FORT: Forward-Only Regression Training of Normalizing Flows

Danyal Rehman [1 2 3] **Oscar Davis** [4] **Jiarui Lu** [1 2] **Jian Tang** [1 5] **Michael Bronstein** [4 6] **Yoshua Bengio** [1 2]
Alexander Tong [1 2 *] **Avishek Joey Bose** [1 4 *]

Abstract

Simulation-free training frameworks have been at the forefront of the generative modelling revolution in continuous spaces, leading to neural dynamical systems that encompass modern large-scale diffusion and flow-matching models. Despite the scalability of training, the generation of high-quality samples and their corresponding likelihood under the model requires expensive numerical simulation—inhibiting adoption in numerous scientific applications such as equilibrium sampling of molecular systems. In this paper, we revisit classical normalizing flows as one-step generative models with exact likelihoods and propose a novel, scalable training objective that does not require computing the expensive change of variable formula used in conventional maximum likelihood training. We propose FORWARD-ONLY REGRESSION TRAINING (FORT), a simple ℓ_2-regression objective that maps prior samples under our flow to specifically chosen targets. We demonstrate that FORT supports a wide class of targets, such as optimal transport targets and targets from pre-trained continuous-time normalizing flows (CNF). We further demonstrate that by using CNF targets, our one-step flows allow for larger-scale training that *exceeds* the performance and stability of maximum likelihood training, while unlocking a broader class of architectures that were previously challenging to train. Empirically, we elucidate that our trained flows can perform equilibrium conformation sampling in Cartesian coordinates of alanine dipeptide, alanine tripeptide, and alanine tetrapeptide.

[*]Equal advising [1]Mila – Québec AI Institute [2]Université de Montréal [3]Massachusetts Institute of Technology [4]University of Oxford [5]HEC Montréal [6]AITHYRA. Correspondence to: Danyal Rehman <danyal.rehman@mila.quebec>.

Proceedings of the Workshop on Generative AI for Biology at the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

1. Introduction

The landscape of modern simulation-free generative models in continuous domains, such as diffusion models and flow-matching, has led to state-of-the-art generative quality across a spectrum of domains (Betker et al., 2023; Brooks et al., 2024; Huguet et al., 2024; Geffner et al., 2025). Despite the scalability of simulation-free training, generating samples and computing model likelihoods from these model families requires computationally expensive inference—often hundreds of model calls—through the numerical simulation of the learned dynamical system. The search for efficient inference schemes has led to a new wave of approaches that seek to learn *one-step* generative models, either through distillation (Yin et al., 2024; Lu & Song, 2024; Sauer et al., 2024; Zhou et al., 2024), shortcut training (Frans et al., 2024), or Inductive Moment Matching (IMM) (Zhou et al., 2025) — methods that are able to retain the impressive sample quality of full simulation. However, many highly sensitive applications—for instance, in the natural sciences (Noé et al., 2019; Wirnsberger et al., 2020)—require more than just high-fidelity samples: they also necessitate accurate estimation of probabilistic quantities, the computation of which can be facilitated by having access to cheap and exact model likelihoods. Consequently, for one-step generative models to successfully translate to scientific applications, they must additionally provide faithful *one-step exact likelihoods* for generated samples.

Given their capacity to compute exact likelihoods, classical normalizing flows (NF) have remained the *de facto* method for generative modelling in scientific domains (Dinh et al., 2014; 2016; Rezende & Mohamed, 2015). For instance, in tasks such as conformation sampling with Boltzmann generators (Noé et al., 2019), rapid and exact likelihood evaluation is critical both for asymptotically debiasing generated samples, and for refining them via annealed importance sampling (Tan et al., 2025). Historically, NFs employed in conventional generative modelling domains (such as images) are trained with the maximum likelihood estimation (MLE) objective, which has empirically lagged behind the expressiveness, scalability, and ease of training of modern continuous normalizing flows (CNFs) trained with regression-based objectives like flow-matching (Peluchetti, 2023; Liu, 2022; Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023).

Table 1. Overview of various generative models and their relative trade-offs with respect to number of inference steps, ability to provide exact likelihoods, and training objective for learning.

Method	One-step	Exact likelihood	Regression training
CNF (MLE)	✗	✓	✗
Flow Matching	✗	✓	✓
Shortcut (Frans et al., 2024)	✓	✗	✓
IMM (Zhou et al., 2025)	✓	✗	✓
NF (MLE)	✓	✓	✗
FORT (ours)	✓	✓	✓

A key driver of the gap between classical flows and CNFs can be attributed to the MLE training objective itself, which requires estimating the Jacobian determinant of the inverse flow-map, which is often costly to evaluate for expressive flow architectures, thus preventing efficient optimization and leading to numerical instability. The tension between MLE training and invertible architectural choices used in flow design has created a plethora of exotic training recipes, even with modernized SOTA flows on images with Transformer backbones (Zhai et al., 2024; Kolesnikov et al., 2024), that run counter to the simplicity of training of current one-step generative models—albeit without efficient exact likelihood. This raises the natural motivating research question:

Q. *Does there exist a performant training recipe for classical Normalizing Flows beyond MLE?*

Present work In this paper, we answer in the affirmative. We investigate how to train an invertible neural network to directly match a predefined invertible function. We introduce FORWARD-ONLY REGRESSION TRAINING (FORT), a novel regression-based training objective for classical normalizing flows that marks a significant departure from the well-established MLE training objective.

Our key insight is that access to coupled samples from any invertible map is sufficient to train a generative model with a regression objective. Moreover, with privileged access to such pairings, a classical NF can then be used to directly regress against the target points by pushing forward the corresponding noise points. As a result, we may view FORT as a flow-matching objective wherein the learnable flow-map is an *exactly invertible* architecture. FORT provides similar benefits to NF training as flow-matching does to continuous NFs. Compared to MLE training of NFs, FORT immediately unlocks a key training benefit: to compute the ℓ_2-regression objective, we only need to compute the NF in the forward direction—removing the need to compute the Jacobian determinant of the inverse flow-map during generation. Furthermore, as outlined in Table 1, unlike other one-step generative methods, FORT provides faithful access to exact log-likelihoods while being cheaper than CNFs.

To train NFs using FORT, we propose a variety of couplings to facilitate simple and efficient training. We propose endpoint targets that are either (I) outputs of a larger pretrained CNF, or (II) the solution to a pre-computed OT map done offline as a pre-processing step. In each case, the designed targets are the result of already invertible mappings, which simplifies the learning problem for NFs and enhances training stability. Empirically, we deploy FORT flows on learning equilibrium sampling for short peptides in Alanine di-, tri-, and tetrapeptide, and find even previously discarded NF architectures, such as affine coupling (Dinh et al., 2016) or Neural Spline Flows (Durkan et al., 2019), can outperform their respective MLE trained counterparts. In particular, we illustrate that in scientific applications where MLE training is unsuccessful, the same model trained using FORT provides higher fidelity proposal samples and their likelihood. Finally, we demonstrate a completely new method of performing Targeted Free Energy Perturbation (Wirnsberger et al., 2020) that avoids costly energy evaluations with FORT that is not possible with conventional MLE training of normalizing flows.

2. Background and Preliminaries

Generative models A generative model can be seen as an (approximate) solution to the distribution matching problem: given two distributions p_0 and p_1, the distributional matching problem seeks to find a push-forward map $f_\theta : \mathbb{R}^d \to \mathbb{R}^d$ that transports the initial distribution to the desired endpoint $p_1 = [f_\theta]_\#(p_0)$. Without loss of generality, we set $p_{\text{prior}} := p_0$ to be a tractable prior (typically standard normal) and take $p_{\text{data}} := p_1$ the data distribution, from which we have empirical samples. We now turn our attention to solving the generative modelling problem with modelling families that admit exact log-likelihood, $\log p_\theta(x)$, where $p_\theta = [f_\theta]_\#(p_0)$, with a particular emphasis on normalizing flows (Dinh et al., 2014; 2016; Rezende & Mohamed, 2015; Papamakarios et al., 2021).

2.1. Continuous normalizing flows

Learning the pushforward map, f_θ, can be done by converting this problem into the solution to a neural dynamical system. For example, in a deterministic dynamical system, the pushforward map becomes a time-dependent sufficiently smooth generator $f_\theta : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d, (t, x_0) \mapsto x_t$ and forms the solution pathway to a (neural) ordinary differential equation (ODE) with initial conditions $f_0(x_0) = x_0$. More precisely, a continuous normalizing flow (CNF) models the problem as the following ODE $\frac{d}{dt} f_{t,\theta}(x) = v_{t,\theta}\left(f_{t,\theta}(x_t)\right)$. Here, $v_{t,\theta} : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ is the time-dependent velocity field associated with the (flow) map that transports particles from p_0 to p_1.

As a CNF is the solution to a deterministic dynamical system, it is an invertible map, and as a result, we can compute the exact log-likelihood, $\log p_{t,\theta}(x_t)$, using the instanta-

neous change of variable formula for probability densities (Chen et al., 2018). The overall log-likelihood of a data sample, x_0, under the model can be computed as follows:

$$\log p_{1,\theta}(x_1) = \log p_0(x_0) - \int_1^0 \nabla \cdot v_{t,\theta}(x_t) dt, \quad (1)$$

Maximizing the model log-likelihood in Equation (1) offers one possible method to train CNF's but incurs costly simulation. Instead, modern scalable methods to train CNF's is to employ flow-matching (Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023; Tong et al., 2023; Liu et al., 2023), which learns $v_{t,\theta}$ by regressing against the (conditional) vector field associated with a designed target conditional flow everywhere in space and time, e.g., constant speed conditional vector fields.

Numerical simulation In practice, the simulation of a CNF is conducted using a specific numerical integration scheme that can impact the likelihood estimate's fidelity in Equation (1). For instance, an Euler integrator tends to overestimate the log-likelihood (Tan et al., 2025), and thus it is often preferable to utilize integrators with adaptive step size, such as Dormand–Prince 45 (Hairer et al., 1993). In applications where likelihood estimates suffice, it is possible to employ more efficient estimators such as Hutchinson's trace estimator to get an unbiased—but higher variance—estimate of the divergence. Unfortunately, as we demonstrate in §3.1, such estimators are too high variance to be useful for importance sampling even in the simplest settings, and remain too computationally expensive and unreliable in larger science applications considered in this work.

One-step maps: shortcut models One way to discretize an ODE is to rely on the self-consistency property of ODEs, also exploited in consistency models (Song et al., 2023), namely that jumping Δt in time can be constructed by following the velocity field for two half steps ($\Delta t/2$). This is the core idea behind shortcut models (Frans et al., 2024) that are trained at various jumps by conditioning the vector field network on the desired step-size Δt. Precisely, $f^*_{\text{short},t,2\Delta t}(x_t) = f^*_t(x_t, \Delta t)/2 + f^*_t(x'_{t+\Delta t}, \Delta t)/2$, where $x'_{t+\Delta t} = x_t + f^*_t(x_t, \Delta t)\Delta t$. In their extreme, shortcut models define a one-step mapping which has been shown to generate high-quality images but it remains an open question whether these models can reliably estimate likelihoods.

2.2. Normalizing flows

The generative modelling problem can also be tackled using time-agnostic generators. One such prominent example are Normalizing Flows (NFs) (Dinh et al., 2016; Rezende & Mohamed, 2015), which parameterize diffeomorphisms (continuously differentiable bijective functions, with continuously differentiable inverse), $f_\theta : \mathbb{R}^d \to \mathbb{R}^d$. Those are typically trained using an MLE objective. It is important to highlight that for arbitrary invertible maps f_θ, computing the absolute value of the log Jacobian determinant of the flow exerts a prohibitively expensive cost that scales $O(d^3)$. Consequently, it is popular to build f_θ using a composition of M elementary diffeomorphisms, each with an easier to compute Jacobian determinant: $f_\theta = f_{M-1} \circ \cdots \circ f_0$ (Papamakarios et al., 2021). Through function composition, simple invertible blocks can lead to flows that are universal density approximators (Teshima et al., 2020), and the resulting MLE objective for training is simply:

$$\log p_\theta(x_1) = \log p_0(x_0) - \sum_{i=0}^{M-1} \log\det \left| \frac{\partial f_{i,\theta}(x_i)}{\partial x_i} \right|. \quad (2)$$

Despite the theoretical expressive power of certain classes of NFs (Teshima et al., 2020; Ishikawa et al., 2023; Kong & Chaudhuri, 2021; Zhang et al., 2020; Bose et al., 2021), training using the MLE objective does not offer any insight into the ease and/or practical optimization of f_θ during the learning process.

3. Forward-Only Regression Training

We seek to build one-step transport maps that both push forward samples $x_0 \sim p_0$ to $x_1 \sim p_1$, and also permit exact likelihood evaluation. Such a condition necessitates that this learned map is a bijective function—i.e. an invertible map—and enables us to compute the likelihood using the change of variable formula. While using an MLE objective is always a feasible solution to learn this map, it is often not a scalable solution for both CNFs and classical NFs. Beyond architectural choices and differentiating through a numerical solver, learning flows using MLE is intuitively harder as the process of learning must *simultaneously* learn the forward mapping, f_θ, and the inverse mapping, f_θ^{-1}, without knowledge of pairings $(x_0, x_1) \sim \pi(x_0, x_1)$ from a coupling.

To appreciate this nuance, consider the set of invertible mappings $\mathcal{I}$ and the subset of flows $\mathcal{F} \subset \mathcal{I}$, that solve the generative modelling problem. For instance, there may exist multiple ODEs (possibly infinitely many) that push forward p_0 to p_1. It is clear then that the MLE objective allows the choice of multiple equivalent solutions $f \in \mathcal{F}$. However, this is precisely what complicates learning f_θ, as *certain* solutions are harder to optimize since there is no prescribed coupling $\pi(x_0, x_1)$ for noise x_0 and data targets x_1. That is to say, during MLE optimization of the flow f_θ, the coupling π is learned in conjunction with the flow, which can be challenging to optimize when the pairing between noise and data is suboptimal.

Regression objectives In order to depart from the MLE objective, we may simplify the learning problem by first picking a solution $f^* \in \mathcal{F}$ and fixing the coupling

$\pi^*(x_0, x_1)$ induced under this choice, i.e. $p_1 = [f^*]_\#(p_0)$. Given privileged access to f^*, we can form a simple regression objective that approximates this in continuous time using our choice of learnable flow:

$$\mathcal{L}(\theta) = \mathbb{E}_{t,x_0,x_1,x_t}\left[\|f_{t,\theta}(x_t) - f_t^*(x_t)\|^2\right], \quad (3)$$

where $(x_0, x_1) \sim \pi^*(x_0, x_1)$ and $x_t \sim p_t(\cdot|x_0, x_1)$ is drawn from a known conditional noising kernel such as a Gaussian distribution. We highlight that a key benefit unlocked with using Equation (3) is that we only need to evaluate the flow $f_{t,\theta}$ in the *forward* direction on samples drawn from the fixed coupling π^*. We further note that the regression objective in Equation (3) is more general than just flows in $\mathcal{I}$, and, at optimality, the learned function behaves like f_t^* on the support of p_0, under mild regularity conditions. We formalize this intuition more precisely in the next proposition.

Proposition 3.1. *Suppose that $f_t^\star$ is invertible for all t, that $(f_t^\star)^{-1}$ is continuous for all t. Then, as $\mathcal{L}(\theta) \to 0$, it holds that $((f_t^\star)^{-1} \circ f_{t,\theta})(x) \to x$ for almost all (with respect to p_0) x.*

The proof for Proposition 3.1 can be found in §A, and illuminates that solving the original generative modelling through MLE can be reformulated as a *matching* problem to a known invertible function f^*. Indeed, many existing generative models already fit into this general regression objective based on the choice of f^*, such as conditional flow-matching (CFM) (Tong et al., 2023), Rectified flow (Liu et al., 2023), and (perfect) shortcut models (Frans et al., 2024). This proposition also shows why these models work as generative models: they converge in probability to the prespecified map.

3.1. Warmup: one-step generative models without likelihood

As there exists powerful one-step regression-based generative models in image applications, it is tempting to consider whether they already solve the thesis of this paper in that they already provide faithful one-step likelihoods. As a warmup, we investigate the invertibility of current state-of-the-art one-step generative models in shortcut models (Frans et al., 2024) and Inductive Moment Matching (Zhou et al., 2024) (see §B for details). Intuitively, both these model classes progressively learn a time-discretized map of the probability path constructed under a typical diffusion or CNF using bootstrap targets and other measures of self-consistency.

Synthetic experiments We instantiate both model classes on a simple generative modelling problem where the data distribution corresponds to a synthetic checkerboard density p_{syn} in 2D. We choose this setup as the target distribution admits analytic log-likelihoods, allowing us to compute the weights needed to perform importance sampling. For example, given a trained model p_θ, and the collection of importance weights, we aim to compute a Monte–Carlo approximation to any test function $\phi(x)$ of interest under p_{syn} using self-normalized importance sampling (SNIS) (Liu, 2001) as follows:

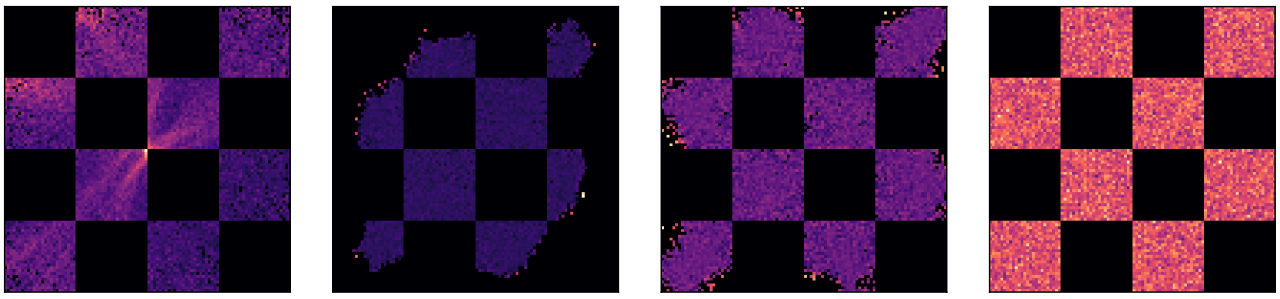
$$\mathbb{E}_{p_{\text{syn}}(x)}[\phi(x)] = \mathbb{E}_{p_\theta(x)}[\phi(x)\bar{w}(x)] \approx \frac{\sum_{i=1}^K w(x^i)\phi(x^i)}{\sum_{i=1}^K w(x^i)}. \quad (4)$$

In addition, computing importance weights also enables resampling the pool of samples according to the collection of normalized importance weights $W = \{\bar{w}(x^i)\}_{i=1}^K$.

In Figure 1, we plot the results of shortcut models and IMM with non-invertible networks and IMM with an invertible network, a Neural Spline Flow (NSF) (Durkan et al., 2019). Given access to correct likelihoods, we can correct generated samples using the self-normalized important IS in Equation (4) to produce asymptotically exact samples that resemble the ground truth checkerboard. As observed, we find that non-invertible shortcuts are imperfect at learning the target and are unable to be corrected to p_{syn} after resampling. Similarly, for the non-invertible IMM, we observe better initial samples but resampling—while better—still presents inaccuracies compared to the ground truth. Finally, when IMM is equipped with an invertible backbone, we see samples that almost perfectly match p_{syn}. While this may initially suggest that IMM with an invertible backbone is ideal, we show that such an approach is difficult to learn even in simple problems at the scale of MNIST (see Appendix B.2).

This puts spotlight on a counter-intuitive question given Proposition 3.1: *Why do shortcut models have incorrect likelihoods?* Shortcut models have incorrect likelihoods for two reasons: (1) invertibility implied under Proposition 3.1 only holds at convergence, and (2) even if the model has converged, Proposition 3.1 is only sufficient for accurate generation. Accurate likelihood estimation requires an invertible map and regularity of higher-order gradients, because the likelihood, as given in Equation (2), requires the computation of the log determinant of the Jacobian. While Proposition 3.1 implies pointwise convergence of f_θ to f^*, this does not imply convergence or regularity of the gradients of f_θ and thus shortcut models can still achieve high quality generations without the need to provide faithful likelihoods.

Insufficiency of uniform convergence While it might seem reasonable to infer that the uniform convergence of $f_\theta \to f^\star$ on a sub-domain $D \subseteq \mathbb{R}^d$, implies pointwise convergence of gradients $\nabla f_\theta \to \nabla f^\star$, this is not generally true. For illustrative purposes, consider the toy example: $f_m(x) = \frac{1}{m}\sin(mx) + x$ and $f^\star(x) = x$. As



(a) Non-invertible shortcut. (b) Non-invertible IMM. (c) IMM with an NF. (d) Ground truth.

Figure 1. Evaluation of IMM and shortcut models with exact likelihood on the synthetic checkerboard experiment, with depictions of the 2D histograms after SNIS resampling.

$m \to \infty$, f_m converges uniformly to $f^\star$; however, the gradient $\nabla f_m(x) = \cos(mx)$ does not converge. Importantly, this means that while f_θ may produce increasingly accurate samples, its likelihoods derived through Equation (2) may not converge to those of the base model. This counterexample demonstrates the need for other methods to ensure correct likelihoods for models with high-quality samples.

3.2. Training normalizing flows using FORT

We now outline our FORT framework to train a one-step map in a classical NF. To remedy the issue found in shortcut models and IMM in Section 3.1, we judiciously choose f_θ to be an already exactly invertible mapping—i.e., a classical NF. Since NFs are one-step maps by construction Equation (3) is instantiated as a simple ℓ_2 regression objective as follows:

$$\begin{aligned}\mathcal{L}(\theta) &= \mathbb{E}_{x_0,x_1}\left[\left\|f_{0,\theta}(x_0) - f_1^*(x_0)\right\|^2\right] + \lambda_r \mathcal{R} \\ &= \mathbb{E}_{x_0,x_1}\left[\left\|\hat{x}_1 - x_1\right\|^2\right] + \lambda_r \mathcal{R}, \end{aligned} \tag{5}$$

where $\mathcal{R}$ is a regularization strategy and $\lambda_r \in \mathbb{R}^+$ is the strength of the regularization applied.

Explicit in Equation (5) is the need to procure *one-step* targets $x_1 = f_1^*(x_0)$ from a known invertible mapping f_1^*. We outline the choice of such functions in §3.3. We also highlight that the one-step targets in Equation (5) differ from the typical flow-matching objective where the continuous targets $f^*_{t,\text{cfm}} = \frac{\partial}{\partial} p_t(x_t|x_0, x_1)$ (see §A.3 for a detailed discussion). Furthermore, note that training an NF within FORT only requires to evaluate the forward direction during training and acts as the closest invertible approximation to the already invertible map f_1^*. Consequently, for NFs that are universal density approximators (Teshima et al., 2020; Kong & Chaudhuri, 2021; Zhang et al., 2020), the learning problem includes a feasible solution.

Training recipe We provide the full training pseudocode in Algorithm 1. In practice, we find that $f^\star$ is often ill-conditioned, with the target distribution often centered around some lower-dimensional subspace of $\mathbb{R}^d$ similar to prior work (Zhai et al., 2024). This may cause f_θ to become numerically ill-conditioned. To combat this, we use three tricks to maintain numerical stability. Specifically, we regularize the log determinant, add small amounts of Gaussian noise to the target distribution similar to (Hui et al., 2025; Zhai et al., 2024), and, finally, include weight decay.

Algorithm 1 FORWARD-ONLY REGRESSION TRAINING

input Prior p_0, empirical samples from p_1, regularization weight λ_r, noise scale λ_n, network f_θ
  while training **do**
    $(x_0, x_1) \sim \pi(x_0, x_1)$
    $x_1 \leftarrow x_1 + \lambda_n \cdot \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, I)$
    $\mathcal{L}(\theta) \leftarrow \|f_\theta(x_0) - x_1\|_2^2 + \lambda_r \mathcal{R}$
    $\theta \leftarrow \text{Update}(\theta, \nabla_\theta \mathcal{L}(\theta))$
  end while
  return f_θ

3.3. FORT targets

To construct useful one-step targets in FORT, we must find a discretization of a true invertible function—e.g., an ODE solution—at longer time horizons. More precisely, we seek a discretization of an ODE such that each time point $t + \Delta t$ where the regression objective is evaluated corresponds to a true invertible function $f^*_{t+\Delta t}$. Consequently, if we have access to an invertible map such that $t + \Delta t = 1$, we can directly regress our parametrized function as a one-step map, $f_{0,\theta}(x_0) = \hat{x}_1$. This motivates the search and design of other invertible mappings that give us invertibility at longer time horizons, for which we give two examples next.

Optimal transport targets Optimal transport in continuous space between two distributions defines a continuous and invertible transformation expressible as the gradient of some convex function (Villani, 2021; Peyré & Cuturi, 2019).

This allows us to consider the invertible OT plan:

$$f^*_{\text{ot}} = \arg\min_T \int T(x)c(x, T(x))dp_0(x)$$
$$\text{s.t. } T_{\#}(p_0) = p_1,$$

where $c : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the OT cost and $T : \mathbb{R}^d \to \mathbb{R}^d$ is a transport map. We note that this map is interesting as it requires no training; however, exact OT runs in $O(n^3)$ time and $O(n^2)$ space, which makes it challenging to scale to large datasets. Furthermore, we highlight that this differs from OT-CFM (Tong et al., 2023), which uses mini-batches to approximate the OT-plan. Nevertheless, in applicable settings, full batch OT acts as a one-time offline pre-processing step for training f_θ using FORT.

Reflow targets Another strategy to obtain samples from an invertible map is to use a pretrained CNF, also known as *Reflow* (Liu, 2022). Specifically, we have that:

$$f^*_{\text{reflow}}(x_0) = x_0 + \int_0^1 v^\star_t(x_t)dt = x_1. \tag{6}$$

In other words, the one-step invertible map is obtained from a pre-trained CNF $v^\star_t$, from which we collect a dataset of noise-target pairs, effectively forming $\pi^*(x_0, x_1)$ which we use during FORT. We now prove that training on Reflow targets with FORT reduces the Wasserstein distance to the p_1.

Proposition 3.2. *Let p_{reflow} be a pretrained CNF generated by the vector field v^*_t, real numbers $(L_t)_{t\in[0,1]}$ such that v^*_t is L_t-Lipschitz for all $t \in [0, 1]$, and a NF f^{nf}_θ trained using Eq. 5 by regressing against $f^\star_{reflow}(x_0)$, where $x_0 \sim \mathcal{N}(0, I)$. Then, writing $p^{nf}_\theta := \text{Law}(f^{nf}_\theta(x_0))$, we have:*

$$\mathcal{W}_2(p_1, p_\theta) \le K \exp\left(\int_0^1 L_t dt\right) + \epsilon,$$
$$K \ge \int_0^1 \mathbb{E}\left(\left[\|v^*_t(x_t) - v_t(x_t)\|_2^2\right]\right)^{\frac{1}{2}} dt,$$

where K is the ℓ_2 approximation error between the velocity field of the CNF and the ground truth generating field v^*_t, $\epsilon^2 = \mathbb{E}_{x_0,x_1}\left[\|f^\star_{reflow}(x_0) - f^{nf}_\theta(x_0)\|_2^2\right]$.

The proof for Proposition 3.2 is provided in §A. Intuitively, the first term captures the approximation error of the pretrained CNF to the actual data distribution p_1, and the second term captures the approximation gap between the flow trained using FORT to the reflow targets of p_{reflow}.

4. Experiments

We evaluate NFs trained using FORT on smaller peptides (ALDP, AL3, and AL4) both on equilibrium sampling and free energy prediction tasks. Classical Amber force fields are used as the energy functions both to generate "Ground Truth" molecular dynamics data, and for SNIS.

4.1. Molecular conformation sampling

We first evaluate FORT on molecular conformation sampling tasks. We test three different architectures across three different molecular systems of increasing length in alanine dipeptide (ALDP) to alanine tripeptide (AL3) and alanine tetrapeptide (AL4), and compare the performance of the same invertible architecture trained using MLE and using FORT. We report the Effective Sample Size (ESS), the 1-Wasserstein distance on the energy distribution, and the 2-Wasserstein distance on the dihedral angles used in the Ramachandran plots and generated samples in §D.

Normalizing flow architectures In our experiments, we use the following NF architectures: the classical RealNVP with a residual network parametrization (Dinh et al., 2016), NSF (Durkan et al., 2019), and our implementation of the transformer-based NF in Jet (Kolesnikov et al., 2024).

Main results We report our main quantitative results in Table 2 and observe that FORT with reflow targets consistently outperforms MLE training of NFs across all architectures for $\mathcal{E}$-$\mathcal{W}_1$ and $\mathbb{T}$-$\mathcal{W}_2$ metrics and slightly underperforms MLE training on ESS. However, this can be justified by the mode collapse that happens in MLE training as illustrated in the Ramachandran plots for ALDP in Figure 2, which artificially increases ESS. We further include energy histograms for proposal samples on ALDP of each flow and their corresponding resampling with IS in Figure 3 and observe NFs trained using FORT more closely match the true energy distribution. Additional results for AL3 and AL4 are reported in §D. Our results demonstrate that FORT is often a compelling and favorable alternative to MLE training *for each classical NF architecture* when we have access to high-quality targets from diverse invertible maps.

Table 2. Quantitative results on alanine dipeptide (ALDP), alanine tripeptide (AL3), and alanine tetrapeptide (AL4).

Datasets →	Dipeptide (ALDP)			Tripeptide (AL3)			Tetrapeptide (AL4)		
Algorithm ↓	ESS ↑	$\mathcal{E}$-$\mathcal{W}_1$ ↓	$\mathbb{T}$-$\mathcal{W}_2$ ↓	ESS ↑	$\mathcal{E}$-$\mathcal{W}_1$ ↓	$\mathbb{T}$-$\mathcal{W}_2$ ↓	ESS ↑	$\mathcal{E}$-$\mathcal{W}_1$ ↓	$\mathbb{T}$-$\mathcal{W}_2$ ↓
NSF (MLE)	**0.055**	13.797	1.243	0.0237	17.596	1.665	**0.016**	20.886	3.885
NSF (FORT)	0.036	**0.519**	**0.958**	**0.0291**	**1.051**	**1.612**	0.010	**6.277**	**3.476**
Res–NVP (MLE)	$< 10^{-4}$	$> 10^3$	> 30	$< 10^{-4}$	$> 10^3$	> 30	$< 10^{-4}$	$> 10^3$	> 30
Res–NVP (FORT)	**0.032**	**2.310**	**0.796**	**0.025**	**3.600**	**1.960**	**0.013**	**2.724**	**4.046**
Jet (MLE)	$< 10^{-4}$	$> 10^3$	> 30	$< 10^{-4}$	$> 10^3$	> 30	$< 10^{-4}$	$> 10^3$	> 30
Jet (FORT)	**0.051**	**6.349**	**0.872**	$< 10^{-4}$	$> 10^3$	**3.644**	$< 10^{-4}$	$> 10^3$	> 30

Ablations In Table 3, we report FORT using OT targets and various amounts of generated reflow targets—a unique advantage of using reflow as the invertible map. As observed, each target choice improves over MLE, outside of ESS for NSF. Importantly, we find that using more samples in reflow consistently improves performance metrics

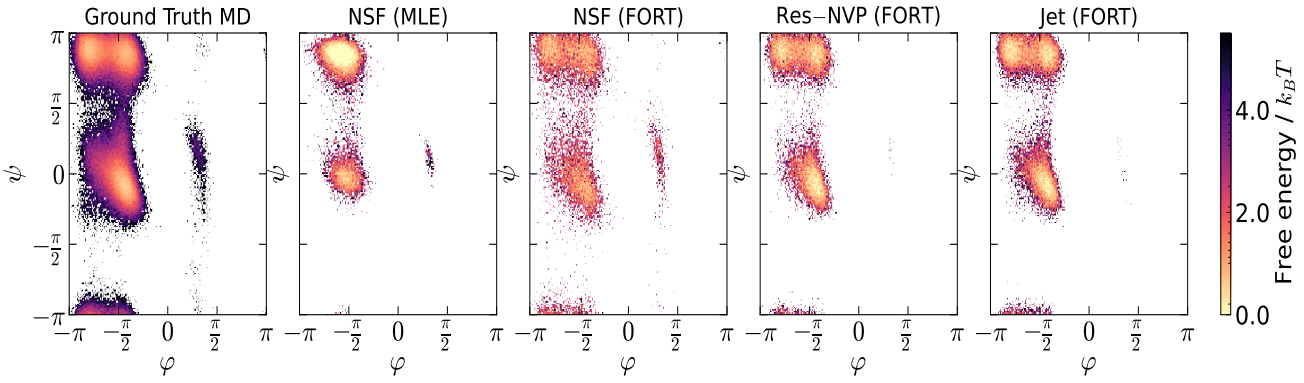


Figure 2. Ramachandran plots for alanine dipeptide (**left to right**: ground truth MD data; most performant MLE-trained model (NSF); NSF (FORT); Res–NVP (FORT); and Jet (FORT).

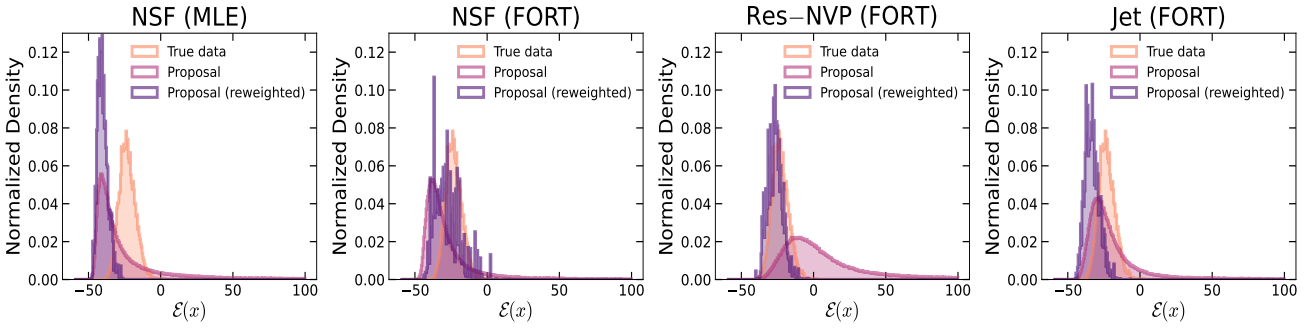


Figure 3. Energy distribution of original and SNIS re-weighted samples generated using various flow-based methods on alanine dipeptide (**left to right**: most performant MLE-trained model (NSF); NSF (FORT); Res–NVP (FORT); and Jet (FORT).

Table 3. Ablations on target types and amount of reflow on ALDP.

Datasets →	Dipeptide (ALDP)		
Algorithm ↓	ESS ↑	$\mathcal{E}$-$\mathcal{W}_1$ ↓	$\mathbb{T}$-$\mathcal{W}_2$ ↓
NSF (MLE)	**0.055**	13.80	1.243
NSF (FORT @ 100k CNF)	0.016	17.39	1.232
NSF (FORT @ 10.4M CNF)	0.036	**0.519**	**0.958**
NSF (FORT @ OT)	0.003	0.604	2.019
Res–NVP (MLE)	$< 10^{-4}$	$> 10^3$	> 30
Res–NVP (FORT @ 100k CNF)	0.009	46.93	1.155
Res–NVP (FORT @ 10.4M CNF)	0.032	2.310	**0.796**
Res–NVP (FORT @ OT)	**0.006**	**0.699**	1.969
Jet (MLE)	$< 10^{-4}$	$> 10^3$	> 30
Jet (FORT @ 100k CNF)	0.017	31.42	1.081
Jet (FORT @ 10.4M CNF)	**0.051**	6.349	**0.872**
Jet (FORT @ OT)	0.003	**2.534**	1.913

for all architectures. In Figure 4 we ablate the impact of regularization and find performance improvements with increasing regularization, up to a certain point. Although regularization beyond this guarantees numerical invertibility, it hampers generation performance across metrics. This trade-off typically occurs between $10^{-6} \leq \lambda_r \leq 10^{-5}$ for all normalizing flow architectures tested in this work.

4.2. Targeted free energy perturbation

Accurate calculations of the free energy difference between two metastable states of a physical system is both ubiquitous and of profound importance in the natural sciences. One approach to tackling this problem is Free Energy Perturbation (FEP) which exploits Zwanzig's identity: $\mathbb{E}_A\left[e^{-\beta\Delta U}\right] = e^{-\beta\Delta F}$ where $\Delta F = F_B - F_A$ is the Helmholtz free energy difference between two metastable states A and B (Zwanzig, 1954). Targeted Free Energy Perturbation (TFEP) improves over FEP by using NFs to learn an invertible map using MLE to increase the distributional overlap between states A and B (Wirnsberger et al., 2020); however, this can be challenging for several reasons. NFs are difficult to learn, especially when the energy function is expensive to compute, or the states occupy small areas.

We propose a new TFEP method that does not require energy function evaluations during training. By using FORT, we can train the normalizing flow solely based on samples from the states A and B. This enables TFEP, where energy evaluations may be costly—a new possibility that is distinct from NFs trained using MLE. To demonstrate this application of FORT, we train an NF solely from samples from two modes of ALDP (see Figure 5) and use OT targets which avoid *any energy function evaluation*. We find we can achieve high-

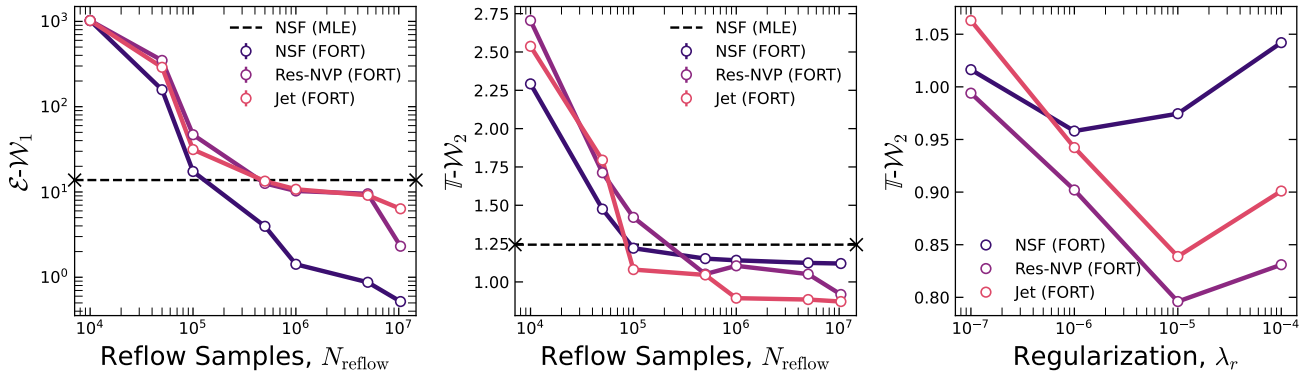


Figure 4. **Left and center**: Ablations demonstrating performance improvements with more reflow samples. **Right**: Increasing regularization improves $\mathbb{T}$-$\mathcal{W}_2$ up to a certain point, beyond which numerical invertibility is guaranteed but sample quality is adversely impacted.

quality free energy estimation in comparison to ground truth molecular dynamics (MD) using only samples during training, as illustrated in Figure 5. We believe this is a promising direction for future applications of free energy prediction.

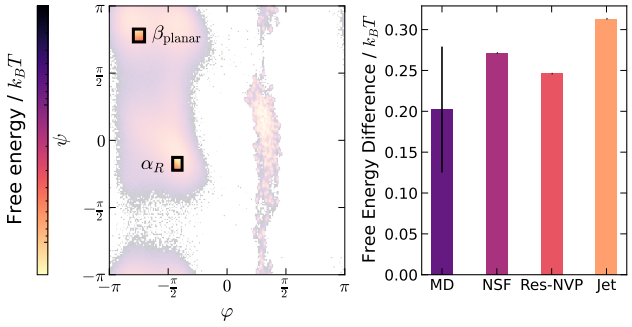


Figure 5. State transitions between β_{planar} and α_R conformations.

5. Related Work

Exact likelihood generative models NFs are generative models with invertible architectures (Rezende & Mohamed, 2015; Dinh et al., 2016) that produce *exact* likelihoods for any given points. Common models include Real NVP (Dinh et al., 2016), Neural Spline Flows (Durkan et al., 2019), and Glow (Kingma & Dhariwal, 2018). Jet (Kolesnikov et al., 2024) and TarFlow (Zhai et al., 2024) are examples of transformer-based flows. Aside from Jet and Tarflow, NFs have generally underperformed compared to diffusion models and flow-matching methods (Ho et al., 2020; Lipman et al., 2023; Albergo et al., 2023; Liu, 2022; Song et al., 2021), partly due to the high cost of evaluating the log-determinants of Jacobians at each training step.

Few-step generative models To avoid costly inference, few-step generative models were introduced as methods to accelerate the simulation of diffusion and CNFs. Common examples include DDIM (Song et al., 2022) and consistency models (Song et al., 2023), which introduced a new training procedure that ensured the model's endpoint prediction remained consistent. Song & Dhariwal (2023); Lu & Song (2024); Geng et al. (2024) have improved this paradigm. Other lines of work proposed related but different training objectives, generalizing consistency training (Frans et al., 2024; Zhou et al., 2025; Kim et al., 2024; Heek et al., 2024). Beyond diffusion and FM, residual networks (He et al., 2015) are a class of neural networks that are invertible if the Lipschitz constant of f_θ is at most one (Behrmann et al., 2019). The log-determinant of the Jacobian is then approximated by truncating a series of traces (Behrmann et al., 2019)—an approximation improved in Chen et al. (2020).

6. Conclusion

In this work, we present FORT, a method for generating high-quality samples alongside exact likelihoods in a single step. Using a base coupling between the dataset samples and the prior, provided by either pre-computed optimal transport or a base CNF, we can train a classical NF using a simple regression objective that avoids computing Jacobians at training time, as opposed to typical MLE training. We have shown, in theory and in practice, that the learned model produces faithful samples, the likelihoods of which empirically allow us to produce state-of-the-art results on several molecular datasets, using importance-sampling resampling. Limitations include the quality of the proposal samples, which substantially improve on MLE-trained NFs, but are not on-par with state-of-the-art CNFs or variants thereof. Moreover, while producing accurate and high-quality likelihoods, they do not in theory match those of the base coupling, which can be a desirable property.

Acknowledgements

DR received financial support from the Natural Sciences and Engineering Research Council's (NSERC) Banting Postdoctoral Fellowship under Funding Reference No. 198506. OD is supported by both Project CETI and Intel. AJB is supported by an NSERC Postdoctoral Fellowship and the EPSRC Turing AI World-Leading Research Fellowship No. EP/X040062/1 and EPSRC AI Hub No. EP/Y028872/1. JT acknowledges funding from the Canada CIFAR AI Chair Program and the Intel-Mila partnership program. The authors acknowledge funding from UNIQUE, CIFAR, NSERC, Intel, and Samsung. The research was enabled in part by computational resources provided by the Digital Research Alliance of Canada (https://alliancecan.ca), Mila (https://mila.quebec), and NVIDIA.

Impact Statement

This work investigates a new regression-based objective for training normalizing flows, targeting applications in AI4Science. The primary focus is on equilibrium conformation sampling and free energy prediction, which has potential applications to material/drug discovery. Although no immediate negative impacts are anticipated, caution is advised when scaling these methods to avoid potential misuse.

References

Albergo, M. S. and Vanden-Eijnden, E. Building normalizing flows with stochastic interpolants. *International Conference on Learning Representations (ICLR)*, 2023.

Albergo, M. S., Boffi, N. M., and Vanden-Eijnden, E. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint 2303.08797*, 2023.

Behrmann, J., Grathwohl, W., Chen, R. T. Q., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks, 2019. URL https://arxiv.org/abs/1811.00995.

Betker, J., Goh, G., Jing, L., Brooks, T., Wang, J., Li, L., Ouyang, L., Zhuang, J., Lee, J., Guo, Y., et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3): 8, 2023.

Bose, A. J., Brubaker, M., and Kobyzev, I. Equivariant finite normalizing flows. *arXiv preprint arXiv:2110.08649*, 2021.

Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., and Ramesh, A. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. *Neural Information Processing Systems (NeurIPS)*, 2018.

Chen, R. T. Q., Behrmann, J., Duvenaud, D., and Jacobsen, J.-H. Residual flows for invertible generative modeling, 2020. URL https://arxiv.org/abs/1906.02735.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

Etmann, C., Ke, R., and Schönlieb, C.-B. iunets: Fully invertible u-nets with learnable up- and downsampling, 2020. URL https://arxiv.org/abs/2005.05220.

Frans, K., Hafner, D., Levine, S., and Abbeel, P. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

Geffner, T., Didi, K., Zhang, Z., Reidenbach, D., Cao, Z., Yim, J., Geiger, M., Dallago, C., Kucukbenli, E., Vahdat, A., et al. Proteina: Scaling flow-based protein structure generative models. *arXiv preprint arXiv:2503.00710*, 2025.

Geng, Z., Pokle, A., Luo, W., Lin, J., and Kolter, J. Z. Consistency models made easy, 2024. URL https://arxiv.org/abs/2406.14548.

Hairer, E., Nørsett, S. P., and Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Computational Mathematics*. Springer-Verlag, 2nd edition, 1993. ISBN 978-3-540-56670-0.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Heek, J., Hoogeboom, E., and Salimans, T. Multistep consistency models, 2024. URL https://arxiv.org/abs/2403.06807.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

Huguet, G., Vuckovic, J., Fatras, K., Thibodeau-Laufer, E., Lemos, P., Islam, R., Liu, C.-H., Rector-Brooks, J., Akhound-Sadegh, T., Bronstein, M., et al. Sequence-augmented se (3)-flow matching for conditional protein backbone generation. *arXiv preprint arXiv:2405.20313*, 2024.

Hui, K.-H., Liu, C., Zeng, X., Fu, C.-W., and Vahdat, A. Not-so-optimal transport flows for 3d point cloud generation. In *ICLR*, 2025.

Ishikawa, I., Teshima, T., Tojo, K., Oono, K., Ikeda, M., and Sugiyama, M. Universal approximation property of invertible neural networks. *Journal of Machine Learning Research*, 24(287):1–68, 2023.

Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ode trajectory of diffusion, 2024. URL https://arxiv.org/abs/2310.02279.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions, 2018. URL https://arxiv.org/abs/1807.03039.

Kolesnikov, A., Pinto, A. S., and Tschannen, M. Jet: A modern transformer-based normalizing flow. *arXiv preprint arXiv:2412.15129*, 2024.

Kong, Z. and Chaudhuri, K. Universal approximation of residual flows in maximum mean discrepancy. *arXiv preprint arXiv:2103.05793*, 2021.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *International Conference on Learning Representations (ICLR)*, 2023.

Liu, J. S. *Monte Carlo Strategies in Scientific Computing*. Springer, 2001.

Liu, Q. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *International Conference on Learning Representations (ICLR)*, 2023.

Lu, C. and Song, Y. Simplifying, stabilizing and scaling continuous-time consistency models. *arXiv preprint arXiv:2410.11081*, 2024.

Noé, F., Olsson, S., Köhler, J., and Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

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. *arXiv preprint arXiv:2312.14589*, 2023.

Peyré, G. and Cuturi, M. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5-6): 355–607, 2019.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Sauer, A., Lorenz, D., Blattmann, A., and Rombach, R. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

Song, Y. and Dhariwal, P. Improved techniques for training consistency models, 2023. URL https://arxiv.org/abs/2310.14189.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models, 2023. URL https://arxiv.org/abs/2303.01469.

Tan, C. B., Bose, A. J., Lin, C., Klein, L., Bronstein, M. M., and Tong, A. Scalable equilibrium sampling with sequential boltzmann generators. *arXiv preprint arXiv:2502.18462*, 2025.

Teshima, T., Ishikawa, I., Tojo, K., Oono, K., Ikeda, M., and Sugiyama, M. Coupling-based invertible neural networks are universal diffeomorphism approximators. *Advances in Neural Information Processing Systems*, 33:3362–3373, 2020.

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. *arXiv preprint arXiv:2302.00482*, 2023.

Villani, C. *Topics in optimal transportation*, volume 58. American Mathematical Soc., 2021.

Wirnsberger, P., Ballard, A. J., Papamakarios, G., Abercrombie, S., Racanière, S., Pritzel, A., Jimenez Rezende, D., and Blundell, C. Targeted free energy estimation via learned mappings. *The Journal of Chemical Physics*, 153 (14), 2020.

Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024.

Zhai, S., Zhang, R., Nakkiran, P., Berthelot, D., Gu, J., Zheng, H., Chen, T., Bautista, M. A., Jaitly, N., and Susskind, J. Normalizing flows are capable generative models. *arXiv preprint arXiv:2412.06329*, 2024.

Zhang, H., Gao, X., Unterman, J., and Arodz, T. Approximation capabilities of neural odes and invertible residual networks. In *International Conference on Machine Learning*, pp. 11086–11095. PMLR, 2020.

Zhou, L., Ermon, S., and Song, J. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025.

Zhou, M., Zheng, H., Wang, Z., Yin, M., and Huang, H. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *Forty-first International Conference on Machine Learning*, 2024.

Zwanzig, R. W. High-temperature equation of state by a perturbation method. i. nonpolar gases. *The Journal of Chemical Physics*, 22(8):1420–1426, 08 1954. ISSN 0021-9606. doi: 10.1063/1.1740409. URL https://doi.org/10.1063/1.1740409.

A. Proofs

A.1. Proof of Proposition 3.1

We first recall Proposition 3.1 below.

Proposition A.1. *Suppose that $f_t^\star$ is invertible for all t, that $(f_t^\star)^{-1}$ is continuous for all t. Then, as $\mathcal{L}(\theta) \to 0$, it holds that $((f_t^\star)^{-1} \circ f_{t,\theta})(x) \to x$ for almost all (with respect to p_0) x.*

To prove Proposition 3.1, we first prove the following lemma, which is essentially the same as the proposition, but it abstracts out the distribution of x_t, which depends on x_0, x_1, and t.

Lemma A.2. *For functions $(f_n)_{n\geq 1}$ and g, where g is invertible and has a continuous inverse, $x_0 \sim p_0$, if $\mathrm{MSE}(f_n, g) \coloneqq \mathbb{E}_{x_0} \|f_n(x_0) - g(x_0)\|_2^2 \to 0$, then $\lim_{n\to\infty} g^{-1}(f_n(x)) = x$ for almost all (wrt to p_0) x.*

Proof. Let $Y_n = \|f_n(x_0) - g(x_0)\|_2$. We know that $\lim_{n\to\infty} \mathbb{E}[Y_n^2] = 0$ (as it corresponds to the MSE), which implies that $\lim_{n\to\infty} \mathrm{Var}(Y_n) = 0$. Consequently, $Y_n \to c$ for some constant $c \in \mathbb{R}$. Moreover, by Jensen's inequality and the convexity of $x \mapsto x^2$, we find that $(\mathbb{E}[Y_n])^2 \leq \mathbb{E}[Y_n^2]$, meaning that $c = 0$. This implies that $\lim_{n\to\infty} \|f_n(x) - g(x)\|_2^2 = 0$ almost everywhere, and thus that $\lim_{n\to\infty} f_n(x) = g(x)$. Finally, since g^{-1} is continuous, we can apply the function to both sides of the limit to find that $\lim_{n\to\infty} g^{-1}(f_n(x)) = x$, almost everywhere. □

It suffices to apply the above lemma to $x_t \sim p_t(\,\cdot \mid x_0, x_1) p_1(x_1 \mid x_0) p_0(x_0)$.

A.2. Proof of Proposition 3.2

We now prove Proposition 3.2. The proposition reuses the following regularity assumptions, as introduced in Benton et al. (2023), which we recall verbatim below for convenience:

(Assumption 1) Let v_{true} be the true generating velocity field for the CNF with field v^* trained using flow matching. Then the true and learned velocity v^* are close in ℓ_2 and satisfy: $\int_0^1 \mathbb{E}_{t,x_t}[\|v_{t,\text{true}}(x_t) - v_t^*(\,x_t)\|^2]dt \leq K^2$.

(Assumption 2) For each $x \in \mathbb{R}^d$ and $s \in [0,1]$, there exists unique flows $(f_{s,t}^*)_{t\in[s,1]}$ and $(f_{(s,t),\text{true}})_{t\in[s,1]}$, starting at $f_{(s,s)}^* = x$ and $f_{(s,s),\text{true}} = x$ with velocity fields $v_t^*(x_t)$ and $v_{t,\text{true}}(x_t)$, respectively. Additionally, f^* and f_{true} are continuously differentiable in x, s and t.

(Assumption 3) The velocity field $v_t^*(x_t)$ is differentiable in both x and t, and also for each $t \in [0,1]$ there exists a constant L_t such that $v_t^*(x_t)$ is L_t-Lipschitz in x.

Proposition A.3. *Let $p_{\textit{reflow}}$ be a pretrained CNF generated by the vector field v_t^*, real numbers $(L_t)_{t\in[0,1]}$ such that v_t^* is L_t-Lipschitz for all $t \in [0,1]$, and a NF $f_\theta^{\textit{nf}}$ trained using Eq. 5 by regressing against $f_{\textit{reflow}}^\star(x_0)$, where $x_0 \sim \mathcal{N}(0, I)$. Then, writing $p_\theta^{\textit{nf}} \coloneqq \mathrm{Law}(f_\theta^{\textit{nf}}(x_0))$, we have:*

$$\mathcal{W}_2(p_1, p_\theta) \leq K \exp\left(\int_0^1 L_t dt\right) + \epsilon,$$

$$K \geq \int_0^1 \mathbb{E}\left(\left[\|v_t^*(x_t) - v_t(x_t)\|_2^2\right]\right)^{\frac{1}{2}} dt,$$

where K is the ℓ_2 approximation error between the velocity field of the CNF and the ground truth generating field v_t^*, $\epsilon^2 = \mathbb{E}_{x_0,x_1}\left[\|f_{\textit{reflow}}^\star(x_0) - f_\theta^{\textit{nf}}(x_0)\|_2^2\right]$.

Proof. We begin by first applying the triangle inequality to $\mathcal{W}_2(p_1, p_\theta)$ and obtain:

$$\mathcal{W}_2(p_1, p_\theta) \leq \mathcal{W}_2(p_1, p_{\text{reflow}}) + \mathcal{W}_2(p_{\text{reflow}}, p_\theta^{\text{nf}}). \tag{7}$$

The first term is an error in Wasserstein-2 distance between the true data distribution and our reflow targets, which is still a

CNF. A straightforward application of Theorem 1 in Benton et al. (2023) gives a bound on this first Wasserstein-2 distance[1]:

$$\mathcal{W}_2(p_1, p_{\text{reflow}}) \leq K \exp\left(\int_0^1 L_t dt\right). \tag{8}$$

To bound $\mathcal{W}_2(p_{\text{reflow}}, p_\theta)$, recall that the following inequality holds $\mathcal{W}_2(\text{Law}(X), \text{Law}(Y)) \leq \mathbb{E}\left[\|X - Y\|_2^2\right]^{\frac{1}{2}}$, for any two random variables X and Y. In our case, these random variables are $p^*_{\text{reflow}} = \text{Law}(f^*_{\text{reflow}}(x_0))$ and $p^{\text{nf}}_\theta = \text{Law}(f^{\text{nf}}_\theta(x_0))$. This gives:

$$\mathcal{W}_2(p_{\text{reflow}}, p^{\text{nf}}_\theta) \leq \mathbb{E}_{x_0,x_1}\left[\left\|f^*_{\text{reflow}}(x_0) - f^{\text{nf}}_\theta(x_0)\right\|_2^2\right]^{\frac{1}{2}}. \tag{9}$$

Combining Equation (8) and Equation (9) achieves the desired result and completes the proof.

$$\mathcal{W}_2(p_1, p_\theta) \leq K \exp\left(\int_0^1 L_t dt\right) + \mathbb{E}_{x_0,x_1}\left[\left\|f^*_{\text{reflow}}(x_0) - f^{\text{nf}}_\theta(x_0)\right\|_2^2\right]^{\frac{1}{2}}. \tag{10}$$

□

Note that the bound on $\mathcal{W}_2(p_{\text{reflow}}, p^{\text{nf}}_\theta)$ is effectively the square-root of the FORT objective and thus optimization of the NF using this loss directly minimizes the upper bound to $\mathcal{W}_2(p_1, p^{\text{nf}}_\theta)$.

A.3. FORT in continuous time

Current state-of-the-art CNFs are trained using "flow-matching" (Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023; Liu et al., 2023), which attempts to match the vector field associated with the flow to a target vector field that solves for mass transportation everywhere in space and time. Specifically, we can cast conditional flow matching (CFM) (Tong et al., 2023) from the perspective of FORT. To see this explicitly, consider a pre-specified probability path, $p_t(x_t)$, and the following $f^*_{t,\text{fm}} = \frac{\partial}{\partial t} p_t(x_t)$. However, since it is generally computationally challenging to sample from p_t directly, the marginalization trick is used to derive an equivalent objective with a conditional $f^*_{t,\text{cfm}}$. We note that FORT requires $f^*_{t,\text{cfm}}$ to be invertible therefore this assumes regularity on $\frac{\partial}{\partial t} p_t(x_t)$. This is generally satisfied by adding a small amount of noise to the following. We present this simplified form for clarity.

$$p_t(x_t) := \int p_t(x_t|x_0, x_1) d\pi(x_0, x_1), \quad p_t(x_t|x_0, x_1) = \delta(x_t; (1-t)x_0 + t x_1). \tag{11}$$

Then setting $f^*_{t,\text{cfm}} = \frac{\partial}{\partial t} p_t(x_t|x_0, x_1)$ it is easy to show that:

$$\begin{aligned}
\mathcal{L}(\theta) &= \mathbb{E}_{t,x_0,x_1,x_t}\left[\left\|v_{t,\theta}(x_t) - \frac{\partial}{\partial t} p_t(x_t|x_0, x_1)\right\|^2\right] \\
&= \mathbb{E}_{t,x_t}\left[\left\|v_{t,\theta}(x_t) - \frac{\partial}{\partial t} p_t(x_t)\right\|^2\right] + C \\
&= \mathbb{E}_{t,x_0,x_1,x_t}\left[\lambda_t \left\|f_{t,\theta}(x_t) - f^*_{t,\text{cfm}}(x_t)\right\|^2\right]
\end{aligned}$$

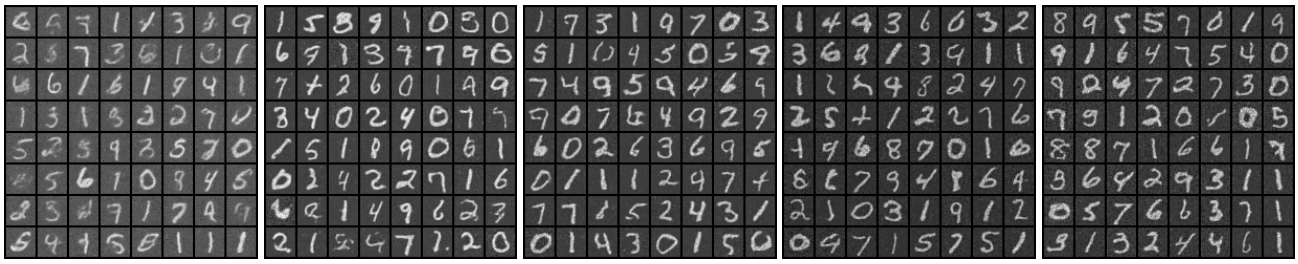
with C independent of θ (Lipman et al., 2023), and λ_t is a loss weighting, which fits within the FORT framework in the continuous-time setting with the last equality known as target/end-point prediction.

B. Additional Background

B.1. Inductive Moment Matching

Introduced in Zhou et al. (2025), Inductive Moment Matching (IMM) defines a training procedure for one-step generative models, based on diffusion/flow-matching. Specifically, IMM trains models to minimize the difference in distribution

[1] A sharper bound can be obtained with additional assumptions, as demonstrated in Benton et al. (2023), but it is not critically important in our context.



(a) 1-step. (b) 2-step. (c) 4-step. (d) 8-step. (e) 16-step.

Figure 6. Generations of IMM trained with an iUNet with a variable number of steps.

between different points in time induced by the model. As a result, this avoids direct optimization for the predicted endpoint, in contrast to conventional diffusion.

More precisely, let $f_\theta : \mathbb{R}^d \times [0,1]^2 \to \mathbb{R}^d, (x,s,t) \mapsto f_\theta(x,s,t)$ be a function parameterized by θ. IMM minimizes the following maximum mean discrepancy (MMD) loss:

$$\mathcal{L}(\theta_n) = \mathbb{E}_{s,t,x_0,x_1}\left[w(s,t)\mathrm{MMD}^2\left(p_{\theta_{n-1},(s|r)}(x_s), p_{\theta_n,(s|t)}(x_s)\right)\right], \tag{12}$$

where $0 \le r \le r(s,t) := r \le s \le 1$, with $s,t \sim \mathcal{U}(0,1)$ iid, $w \ge 0$ is a weighting function, x_1 is a sample from the target distribution, $x_0 \sim \mathcal{N}(0,I)$, x_s is some interpolation between x_0 and x_1 at time s (typically, using the DDIM interpolation (Song et al., 2022)), the subscript $n \in \mathbb{N}$ of parameter θ refers to its training step, and MMD is some MMD function based on a chosen kernel (typically, Laplace).[2] Essentially, the method uses as a target the learned distribution of the previous step at a higher time to train the current distribution at lower times. With a skip parameterization, the higher time distribution is by construction close to the true solution, as $p_\theta(x_s \mid x_r) \approx p(x_s \mid x_r)$ when $r \approx s$, and x_s is known. (Or, in other terms, $f_\theta(x,s,r \approx s) \approx x$ with the skip parameterization.) When the distributions match (when the loss is zero), $\mathrm{MMD}^2(p_{1,\theta}, p_1) = 0$, and so the generative model's and the target distribution's respective moments all match.

This training procedure allows for variable-step sampling. For chosen timesteps, $(t_i)_{i=1}^n$, one can sample from $p_{1,\theta}$ by sampling $x_0 \sim \mathcal{N}(0,I)$ and performing the steps

$$x_{t_{i+1}} \leftarrow \mathrm{DDIM}(f_\theta(x_{t_i}, t_{i+1}, t_i), x_{t_i}, t_i, t_{i+1}), \tag{13}$$

where DDIM is the DDIM interpolant.

B.2. Inductive Moment Matching negative results

We detail in Appendix B.1 the Inductive Moment Matching (IMM) framework (Zhou et al., 2025). Observing the sampling procedure, which we give in Equation (13), one can make this procedure invertible by constraining the Lipschitz constant of the model, or by using an invertible model. For the first case, if we use the "Euler" (skip) parameterization alongside the DDIM interpolation, it is shown that the reparameterized model g_θ can be written as:

$$\forall x,s,t, \qquad g_\theta(x,s,t) = x - (s-t)f_\theta(x,s,t). \tag{14}$$

Moreover, $0 \le s - t \le 1$, and so if the Lipschitz constant of f_θ is strictly less than one, then the overall model is invertible, using the argument of residual flows (Behrmann et al., 2019); so the change of variables formula applies as follows (using the time notation of IMM/diffusion):

$$\log p_1^\theta(x) = \log p_0(x_0) - \sum_i \log\left[(t_{i+1} - t_i)\det(J_{f_\theta(\cdot,t_{i+1},t_i)}(x_{t_i}))\right], \tag{15}$$

The difficulty of evaluating the log-determinant of the Jacobian remains. Note, however, that we do not need to find the inverse of the function to evaluate the likelihood of generated samples, since we know each $(x_{t_i})_i$. The second path (of using an invertible model) is viable only for one-step sampling with no skip parameterization (which, according to Zhou et al. (2025), tends to under-perform, empirically), since the sampling procedure then boils down to $x_1 = f(x_0, 1, 0)$ for $x_0 \sim \mathcal{N}(0,I)$.

[2] Note that we have adapted IMM's notation to our time notation, with noise at time zero, and clean data at time one.

(a) Using the ResFlow architecture proposed in Chen et al. (2020).

(b) Using the TarFlow architecture (Zhai et al., 2024), $m = 4$.

(c) Using the TarFlow architecture (Zhai et al., 2024), $m = 16$.

Figure 7. One-step generation results with a Lipschitz-constrained (ResFlow) model and an invertible model (TarFlow) for IMM. The m parameter is the group size in IMM used to approximate the MMD.

While both approaches succeeded in synthetic experiments, they fail to scale to datasets such as MNIST, the results of which we include here in Figure 6 and in Figure 7. We have tried iUNet (Etmann et al., 2020) and TarFlow (Zhai et al., 2024), an invertible UNet and a Transformer-based normalizing flow, respectively, for invertible one-step models; and we have tried the ResFlow architecture in (Chen et al., 2020) for the Lipschitz-constrained approach. As observed, TarFlow fails to produce images of high quality; iUNets produced significantly better results, albeit still not sufficient, especially for the one-step sampling, which is the only configuration which guarantees invertibility; the Lipschitz-constrained ResFlow entirely failed to produce satisfactory results, although the loss did diminish during training. In general, an even more important limitation to consider is the difficulty of designing invertible or Lipschitz-constrained models for other data types, for instance, 3D coordinates. Perhaps further research on the architectural side could allow for higher performance with invertible sampling.

C. Experimental Details

C.1. Metrics

The performance metrics considered across the investigated flows were the effective sample size, ESS, Wasserstein-1 energy distance, $\mathcal{E}$-$\mathcal{W}_1$, and the Wasserstein-2 distance on dihedral angles, $\mathbb{T}$-$\mathcal{W}_2$.

Effective Sample Size (ESS) We compute the effective sample size (ESS) using Kish's formula, normalized by the number of samples generated:

$$\mathrm{ESS}\left(\{w_i\}_{i=1}^N\right) = \frac{1}{N}\frac{\left(\sum_{i=1}^N w_i\right)^2}{\sum_{i=1}^N w_i^2}. \tag{16}$$

where w_i is the unnormalized weight of each particle indexed by i over N particles. Effective sample size measures the variance of the weights and approximately how many more samples would be needed as compared to an unbiased sample. For us, this captures the local quality of the proposal relative to the ground truth energy. It does not rely on a ground truth test set, however is quite sensitive and may be misleading in the case of dropped modes or incomplete coverage as it only measures agreement on the support of the generated distribution.

Wasserstein-1 Energy Distance ($\mathcal{E}$-$\mathcal{W}_1$) The Wasserstein-1 energy distance measures how well the generated distribution matches some ground truth sample (often generated using MD data) by calculating the Wasserstein-1 distance between the energy histograms. Specifically:

$$\mathcal{E}\text{-}\mathcal{W}_1(x, y) = \min_{\pi} \int_{x,y} |x - y| d\pi(x, y) \tag{17}$$

where π is a valid coupling of $p(x)$ and $p(y)$. For discrete distributions of equal size, π can be thought of as a permutation matrix. This measures the model's ability to generate very accurate structures as the energy function we use requires extremely accurate bond lengths to obtain reasonable energy values. When the bond lengths have minor inaccuracies, the energy can blow up extremely quickly.

Torus Wasserstein ($\mathbb{T}$-$\mathcal{W}_2$) The torus Wasserstein distance measures the Wasserstein-2 distance on the torus defined by the main torsion angles of the peptide. That is for a peptide of length l, there are $2(l-1)$ torsion angles defining the *dihedrals* along the backbone of interest $((\phi_1, \psi_1), (\phi_2, \psi_2), \dots (\phi_l, \psi_l))$. We define the torus Wasserstein distance over these backbone angles as:

$$\mathbb{T}\text{-}\mathcal{W}_2(p,q)^2 = \min_{\pi} \int_{x,y} c_{\mathcal{T}}(x,y)^2 d\pi(x,y) \tag{18}$$

where π is a valid coupling between p and q, and $c_{\mathcal{T}}(x,y)^2$ is the shortest distance on the torus defined by the dihedral angles:

$$c_{\mathcal{T}}(x,y)^2 = \sum_{i=0}^{2(L-1)} \left[(\text{Dihedrals}(x)_i - \text{Dihedrals}(y)_i + \pi) \bmod 2\pi - \pi\right]^2. \tag{19}$$

The torus Wasserstein distance measures large scale changes and is quite important for understanding mode coverage and overall macro distribution. We find FORT does quite well in this regard.

C.2. Additional details on experimental setup

To accurately compute the previously defined metrics, 250k proposal samples were drawn and re-weighted for alanine dipeptide, tripeptide, and tetrapeptide.

Data normalization We adopt the same data normalization strategy proposed in (Tan et al., 2025), in which the center of mass of each atom is first subtracted from the data, followed by scaling using the standard deviation of the training set.

Exponential moving average We apply an exponential moving average (EMA) on the weights of all models, with a decay of 0.999, as commonly done in flow-based approaches to improve performance.

Training details and hardware All models were trained on NVIDIA L40S 48GB GPUs for 5000 epochs, except those using OT targets, which were trained for 2000 epochs. Convergence was noted earlier in the OT experiments, leading to early stopping. The total training time for all models is summarized in Table 4. The time taken to compute the OT map is also provided; since computing the OT map is independent of the feature dimension, but only on the number of data points used, the compute time was relatively consistent across all datasets. A total of 100k points was used for training the CNF, performing MLE training, and computing the OT map.

Table 4. FORT training time (in hours) on ALDP, AL3, and AL4.

Model	ALDP	AL3	AL4
OT map	3.6	3.8	3.8
DiT CNF	27.6	40.7	48.6
NSF	21.0	23.8	26.8
Res–NVP	15.7	15.6	15.0
Jet	19.1	19.2	20.1

Reflow targets Ablations were done to investigate the influence of synthetic data quantity on all metrics. For all benchmarking performed against MLE training, the largest amount of synthetic data was used. For ALDP, AL3, and AL4, this constituted 10.4M, 10.4M, and 10M samples, respectively.

Determinant regularization During FORT, it was initially observed that as proposal sample quality improved, the re-weighted samples progressively deteriorated across all metrics due to the models becoming numerically non-invertible. This was partially addressed by adding regularization to the loss in the form of a log determinant penalty. Sweeps were conducted using multiple regularization weights ranging between 10^{-7} and 10^{-4} to prevent hampering sample performance. The amount of regularization added was a function of the flow and dataset. The final weights are summarized in Table 5.

Table 5. Regularization weights used across datasets and flows.

Model	ALDP	AL3	AL4
NSF	10^{-6}	10^{-5}	10^{-5}
Res–NVP	10^{-5}	10^{-5}	10^{-6}
Jet	10^{-5}	10^{-6}	10^{-5}

Target noise To discourage numerical non-invertibility of the trained flows, Guassian noise was also introduced to the target samples. Experiments were conducted with noise magnitudes of 0.01, 0.05, 0.1, and 0.25, with a final value of 0.05 being selected for use across models and datasets.

FORT implementation details A summary of all trained model configurations is provided in Table 6. To maintain a fair comparison, the configurations reported below were unchanged for MLE training and FORT. Adam was used as the optimizer with a learning rate of 5×10^{-4} and a weight decay of 0.01. We also included a varying cosine schedule with warmup in line with the approach suggested in (Tan et al., 2025).

Table 6. Model configurations for the DiT CNF, NSF, Res–NVP, and Jet across all datasets (ALDP, AL3, AL4). A dash (–) indicates the parameter is not applicable to the respective model.

Model	hidden features	transforms	layers	blocks per layer	conditioning dim.	heads	dropout	**# parameters (M)**
DiT CNF	768	–	6	–	128	12	0.1	46.3
NSF	256	24	–	5	–	–	–	76.8
Res–NVP	512	–	8	6	–	–	0.1	80.6
Jet	432	–	4	12	128	12	0.1	77.6

Quality of CNF targets To maximize the likelihood that models trained with FORT have the potential to outperform MLE, securing high quality targets is essential. In line with this pursuit, a CNF with a diffusion transformer backbone was used. In Figure 8, the true data and the CNF proposal are shown, where it can be seen that the learned energy distributions across all three peptides are nearly perfect. Re-weighted samples are not included as obtaining likelihoods from the CNF requires estimating the trace of the divergence, which is often an expensive operation with a large time and memory cost. Although many unbiased approaches for approximating the likelihood exist (Hutchinson, 1989), these methods are typically unusable for Boltzmann generators due to their variance, which introduces bias into the weights needed for SNIS.

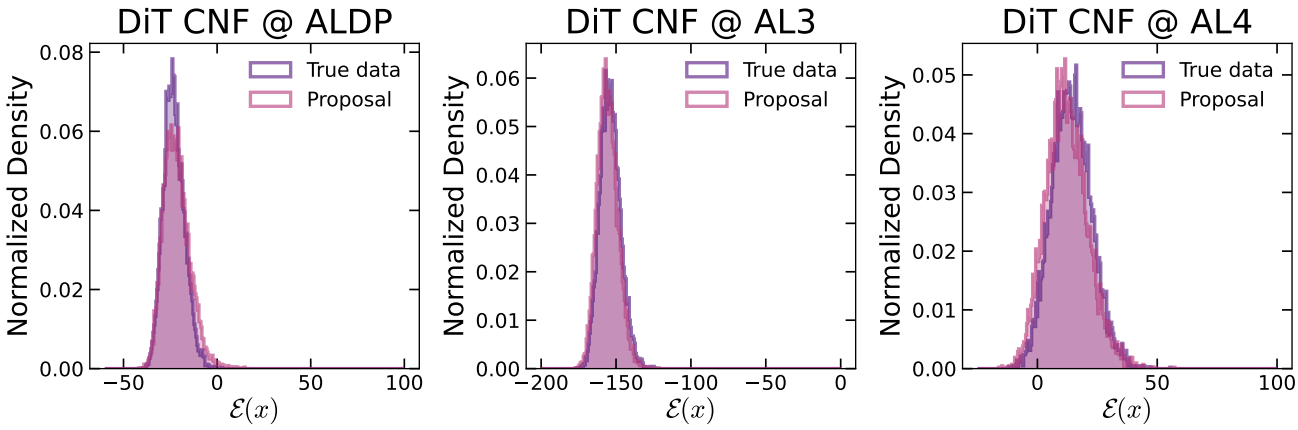


Figure 8. True energy distribution and learned proposal using the DiT-based CNF. *The re-weighted proposal is not present because it was too computationally expensive to compute for a sufficient number of points.

D. Additional Results

D.1. Generated samples of peptide conformations

Samples of generated peptides Below we provide sample conformations of alanine dipeptide, alanine tripeptide, and alanine tetrapeptide generated using both MLE training and FORT.

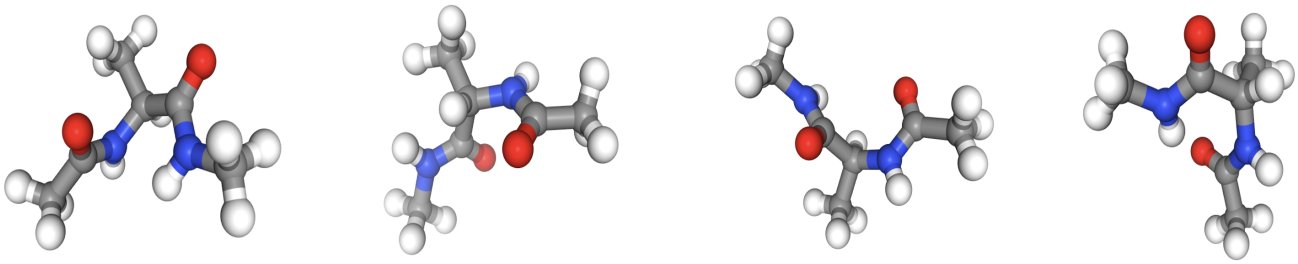


Figure 9. Generated conformations of alanine dipeptide across various flow-based methods (**left**: NSF with MLE; **center left**: NSF with FORT; **center right**: Res–NVP with FORT; **right**: Jet with FORT).

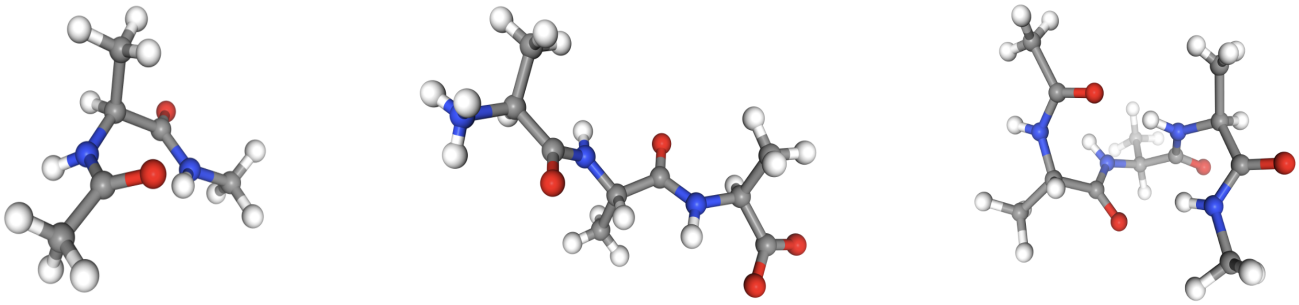


Figure 10. Generated samples of various sized peptides using the most performant FORT flow (**left**: alanine dipeptide; **center**: alanine tripeptide; **right**: alanine tetrapeptide).

D.2. Improved proposals using FORT for larger peptides

FORT outperforms MLE for AL3 and AL4 In Figure 11, we demonstrate that the energy distribution of the re-weighted samples using FORT yields a more favourable energy distribution over MLE-trained flows. For the tripeptide, the results are in strong agreement with the MD data. For the tetrapeptide, the re-weighted samples are superior than their MLE counterparts, but have some room for improvement in perfectly matching the true MD energy distribution.

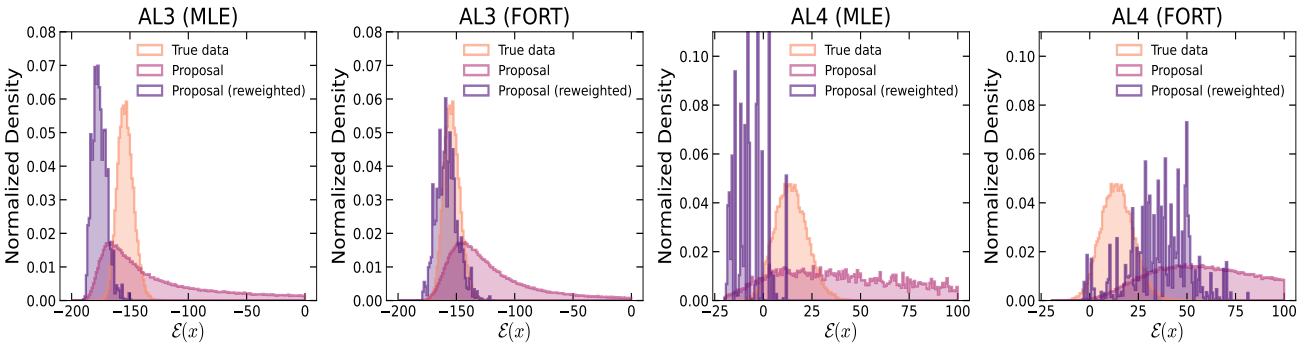


Figure 11. Energy distribution of original and re-weighted samples generated for the most performant MLE and FORT models on alanine tripeptide (**left** and **center left**) and alanine tetrapeptide (**center right** and **right**).

D.3. Ramachandran plots for larger peptides

Alanine tripeptide (AL3) We demonstrate the learned distributions of the two pairs of dihedral angles that parameterize AL4 using our best MLE-trained and FORT flows. The inability to capture the modes using MLE is elucidated, where multiple modes appear to blend together in both sets of dihedral angles. Conversely, using FORT, most modes are accurately captured and the general form of the Ramachandran plots conforms well to that of the true distribution obtained from MD.

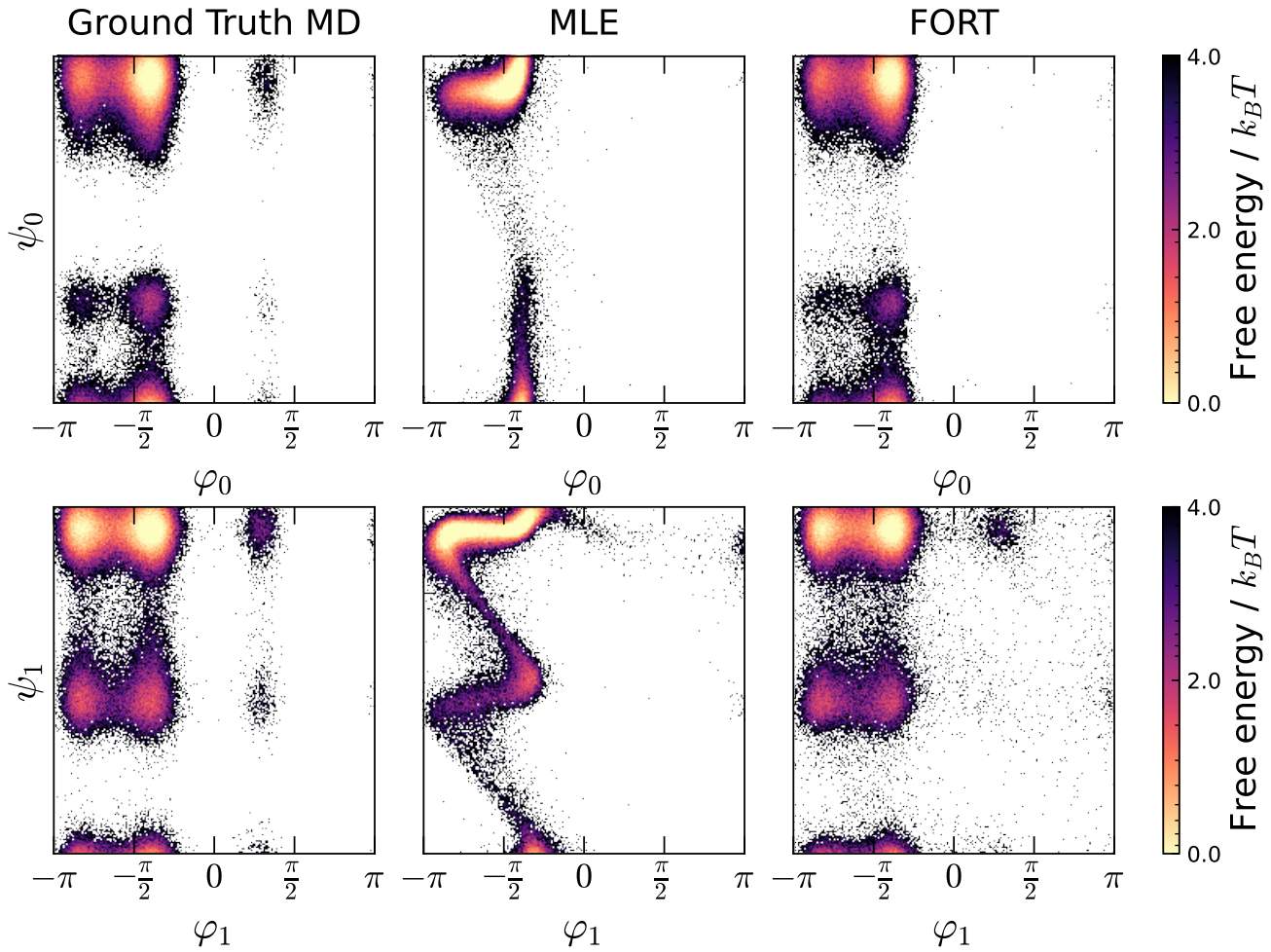


Figure 12. Ramachandran plots for the dihedral angles in AL3 (**left**: ground truth, **middle**: best MLE-trained flow, **right**: best FORT flow.

Alanine tetrapeptide (AL4) The findings observed with alanine tripeptide are even more pronounced with alanine tetrapeptide, where certain modes are entirely missed when MLE-trained flows are used. With FORT, however, most modes are accurately captured and the density distribution is in strong agreement with the ground truth data. These findings support the utility of regression-based training over conventional MLE for applications to equilibrium conformation sampling.

D.4. Targeted free energy perturbation

Generating regression targets Using the available MD data, two conformations of alanine dipeptide were selected: β_{planar} and α_R (Ghamari et al., 2022). The (ϕ, ψ) ranges for the β_{planar} conformation were chosen as $(-2.5, -2.2)$ and $(2.3, 2.6)$, and for the α_R conformation as $(-1.45, -1.2)$ and $(-0.7, -0.4)$, respectively. The dataset was then truncated to 82,024 source-target conformation pairs, which were used to compute the OT pairing and generate an invertible map. These pairs were subsequently trained using FORT, with the same model configurations and settings outlined in Table 6.

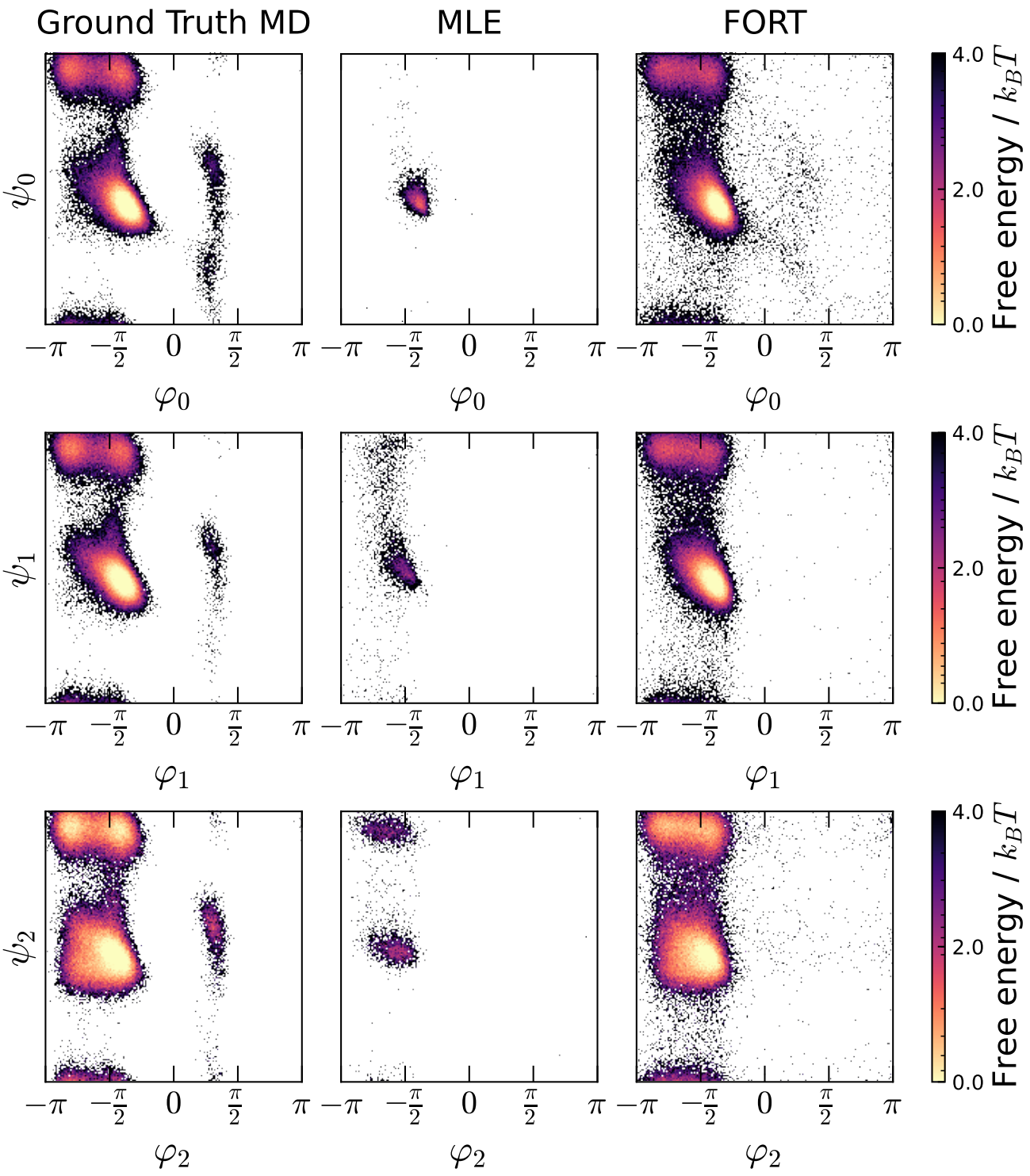


Figure 13. Ramachandran plots for the dihedral angles in AL4 (**left**: ground truth, **middle**: best MLE-trained flow, **right**: best FORT flow.

Supplement References

J. Benton, G. Deligiannidis, and A. Doucet. Error bounds for flow matching methods. *arXiv preprint arXiv:2305.16860*, 2023.

D. Ghamari, P. Hauke, R. Covino, and P. Faccioli. Sampling rare conformational transitions with a quantum computer. *Scientific Reports*, 12(1):16336, 2022.

M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.