon \mathbb{R}^d \to \mathbb{R}.$$

The data pair (X_0, X_1) can be an arbitrary coupling of π_0, π_1, typically independent (i.e., $(X_0, X_1) \sim \pi_0 \times \pi_1$) as dictated by the lack of meaningfully paired observations in practical problems. In comparison, the rectified coupling (Z_0, Z_1) has a deterministic dependency as it is constructed from an ODE model. Hence, the rectification procedure above converts an arbitrary coupling into a deterministic coupling with lower or equal convex transport costs.

Straight line flows yield fast inference Following Algorithm 1, denote by $\boldsymbol{Z} = \texttt{RectFlow}((X_0, X_1))$ the rectified flow induced from (X_0, X_1). Applying this operator recursively

Algorithm 1 Rectified Flow: Main Algorithm

Procedure: $\boldsymbol{Z} = \texttt{RectFlow}((X_0, X_1))$:

Inputs: Draws from a coupling (X_0, X_1); velocity model $v_\theta \colon \mathbb{R}^d \to \mathbb{R}^d$ with parameter θ.

Training: $\hat{\theta} = \arg\min_{\theta} \mathbb{E}\left[\|X_1 - X_0 - v(tX_1 + (1-t)X_0,\ t)\|^2\right]$, with $t \sim \text{Uniform}([0,1])$.

Sampling: Draw (Z_0, Z_1) from $\mathrm{d}Z_t = v_{\hat{\theta}}(Z_t, t)\mathrm{d}t$ with $Z_0 \sim \pi_0$ (or backwardly $Z_1 \sim \pi_1$).

Return: $\boldsymbol{Z} = \{Z_t \colon t \in [0,1]\}$.

Reflow (optional): $\boldsymbol{Z}^{k+1} = \texttt{RectFlow}((Z_0^k, Z_1^k))$, starting from $(Z_0^0, Z_1^0) = (X_0, X_1)$.

Distill (optional): Learn a neural network $\hat{T}$ to distill the k-rectified flow, such that $Z_1^k \approx \hat{T}(Z_0^k)$.

yields a sequence of rectified flows $\boldsymbol{Z}^{k+1} = \texttt{RectFlow}((Z_0^k, Z_1^k))$ with $(Z_0^0, Z_1^0) = (X_0, X_1)$, where $\boldsymbol{Z}^k$ is the k-th rectified flow, or simply k*-rectified flow*, induced from (X_0, X_1). This *reflow* procedure not only decreases transport cost, but also has the important effect of straightening paths of rectified flows, that is, making the paths of the flow more straight. This is highly attractive computationally as flows with nearly straight paths incur small time-discretization error in numerical simulation. Indeed, perfectly straight paths can be simulated exactly with a single Euler step and is effectively a one-step model, addressing the very bottleneck of slow inference of ODE/SDE models.

3 Experiments

Unconditional Image Generation on CIFAR-10 Among one-step generative models, the distilled 2-rectified flow achieves an FID of 4.85, beating the best known one-step generative model with U-net architecture, 8.91 (TDPM, Table 1 (b)). The recalls of both 2-rectified flow (0.50) and 3-rectified flow (0.51) outperform the best known results of GANs (0.49 from StyleGAN2+ADA) showing an advantage in diversity.

Image-to-Image Translation As shown in Figure 3, rectified flow can effectively transfer images between different domains in a unsupervised manner, without the burdensome cycle consistency regularization in CycleGAN [21].

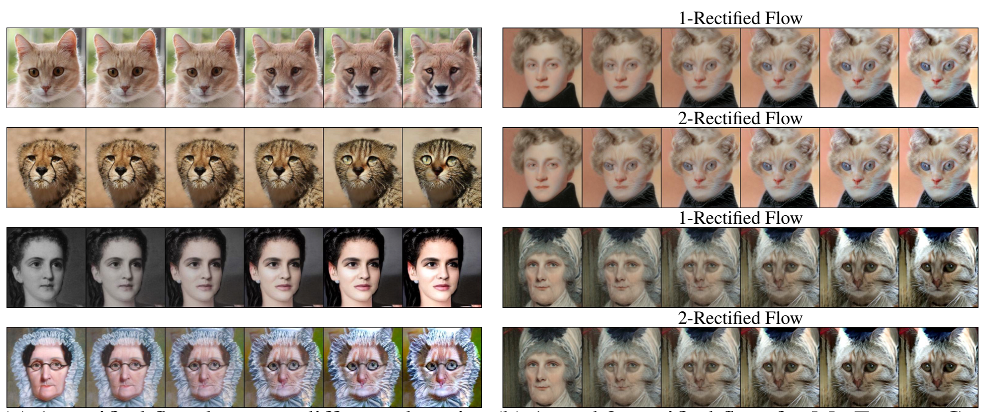
Method	NFE(↓)	IS (↑)	FID (↓)	Recall (↑)
ODE	*One-Step Generation (Euler solver, N=1)*			
1-Rectified Flow (+*Distill*)	1	1.13 (***9.08***)	378 (*6.18*)	0.0 (*0.45*)
2-Rectified Flow (+*Distill*)	1	8.08 (*9.01*)	12.21 (***4.85***)	0.34 (*0.50*)
3-Rectified Flow (+*Distill*)	1	8.47 (*8.79*)	8.15 (*5.21*)	0.41 (***0.51***)
VP ODE [16] (+*Distill*)	1	1.20 (*8.73*)	451 (*16.23*)	0.0 (*0.29*)
sub-VP ODE [16] (+*Distill*)	1	1.21 (*8.80*)	451 (*14.32*)	0.0 (*0.35*)
ODE	*Full Simulation (Runge–Kutta (RK45), Adaptive N)*			
1-Rectified Flow	127	**9.60**	**2.58**	**0.57**
2-Rectified Flow	110	9.24	3.36	0.54
3-Rectified Flow	104	9.01	3.96	0.53
VP ODE [16]	140	9.37	3.93	0.51
sub-VP ODE [16]	146	9.46	3.16	0.55
SDE	*Full Simulation (Euler solver, N=2000)*			
VP SDE [16]	2000	9.58	2.55	0.58
sub-VP SDE [16]	2000	9.56	2.61	0.58

(a) Results using the DDPM++ architecture.

Method	NFE(↓)	IS (↑)	FID (↓)	Recall (↑)
GAN	*One-Step Generation*			
SNGAN [12]	1	8.22	21.7	0.44
StyleGAN2 [7]	1	9.18	8.32	0.41
StyleGAN-XL [14]	1	-	1.85	0.47
StyleGAN2 + ADA [7]	1	9.40	2.92	0.49
StyleGAN2 + DiffAug [19]	1	9.40	5.79	0.42
TransGAN + DiffAug [6]	1	9.02	9.26	0.41
GAN with U-Net	*One-step Generation*			
TDPM (T=1) [20]	1	8.65	8.91	0.46
Denoising Diffusion GAN (T=1) [18]	1	8.93	14.6	0.19
ODE	*One Step Generation (Euler solver, N=1)*			
DDIM Distillation [10]	1	8.36	9.36	0.51
NCSN++ (VE ODE) [16] (+*Distill*)	1	1.18 (*2.57*)	461 (*254*)	0.0 (*0.0*)
ODE	*Full Simulation (Runge–Kutta (RK45), Adaptive N)*			
NCSN++ (VE ODE) [16]	176	9.35	5.38	0.56
SDE	*Full Simulation (Euler solver)*			
DDPM [5]	1000	9.46	3.21	0.57
NCSN++ (VE SDE) [16]	2000	9.83	2.38	0.59

(b) Recent results with different architectures reported in literature.

Table 1: Results on CIFAR10 unconditioned image generation. Fréchet Inception Distance (FID) and Inception Score (IS) measure the quality of the generated images, and recall score [9] measures diversity. The number of function evaluation (NFE) denotes the number of times we need to call the main neural network during inference. It coincides with the number of discretization steps N for ODE and SDE models.



(a) 1-rectified flow between different domains (b) 1- and 2-rectified flow for MetFace $\rightarrow$ Cat.

Figure 3: Samples of trajectories z_t of 1- and 2-rectified flow for transferring between different domains.

References

[1] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-DPM: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *arXiv preprint arXiv:2201.06503*, 2022.

[2] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[3] R Flamary, N Courty, D Tuia, and A Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell*, 1, 2016.

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[6] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. TransGAN: Two pure transformers can make one strong GAN, and that can scale up. *Advances in Neural Information Processing Systems*, 34, 2021.

[7] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[9] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019.

[10] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

[11] Zhaoyang Lyu, Xudong Xu, Ceyuan Yang, Dahua Lin, and Bo Dai. Accelerating diffusion models via early stop of the diffusion process. *arXiv preprint arXiv:2205.12524*, 2022.

[12] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[13] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[14] Axel Sauer, Katja Schwarz, and Andreas Geiger. StyleGAN-XL: Scaling StyleGAN to large diverse datasets. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings*, pages 1–10, 2022.

[15] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

[16] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

[17] Giulio Trigila and Esteban G Tabak. Data-driven optimal transport. *Communications on Pure and Applied Mathematics*, 69(4):613–648, 2016.

[18] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion GANs. *arXiv preprint arXiv:2112.07804*, 2021.

[19] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient GAN training. *Advances in Neural Information Processing Systems*, 33: 7559–7570, 2020.

[20] Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Truncated diffusion probabilistic models. *arXiv preprint arXiv:2202.09671*, 2022.

[21] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.