# Learning Straight Flows by Learning Curved Interpolants

**Shiv Shankar**
University of Massachusetts
sshankar@cs.umass.edu

**Tomas Geffner**
NVIDIA
tgeffner@nvidia.com

## Abstract

Flow matching models typically use linear interpolants to define the forward/noise addition process. This, together with the independent coupling between noise and target distributions, yields a vector field which is often non-straight. Such curved fields lead to a slow inference/generation process. In this work, we propose to learn flexible (potentially curved) interpolants in order to learn straight vector fields to enable faster generation. We formulate this via a multi-level optimization problem and propose an efficient approximate procedure to solve it. Our framework provides an end-to-end and simulation-free optimization procedure, which can be leveraged to learn straight line generative trajectories.

## 1 Introduction

In recent years, neural generative models have become remarkably successful in a variety of domains, including computer vision (Xing et al., 2023), data generation (Karras et al., 2020), robotics (Firoozi et al., 2023), and scientific applications (Guo et al., 2024). This success can be broadly attributed to the development of generative models based on learning vector fields (Ho et al., 2020; Yang et al., 2023b; Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022). Among these methods, flow matching (Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022) is a recent simulation-free method that has gained traction due to its robust performance. Flow matching is a version of Continuous Flows (CNFs) (Chen et al., 2018) that uses deep neural networks to learn a velocity vector field that transports samples from a source distribution to the target distribution. Peluchetti (2022); Lipman et al. (2022) show that a regression objective on conditional probability paths between source and target samples can be used to learn a model of the velocity field without an explicit target field.

Despite their success, flow-matching models (as well as other vector-field-based models) often suffer from slow sampling. Specifically, one has to simulate an ODE (or SDE) with a numerical solver to generate samples. Since these models learn a vector field that needs to be numerically integrated, the overall process can be slow and rife with discretization errors (Song et al., 2023). This is because the learnt vector fields often have curved trajectories, and therefore small discretization step-sizes are needed for accurate simulation.

To alleviate this issue, many methods have been proposed to learn more easily integrable vector fields. One common approach tries to incorporate 'straightness' to the field (Liu et al., 2022; 2023; Kornilov et al., 2024; Tong et al., 2023), as straight paths allow for 1-step and discretization-free inference. While few proposals learn straight approximations to the true field (Yang et al., 2024; Zhang et al., 2024; Lee et al., 2023), most of these share a common flaw. Most of these these methods use linear conditional paths and independent coupling during training. While easy to implement, the underlying vector field in these scenarios is fundamentally curved, and any 'straight' version will be necessarily approximate.

In this work, we propose a new method to learn straight flows using flow matching by tuning interpolants. We propose explicitly training the conditional probability paths (or interpolants) used in flow matching to enforce straightness in the velocity field. While mathematically simple, the corresponding bi-level optimization problem is intractable. We address this by proposing an approximate approach to solve this optimization problem, which relies on an an analytic form for the target vector field of a Conditional Flow Matching objective (Lipman et al., 2022), and enforcing straightness on the target field tuning the interpolant. Empirically, we observe that our method outperforms recent models on standard datasets.

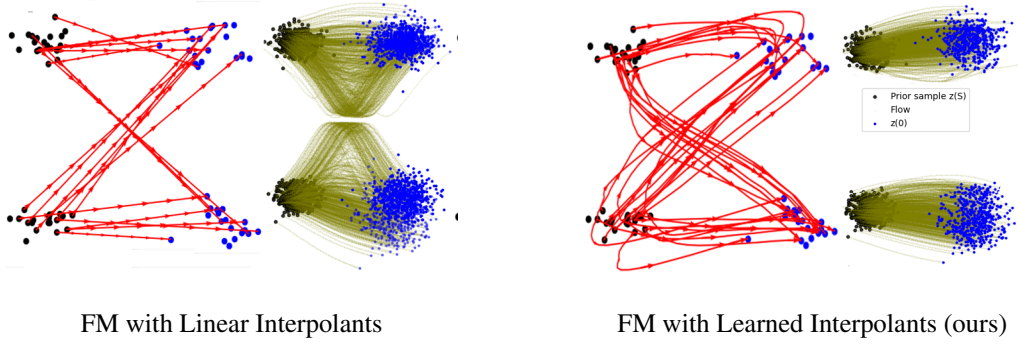| FM with Linear Interpolants | FM with Learned Interpolants (ours) |

Figure 1: Illustration of training difference flow matching (FM) (Lipman et al., 2022) and our method on an illustrative 2 gaussian (source, black) to 2-gaussian (target, blue) problem. On the left we have standard FM with linear interpolants (red lines) and the resulting flow vector. On the right we show our approach. By allowing non-linear interpolants we are able to learn a flow velocity field with significantly less curvature.

**Contributions** (a) We propose a new bi-level formulation to learn straight flows that explicitly forces straightness by learning interpolants, (b) We derive an analytic form of target vector field that allows training interpolants without differentiable optimization methods, (c) We present scalable parametric models for conditional paths that enable efficient training, (d) We show significant improvements for low-shot (i.e., low number of function evaluations) generation quality.

## 2 PRELIMINARIES

Let $p_0$ and $p_1$ be two distributions on $\mathbb{R}^d$. Usually, $p_1$ is our target distribution, only known through samples, and $p_0$ is the source distribution, usually chosen to be tractable (e.g., a Gaussian). Our goal is to generate samples from $p_1$. One way to generate these samples is by transporting the initial distribution $p_0$ via a vector field to the target $p_1$. Specifically, consider a vector field $u(t, x) : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d$ such that the differential equation

$$\frac{d\gamma_x(t)}{dt} = u(t, \gamma_x(t)), \qquad \gamma_x(0) \sim p_0 \tag{1}$$

produces sample from $p_1$. Under some topological constraints on the distributions, the existence of such fields is well known (Ambrosio & Crippa, 2014). For a given starting sample $x$, we denote the solution of Equation (1) as $\gamma_x(t)$. This solution, also called a flow, is the trajectory of point $x$ as it evolves under $u$. We denote by $p_t$ the probability distribution obtained at time $t$ by moving the samples of $p_0$ via the field $u$.

### FLOW MATCHING

Flow matching (FM) (Lipman et al., 2022) is a simulation-free method for learning one such vector field using a neural network $v_\theta(t, x)$. It does that by optimizing the FM objective

$$\mathcal{L}_{FM} = E_{t,p_t} ||v_\theta(t, x_t) - u(t, x_t)||_2^2. \tag{2}$$

Unfortunately, this objective is not tractable given only the source and target distributions, as the target vector field $u(t, x_t)$ is unknown. Lipman et al. (2022) show that one can optimize the conditional flow matching objective instead

$$\mathcal{L}_{CFM} = E_{t,q(z)} E_{p_t(x_t|z)} ||v_\theta(t, x_t) - u(t, x_t|z)||_2^2, \tag{3}$$

where $u(t, x_t|x_1)$ is a conditional vector field and corresponding probability path $p_t(x_t|z)$. Lipman et al. (2022) considered $z$ to be a sample from the target $x_1$. While Lipman et al. (2022) considered a family linear Gaussian paths, more generalized variants of this problem have been proposed (Tong et al., 2023). Any suitable conditioning variable $z$ can be chosen if the objective remains tractable

(Pooladian et al., 2023; Tong et al., 2023). A related problem is that of the Schrodinger Bridge (De Bortoli et al., 2021; Wang et al., 2021), which seeks the vector field whose probability law is close to that of the standard Brownian diffusion process.

# 3 LEARNING STRAIGHT FLOWS BY TUNING INTERPOLANTS

In this section we describe our method to learn straight flows. We start with the standard CFM objective, but introduce parametric interpolants (parameterized by $\phi$), instead of the linear ones often used in flow matching. Once we do so, the learned flow model $v_\theta$ becomes dependent on the interpolant parameters $\phi$. We thus formulate the problem of learning straight flows as a bilevel optimization problem, where the interpolants are tuned to optimize the straightness of $v_\theta$ (Section 3.1). Next, we show how one can solve this bilevel optimization without using differentiable optimization methods. For this purpose, we derive an analytic expression for the flow field in terms of the parameters $\phi$ (Section 3.2) together with a measure of the flow's straightness (Section 3.3). Finally, we describe a specific family of interpolants which enables scalable computation with high-dimensional datasets (Section 3.4).

## 3.1 BI-LEVEL FORMULATION FOR STRAIGHT FLOWS

We consider the commonly used CFM objective with $z = (x_0, x_1)$ (Tong et al., 2023). While CFM models often use simple linear interpolants, the consistency of the CFM approach only requires smooth enough conditional fields. Crucially the linear interpolant $x_t$ can be replaced by any other interpolant (stochastic or otherwise), as long as the conditional $u$ (Equation (3)) is modified to be the corresponding tangent vector. Specifically, choosing the interpolant to be $x_t = \phi_{t,x_1}(x_0)$ we can rewrite the CFM objective from Equation (3) as

$$E_{t,p_0(x_0),p_1(x_1)}||v_\theta(t, x_t) - \partial_t \phi_{t,x_1}(x_0)||_2^2, \tag{4}$$

which recovers standard CFM for $\phi := tx_1 + (1-t)x_0$.

We propose learning straight flows by choosing $\phi$ such that the resulting vector field $v$ is as straight as possible. We formulate this as a bi-level optimization problem

$$\min_\phi |v_{\theta^*}(.; \phi)|_{\text{Straight}} \qquad \text{s.t.} \tag{5}$$

$$v_{\theta^*} = \arg\min_\theta E_{t,p_0(x_0),p_1(x_1)}||v_\theta(t, x_t; \phi) - \partial_t \phi_{t,x_1}(x_0)||_2^2,$$

where $|\cdot|_{\text{Straight}}$ is a measure of straightness of the vector field. As an example, previous work by Pooladian et al. (2023); Liu et al. (2022) suggest measuring curvature using

$$|v|_{\text{Straight}} = \mathbb{E}_{\substack{t \sim U(0,1) \\ x_0 \sim p_0}} \left[ ||v_t(\gamma_{x_0}(t))||^2 - ||\gamma_{x_0}(1) - x_0||^2 \right],$$

where $\gamma$ is the flow, that is, the solution to Equation 1.

**Remark.** *Note that $v_\theta$ is a function of $\theta$, but also has an implicit dependence on $\phi$. Due to the inner optimization, the optimal parameter $\theta^*$ depends on $\phi$. We made this dependence explicit in Equation (5).*

While mathematically straightforward, solving the optimization problem from Equation (5) is computationally intensive. This is a bi-level problem, where each inner objective requires solving a FM objective. Our insight is that one is not required to solve the FM objective to get $v_{\theta^*}$, and can instead leverage the optimal vector field $v_{\theta*}$ in a different manner directly in terms of the interpolant functions $\phi$.

## 3.2 REFORMULATING FLOW MATCHING

This section shows how to connect $v_\theta$ to the interpolant function $\phi$ without solving the inner optimization iteratively, by presenting an analytic form for the optimal $v_\theta^*$ (Equation (6)) in terms of $\phi$. We will then use this result in Section 3.3 and Section 3.4 to derive our final optimization algorithm.

**Proposition 1.** *Let $\mathcal{J}$ be the determinant of the Jacobian of the interpolant's inverse, i.e., $\left|\frac{d\phi_{t,x_1}^{-1}(x)}{dx}\right|_\Delta$. The optimal velocity field for an interpolant $\phi$ is given by*

$$v_\phi^*(t, x_t) = \int \frac{\partial_t \phi_{t,x_1}(\phi_{t,x_1}^{-1}(x_t)) p_0(\phi_{t,x_1}^{-1}(x_t)) |\mathcal{J}| p_1(x_1) dx_1}{\int p_0(\phi_{t,x_1}^{-1}(x_t)) |\mathcal{J}| p_1(x_1) dx_1}. \tag{6}$$

Specific simple parameterisations, like paths of the form $x_0 = \frac{x_t - g(t)x_1}{1-h(t)}$ with the coefficients $g, h$ being non-linear in $t$, yield efficient versions of Equation (6). For such paths, Equation (6) simplifies as the Jacobians do not depend on $x_1$. However, such parameterisations, while simple to use, may not be powerful enough for complex datasets.

While exact, Equation (6) is intractable, as it involves complex integrals. In practice, our final algorithm will leverage empirical estimates for this quantity using samples from $p(x_1)$, and flexible interpolants $\phi$ parameterized by neural networks, as explained in Section 3.4.

### 3.2.1 DERIVATION OF EQUATION (6)

A key challenge in analyzing the FM objective is that both terms involved in the $L2$ loss (Equation (4)) depend on $x_0$ and $x_1$. This can be addressed by writing out the distribution over $x_t$ explicitly

$$E_{t,p_0(x_0),p_1(x_1)} ||v_\theta(t, x_t) - \partial_t \phi_{t,x_1}(x_0)||_2^2 \tag{7}$$

$$= E_{t,p(x_0,x_1)} ||v_\theta(t, x_t) - \partial_t \phi_{t,x_1}(x_0)||_2^2 \tag{8}$$

$$= E_{t,p_t(x_t),p(x_0,x_1|x_t,t)} ||v_\theta(t, x_t) - \partial_t \phi_{t,x_1}(x_0)||_2^2, \tag{9}$$

where it can be observed that the global minimizer is

$$v^*(t, x_t) = \mathbb{E}_{p(x_0,x_1|x_t,t)}[\partial_t \phi_t(x_0, x_1)]. \tag{10}$$

This expression can also be directly obtained from the generalized CFM objective (Pooladian et al., 2023), using $z = x_t$.

While simple, Equation (10) cannot be used in practice since we do not have access to the required distributions. For instance, $x_1$ is only available via samples, and the posterior $p(x_0, x_1|x_t)$ is intractable. This can be addressed by some analysis and a few judicious choices conditional paths. First we write the conditional distribution as:

$$p(x_0, x_1|x_t, t) = \frac{p(x_0, x_1, x_t|t)}{p(x_t|t)} = \frac{p(x_0, x_1, x_t|t)}{\int p(x_0, x_1, x_t|t) dx_0 dx_1}. \tag{11}$$

Specifically, using $x_t = \phi_{t,x_1}(x_0)$, Equation (11) becomes

$$p(x_0, x_1|x_t, t) = \frac{p(x_0, x_1, \phi_{t,x_1}(x_0)|t)}{\int p(x_0, x_1, \phi_{t,x_1}(x_0)|t) dx_0 dx_1}. \tag{12}$$

Since the maps are deterministic, we get

$$p(x_0, x_1 \mid x_t, t) = \frac{\delta(\tilde{x}_t - x_t) p_0(x_0) p_1(x_1)}{p(x_t|t)}, \tag{13}$$

where $\tilde{x}_t = \phi_{t,x_1}(x_0)$ is an auxiliary variable introduced to distinguish the conditioning variable from the conditioned value, $\delta$ is the Dirac function (Dirac, 1981), and $p(x_t)$ is the marginal likelihood (normalization constant), given by

$$p(x_t|t) = \int p(x_0, x_1, x_t|t) dx_0 dx_1 = \int \delta(\tilde{x}_t - x_t) p_0(x_0) p_1(x_1) dx_0 dx_1. \tag{14}$$

While the above integral can be estimated with infinite data, the expression is not directly practically useful, since the estimator will often just be zero when relying on empirical samples (due to the Dirac function). However, a useful expression can be obtained noting that the Dirac delta function allows us to integrate over $x_0$ using the change in variables formula (Halperin & Schwartz, 1952), yielding

$$p(x_t|x_1, t) = p_0(\phi_{t,x_1}^{-1}(x_t)) |\mathcal{J}|, \tag{15}$$

where $\mathcal{J}$ is the determinant of the Jacobian of the interpolant's inverse, i.e., $\left|\frac{d\phi_{t,x_1}^{-1}(x)}{dx}\right|_\Delta$. Substituting this in Equation (14) and using the definition of $v^*$ (the expectation of $\partial_t \phi_{t,x_1}$ under $p(x_0, x_1|x_t)$) yields Equation (6).

### 3.3 Straightening the Flow

Equation (6) yields powerful way to manipulate the target velocity field $v^*$ to enforce desired properties on the flow. When we want the flows to be straight, the vector field $v^*$ should be constant everywhere on a trajectory. This implies that a straight velocity field $u_{\text{str}}$ must satisfy

$$u_{\text{str}}(t, \gamma_{x_0}(t)) = u_{\text{str}}(s, \gamma_{x_0}(s)) = u_{\text{str}}(0, \gamma_{x_0}(0)) \ \forall s, t,$$

where $\gamma_{x_0}$ is the flow generated by $u_{\text{str}}$ from point $x_0$. Since the vector field is constant, we can differentiate the above expression with respect to $t$ to get

$$\partial_t u_{\text{str}} = -\nabla_x u_{\text{str}} \cdot u_{\text{str}} \tag{16}$$

for all points on the trajectory, where $\cdot$ refers to the matrix-vector product. Therefore, since we know the optimal field is given by $v^*$, we can learn straight flows by optimizing

$$\phi^* = \arg \min_\phi ||\partial_t v_\phi^* + \nabla_x v_\phi^* \cdot v_\phi^*||^2. \tag{17}$$

We note here that this criteria for straight flows has been observed before Liu et al. (2022); Yang et al. (2024).

While the above expression is correct, optimizing it is computationally intensive, as we need to differentiate through $v_\phi^*$, which in turns requires the Jacobian of $\phi_{t,x_1}^{-1}$. However, using a flexible neural network for $v_\theta$, under ideal training we expect $v_\theta \approx v_\phi^*$. Therefore, we propose to replace the derivatives of $v^*$ with derivatives of $v_\theta$, which yields

$$\min_{\theta,\phi} \mathbb{E}_t \mathbb{E}_{x_0,x_1} ||v_\theta(x_t, t) - \text{sg}(\partial_t \phi_{t,x_1}(x_0))||^2 + \lambda ||\partial_t v_\theta(x_t', t) + \nabla_x v_\theta(x_t', t) \cdot v^*(x_t', t)||^2. \tag{18}$$

Here "sg" refers to stop gradient and $\lambda > 0$ is a hyper-parameter. We suppress the dependence of $v^*$ on $\phi$ for notational convenience.

### 3.4 Scalable Computation

While Equation (18) is 'analytical', in general, it has poor scaling in the dimensionality of the output, specifically due to the presence of the inverse-derivative as well as the Jacobian, both of which do not generally scale with the dimension of the data. Additionally, we need to ensure that the function $\phi_{t,x_1}$ remains invertible. Fortunately, researchers have developed models that are amenable to these considerations. Specifically, the literature on normalising flows and invertible models (Papamakarios et al., 2021) has proposed several families of expressive neural networks that support such operations. In this work we use GLOW/1x1 convolution model (Kingma & Dhariwal, 2018) to parameterise the function $\phi$. This family of neural networks learns any linear transform's parameters in the "PLU" format. It initializes any matrix parameter $W$ of a linear layer, finds its PLU decomposition, fixes $P$ and optimizes the lower and upper diagonal matrices $L$ and $U$. Normalizations and activations are also chosen in a way that ensures easy inversion and memory efficiency.

**Computing Expectations** The expression for $v^*$ from Equation (6) requires computing expectations (or integrals) over the data distribution $p(x_1)$. Since $p_0$ is Gaussian (or a similar tractable distribution), the value of $p_0(\phi_{t,x_1}(x_t))$ can be computed exactly. Therefore, one can estimate both the numerator and the denominator as empirical expectations over the target distribution. The resulting estimator is consistent, with the bias going to zero as we aggregate more data samples. Additionally, this estimation process, which combines vector fields from multiple paths at the same time, often produces a lower variance estimate of the optimal $v^*$. In fact, Xu et al. (2023) used this idea to reduce the variance of the gradient estimates used to train diffusion models, observing that it often led to more robust optimization, thanks to the variance reduction, despite the bias introduced.

## 4 Related Work

Multisample FM (Pooladian et al., 2023) proposes to generalize the independent coupling of the data distribution $p_1(x_1)$ and prior distribution $p_0(x_0)$ to the optimal transport coupling plan $\pi(x_0, x_1)$.

---

**Algorithm 1** Vector field model training algorithm

---

**Require:** Sampler for $p_0$ (usually Gaussian), Empirical samples from $p_1$, batch size $N$, averaging size $M (\geq N)$ (to estimate Equation (6)); models $v_\theta(x, t)$ and $\phi$
1: **while** not converged **do**
2:     Sample $N$ points $\{t^i\}_{i=1}^N$ from $\mathcal{U}[0, 1]$
3:     Sample $N$ pairs $\{x_0^i, x_1^i\}_{i=1}^n$ from $p(x_0, x_1) = p_0(x_0)p_1(x_1)$
4:     Sample $M - N$ points $\{\hat{x}_1^j\}_{i=N}^{M-N}$ from $p_1$
5:     Compute $N$ interpolants $x_t^i$ from $(t^i, x_0^i, x_1^i)$
6:     Estimate $v^*(x_t^i)$ (Eq. 6) replacing integrals $\int f(x_t, x_1, t)p_1(x_1)dx_1$ by empirical estimates $\frac{1}{M}\sum_{j=1}^M f(x_t^i, x_1^j, t^i)$
7:     Calculate the empirical loss using Equation (18)
8:     Update parameters $\theta, \phi$ (e.g., using SGD or Adam Kingma (2014))
9: **end while**

---

Under the optimal transport plan, the learned trajectory of ODE are straight (Pooladian et al., 2023). However, this requires constructing the optimal transport plan for the data which is computationally prohibitive. Minibatch OT and Minibatch Sinkhorn coupling have been suggested to lower the cost of computing such couplings (Tong et al., 2023).

Liu et al. (2022) suggested a rectified flow matching method which uses a pretrained FM to learn a straighter approximation. Based on this insight, other methods to learn straight flows have also been proposed (Liu et al., 2022; 2023; Lee et al., 2023). These methods however are not simulation-free (requiring sampling during training) and often require iteratively distilling from flows, both of which are computationally intensive.

**Non-Linear Interpolants**   Rectified and distilled flow methods (Lee et al., 2023; Liu et al., 2022) rely on the coupling given by a "teacher" (pre-trained) flow. These methods often rely on linear interpolants, and require multiple training stages, where each stage requires generating training pairs of noise and data samples by simulating the ODE with the vector field learnt in the previous stage. Our method, on the other hand, is simulation-free and uses the independent coupling, though with learned interpolants. Kapuśniak et al. (2024) also proposed using non-linear interpolants. They learn an interpolant such that the conditional paths stay close to a manifold, but do not directly optimize any property of $v^*$. Bartosh et al. (2024) proposed learning the forward process in diffusion models, akin to learning the interpolant in flow matching, and also rely on interpolants parameterized via neural networks and adaptations of normalizing flows. While closely related to our method, the approach proposed in prior work differs in its formulation for learning the interpolants. Specifically, they employ a specific parameterization for both the interpolant $\phi$ and the flow model $v_\theta$, which share parameters. This design choice, can limit the flexibility of the model. Additionally, since they do not use a bi-level objective, the optimization procedure does not account for the relationship between the optimal $\theta$ and optimal $\phi$. In other words, the dependence of $\theta$ on $\phi$ is ignored. Our method does not have these restrictions. Empirically, we observe that our approach achieves improved performance (Section 5), highlighting the benefits of explicitly modeling this relationship.

**Consistency Models**   This family of models (Yang et al., 2024; 2023a; Song et al., 2023) aims to learn a function that directly solves Equation (1). One can choose to solve the ODE numerically and learn the function using a distilled dataset consisting of pairs of noise and the corresponding data-samples, though more common and scalable approaches use a simulation free approach. We solve the problem of learning straight fields, which is orthogonal to the problem that consistency models aim to solve, which involves learning a function that correctly integrates the vector field. In principle, these ideas could be combined to produce more efficient sampling methods.

## 5 EXPERIMENTS

We conduct two sets of experiments one on low-resolution datasets, including CIFAR-10 (Alex, 2009) and ImageNet 32x32 (Chrabaszcz et al., 2017), and another on higher-resolution datasets, namely CelebA-HQ (Karras et al., 2017) and AFHQ-Cat (Choi et al., 2020). Following the methodology of

| CIFAR-10 | ImageNet 32x32 | CelebA 256x256 |

Figure 2: Generated samples from CIFAR, ImageNet and CelebA.

Song et al. (2023), we evaluate the model across varying numbers of function evaluations (NFE). The flow field is learned using a U-Net architecture based on DDPM++ (Song et al., 2020). To assess the quality of generated images, we employ the Fréchet Inception Distance (FID) score (Heusel et al., 2017).

**Baselines**   We follow Song et al. (2023); Yang et al. (2024) and compare our method against several baselines comprising of representative diffusion models and flow models. The baseline models include Consistency Models (Song et al., 2023), Rectified Flow (Liu et al., 2022), Rectified Flow with Bellman Sampling (Nguyen et al., 2024), Neural Flow Diffusion Models (Bartosh et al., 2024), and Consistency-FM (Yang et al., 2024). We did not run these baselines ourselves and have reported results from literature. Since not all earlier works have reported results on all the datasets, for each dataset the set of baselines is not always identical.

**Results** The results for the CIFAR dataset are presented in Table 1. Our method demonstrates superior performance compared to models such as Consistency FM (Yang et al., 2024), Rectified Flow (Liu et al., 2022), and Consistency Model (Song et al., 2023). We also see that our model matches or outperforms mainstream diffusion models while using a low number of function evaluations (NFE).

Table 3 shows results on the ImageNet 32x32 dataset, where we compare against plain flow matching (Lipman et al., 2022) (with a large number of NFEs), multisample flow matching (Pooladian et al., 2023), and Neural Flow Diffusion models (Bartosh et al., 2024). Our approach yields better performance than competing approaches using a low number of NFEs.

Table 1: Comparison with baseline models on CIFAR-10. Results for other models are obtained from previous work.

| Method | NFE ($\downarrow$) | FID ($\downarrow$) |
|---|---|---|
| Score SDE Song et al. (2020) | 2000 | 2.20 |
| DDPM Ho et al. (2020) | 1000 | 3.17 |
| LSGM Vahdat et al. (2021) | 147 | 2.10 |
| PFGM Xu et al. (2022) | 110 | 2.35 |
| EDM Karras et al. (2022) | 35 | **2.04** |
| 1-Rectified Flow /ReFlowLiu et al. (2022) | 1 | 378 |
| Glow Kingma & Dhariwal (2018) | 1 | 48.9 |
| Residual Flow Chen et al. (2019) | 1 | 46.4 |
| GLFlow Xiao et al. (2019) | 1 | 44.6 |
| DenseFlow Grcić et al. (2021) | 1 | 34.9 |
| Consistency Model (Song et al., 2023) | 2 | 5.83 |
| Consistency Flow Matching (Yang et al., 2024) | 2 | 5.34 |
| Ours | 2 | **4.61** |

We further evaluate our method on high-resolution image generation tasks, specifically $256{\times}256$ images from AFHQ-Cat and CelebA-HQ. Following Yang et al. (2024), we compare against baseline methods, including Consistency FM (Yang et al., 2024), ReFlow (Liu et al., 2022), and ReFlow with Bellman sampling (Nguyen et al., 2024). All baseline results are taken from Yang et al. (2024). Our method outperforms baseline approaches such as Rectified Flow (Liu et al., 2022) and Rectified Flow with Bellman sampling (Nguyen et al., 2024) by a significant margin.

Table 2: Comparison with flow matching models on CelebA.

| Method | NFE ($\downarrow$) | FID ($\downarrow$) |
|---|---|---|
| ReFlow (Liu et al., 2022) | 8 | 109.4 |
| | 6 | 127.0 |
| ReFlow + Bellman Sampling (Nguyen et al., 2024) | 8 | 49.8 |
| | 6 | 72.5 |
| Consistency Flow Matching (Yang et al., 2024) | 6 | 36.4 |
| Ours | 6 | **28.6** |

Table 3: Comparison with FM models on on ImageNet-32x32.

| Method | NFE ($\downarrow$) | FID ($\downarrow$) |
|---|---|---|
| Flow Matching (Lipman et al., 2022) | 120 | 5.0 |
| MultiSample FM (Pooladian et al., 2023) | 4 | 17.3 |
| | 12 | 7.2 |
| NFDM (Bartosh et al., 2024) | 4 | 6.1 |
| | 12 | 4.1 |
| Ours | 4 | **5.58** |
| | 12 | **3.84** |

## 6 CONCLUSION

We propose a novel approach to learn flow matching vector fields that; unlike existing methods, which try to learn straight approximations to a curved vector field; learns a straight vector field directly. We use non-linear interpolants in the CFM objective and show how one can optimize the corresponding vector field solutions. In the process, we provide analytical expressions for the general solution to a CFM model and show how it can be tuned to adjust the "straightness" of the vector field. We present a way to parametrize the interpolants using a GLOW model, allowing fast inversion and determinant computations. Our approach outperforms recent methods when using a low number of NFEs.

## REFERENCES

Albergo, M. S. and Vanden-Eijnden, E. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2022.

Alex, K. Learning multiple layers of features from tiny images. *https://www. cs. toronto. edu/kriz/learning-features-2009-TR. pdf*, 2009.

Ambrosio, L. and Crippa, G. Continuity equations and ode flows with non-smooth velocity. *Proceedings of the Royal Society of Edinburgh Section A: Mathematics*, 144(6):1191–1244, 2014.

Bartosh, G., Vetrov, D., and Naesseth, C. A. Neural flow diffusion models: Learnable forward process for improved diffusion modelling. *arXiv preprint arXiv:2404.12940*, 2024.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Chen, R. T., Behrmann, J., Duvenaud, D. K., and Jacobsen, J.-H. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

Choi, Y., Uh, Y., Yoo, J., and Ha, J.-W. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8188–8197, 2020.

Chrabaszcz, P., Loshchilov, I., and Hutter, F. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34: 17695–17709, 2021.

Dirac, P. A. M. *The principles of quantum mechanics*. Number 27. Oxford university press, 1981.

Firoozi, R., Tucker, J., Tian, S., Majumdar, A., Sun, J., Liu, W., Zhu, Y., Song, S., Kapoor, A., Hausman, K., et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, pp. 02783649241281508, 2023.

Grcić, M., Grubišić, I., and Šegvić, S. Densely connected normalizing flows. *Advances in Neural Information Processing Systems*, 34:23968–23982, 2021.

Guo, Z., Liu, J., Wang, Y., Chen, M., Wang, D., Xu, D., and Cheng, J. Diffusion models in bioinformatics and computational biology. *Nature reviews bioengineering*, 2(2):136–154, 2024.

Halperin, I. and Schwartz, L. *Introduction to the Theory of Distributions*. University of Toronto Press, 1952.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *NeurIPS*, volume 33, pp. 6840–6851, 2020.

Kapuśniak, K., Potaptchik, P., Reu, T., Zhang, L., Tong, A., Bronstein, M., Bose, A. J., and Di Giovanni, F. Metric flow matching for smooth interpolations on the data manifold. *arXiv preprint arXiv:2405.14780*, 2024.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of stylegan. In *CVPR*, pp. 8110–8119, 2020.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

Kingma, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Kornilov, N., Gasnikov, A., and Korotin, A. Optimal flow matching: Learning straight trajectories in just one step. *arXiv preprint arXiv:2403.13117*, 2024.

Lee, S., Kim, B., and Ye, J. C. Minimizing trajectory curvature of ode-based generative models. In *International Conference on Machine Learning*, pp. 18957–18973. PMLR, 2023.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2022.

Liu, X., Gong, C., et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2022.

Liu, X., Zhang, X., Ma, J., Peng, J., and Liu, Q. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. *arXiv preprint arXiv:2309.06380*, 2023.

Nguyen, B., Nguyen, B., and Nguyen, V. A. Bellman optimal stepsize straightening of flow-matching models. In *The Twelfth International Conference on Learning Representations*, 2024.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22 (57):1–64, 2021.

Peluchetti, S. Non-denoising forward-time diffusions, 2022. URL https://openreview.net/forum?id=oVfIKuhqfC.

Pooladian, A.-A., Ben-Hamu, H., Domingo-Enrich, C., Amos, B., Lipman, Y., and Chen, R. T. Multisample flow matching: Straightening flows with minibatch couplings. 2023.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. In *International Conference on Machine Learning*, pp. 32211–32252. PMLR, 2023.

Tong, A., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., FATRAS, K., Wolf, G., and Bengio, Y. Improving and generalizing flow-based generative models with minibatch optimal transport. In *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*, 2023.

Vahdat, A., Kreis, K., and Kautz, J. Score-based generative modeling in latent space. *Advances in neural information processing systems*, 34:11287–11302, 2021.

Wang, G., Jiao, Y., Xu, Q., Wang, Y., and Yang, C. Deep generative learning via schrödinger bridge. In *International conference on machine learning*, pp. 10794–10804. PMLR, 2021.

Xiao, Z., Yan, Q., and Amit, Y. Generative latent flow. *arXiv preprint arXiv:1905.10485*, 2019.

Xing, Z., Feng, Q., Chen, H., Dai, Q., Hu, H., Xu, H., Wu, Z., and Jiang, Y.-G. A survey on video diffusion models. *ACM Computing Surveys*, 2023.

Xu, Y., Liu, Z., Tegmark, M., and Jaakkola, T. Poisson flow generative models. *Advances in Neural Information Processing Systems*, 35:16782–16795, 2022.

Xu, Y., Tong, S., and Jaakkola, T. Stable target field for reduced variance score estimation in diffusion models. *arXiv preprint arXiv:2302.00670*, 2023.

Yang, L., Liu, J., Hong, S., Zhang, Z., Huang, Z., Cai, Z., Zhang, W., and Bin, C. Improving diffusion-based image synthesis with context prediction. In *NeurIPS*, 2023a.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Zhang, W., Cui, B., and Yang, M.-H. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023b.

Yang, L., Zhang, Z., Zhang, Z., Liu, X., Xu, M., Zhang, W., Meng, C., Ermon, S., and Cui, B. Consistency flow matching: Defining straight flows with velocity consistency. *arXiv preprint arXiv:2407.02398*, 2024.

Zhang, D., Han, Q., Xiong, Y., and Du, H. Mutli-modal straight flow matching for accelerated mr imaging. *Computers in Biology and Medicine*, pp. 108668, 2024.