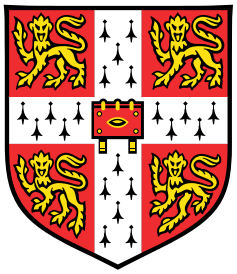
Optimal Path Flow for Multimodal Generation



Krisztina Sinkovics

Supervisor: Prof. Pietro Liò

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy in Machine Learning and Machine Intelligence

Trinity College August 2024

I would like to dedicate this thesis to the relentless pursuit of one's dreams,
regardless of background, age, gender, ethnicity, nationality or faith.

Declaration

I, Krisztina Sinkovics (Kristina Shinkovych) of Trinity College, being a candidate for the Master of Philosophy in Machine Learning and Machine Intelligence, hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this or any other university. In preparation of this project report I did not use text from AI-assisted platforms generating natural language answers to user queries, including but not limited to ChatGPT.

This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 10,761 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

The code for this thesis build on the basis of FlowMol codebase (Dunn and Koes, 2024).

Krisztina Sinkovics
August 2024

Acknowledgements

I would like to express my gratitude to my supervisor Professor Pietro Lio, for his guidance and support throughout the process of writing this thesis. I would also like to thank my cosupervisors Tor Fjelde, Charlie Harris and Emile Mathieu, for their patient explanations and valuable feedback.

I must also give a heartfelt thanks to my mentors Sergei and Ali for helping me navigate this process. I am grateful to my brother Rikhard for his support and jokes and to my dear friends Luca, Vasyl, Viktor, Miguel and Alicia, without whom this journey would not have been the same.

Abstract

Generating high-quality biomolecules is an important step in advancing the drug discovery process and broader domains in natural sciences. This task becomes particularly challenging when an increasing number of data modalities are involved (e.g. atom type, positions etc), growing combinatorial in complexity due to the need for choosing effective interpolation schedules for multiple intertwined modalities at training and inference time. Current methods rely on manual tuning and heuristics. We propose to automate this process by optimizing the transport path with respect to an objective derived on the basis of dynamic optimal transport (OT) by training an interpolation schedule parametrized by a neural network.

Our approach, based on the work of Albergo and Vanden-Eijnden (2023) and Albergo et al. (2023a), decouples the choice of path from the design of vector field, allowing a single vector field to be trained over all possible paths. This vector field is then leveraged to learn an optimal path by minimizing the Optimal Transport loss to generate straighter trajectories in Euclidean space.

We validated our method on a toy Gaussian problem, confirming that optimizing the path reduced OT loss and resulted in straighter trajectories. Extending this approach to small molecule generation, we found that path optimization lead to consistent interpolation schedules that reduce the OT loss. However, we observed that the reductions in OT loss did not automatically translate into significant improvements in sampling efficiency or sample quality, revealing areas for future investigation.

We proposed strategies that address potential sources of error, namely improving the accuracy of the vector field and addressing the relationship between the path optimization objective and integration errors, including exploring alternative path optimization norms. Additionally, we highlighted the importance of considering local strain energy in molecular design, as positional inconsistencies can impact quality. Since most drug discovery efforts work in the low data domains under resource constraints, focusing on efficient, simulation-free methods that can jointly model multiple modalities remains crucial.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms / Abbreviations

CFM Conditional Flow Matching

EGNN Equivariant Graph Neural Network

FM Flow Matching

GDL Geometric Deep Learning

GNN Graph Neural Network

GVP Geometric Vector Perceptron

MPNN Message Passing Neural Network

ODE Ordinary Differential Equation

OT Optimal Transport

PB Posebusters (referred to molecular validity test)

SDE Stochastic Differential Equation

TE Transport Equation

VF Vector Field

Chapter 1

Introduction

1.1 Motivation and Overview

Deep generative models aim to generate novel samples by learning from data. Recent advances in diffusion models for image and video generation have significantly increased the popularity of these methods and triggered extensions to other domains. Denoising diffusion probabilistic models and score-based diffusion models rely on a forward noising process over samples from the data distribution and a time-reversal stochastic process to map samples from a noise distribution back to the desired target distribution. However, these methods require solving a stochastic differential equation (SDE) at inference time, limiting the possibility to get high quality samples in few discretisation steps. What is more, they are constrained in the choice of the noise source, which needs to be an invariant distribution of the noising process. This creates challenges when dealing with a combination of continuous and discrete data, a common scenario in real-world problems across the natural sciences.

These limitations were addressed in Flow Matching (FM) (Lipman et al., 2023, Albergo et al., 2023b, Liu, 2022), which is a computationally efficient generative paradigm based on continuous flows induced by ordinary differential (ODE) equations, but alleviates the need to simulate the ODE during training by directly regressing the vector field that gives rise to the generative paths. Further extensions by Tong et al. (2024) and Albergo and Vanden-Eijnden (2023) propose more general paths which allow for an arbitrary source density, rather than requiring a Gaussian base by the construction of the path. The simplicity of their framework allows for much more flexibility in the design of interpolants. This makes Flow Matching a perfect candidate for building generative models over multiple modalities to tackle problems prevalent in drug discovery and material sciences.

Such a generative model can learn the distribution over the space of biomolecules and, therefore, generate valid biomolecules. By conditioning these models, one can tackle com-

plex tasks, such as designing binders with specific properties for particular targets. Sampling molecular structures with desired properties have the potential to accelerate chemical discovery by reducing the need to engage in resource-intensive screening-based based discovery paradigms. Application of generative models alongside geometric deep learning techniques has been particularly successful in generating credible biomolecule structures under external constraints, leading to a rapid expansion of this field of research (Torge et al., 2023, Martinkus et al., 2023, Corso et al., 2023, Didi et al., 2024). Yet the challenge of making better biomolecules that exhibit desired properties and adhere to desired constraints remains.

Jointly modeling multiple modalities would enable the generation of meaningful samples without the need run the generated 3D structure through a Protein Message Passing Neural Network to get amino acids (Dauparas J, 2022). Current approaches such as FrameDiff (Yim et al., 2023b), FrameFlow (Campbell et al., 2024), (Bose et al., 2024), MiDi (Vignac et al., 2023a), FlowMol (Dunn and Koes, 2024) require manually selecting an interpolation schedule between the source and the target density for each of these modalities. Doing so has a significant impact on model performance. This becomes increasingly difficult since the problem of choosing schedules grows combinatorially with the number of modalities, further complicated by the fact that schedules live in the space of functions, and at present most methods rely on different heuristics (Karras et al., 2022), (Vignac et al., 2023a), (Dunn and Koes, 2024), (Yim et al., 2023a). Learning a schedule over all modalities jointly would allow one to bypass manually tuning the schedule, but also to find one that could yield even better samples.

Optimizing interpolation paths has been initially explored in (Albergo et al., 2023a) in a multi-marginal setting with one modality. We build on their word to naturally extend this framework to the multi modalities setting. This requires decoupling the path from the vector field along the theoretical findings by Albergo et al. (2023a). This approach enables training a vector field over all possible paths for every modality, offering the advantage of training a single vector field rather than separate fields for each combination of paths. Then at inference time, one can choose any path of their liking. In particular, we suggest to learn the transport path which minimizes the OT loss, sinceLipman et al. (2023), Tong et al. (2024) demonstrate that using Optimal Transport (OT) displacement interpolation to define the conditional probability paths leads to faster training and sampling and results in better generalization.

1.2 Main Contributions

The thesis presents several key contributions to the field of generative modeling for small molecules:

1. Our approach allows us to select any path at inference time by amortising the vector field neural network trained over all possible paths. This framework, applied to generating small molecules, extends the original methodology introduced by Albergo et al. (2023a) to the multiple modalities setting.

2. This work proposes a method for automatically learning an optimal interpolation path for each modality by minimizing the Optimal Transport (OT) loss, which extends the methodology proposed by Albergo et al. (2023a) into multimodal setup.

3. We investigate the effect of the choice of path at training and inference time on the sampling efficiency and quality of small molecule generation evaluated through the lens of relevant metrics such as Posebusters (PB) validity and Strain Energy (SE).

4. Finally, we outline future research directions, focusing on potential improvements in vector field approximation and alternative path optimization strategies for more efficient and accurate molecular generation.

1.3 Thesis Outline

This thesis is structured as follows:

Chapter 1: Introduction. This chapter introduces the core motivation behind our research. It explains the challenges associated with interpolation schedules in multimodal settings, particularly the complexity that arises when dealing with multiple entangled modalities. This chapter gives an overview the proposed methodology, which seeks to address these issues through an automatically learned schedule using neural networks. Additionally, it summarizes the main contributions of the thesis, which focus on improving the efficiency and quality of sample generation.

Chapter 2: Background. The background chapter provides an overview of the theoretical foundations that inform this research. It discusses the flow matching framework and the role of neural ODEs and probability flows between distributions. The chapter also covers the background of optimal transport, introduces stochastic interpolants, and explains how they allow to decouple the choice of path from the vector field and lays the foundations for our methodology. It also provides an overview of Geometric Deep Learning (GDL), symmetry

groups and their importance in the design of equivariant Graph Neural Networks (GNNs). These concepts prepare the basis for applications of generative models in biomolecular design.

Chapter 3: Optimal Path Flow: Methodology and Evaluation. In this chapter, the core methodology is introduced, focusing on the two-step approach of training a vector field over all possible paths and then using this vector field to optimizing interpolation schedule parametrized by a neural network. The chapter illustrates the performance of the proposed method in two main case studies. First, a simple scenario involving 2D Gaussian distributions is used to validate the method and demonstrate its effectiveness. The second case study applies the methodology to the more complex task of generating small molecules, investigating the relationship between the learned interpolation paths, OT loss, sampling efficiency, and sample quality.

Chapter 4: Concluding Remarks. This chapter discusses the strengths and limitations of the proposed method, reflecting on the results from both the 2D Gaussian and molecular generation case studies. It explains the reasons why some of the findings from the 2D Gaussian setup did not translate into the gains in sampling efficiency and sample quality and suggests approaches to address potential sources of error. Future research directions are outlined, focusing on refining vector field learning, exploring alternative objectives for path optimization, and considering local strain energy in molecular design.

Chapter 2

Background

2.1 Flow Matching Framework

Flow Matching is a generative modeling paradigm introduced by Lipman et al. (Lipman et al., 2023). It combines ideas from Continuous Normalizing Flows (Chen et al., 2019) and Diffusion Models Song and Ermon (2019), Ho et al. (2020). The former requires expensive ODE simulations during training time, which makes them very slow. Score-based diffusion models, on the other hand, requires expensive SDE simulations during inference. Flow Matching is a faster simulation-free method that alleviates the need for expensive Ordinary Differential Equation (ODE) integration during training or SDE simulations during inference. Moreover, conditional Flow matching does not require the source distribution to be Gaussian, nor does it require evaluation of p_0 density.

2.1.1 Neural ODEs and Probability Flows Between Distributions

Generative modeling is a rapidly growing field, offering a range of solutions to the task of fitting a mapping f between the source distribution q_0 and the target distribution q_1, such that if $x_0 \sim q_0$, then $f(x_0) \sim q_1$. Typically, the source distribution is either Gaussian or other distribution that is easy to sample from. However, cases when both q_0 and q_1 are empirical distributions represented by a finite set of samples are also considered.

Neural ODEs provide a powerful method to model continuous transformation of the data, parameterized by differential equations, smoothly evolving samples from the source density to the target density.

Definition: A smooth time-varying *vector field*[a] $u : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ defines the following ODE:

$$\frac{d}{dt}\phi_t = u_t(\phi_t(x)), \quad \forall t \in [0,1] \tag{2.1}$$

where the *flow maps* $\phi_t(x)$ are solutions to ODE with the initial condition $\phi_0(x) = x$.

Transporting a sample x along the vector field u from time 0 to time t gives $\phi_t(x)$.

[a]Vector field is also known as velocity (velocity field) or drift in diffusion and SDE literature.

We are interested in learning a sequence of generative models (i.e. probability densities) $(p_t)_{t=0}^1$ for $t \in [0,1]$ such that $p_0 = q_0$ represents the source density and $p_1 = q_1$, the data density.

Definition: *Probability density path* p_t is said to be generated by a vector field u_t if, given a source density p_0, its flow ϕ_t satisfies the push-forward equation:

$$p_t = [\phi_t]_{\#} p_0 \tag{2.2}$$

The push-forward (also known as change of variables operator) is defined as

$$[\phi_t]_{\#} p_0 = p_0(\phi_t^{-1}(x)) \det \left[\frac{\partial \phi_t^{-1}}{\partial x}(x) \right]$$

The flow $\phi_t : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ transports the initial density of points $x \sim p_0$ along the vector field u form time 0 to time t:

$$x_t \triangleq \phi_t(x_0) = x_0 + \int_0^t u_s(x_s) ds$$

which is the ODE solution at time t.

Remark: We require $u_t(x)$ to be smooth to ensure the invertibility of the resulting flow ϕ_t. This implies that u_t must be at least locally Lipschitz in x and Bochner integrable[1] in t (Tong et al., 2024).

The time-evolution of the density p_t, which can also be viewed as a function $p : [0,1] \times \mathbb{R}^d \to \mathbb{R}$, is characterized by *Continuity Equation*, also generally known as the *Transport Equation (TE)*[2]:

$$\frac{\partial p_t(x)}{\partial t} = -\nabla \cdot (u_t(x) p_t(x)) \tag{2.3}$$

[2]From here onwards, we will use the term Transport Equation instead of Continuity Equation since our focus is transporting points from source to target density.

and the initial conditions p_0.

Remark: The Transport Equation (2.3) is a Partial Differential Equation (PDE) that provides a necessary and sufficient condition for vector field u_t to generate probability path p_t (Villani, 2008).

Under these conditions, $p_t(x)$ represents the marginal probability path generated by an ODE defined by $u_t(x)$ with initial state sampled from p_0.

To approximate this ODE, we can employ a neural network to model a time-dependent vector field $v^\theta : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ parameterized by θ. The vector field network must be designed with Lipschitz continuity to ensure the rate of change remains bounded, preventing abrupt shifts between points and preserving smooth transitions.

In *Continuous Normalizing Flows (CNFs)* this parametric vector field $v^\theta(t,x)$ is trained via maximum likelihood estimation

$$\mathscr{L}(\theta) = \mathbb{E}_{x\sim q_1}[\log p_1(x)]$$

This training process requires integrating both the time evolution of x_t and the log-likelihood $\log p_t$, both of which are derived from $v^\theta(t,x)$. The drawbacks of CNFs are that they involve expensive ODE simulations during training time and the method does not scale well to high-dimensional spaces.

On the other hand, *Flow Matching (FM)* offers an alternative by circumventing the need for ODE simulations during training time. Instead, it directly regresses the vector field $u_t(x)$ using the following **Flow Matching objective (Loss-FM)**:

$$\mathscr{L}_{\text{FM}} = \mathbb{E}_{t\sim U(0,1), x\sim p_t(x)}\left[\|u_t(x) - v^\theta(t,x)\|^2\right] \tag{2.4}$$

However, this approach assumes that the probability path $p_t(x)$ and the vector field $u_t(x)$ generating it are both known and that $p_t(x)$ can be tractably sampled, which is not feasible in most cases.

2.1.2 Conditional Flow Matching

In practice, we do not have direct access to the marginal vector field u_t which generates the desired density p_t. Therefore, Lipman et al. (2023) proposed to construct both p_t and u_t by aggregating a mixture of simpler probability paths and vector fields defined per sample, referred to as *conditional probability paths* $p_t(x \mid x_1)$ and the *conditional vector fields* $u_t(x \mid x_1)$.

Remark: While Lipman et al. (2023) constructed Conditional Flow Matching assuming that the source distribution q_0 is a Gaussian and conditioning the per-sample vector fields and probability paths on the target samples x_1, Tong et al. Tong et al. (2024) extended the Conditional Flow Matching framework to a more general setting. Their approach relaxes the constraints on the form of q_0 and generalizes the conditioning to some variable to $z \sim q(z)$. In Tong et al.'s notation, the *conditional probability paths* have the form $p_t(x \mid z)$ and the *conditional vector fields* are $u_t(x \mid z)$, where z is a combination of the initial and terminal points (x_0, x_1). The coupling $q(z)$ can take a flexible form as long as it has marginals $q(x_0)$ and $q(x_1)$. In this work, we adopt Tong et al.'s more general formulation.

The conditional probability path, generated by the conditional vector field, also satisfies the transport equation:

$$\frac{\partial p_t(x \mid z)}{\partial t} = -\nabla \cdot (u_t(x \mid z) p_t(x \mid z))$$

Definition: The *marginal probability path* p_t can be obtained from the *conditional probability paths* by marginalizing over $q(z)$ in the following way:

$$p_t(x) = \int p_t(x \mid z) q(z) dz, \tag{2.5}$$

At time $t = 1$, the marginal probability p_1 is a mixture distribution that closely approximates the data distribution q_1,

$$p_1(x) = \int p_1(x \mid z) q(z) dz \approx q_1(x).$$

Definition: The *marginal vector field* $u_t(x)$ can be obtained by marginalizing over the *conditional vector fields* $u_t(x \mid z)$ in the following way:

$$u_t(x) = \mathbb{E}_{z \sim p_{1|t}}[u_t(x \mid z)] = \int u_t(x \mid z) \frac{p_t(x \mid z) q(z)}{p_t(x)} dz \tag{2.6}$$

Theorem: Given the conditional vector fields $u_t(x \mid z)$ which generate the conditional probability paths $p_t(x \mid z)$ for any conditioning distribution q, the marginal vector field $u_t(x)$ obtained via (2.6) generates the marginal probability path $p_t(x)$ obtained via equation 2.5.

The proof relies on demonstrating that marginal $u_t(x)$ and $p_t(x)$ satisfy the transport (2.3). See appendix A.1 for detailed proof.

Example: In Conditional Flow Matching, Tong et al. (Tong et al., 2024) models conditionals z as Gaussian flows between x_0 and x_1 with standard deviation σ, characterized by the following:

$$p_t(x \mid z) = \mathcal{N}(x|\mu_t, \sigma_t^2) = \mathcal{N}(x|tx_1 + (1-t)x_0, \sigma_t^2)$$

$$u_t(x \mid z) = x_1 - x_0$$

where $\mu_t = tx_1 + (1-t)x_0$ represents a conditional probability path (or interpolant), and σ_t is a sequence of monotonically decreasing noise levels. As $t \to 0$, the noise approaches zero, i.e. $\sigma_1 \approx 0$, and:

$$\lim_{t\to 1} p_t(x \mid z) \approx \delta_{x_1}(x)$$

Theorem 2.1.2 enables us to replace the intractable vector field $u_t(x)$ in the Flow Matching Objective (2.4) with a tracTable conditional vector field $u_t(x \mid z)$, marginalizing over z to obtain **Conditional Flow Matching objective (Loss-CFM)**:

$$\mathscr{L}_{\text{CFM}} = \mathbb{E}_{t\sim U(0,1), z\sim q(z), x\sim p_t(x|z)} \left[\| u_t(x \mid z) - v^\theta(t,x) \|^2 \right] \tag{2.7}$$

where $v^\theta(t,x)$ is a neural network parameterized vector field.

Because both $p_t(x \mid z)$ and $u_t(x \mid z)$ are defined on a per-sample basis, we can efficiently sample from $p_t(x \mid z)$ and compute $u_t(x \mid z)$, which allows us to obtain an unbiased estimator using the tracTable Loss-CFM. Moreover, Loss-FM (2.4) and Loss-CFM (2.7) have identical gradients w.r.t θ, which is formulated in theorem 2.1.2 in accordance with Lipman et al. (2023), Theorem 2.

Theorem: Assuming that $p_t(x) > 0, \forall x \in \mathbb{R}^d$ and $t \in [0,1]$, then, up to a constant independent of θ, objectives Loss-CFM and Loss-FM are equivalent in the sense that

$$\nabla_\theta \mathscr{L}_{\text{CFM}}(\theta) = \nabla_\theta \mathscr{L}_{\text{FM}}(\theta)$$

Thus, we can use Loss-CFM to train the parametric vector field $v^\theta(t,x)$.

2.1.3 Static and Dynamic Optimal Transport

Optimal Transport (OT) is a problem that seeks to devise a mapping from one measure to another that minimizes the cost of displacing mass. This concept is particularly useful

when the goal is to find the most efficient mapping between two distributions q_0 and q_1. In *Static Optimal Transport*, this task is approached without explicitly considering time or the evolution of the transformation process. The 2-Wasserstein distance offers a convenient framework for optimizing the displacement cost between q_0 and q_1 in $\mathbb{R}^d$ using the Euclidean distance $c(x_0, x_1) = \|x_0 - x_1\|$ as a cost of moving mass from x_0 to x_1. Static OT is formulated as the following optimization problem:

$$W(q_0, q_1)^2 = \inf_{\pi \in \Pi(q_0, q_1)} \int_{\mathbb{R}^d \times \mathbb{R}^d} c(x_0, x_1)^2 \, d\pi(x_0, x_1),$$

where π is a coupling between a pair (x_0, x_1), Π represents the set of all joint probability measures on $\mathbb{R}^d \times \mathbb{R}^d$ whose marginals are q_0 and q_1. This conveniently aligns with the requirement on conditional flow matching coupling $q(z)$ between the source q_0 and the target q_1 in remark 2.1.2.

Static OT provides a transport plan or map that moves mass directly from one point in q_0 to a point in q_1, without explicitly modeling the time evolution of probability distribution. In contrast, *Dynamic Optimal Transport* introduces a time component to the transport problem, making it a continuous path problem, often described by a flow of mass or trajectory between distributions.

Definition: *Dynamic Optimal Transport* seeks to minimize the cost of transporting the mass over the entire path, subject to continuity constraints. The *2-Wasserstein distance in a dynamic setting* can be expressed as an optimization problem over vector fields u_t that transform q_0 into q_1

$$W(q_0, q_1)^2 = \inf_{p_t, u_t} \int_{\mathbb{R}^d} \int_0^1 p_t(x) \|u_t(x)\|^2 \, dt \, dx, \tag{2.8}$$

where $p_t \geq 0$ and subject to boundary conditions $p_0 = q_0$ and $p_1 = q_1$ as well as the Transport Equation 2.3

It has been proven by Benamou and Brenier (2000) that, under the assumption that q_0 and q_1 are compactly supported distributions with bounded densities, the static and dynamic OT formulations are equivalent.

In their Conditional Flow Matching framework, Tong et al. (2024) consider both independent coupling as well as OT coupling when designing the conditioning variable z for the velocity field $u_t(x \mid z)$ and the resulting path $p_t(x \mid z)$ that it generates.

Definition: *Independent Coupling* is defined as $q(x_0)q(x_1)$, with marginals q_0 and q_1.

OT Coupling is a coupling $\pi(x_0,x_1)$ given by OT plan between the source and target points, with marginals q_0 and q_1.

In OT-CFM, Tong et al. propose using the 2-Wasserstein optimal transport map π as a conditional distribution, i.e. $q(z) := \pi(x_0,x_1)$, where pairs (x_0,x_1) are jointly sampled according to π. While static OT plan can be computed exactly for small datasets, such as single-cell data, however, it becomes impractical for larger datasets due to the computational complexity. Specifically, computing OT scales cubically in time and quadratically in memory with respect to the number of samples (Cuturi, 2013). To address this, Tong et al. (2024) propose approximating the transport map using minibatch OT, where transport plan π_{batch} is computed for each batch of data, significantly reducing the computational burden.

2.2 Interpolation Between Densities and Optimal Transport

2.2.1 Stochastic Interpolants

From the perspective of optimal transport, the challenge in generative modeling lies in designing a computationally efficient structure imposed on the transport between a simple source density and a complex target density. This challenge can be approached by devising a time-dependent map that interpolates between two arbitrary densities. Same as previously, our goal is to construct probability path p_t, in particular, a sequence of probability densities $(p_t)_{t=0}^1$ for $t \in [0,1]$ such that $p_0 = q_0$ and $p_1 = q_1$, and we achieve this through a sequence of conditional probability paths referred to as interpolants x_t, which connect samples from p_0 to samples from p_1. This time-dependent map x_t functions as s push-forward that transports the base distribution at time $t = 0$ to the target distribution at time $t = 1$.

$$\dot{x_t} = u_t(x_t), \quad x_0 = x \tag{2.9}$$

where the dot $\dot{x_t}$ denotes derivative wrt t and $u_t(x)$ is the vector field governing the transport (Albergo and Vanden-Eijnden, 2023).

Viewing the generative process from the perspective of interpolants offers several benefits:

- Flexibility in the design choice of the interpolant, depending on the task at hand, form of the source and target distributions, whether we are interested in the intermediate path.

- Dtochastic interpolant formulation allows expressing both deterministic (ODE-based) as well as stochastic (SDE-based) generative models under one paradigm.
- Decoupling velocity from the path, which allows us to minimize the path and further optimize the transport.

Definition: *Stochastic Interpolant* between q_0 and q_1 is a stochastic process x_t defined (as per Albergo et al. (2023b)):

$$x_t = \underbrace{I(t,x_0,x_1)}_{\text{deterministic component}} + \underbrace{\gamma(t)z}_{\text{stochastic component}}, \quad t \in [0,1] \tag{2.10}$$

where

1. the deterministic component $I \in C^2([0,1],(C^2(\mathbb{R}^d \times \mathbb{R}^d))^d)$ in the space of twice continuously differentiable functions, satisfies the boundary conditions the boundary conditions $I(0,x_0,x_1) = x_0$ and $I(1,x_0,x_1) = x_1$,
2. the stochastic component $\gamma : [0,1] \to \mathbb{R}$ satisfies $\gamma(0) = \gamma(1) = 0, \gamma(t) > 0$ for all $t \in (0,1)$, and $\gamma^2 \in C^2([0,1])$
3. z is a Gaussian random variable independent of the pair (x_0,x_1) drawn according to the coupling with marginals q_0 and q_1 (see Definition 2.1.3) , i.e. $z \sim \mathcal{N}(0,I)$ and $z \perp (x_0,x_1)$.

See (Albergo and Vanden-Eijnden, 2023) for proof that this formulation satisfies the Transport Equation (2.3).

Generally, x_t are designed as linear interpolants under the assumption of Euclidean geometry; however, recent work (Kapusniak et al., 2024) has been done to learn interpolants that approximate geodesics on non-Euclidean manifolds.

Definition: *Spatially Linear Interponalts* are a special case of stochastic interpolants from equation 2.10 where the function I is linear is both x_0 and x_1, i.e.:

$$\begin{aligned} x_t^{\text{lin}} &= \underbrace{I^{\text{lin}}(t,x_0,x_1)}_{\text{deterministic component}} + \underbrace{\gamma(t)z}_{\text{stochastic component}} \\ &= \underbrace{\alpha_0(t)x_0 + \alpha_1(t)x_1}_{\text{deterministic component}} + \underbrace{\gamma(t)z}_{\text{stochastic component}}, \end{aligned} \tag{2.11}$$

where the curve $\alpha(t)$ is the *interpolant schedule*, which we also refer to as *path*, given that it governs the transport trajectory. The start and end point (x_0, x_1) are sampled from a coupling with marginals q_0 and q_1 (see Definition 2.1.3), $z \sim \mathcal{N}(0, I)$ and $z \perp (x_0, x_1)$, and $\alpha_0, \alpha_1, \gamma^2 \in C^2([0,1])$ satisfy the conditions

$$\alpha_0(0) = \alpha_1(1) = 1; \alpha_0(1) = \alpha_1(0) = \gamma(0) = \gamma(1) = 0; \forall t \in (0,1) : \gamma(t) > 0. \quad (2.12)$$

Remark: Despite their simplicity, spatially linear interpolants offer substantial design flexibility. Consider direct parallels between the linear interpolant formulation and the formulation used different variations of Conditional Flow Matching:

- We can recover CFM formulation by Tong et al. (Tong et al., 2024) from example 2.1.2 setting $\mu_t = I^{\text{lin}}(t, x_0, x_1) = \alpha_0(t)x_0 + \alpha_1(t)x_1 = (1-t)x_0 + tx_1$ and $\sigma_t = \gamma(t)$.
- We can recover the original Flow Matching variant by Lipman et al., which only uses one-sided conditioning $(x \mid x_1)$. Since in their formulation $p_0 = q_0 = \mathcal{N}(0, \sigma_0^2)$ is a known Gaussian, we can absorb x_0 into the Gaussian component z leaving us with a one-sided stochastic interpolant $\mu_t = I^{\text{lin}}(t, x_1) = \alpha_1(t)x_1 = tx_1$ and $\sigma_t = \gamma(t)$. By choosing $\sigma_t = 0$, we recover Rectified Flow formulation (Liu, 2022)
- In the case of generalized variance preserving (GVP) stochastic interpolant by (Albergo and Vanden-Eijnden, 2023), $\mu_t = I^{\text{lin}}(t, x_0, x_1) = \alpha_0(t)x_0 + \alpha_1(t)x_1 = cos(\frac{1}{2}\pi t)x_0 + sin(\frac{1}{2}\pi t)x_1$ and $\sigma_t = 0$.

An extended comparison between interpolant formulation and Flow Matching variants can be found in Table 2.1

Table 2.1 **Probability path definitions for existing methods which fit in the generalized conditional flow matching framework**. Adapted on the basis of Tong et al. (Tong et al., 2024).

CFM approach	coupling (x_0, x_1)	$I^{\text{lin}}(t, x_0, x_1) = \mu_t$	$\gamma(t) = \sigma_t$
Var. Exploding (Song and Ermon, 2019)	$q(x_1)$	x_1	σ_{1-t}
Var. Preserving (Ho et al., 2020)	$q(x_1)$	$\alpha_{1-t}x_1$	$\sqrt{1-\alpha_{1-t}^2}$
Flow Matching (Lipman et al., 2023)	$q(x_1)$	tx_1	$t\sigma - t + 1$
Rectified Flow (Liu, 2022)	$q(x_0)q(x_1)$	$tx_1 + (1-t)x_0$	0
Generalized Var. Pres. SI (Albergo and Vanden-Eijnden, 2023)	$q(x_0)q(x_1)$	$\cos(\frac{1}{2}\pi t)x_0 + \sin(\frac{1}{2}\pi t)x_1$	0
Independent CFM (Tong et al., 2024)	$q(x_0)q(x_1)$	$tx_1 + (1-t)x_0$	σ
Optimal Transport CFM (Tong et al., 2024)	$\pi(x_0, x_1)$	$tx_1 + (1-t)x_0$	σ
Schrödinger Bridge CFM (Tong et al., 2024)	$\pi_{2\sigma^2}(x_0, x_1)$	$tx_1 + (1-t)x_0$	$\sigma\sqrt{t(1-t)}$

2.2.2 Interpolants Allow to Decouple Velocity from the Path

In conventional maximum likelihood training of flows one is forced to couple the minimising the objective with the choice of path, leading to complicated optimization processes. Karras et al. (2022) in their overview on noise schedule designs in both flows (ODE-based methods) and diffusions (SDE-based methods), highlighted that the choice of α_0 and α_1 has major practical implications on the integration of the resulting vector field. Albergo and Vanden-Eijnden (2023) and Albergo et al. (2023a) showed that using the formalism of stochastic interpolants allows decoupling the CFM optimization problem from choosing a transport path governed by the interpolant schedule $\alpha(t)$, which paves the way to shortening the path length. Since this work is aiming at optimizing the transport path, the implications of their finding are important for setting up the methodology.

Definition: Consider a *one-sided stochastic interpolant*, a variant where $\gamma(t) = 0$:

$$x_t = \alpha_0(t)x_0 + \alpha_1(t)x_1$$

Extending x_t from the Definition 2.2.1 w.r.t. scalar $t \in [0,1]$ to a more general definition w.r.t. *interpolation coordinate* $\alpha = \alpha_0, \alpha_1$, and specifying a curve $\alpha(t)$ governing the transport path, paves way for a *more general definition of the process* $x(\alpha)$.

$$x(\alpha(t)) = \alpha_0(t)x_0 + \alpha_1(t)x_1 \tag{2.13}$$

where $\alpha_0(t)$ and $\alpha_1(t)$ are differentiable functions of $t \in [0,1]$ subject to constraints $\alpha_0(0) = \alpha_1(1) = 1$ and $\alpha_0(1) = \alpha_1(0) = 0$ which guarantee that $x_{t=0} = x_0$ and $x_{t=1} = x_1$. Hence the density $p(\alpha, x)$ of $x(\alpha)$ reduces to $p_0(x)$ for $\alpha = (1,0)$ and to $p_1(x)$ for $\alpha = (0,1)$ by construction.

While in (Albergo et al., 2023a) Albergo et al. studies barycentric interpolants defined on the simplex, this is not a necessary condition for their findings to hold for arbitrary interpolants, and the only requirement is that $\alpha_0(t)^2 + \alpha_1(t)^2 > 0$

If x_t is the solution to ODE equation 2.9, Albergo and Vanden-Eijnden (2023) and Albergo et al. (2023b) establish that under the stochastic interpolant formulation $x_t = \alpha_0(t)x_0 + \alpha_1(t)x_1$, the vector field factories into conditional expectations, which is also true for the generalized process $x(\alpha)$ (Albergo et al., 2023a):

$$\begin{aligned} u(\alpha, x) &= \mathbb{E}[\dot{x}_\alpha \mid x(\alpha) = x] \\ &= \dot{\alpha}_0(t)\mathbb{E}[x_0 \mid x(\alpha) = x] + \dot{\alpha}_1(t)\mathbb{E}[x_1 \mid x(\alpha) = x] \\ &= \dot{\alpha}_0(t)\eta_0(\alpha, x) + \dot{\alpha}_1(t)\eta_1(\alpha, x) \end{aligned} \tag{2.14}$$

where the expectation is taken over $p_0(x_0)p_1(x_1)$. $\eta_1(\alpha,x) = \mathbb{E}[x_1 \mid x(\alpha) = x]$ is commonly referred to as *denoiser*. Theorem 1 from Albergo et al. (2023a) and its proof stipulate that each η is a unique minimizer of its respective objective:

$$\begin{aligned}\mathscr{L}_{\eta_t}(\eta_0^\theta) = &\int_{[0,1]^2} \mathbb{E}_{(x_0,x_1)\sim\nu(q_0,q_1)} \left[\|\eta_0^\theta(\alpha,x(\alpha))\|^2 - 2x_0\eta_0^\theta(\alpha,x(\alpha))\right] d\alpha \\ &+ \text{const. not dependent on } \theta\end{aligned} \tag{2.15}$$

$$\begin{aligned}\mathscr{L}_{\eta_\infty}(\eta_1^\theta) = &\int_{[0,1]^2} \mathbb{E}_{(x_0,x_1)\sim\nu(q_0,q_1)} \left[\|\eta_1^\theta(\alpha,x(\alpha))\|^2 - 2x_0\eta_1^\theta(\alpha,x(\alpha))\right] d\alpha \\ &+ \text{const. not dependent on } \theta\end{aligned} \tag{2.16}$$

where $\nu(q_0,q_1)$ is a coupling whose marginals are q_0 and q_1. We can see from (2.15) and (2.16) that to minimize the objective for each conditional expectation η, we are not integrating over the scalar time $t \in [0,1]$ but marginalizing over the interpolation coordinate $\alpha \in [0,1]^2$. This means that in order to find the optimal $\alpha^*(t)$, we do not need to train the vector field for each $\alpha(t)$ curve separately, but we can train vector field $u(\alpha,x)$ (or conditional expectations η^θ) for all possible coordinates $\alpha = (\alpha_0,\alpha_1)$ on the unit cube and then amortize this network to find the optimal path.

Using these results Albergo et al. (2023a) then defines a velocity field in corollary 2:

$$u(\alpha,x) = \dot{\alpha}_0(t)\eta_0(\alpha(t),x) + \dot{\alpha}_1(t)\eta_1(\alpha(t),x) \tag{2.17}$$

which shows that the 2-Wasserstein loss (2.8) can be minimized with respect to the path $\alpha(t)$.

Having demonstrated that using the spatially linear interpolant formulation, the learning problem for the vector field u can be decoupled from the choice of path $\alpha(t)$, Albergo et al. (2023a) shows that the solution to

$$\begin{aligned}C(\alpha^\phi)) &= \min_{\alpha^\phi} \int_0^1 \mathbb{E}\left[\left\|u(\alpha^\phi(t),x(\alpha^\phi(t)))\right\|^2\right] dt \\ &= \min_{\alpha^\phi} \int_0^1 \mathbb{E}\left[\left\|\dot{\alpha^\phi}_0(t)\eta_0(\alpha^\phi(t),x(\alpha^\phi(t))) + \dot{\alpha^\phi}_1(t)\eta_1(\alpha^\phi(t),x(\alpha^\phi(t)))\right\|^2\right] dt\end{aligned} \tag{2.18}$$

gives the transport with shortest path length in Wasserstein-2 metric over the class of velocities $u^\theta(t,x) = \dot{\alpha^\phi}_0(t)\eta_0(\alpha^\phi(t),x) + \dot{\alpha^\phi}_1(t)\eta_1(\alpha^\phi(t),x)$, where the expectation is taken over $(x_0,x_1) \sim q_0q_1$, and the minimization is over all paths $\alpha^\phi \in C^1([0,1])$ such that $\alpha^\phi(t) \in [0,1]^2$.

Campbell et al. (2024) were the first to use the above approach to train a multimodal generative model for protein design called *MultiFlow*, jointly modelling continuous protein structure and discrete sequence by defining factorized flows for each data modality.

2.3 Equivariant Generative Models

2.3.1 Graph Neural Networks

Graph Neural Networks are the underlying blocks of Geometric Deep Learning (GDL) when dealing with graphs that represent structures with nodes and edges (Bronstein et al., 2021). Nodes and edges have specific attributes called node features and edge features. GNNs provide a mechanism which models the exchange of information between the nodes through the edges. Figure 2.1 explains dataflow in the three flavours of GNNs.

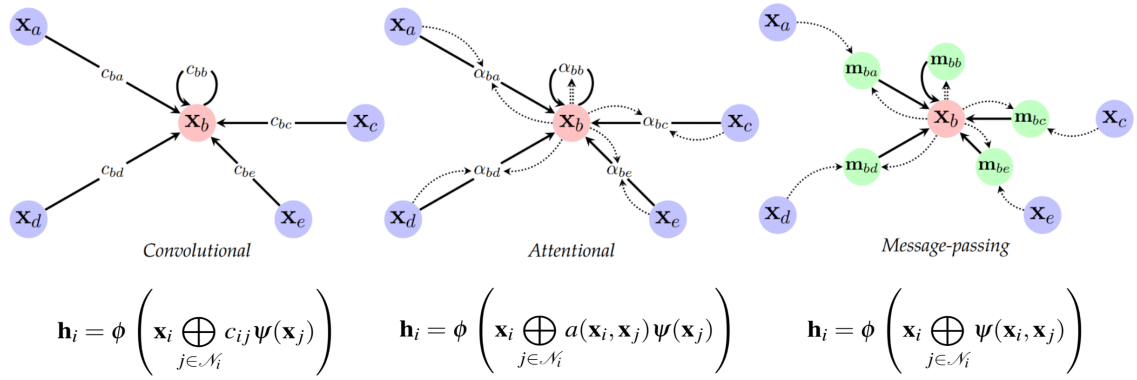


Fig. 2.1 **The three flavours of GNN layers** by Bronstein et al. (2021)

2.3.2 Equivariance and SE(3) Equivariant Graph Neural Networks

A **symmetry** of an object or system is a transformation that preserves some property or structure of that object or system (leaving them unchanged or invariant). These transformations can be smooth, continuous, or discrete, and they leave certain properties invariant, meaning that the object looks the same after the transformation. Symmetries can be found in many disciplines, from geometry to physics and machine learning. They are especially relevant in the context of Geometric Deep Learning, which concerns itself with unstructured sets, grids, graphs, and manifolds. Bronstein et al. (2021) describe GDL as a unified collection of methods that respect the structure and symmetries of objects in these domains.

Symmetries can be combined to obtain new symmetries, in particular, if we have two symmetries $\mathfrak{g}$ and $\mathfrak{h}$, since each of them leaves an object invariant, so does their combination $\mathfrak{g} \circ \mathfrak{h}$[3] or $\mathfrak{h} \circ \mathfrak{g}$, hence their composition is also a symmetry. Moreover, symmetries are invertible and such an inverse is also considered to be a symmetry. Consequently we can define a **group of symmetries** as the set of all possible transformations that preserve the structure or property of an object.

Definition: A *symmetry group* is a set $\mathfrak{G}$ together with a binary operation $\circ : \mathfrak{G} \times \mathfrak{G} \to \mathfrak{G}$, called *composition*, that satisfies the following properties:

- *Associativity*:$(\mathfrak{g} \circ \mathfrak{h}) \circ \mathfrak{f} = \mathfrak{g} \circ (\mathfrak{h} \circ \mathfrak{f})$ for all $\mathfrak{g}, \mathfrak{h}, \mathfrak{f} \in \mathfrak{G}$.
- *Identity*: there exists a unique $\mathfrak{e} \in \mathfrak{G}$ satisfying $\mathfrak{e} \circ \mathfrak{g} = \mathfrak{g} \circ \mathfrak{e} = \mathfrak{g}$ for all $\mathfrak{g} \in \mathfrak{G}$.
- *Inverse*: For each $g \in \mathfrak{G}$ there is a unique inverse $\mathfrak{g}^{-1} \in \mathfrak{G}$ such that $\mathfrak{g} \circ \mathfrak{g}^{-1} = \mathfrak{g}^{-1} \circ \mathfrak{g} = \mathfrak{e}$.
- *Closure*: The group is closed under composition, i.e., for every $\mathfrak{g}, \mathfrak{h} \in \mathfrak{G}$, we have $\mathfrak{g} \circ \mathfrak{h} \in \mathfrak{G}$.

Note that not all groups are commutative, and we may have groups for which $\mathfrak{g} \circ \mathfrak{h} \neq \mathfrak{h} \circ \mathfrak{g}$

In the applications for the physical world, the **Euclidean group**, denoted by $E(n)$, is the most prevalent since it preserves distances and angles in Euclidean geometry. The most noTable subgroups of $E(n)$ are $E(2)$ in 2D space and $\mathbf{E}(\mathbf{3})$ in 3D space, which include:

- *Translations*: moving a point or object in space without altering its shape or orientation.
- *Rotations*: rotating a point or object around a fixed point without changing distances between points.
- *Reflections*: Flipping an object along a line (2D) or a plane (3D), resulting in a mirror image.

The Euclidean Group $E(3)$ is crucial in microbiology, particularly for analyzing 3D microbiological structures while preserving their geometric properties. For instance, the structure of proteins plays a fundamental role in determining their function, making it essential to study these structures without distorting their spatial characteristics.

Studying molecules require **Special Euclidean group SE**$(\mathbf{3})$, which only includes **rotations and translations**, but not reflections, due to their stereochemical feature called chirality

[3] $\mathfrak{g} \circ \mathfrak{h}$ should be read right-to-left: $\mathfrak{h}$ is applied first, $\mathfrak{g}$ is applied afterwards.

("hand" from Greek), which can arise due to the tetrahedral geometry of saturated carbon and the associated three-dimensional properties. A molecule is called chiral if it cannot be superposed on its mirror image by any combination of rotations, translations, and some conformational changes, similar to human hands, as illustrated in Figure 2.2

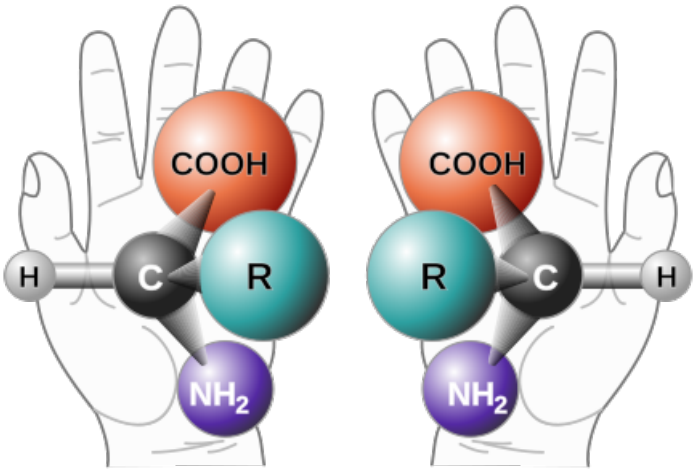


Fig. 2.2 **Illustration of chirality**: two enantiomers of a generic amino acid that are chiral, i.e. right-hand-side cannot be superimposed onto the left-hand-side reflection (NASA, 2018).

Equivariant Graph Neural Network (EGNN) proposed by Satorras et al. (2022) was the first to mode which could learn graph neural networks equivariant to rotations, translations, reflections and permutations called E(n)-Equivariant Graph Neural Networks (EGNNs). Geometric Vector Perceptrons (GVP) (Jing et al., 2021) enabled even better performance on biomolecular structures by extending standard dense layers to operate on collections of Euclidean vectors. GNNs equipped with GVP layers are able to perform both geometric and relational reasoning on efficient representations of macromolecules, which gave rise to their widespread application across a range of drug discovery tasks, such as designing antibodies (Boom et al., 2023), proteins and molecules (Dunn and Koes, 2024). Given their superior performance for molecular tasks we will be using GVP layers in the vector field architecture along the lines of Flowmol (Dunn and Koes, 2024).

2.3.3 Co-design of Molecular Topology and Conformations

The success of generative models of 3D molecules based on diffusion or flow matching relies on their ability to utilize GNNs that perform message-passing with geometric quantities to generate valid and energetically-stable large molecules. Physical inductive biases such as invariant graph attention and molecular chirality both play significant roles in generating valid 3D molecules (Morehead and Cheng, 2024). There is a significant number of studies

that leverage diffusion for structure-based drug design (Peng et al., 2023, Torge et al., 2023, Morehead and Cheng, 2024, Schneuing et al., 2023) with the following state-of-the-art diffusion models for3D de novo small molecule generation MiDi (Vignac et al., 2023b), MolDiff (Peng et al., 2023), (Huang et al., 2023). Recent studies started exploring the use of Flow matching for these purposed, notably FlowMol (Dunn and Koes, 2024) which leverages Geometric Vector Perceptrons to generate valid small molecules effectively.

Chapter 3

Optimal Path Flow: Methodology and Evaluation

3.1 Methodology

Below, we outline the framework for optimizing the path $\alpha^m(t)$ when jointly modeling multiple modalities.

Definition:

Let M denote the number of modalities, $m = \{m_1, m_2, \ldots, m_M\}$ is a combination of continuous or discrete (but relaxed, so in effect continuous) modalities that we aim to generate concurrently.

Consider a generalized one-sided interpolant as defined in equation 2.13

$$x(\alpha) = I(\alpha, x_0, x_1) = \alpha_0(t)x_0 + \alpha_1(t)x_1$$

Same as previously $\alpha_0(t)$ and end $\alpha_1(t)$ are differentiable functions of $t \in [0,1]$ subject to constraints $\alpha_0(0) = \alpha_1(1) = 1$ and $\alpha_0(1) = \alpha_1(0) = 0$. However, in the scenario with multiple modalities, we have a separate curve $\alpha^{m_i}(t)$ per each modality m_i, where $i \in [0, ...M]$ α^{m_i}.

Then for a spatially linear interpolant of the type $x_t = (1-t)x_0 + tx_1$, generalized to the process $x(\alpha(t))$, the path $\alpha(t) = (\alpha_0(t), \alpha_1(t)) = (1-\alpha_1(t), \alpha_1(t))$ for M modalities can be defined using a block diagonal matrix $\alpha_1(t) = \texttt{block\_diag}\left([\alpha_1^{m_1} I, \ldots, \alpha_1^{m_M} I]\right)$ with $\alpha_1^{m_i}(t) \in [0,1]$ i.e. scalar and boundary conditions $\alpha_1(0) = 0$ and $\alpha_1(1) = 1$.

Example: We illustrate the above definition for an example with a generalized spatially linear interpolant $x(\alpha)$ with two modalities, i.e. $m = \{m_1, m_2\}$, where m_1 (e.g., atom types) and m_2 (e.g., atom positions) as the following:

$$\begin{aligned} x(\alpha(t)) &= \alpha_0(t)x_0 + \alpha_1(t)x_1 = (1-\alpha_1(t)) \begin{pmatrix} x_0^{m_1} \\ x_0^{m_2} \end{pmatrix} + \alpha_1(t) \begin{pmatrix} x_1^{m_1} \\ x_1^{m_2} \end{pmatrix} \\ &= \begin{pmatrix} (1-\alpha_1(t))x_0^{m_1} \\ (1-\alpha_1(t))x_0^{m_2} \end{pmatrix} + \begin{pmatrix} \alpha_1(t)x_1^{m_1} \\ \alpha_1(t)x_1^{m_2} \end{pmatrix} \end{aligned} \tag{3.1}$$

By employing the interpolant $x(\alpha)$ as defined in the equation 3.1, we effectively decouple the problem of learning the vector field $u(\alpha, x)$ from the challenge of designing a path $\alpha(t)$, that governs the transport process. Building on the insights of Albergo et al. (2023a) as discussed in Section 2.2.2, we propose a structured methodology for learning the optimal path $\alpha^*(t)$ in two key stages:

1. **Training the vector field network** $u^\theta(\alpha, x)$ over all possible coordinates $\alpha = (\alpha_0, \alpha_1)$ within the unit hypercube. In this phase, we sample $\alpha_1^{m_1}, \ldots, \alpha_1^{m_M} \in U[0,1]^{\otimes M}$ within the output space of the functions $\alpha_1^{m_i}(t)$ for each modality m_i independently $\alpha_1^{m_i} \perp \alpha_1^{m_j}$ to cover all possible realizations of interpolant coordinates α. For illustration, see Figure 3.1a.

2. **Leveraging the trained vector field network** $u^\theta(\alpha, x)$ from the first step to **optimize the path** $\alpha^\phi(t)$ **parametrized by a neural network**. This step aims to approximate the optimal curve $\alpha^*(t)$ for each modality, while modeling the modalities jointly to determine the most efficient transport path. For illustration, see Figure 3.1b.

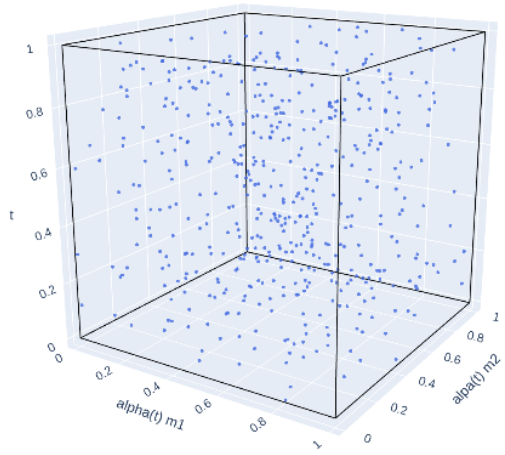
Below we outline the details for each of the two steps for the example with two modalities $m = \{m_1, m_2\}$.

3.1.1 Step 1: Training the vector field $u^\theta(\alpha, x)$

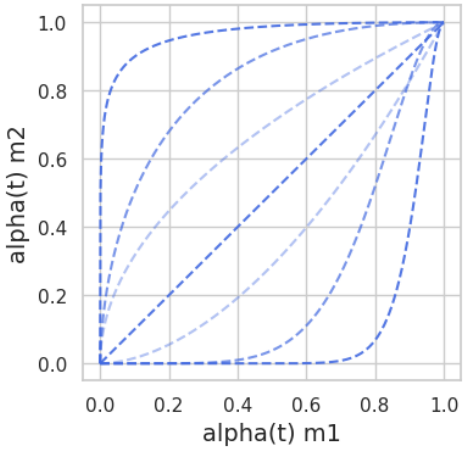
In order to train the vector field, we need to sample conditional path $x(\alpha)$. This requires sampling for α in the following way:

$$\alpha = [\alpha_0, \alpha_1] = [1-\alpha_1, \alpha_1] = \left[(1-\alpha_1^{m_1}, \ldots, 1-\alpha_1^{m_M}), \quad (\alpha_1^{m_1}, \ldots, \alpha_1^{m_M})\right] \tag{3.2}$$

which results in $\alpha_1^{m_1}, \ldots, \alpha_1^{m_M} \in U[0,1]^{\otimes M}$



(a) Step 1: α sampled in the output space on the unit cube.



(b) Step 2: Curves $\alpha(t)$ as examples of learned maps $t : [0,1] \to [0,1]^{2 \text{ modalities}}$.

Fig. 3.1 **Illustration of $\alpha(t)$ in the two steps of Optimal Path Flow training for two modalities**. In step 1 (Figure a), during the vector field training α is sampled in the output space on the unit cube. The coordinates are t as the input and $\alpha^{m_i}(t)$ are coordinates for the output for each modality m_i. In step 2 (Figure b), the vector field network is kept fixed while we jointly optimize the path $\alpha^{\phi}(t)$ for each modality m_i. The dashed lines represent examples of potential learned curves.

Recall from Section 2.1.2 conditional flow matching framework specifies the probability density $p_t(x)$ by marginalizing the conditional probability densities $p_t(x \mid z)$:

$$\begin{aligned} p_t(x) &= \int p_t(x \mid z) q(z)\, dz \\ &= \int p_t\left(x \mid x_0, x_1\right) p_0(x_0) p_1(x_1)\, dx_0\, dx_1 \end{aligned}$$

Similarly, we establish the probability density $p_t^{\alpha}\left(x(\alpha(t))\right)$ by marginalizing the conditional probability densities $p_t^{\alpha}\left(x(\alpha(t)) \mid x_0, x_1\right)$

$$p_t^{\alpha}\left(x(\alpha(t))\right) \triangleq \int p_t^{\alpha}\left(x(\alpha(t)) \mid x_0, x_1\right) p_0\left(x(\alpha(0))\right) p_1\left(x(\alpha(1))\right) dx_0\, dx_1 \tag{3.3}$$

such that

$$\lim_{t \to 1} p_t(x(\alpha(t)) \mid x_0, x_1) \approx \delta_{x_1}(x(\alpha(t)))$$

Recall that defining the flow through stochastic interpolant $x(\alpha(t))$ enables us to use factorization into conditional expectations:

$$x(\alpha) = \alpha_0(t)\mathbb{E}\left[x_0 \mid x(\alpha(t)) = x\right] + \alpha_1(t)\mathbb{E}\left[x_1 \mid x(\alpha(t)) = x\right] \tag{3.4}$$

From here we can express $\mathbb{E}[x_0 \mid x(\alpha(t)) = x] = \eta_0(\alpha, x)$ in terms of $\mathbb{E}[x_1 \mid x(\alpha(t)) = x] = \eta_1(\alpha, x)$:

$$\mathbb{E}[x_0 \mid x(\alpha(t)) = x] = \frac{1}{\alpha_0(t)}\left(x(\alpha) - \alpha_1(t)\mathbb{E}[x_1 \mid x(\alpha(t)) = x]\right) \tag{3.5}$$

or

$$\eta_0(\alpha, x) = \frac{1}{\alpha_0(t)}\left(x(\alpha) - \alpha_1(t)\eta_1(\alpha, x)\right) \tag{3.6}$$

Using this factorization we can define the vector field $u(\alpha, x)$ from equation 3.7 using only the denoiser $\mathbb{E}[x_1 \mid x(\alpha(t)) = x] = \eta_1(\alpha, x)$:

$$\begin{aligned} u(\alpha, x) &= \mathbb{E}[\dot{x}_\alpha \mid x(\alpha) = x] \\ &= \dot{\alpha}_0(t)\mathbb{E}[x_0 \mid x(\alpha) = x] + \dot{\alpha}_1(t)\mathbb{E}[x_1 \mid x(\alpha) = x] \\ &= \dot{\alpha}_0(t)\eta_0(\alpha, x) + \dot{\alpha}_1(t)\eta_1(\alpha, x) \\ &= \frac{\dot{\alpha}_0(t)}{\alpha_0(t)}x(\alpha) - \frac{\dot{\alpha}_0(t)}{\alpha_0(t)}\alpha_1(t)\eta_1(\alpha, x) + \alpha_1(t)\eta_1(\alpha, x) \\ &= \frac{\dot{\alpha}_0(t)}{\alpha_0(t)}x(\alpha) + \left(\dot{\alpha}_1(t) - \frac{\dot{\alpha}_0(t)}{\alpha_0(t)}\alpha_1(t)\right)\eta_1(\alpha, x) \end{aligned} \tag{3.7}$$

This means that in order to obtain the vector field, it is sufficient to learn the denoiser $\eta_1^\theta(\alpha, x)$ parametrized by a neural network with parameters θ. Using the insights from (2.16) that in order minimize the flow matching objective for the denoiser, we are not integrating over the scalar time $t \in [0, 1]$ but marginalizing over the interpolation coordinate $\alpha \in [0, 1]^M$, we define the objective for the denoiser model $\mathbb{E}[x_1 \mid x(\alpha(t)) = x] = \eta_1^\theta(\alpha(t), x)$ as follows:

$$\begin{aligned} \mathscr{L}(\theta) &= \mathbb{E}_\alpha \mathbb{E}_{x_0, x_1}\left[\|\eta_1^\theta(\alpha, x) - x_1\|^2\right] \\ &= \int_{[0,1]^2} \mathbb{E}_{x_0, x_1}[\|\eta_1^\theta(\alpha(t), x)\|^2 - 2x_1\eta_1^\theta(\alpha(t), x)]\, dx\, d\alpha \\ &\quad + \text{const. not dependent on } \theta \end{aligned} \tag{3.8}$$

The training procedure for Step 1 is summarized in the Algorithm 1.

3.1.2 Step 2: Optimizing the path $\alpha^\phi(t)$ by leveraging the vector field $u^\theta(\alpha, x)$

In this step $\alpha(t)$ is a map $t : [0, 1] \to [0, 1]^M$. We do not need to retrain the vector field, instead, we use the learned denoiser η_1^θ and freeze its parameters. We first sample $t \sim U(0, 1)$ and pass it through the map α_1^ϕ, which is now parametized as a neural network with parameters ϕ, to obtain the conditional path $x(\alpha(t))$. Then we pass this $x(\alpha(t))$ through the denoiser

Algorithm 1 Training Denoiser (to obtain vector field)

Require: Empirical or samplable distributions p_0, p_1, initial network $\eta_1^\theta(\alpha(t), x)$ for approximating $\mathbb{E}[x_1 \mid x(\alpha) = x]$
  while Training **do**
    $x_0 \sim p_0$; $x_1 \sim p_1$
    Sample α in the output space: $\alpha \sim U(0,1)^M$
    $x_\alpha = (1-\alpha_0)x_0 + \alpha_1 x_1$
    Calculate CFM Loss:
    $\mathscr{L}_{\text{CFM}}(\theta) \leftarrow \|\eta_1^\theta(\alpha, x_\alpha) - x_1\|^2$
    $\theta \leftarrow \text{Update}(\theta, \nabla_\theta \mathscr{L}_{\text{CFM}}(\theta))$
  end while
  return η_1^θ

$\eta_1^{\theta_{\text{frozen}}}$ `with torch.no_grad` and calculate the vector field $u(\alpha^\phi(t), x(\alpha^\phi(t)))$ according to the equation 3.7. The training procedure for Step 2 is summarized in the Algorithm 2.

The **requirement for** $\alpha_1^\phi(t)$ is that the curves $\alpha_1^{\phi_{m_i}}(t)$ corresponding to each modality $m_i, i \in [1, \ldots, M]$ must be **continuous, monotonic and adhere to the boundary conditions** $\alpha_1^\phi(0) = 0$ and $\alpha_1^\phi(1) = 1$. We parametrise $\alpha_1^{\phi_{m_i}}(t)$ with a Neural Network similar to Shaul et al. (2023) Appendix B. These networks are trained jointly, but each is parameterized independently to ensure that the resulting function $\alpha_1^{\phi_{m_i}}(t)$ is monotonic for every modality m_i. $\alpha_1^{\phi_{m_i}}(t)$ is defined as follows:

$$\alpha_1^{\phi_{m_i}}(t) = \frac{|f^{\phi_{m_i}}(t) - f^{\phi_{m_i}}(0)|}{|f^{\phi_{m_i}}(1) - f^{\phi_{m_i}}(0)|} \tag{3.9}$$

where $f^{\phi_{m_i}}$ is an MLP defined as:

$$f^{\phi_{m_i}}(t) = \texttt{sigmoid}(L_1^{m_i}(t) + L_3^{m_i}(2\texttt{sigmoid}(L_2^{m_i}(t)) - 1) \tag{3.10}$$

where $L_i^{m_i}, i \in [3]$, are linear layers: $L_1^{m_i} : \mathbb{R} \to \mathbb{R}$; $\quad L_2^{m_i} : \mathbb{R} \to \mathbb{R}^2$; $\quad L_3^{m_i} : \mathbb{R}^2 \to \mathbb{R}$. In total, the model for α uses 10 learnable parameters for each modality m_i.

Based on Equation 2.18 we define the loss function for the model for optimal path to be 2-Wasserstein loss from Equation 2.8 which is minimized with respect to the path $\alpha(t)$. We refer to it as the **OT-Loss**:

$$\mathscr{L}(\phi) = \int_0^1 \mathbb{E}_{x_0, x_1 \sim p_0, p_1} \left[\left\| u(\alpha^\phi(t), x(\alpha^\phi(t))) \right\|^2 \right] dt \tag{3.11}$$

Taking the gradient of the OT loss leads to:

$$\nabla_\phi \int_0^1 \mathbb{E}_{x_0,x_1\sim p_0,p_1}\left[\left\|u(\alpha^\phi(t),x(\alpha^\phi(t)))\right\|^2\right]dt \tag{3.12}$$

$$= \int_0^1 \mathbb{E}_{x_0,x_1\sim p_0,p_1}\left[\left\|\nabla_\phi u(\alpha^\phi(t),x(\alpha^\phi(t)))\right\|^2\right]dt \tag{3.13}$$

Equation 3.12 is providing us with a mechanism to obtain optimal path under the **assumption that the learned vector field from step 1 corresponds to the true process(es)** $(p_t)_{t=0}^1$ **of transporting density from the source** p_0 **to the target** p_1. In this sense, the expectation in the equation 3.12 can also be considered with respect to $p(\alpha(t))$.

To obtain the path $\alpha^*(t)$ the resulting objective is minimized with respect to the path $\alpha^\phi(t)$:

$$\min_{\alpha^\phi}\int_0^1 \left\|u(\alpha^\phi(t),x(\alpha^\phi(t)))\right\|^2 p(\alpha(t),x)dx\,dt \tag{3.14}$$

Algorithm 2 Training the Optimal Path

Require: Empirical or samplable distributions p_0, p_1, trained denoiser network $\eta_1^\theta(\alpha(t),x)$ from step 1, initial f^ϕ network for calculating $\alpha(t)$.
while Training **do**
  $x_0 \sim p_0; x_1 \sim p_1$
  $t \sim U(0,1)$
  $\alpha(t)$ is a map: $[0,1] \to [0,1]^{M \text{ modalities}}$
  $x(\alpha(t)) = (1-\alpha(t))x_0 + \alpha(t)x_1$
  Calculate OT Loss:
  $\mathscr{L}_\alpha(\phi) \leftarrow \int_0^1 \mathbb{E}_{x_0,x_1\sim p_0,p_1}\left[\left\|u(\alpha^\phi(t),x(\alpha^\phi(t)))\right\|^2\right]dt$
  $\phi \leftarrow \text{Update}(\phi, \nabla_\phi \mathscr{L}_\alpha(\phi))$
end while
return α^ϕ

3.2 Optimizing Paths Between 2D Gaussians

In order to demonstrate the efficacy of the proposed methodology, we conduct two case studies. We begin with a simplified scenario involving 2D Gaussian distributions, where both the source and target distributions are Gaussian. This toy example serves as a useful framework for developing insights and intuitions that can be extended to more complex cases, such as molecular conformations discussed in the subsequent case study. Moreover, the analytical tractability of the Optimal Transport (OT) loss in this context allows for a

straightforward validation of the methodology, ensuring that the observed outcomes align with the theoretical expectations.

3.2.1 Setup

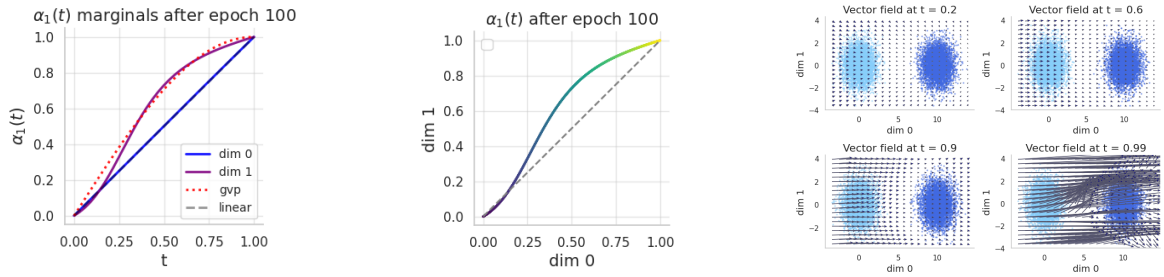
We consider a source distribution $q_0 \sim \mathcal{N}(\mu_0, I)$ and a target distribution $q_1 \sim \mathcal{N}(\mu_1, I)$, where the target is displaced by 10 units along the x-axis. Specifically, the mean values are given as $\mu_0 = (0,0)$ and $\mu_1 = (10,0)$. For this example, the two axes (x and y) represent the two modalities, with the x-axis corresponding to dimension 0 and the y-axis to dimension 1. Accordingly, the methodology outlined in Section 3.1 is applied, with the number of modalities set to two, i.e m_1, m_2. For Step 1 the implementation follows the algorithm 1 to train the denoiser η_1^θ with α sampled in the output space on the unit cube as shown in Figure 3.1a. The architecture of η_1^θ consists of four linear layers, each containing 512 hidden units, with ReLU activation functions applied to introduce non-linearity. In step two, the learned denoiser $\eta_1^{\theta_{\text{frozen}}}$ is used to train the path $\alpha^\phi(t)$ parametrized by a neural network as described in equations 3.9 and 3.10.The network structure includes two separate sub-networks, each corresponding to one of the two modalities, and each sub-network contains 10 parameters, resulting in a total of 20 trainable parameters that are optimized jointly.

3.2.2 Results

Given that there is no displacement along the y-axis (dimension 1), we anticipate more rapid changes in the y-component, as the displacement between the source and target distributions is significantly smaller compared to the x-axis. From Figure 3.2, we can see that the dimension which which require less displacement to match the target is also denoised faster. This is particularly evident in Figure 3.2b, where for $\alpha^{\dim 0}(t) = 0.5$, the x-axis has reached halfway, while the y-axis has already progressed to $\alpha^{\dim 1}(t) \approx 0.8$. Moreover, when $\alpha^{\dim 0}(t) = 0.75$ the y-axis has nearly completed its transition, reaching $\alpha^{\dim 1}(t) = 0.9$

We aim to investigate the impact of path optimization on the learned trajectories, particularly in this simplified setting where the effects are more easily visualized. Figure 3.3a presents the trajectories obtained by transporting 500 samples from the source distribution $p_0 \sim \mathcal{N}(0, I)$ along the vector field $u(t,x)$ obtained from step 1 using a linear path for each dimension. In contrast, Figure 3.3b depicts the trajectories produced by transporting the same 500 points along the same vector field, but following the learned path $\alpha^\phi(t)$ as shown in Figure 3.2b.

It is well known that the OT coupling between two Gaussians differing only in the mean results in all straight lines (Peyré and Cuturi, 2019). This setup allows us to compute the



(a) $\alpha(t)$ marginals (b) Path $\alpha(t)$ (c) Vector field $u(\alpha(t),x)$

Fig. 3.2 **Learned** $\alpha^{\phi}(t)$ for optimal path between two 2D Gaussians with the source distribution mean $\mu_0 = (0,0)$ and target distribution mean $\mu_1 = (10,0)$.

theoretical 2-Wasserstein loss for the dynamic optimal transport problem. For the source and target distributions defined as $p_0 = \mathcal{N}(\mu_0, I)$ and $p_1 = \mathcal{N}(\mu_1, I)$, the OT coupling is the coupling ν_{OT} that minimizes the expected squared distance between the points in the source and target distributions:

$$\mathbb{E}_{(X_0,X_1)\sim\nu_{\text{OT}}}\left[\|X_0 - X_1\|^2\right] = \min_{\nu} \mathbb{E}_{(X_0,X_1)\sim\nu}\left[\|X_0 - X_1\|^2\right]$$

Since it is known that the optimal coupling corresponds to a straight-line path, the OT coupling is

$$\nu_{\text{OT}}: \quad X_0 \sim p_0 \quad \text{and} \quad X_1 = X_0 + (\mu_1 - \mu_0)$$

This coupling preserves the marginals, as detailed in the OT definition 2.1.3. In this case, computing the expected distance is straightforward:

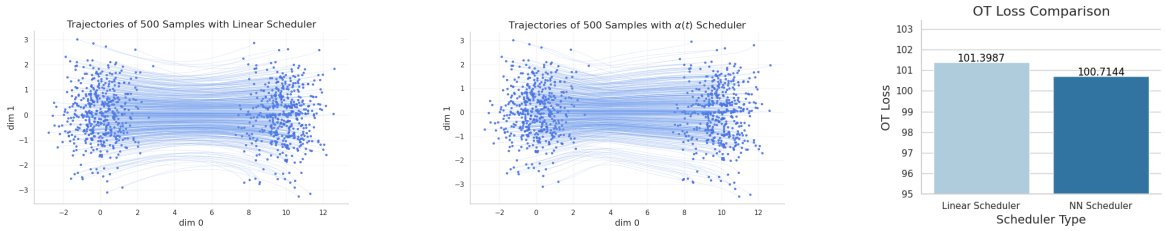
$$\|X_0 - X_1\|^2 = \|X_0 - (X_0 + (\mu_1 - \mu_0))\|^2 = \|\mu_1 - \mu_0\|^2$$

Thus, for the two Gaussians with means $\mu_0 = (0,0)$ and $\mu_1 = (10,0)$ the theoretical OT loss is:

$$\mathbb{E}_{(X_0,X_1)\sim\nu_{\text{OT}}}\left[\|X_0 - X_1\|^2\right] = \|\mu_1 - \mu_0\|^2 = 10^2 = 100$$

Based on the above, the degree to which the learned path approximates a straight line is a strong indicator of its optimality. Visual inspection of the Figures 3.3b and 3.3a confirms this expectation: the trajectories learned via the neural network are noticeably straighter compared to those generated by the linear schedule. To quantitatively validate this observation, we compute the integral of the loss function described in Equation 3.12 over a the grid of 1000 steps and compare the resulting OT loss for the two paths. As seen in Figure 3.3c the results suggest that optimizing the path $\alpha^{\phi}(t)$ reduces the OT loss compared to the linear schedule.

Specifically, the OT loss for the optimized schedule is 100.7, compared to 101.4 for the linear path, which is closer to the theoretical value of 100.



(a) Linear t (b) Learned $\alpha^{\phi}(t)$ (c) OT loss

Fig. 3.3 **Marginal trajectories p_t under linear t vs. learned $\alpha^{\phi}(t)$ with OT loss comparison measured by 2-Wasserstein distance**. OT loss integral from equation 3.12 is approximated on the grid of 1000 steps and evaluated at the end of the training process for optimizing the path. The results indicate that the path $\alpha^{\phi}(t)$ parameterized by a Neural Network shortens the distances between the source and target samples and is closer to the theoretical value of 100, calculated for two Gaussians with means $\mu_0 = (0,0)$ and $\mu_1 = (10,0)$.

Summary

In this Section, we investigated the impact of optimizing the transport path between two 2D Gaussian distributions. By comparing a linear time schedule with curves learned by $\alpha^{\phi}(t)$, we found that the learned path produced straighter trajectories, resulting in a lower OT loss closer to the theoretical 2-Wasserstein distance. We observed that the modality which aligns faster with the target distribution was denoised more quickly. These findings provide valuable insights to inform our hypothesis and guide the design of experiments for generating molecular conformations, as explored in the subsequent Section.

3.3 Optimizing Paths for De Novo Molecule Design

In this Section, we apply the proposed methodology to the more complex task of generating molecular conformations. By examining multiple trajectories of molecules generated from Gaussian noise using the FlowMol model by Dunn and Koes (2024), we observed that atom positions tend to stabilize slightly faster than other modalities, such as atom types, charges, and bond orders. This observation is illustrated in Figure 3.4. Based on the theoretical insights discussed in Section 3.1 and the findings from our earlier case study on 2D Gaussians, we formulate the following **hypotheses**:

- **H1:** *Atom positions are expected to denoise more rapidly than other modalities, and this should be reflected in the learned curves $\alpha^{\phi}(t)$.*

- **H2:** *Optimizing the path will reduce the OT loss during sample generation for design of molecular conformations.*
- **H3:** *Minimizing the OT loss will enable more efficient sampling, allowing to achieve the same quality of samples generated at fewer integration steps.*
- **H4:** *Minimizing the OT loss induces better sample quality (↑ Posebusters validity and ↓ Strain Energy) attributed to reduced integration error.*

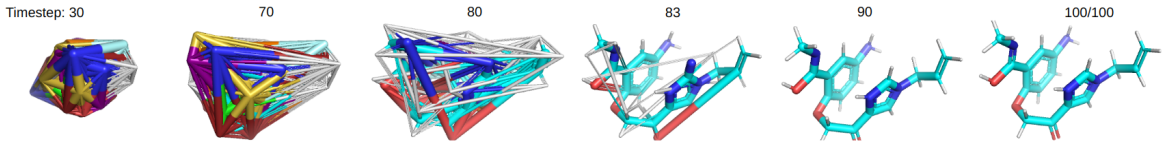


Fig. 3.4 **Trajectory of a valid molecule generated from Gaussian noise in 100 ODE steps**. The generated modalities are atom positions, atom types, charges and bond orders. The molecule is generated using the FlowMol model by Dunn and Koes (2024).

These hypotheses not only reflect the underlying theoretical framework but are also grounded in our observations from both the molecular trajectories and the simpler 2D Gaussian case. The subsequent Sections will detail how these hypotheses are tested in the molecular design context.

3.3.1 Setup and Implementation

A molecule with N atoms is modeled as a fully-connected graph, where each atom is represented as a node. Each atom has a position in space denoted by $X = \{x_i\}_{i=1}^{N} \in \mathbb{R}^{N\times 3}$, an atom type (corresponding to the atomic element) represented by $A = \{a_i\}_{i=1}^{N} \in \mathbb{R}^{N\times n_a}$, and a formal charge given by $C = \{c_i\}_{i=1}^{N} \in \mathbb{R}^{N\times n_c}$. Furthermore, each pair of atoms is associated with a bond order $E = \{eij, \forall i, j \in N \mid i \neq j\} \in \mathbb{R}^{(N^2-N)\times n_e}$. Here, n_a, n_c, and n_e represent the number of possible atom types, charges, and bond orders, respectively. These modalities, being categorical, are encoded as one-hot vectors. The molecular graph is a tuple of the components, denoted by $g = (X, A, C, E)$, forms the basis for subsequent computations, and the varying dimensions across these components will influence how we compute the norm of the vector field, as these components differ in their respective dimensionalities.

We rely on the findings in Flowmol (Dunn and Koes, 2024), which demonstrate the superior performance of a simpler continuous setting for all the modalities over a more complex setting, which involves introducing simplex constraints to account for categorical modalities. Therefore, we set the source distribution g_0 as a Gaussian for each modality.

For training the denoiser $\eta_1^\theta(\alpha, x)$ in step 1, we adopt a similar approach to the FlowMol architecture and Flow Matching Loss specifics introduced by Dunn and Koes (2024) (see Figure 3.5). Certain modifications are made to the neural network architecture and the time sampling procedure to account for the decoupled interpolation schedules for the modalities involved. These modifications were necessary to accommodate the separate time sampling for each modality, as illustrated in Figure 3.1a. Unlike the previous example with 2D Gaussians, where we sampled α on the unit cube with two modalities, the current setup involves four distinct modalities. Each node n_i in a molecular graph has 3 dimensions corresponding to the spatial coordinates, the atom types can be one of five categories (for the QM9 dataset); charges have six distinct categories, each edge e_{ij} has a bond order and bond orders between atoms are classified into five categories (none, single, double, triple, aromatic). The number of edges is $N^2 - N$, where N is the number of nodes. This motivates two distinct approaches for defining the norm of the vector field, which we compute for each batch of molecules:

- norm definition 1 (normalized by modalities), which involves flattening each component of the vector field, taking the norm and then taking a mean with respect to the number of modalities.
- norm definition 2 (normalized by modalities and modality dimensions). This method involves taking the mean of the norms for each component of the vector field and then averaging across modalities. This approach ensures that each modality contributes proportionally to the overall norm, making it more balanced.

Following the terminology introduced by Dunn and Koes (2024), we refer to the denoiser η_1^θ as an "endpoint" (EP) parameterization, as it estimates the expectation of the target point given intermediate point $\mathbb{E}[g_1 \mid g(\alpha) = g]$. For consistency, we adhere to this nomenclature throughout the study. In the context of molecular graphs, the endpoint-parameterized flow matching loss is defined as:

$$\mathscr{L}(EP) = \mathbb{E}_{\alpha, g_\alpha}\left[\frac{\dot{\alpha}_t}{1-\alpha_t}\|\eta_1^\theta(\alpha, g_\alpha) - g_1\|\right] \tag{3.15}$$

where $g_\alpha = \alpha_0(t)g_0 + \alpha_1(t)g_1 = (1-\alpha_1(t))g_0 + \alpha_1(t)g_1$. Similarly we replace the interpolant-dependent loss weight $\frac{\dot{\alpha}_t}{1-\alpha_t}$ with a capped weight function $\omega(t) = \min(\max(0.005, \frac{\dot{\alpha}_t}{1-\alpha_t}), 1.5)$ since when $\alpha(t) \to 1$, the fraction $\frac{\dot{\alpha}_t}{1-\alpha_t}$ explodes.

The final loss for Step 1 is a weighted sum of the losses for each modality, represented as:

$$\mathscr{L} = \omega_X \mathscr{L}_X + \omega_A \mathscr{L}_A + \omega_C \mathscr{L}_C + \omega_E \mathscr{L}_E \tag{3.16}$$

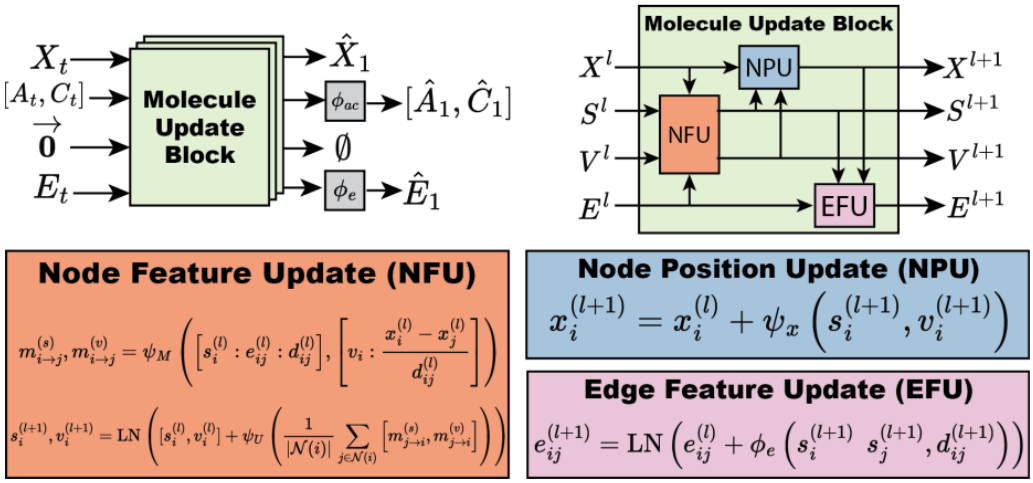


Fig. 3.5 **Modified FlowMol Architecture**. *Top left*: An input molecular graph g_t is transformed into a predicted final molecular graph g_1 by being passed through multiple molecule update blocks. *Top right*: A molecule update block uses NFU, NPU, and EFU sub-components to update all molecular features. *Bottom*: Update equations for graph features. ϕ and ψ are used to denote Multilayer perceptions (MLP) and Geometric Vector Perceptrons (GVP)s, respectively. Source: FlowMol (Dunn and Koes, 2024)

where $\{\omega_X, \omega_A, \omega_C, \omega_E\}$ are scalar weights determining the contribution for each modality to the final endpoint objective.

Optimizing the paths in Step 2 follows the same logic as in the case with two Gaussians presented in Section 3.2, with the number of modalities extended to four, resulting in 40 learnable parameters. When calculating the OT loss (3.12), we use two different ways to calculate norm of the vector field described above.

Dataset: due to the resource limitations, we concentrate on the dataset QM9 (Wu et al., 2018) containing small molecules with up to 9 heavy atoms.

Metrics to evaluate molecular validity:

- Posebusters (PB) validity (Buttenschoen et al., 2023), which measures the validity of the molecules across nine categories measuring chemical validity and consistency as well as intramolecular validity. If a molecule is passing all 9 tests it is considered PB valid.

- Strain energy, which refers to the "internal energy stored within a ligand as a result of conformational changes upon binding. Lower strain energy results in more favourable binding interactions and potentially more effective therapeutics" (Harris et al., 2023).

3.3.2 Results

In this Section, we present the results for optimal path flow following the training process for the vector field in Step 1, and the subsequent joint optimization of the path for each of the four modalities in Step 2, as outlined in Section 3.3.1.

We trained the vector field in Step 1 using two different parameterizations: the endpoint parametrization (referred to as denoiser in the previous case with two Gaussians) $\eta_1^\theta(\alpha, g)$ which is the expectation $\mathbb{E}[g_1 \mid g(\alpha) = g]$; and a direct vector field parameterization. We then evaluated the performance of these models on 1000 small molecules generated using a linear schedule for the ODE integration. The results, presented in Figure 3.6, report the mean pose busters (PB) validity rate and 95% confidence intervals based on five batches of 1000 molecules generated from five different seeds. This evaluation across different ODE integration timesteps helps to guide our further experiments.

A significant difference in performance was observed between the endpoint parameterization and the direct vector field parameterization. Specifically, the endpoint parameterization consistently achieved PB validity rates exceeding 90% starting at 50 ODE timesteps, while the direct vector field parameterization showed notably lower performance, with PB validity remaining below 80%. These results align with previous findings by Dunn and Koes (2024), which suggest that regressing the expectation $\mathbb{E}[g_1 \mid g(\alpha) = g]$ better captures the complex molecular properties required for accurate sample generation. Based on these results, we continue using the endpoint parameterization in subsequent experiments, as our approach relies on the assumption that the vector field from Step 1 is capable of accurately approximating transport paths between the source and target distributions.

Next, we explore whether optimizing the transport path improves performance with fewer ODE steps during inference, as it is designed to make the transport paths more efficient. To this end, we are interested in further evaluating the learned schedules for the points where the performance of the endpoint vector field begins to decline. Figure 3.6 highlights a marked drop in performance between 10 and 20 ODE timesteps, making 15 timesteps a good candidate for further investigation. Additionally, performance begins to plateau around 30 timesteps, identifying this as another point of interest. Finally, 100 timesteps is selected as a third benchmark, in alignment with other flow matching studies such as (Dunn and Koes, 2024).

After establishing that the vector field parameterized via the endpoint method should be used going forward, we freeze the weights of the $\eta_1^\theta(\alpha, x)$ model and amortize it to train $\alpha_1^\phi(t)$ in Step 2 and investigate the effect of path optimization on the OT Loss. The OT loss is evaluated every ten epochs during the training of the interpolant schedule $\alpha^\phi(t)$. The integral in equation 2.18 is approximated on a 100-timestep grid using 1600 samples from

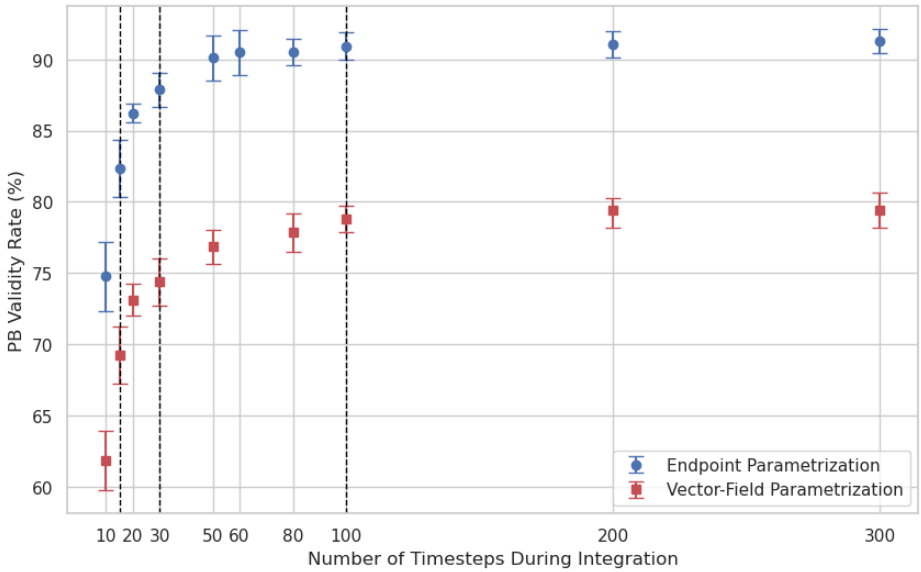


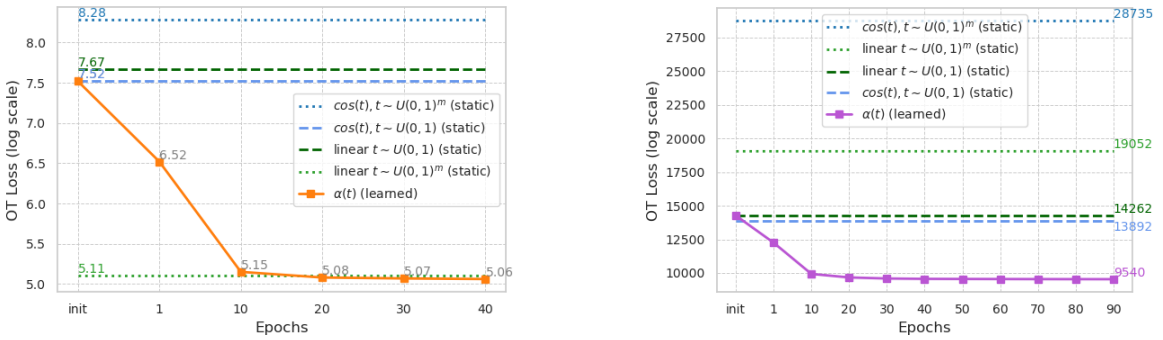
Fig. 3.6 **Posebusters validity of 1000 samples generated with a range of ODE integration timesteps**. Mean and 95% confidence intervals are calculated based on 5 batches of 1000 samples generated with five different seeds. Black dashed lines indicate points of further investigation (15, 30, 100 integration steps) for comparing the learned paths with the baseline model.

the validation split. This evaluation is conducted for two different setups: one with a more balanced vector field norm (see Figure 3.7a) and another with a less balanced norm (see Figure 3.7b).

In both setups, the results demonstrate that optimizing the transport paths leads to a reduction in OT loss, providing clear support for our hypothesis 2. This implies that the learned paths indeed straighten the trajectories of the samples transported from the source Gaussian noise to the target point along the vector field from Step 1. To further contextualize these results, we include reference lines representing the OT loss for both the linear schedule and the per-modality cosine schedule used in the FlowMol model (Dunn and Koes, 2024). Interestingly, for the setup with the more balanced norm, the optimized interpolant schedule reduces the OT loss to a level slightly below that of the linear schedule (see Figure 3.7a). This outcome implies that the optimal schedule for the learned endpoint model may be quite close to the linear schedule.

To ensure that the learned paths are robust to initialization and that no imbalance arises from the joint optimization of multiple modalities, we conduct several ablation studies.

First, we examine the results of training each modality independently, keeping the others fixed to a linear schedule (see Figure 3.8). In this experiment, we use the more balanced norm. The results show that atom types (denoted as a), atom charges (c), and bond orders (e)



(a) OT loss calculated with a more balanced definition of VF norm (normalized across features and feature dimensions).

(b) OT loss calculated with a less balanced definition of VF norm (normalized across features).

Fig. 3.7 **OT loss decreases over the course of training the NN parametrized schedule** $\alpha^{\phi}(t)$. Integral 2.18 approximated on the grid of 100 steps, evaluated every 10 epochs for the same 1600 randomly selected molecules from the validation split.

converge to a near-linear schedule, while atom positions (x) converge to a curve above the diagonal.

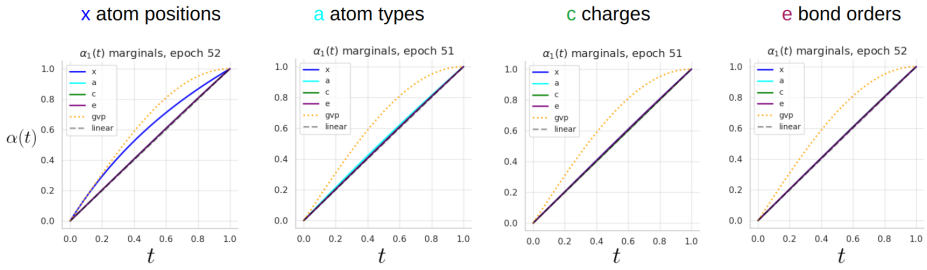


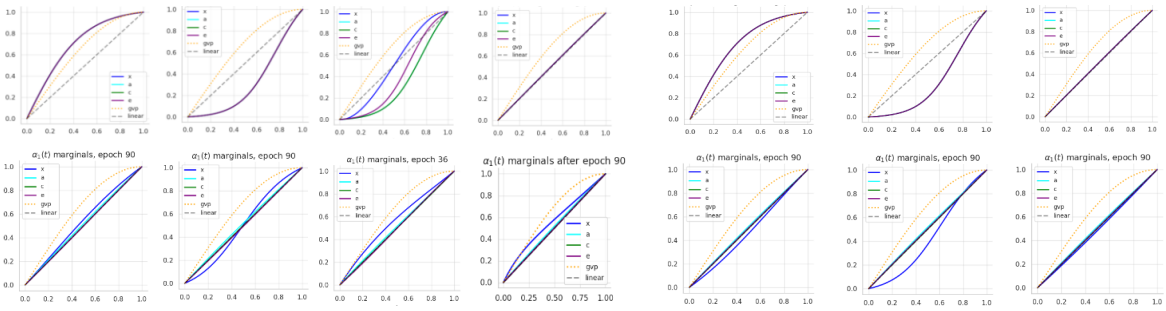
Fig. 3.8 **Independently trained curves one modality at a time** $\alpha_1^{m_i}(t)$, keeping schedule for all other modalities linear. QM9 dataset.

We then proceed to evaluate the sensitivity of the optimal path $\alpha^{\phi}(t)$ to different initializations and vector field norm definitions. The results of these experiments are shown in Figure 3.9. For the case with a more balanced norm, we directly compare the schedules learned independently for each modality against those trained jointly. The resulting curves converge to the same shapes. This further reinforces the consistency and robustness of the learned paths.

In both the less balanced and more balanced norm setups, the atom types, charges, and bond orders follow near-linear schedules, with only the relative position of the curve for atom positions differing. With a more balanced norm, the x curve lies slightly to moderately above the diagonal (see Figure 3.9a), consistent with hypothesis 1 that atom positions should be denoised faster than other modalities. Interestingly, with the less-balanced norm 3.9b,

the x curve is below the diagonal, suggesting slower denoising of atom positions, which contradicts hypothesis 1.

This discrepancy can likely be attributed to the differing contributions of modalities to the overall vector field norm. In the less balanced case, bond orders—having the largest dimensionality ($(N^2 - N) \times 5$), have a dominant weight in the norm calculation, while atom positions, with $N \times 3$ values for N nodes in a batch, contribute with the smallest weight. By contrast, in the more balanced norm, the relative contributions of each modality are proportionate, mitigating these effects.



(a) $\alpha_1^{\phi}(t)$ learned with a more balanced definition of the VF norm.

(b) $\alpha_1^{\phi}(t)$ learned with a less balanced definition of the VF norm.

Fig. 3.9 **Paths $\alpha^{\phi}(t)$ learned with different definitions of the vector files norm starting from different initializations**. *Top row*: initialization. *Bottom row*: learned paths.

After confirming that optimizing the path reduces the OT loss relative to typical choices of paths made in practice and validating the consistency of the learned paths, we now turn to the final part of our investigation, where we examine the impact of reduced OT loss on the efficiency of sample generation and sample quality during inference. For this evaluation, we compare several interpolation schedules, as illustrated in Figure 3.10, namely the cosine schedule "cos(t)" used in the FlowMol paper, "linear t" and two learned interpolation schedules, one with atom positions denoised faster than other modalities (which was learned with a more balanced VF norm, corresponding to the OT loss in Figure 3.7a), and one with atom positions denoised slower than other modalities (learned with a less balanced VF norm, corresponding to the OT loss in Figure 3.7b)

In Table 3.1, we present a comparison of Posebusters validity rates across the four interpolation schedules for five batches of 1000 samples generated with 100, 30 and 15 ODE integration steps. Based on our hypothesis, we anticipated that the samples generated with the optimized path $\alpha(t)$ would show superior performance. However, our findings do not provide evidence in support of this hypothesis; at 100 integration steps, we observed no

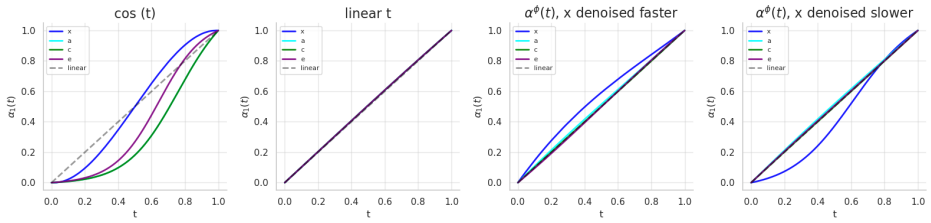


Fig. 3.10 **Interpolation schedules for evaluation on QM9 dataset**. $\alpha^{\phi}(t)$ schedules are the ones learnt during the path optimization.

improvement over the linear or cosine schedules. This implies that reducing OT error does not necessarily lead to reductions in the integration error, challenging hypothesis 4.

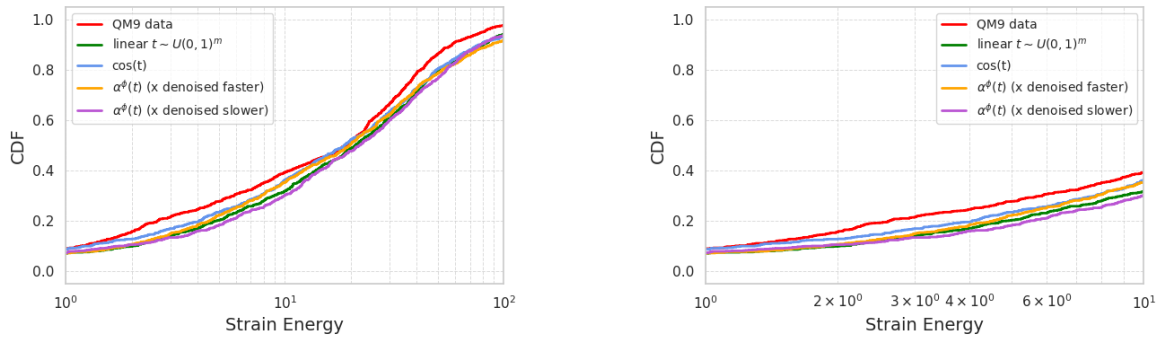
To further investigate the link between OT loss reduction and sample quality, we also analyzed the Strain Energy distributions for the generated molecular conformations. Figure 3.11 shows the cumulative distribution functions (CDFs) of Strain Energy for 1000 molecular conformations generated with different schedules, compared to 1000 random molecules from the QM9 dataset. Zooming in to the range of greater interest 1-10 SE in subFigure 3.11b, we observe that the learned schedule with faster denoising of atom positions, as well as the $cos(t)$ schedule used in FlowMol (Dunn and Koes, 2024), schedule from FlowMol, produce CDFs most similar to the actual QM9 data. Both of these schedules share the characteristic of denoising atom positions faster than other modalities. Additionally, we note a drop in PB validity at 15 integration steps for the learned schedule where atom positions are denoised slower, reinforcing the idea that atom positions may play a critical role in molecular conformation generation and should not be denoised at a slower rate than other modalities.

These findings warrant further investigation, particularly for larger molecules that tend to exhibit higher strain energy stored in rotational bonds, as the effect of atom position denoising on sample quality might be more pronounced.

Turning our attention to the efficiency of sample generation with fewer ODE integration steps, we found no improvement over the linear or cosine schedules for either 30 or 15 integration steps. This result contradicts our hypothesis 3, as reducing the OT loss does not appear to lead to more efficient sampling in these cases.

The observed results, which challenge our expectation that reducing OT loss would lead to improvements in sampling efficiency and sample quality, can be attributed to two potential sources of error in flow matching:

- Vector field approximation.
- Integration error.



(a) Strain Energy 1-100 range

(b) Zoom into the range 1-10

Fig. 3.11 **Strain Energy of 1000 molecules from QM9 dataset vs. 1000 samples generated with different time schedules**.

ODE Integration Steps	Cos(t)	Linear t	Learned $\alpha(t)$	
			x denoised faster	x denoised slower
100	90.69 (89.35, 92.05)	90.94 (89.95, 91.93)	89.92 (87.94, 91.9)	90.44 (89.0, 90.98)
30	87.14 (85.83, 88.45)	87.88 (86.68, 89.08)	87.66 (86.88, 88.43)	87.34 (86.1, 88.56)
15	82.56 (81.9, 83.22)	82.34 (80.35, 84.33)	81.94 (80.38, 83.5)	79.78 (78.0, 81.56)

Table 3.1 **Posebusters Validity (%) for the selected schedules across different numbers of ODE integration steps**. The same endpoint model was used for the vector field. Reported are mean values (with 95% confidence intervals) across 5 batches of 1000 generated molecules.

If the learned vector field from Step 1 does not approximate the true vector field that transports samples between the source and the target density, the insights about the optimal paths learned in Step 2 might be incomplete. Although we trained the vector field for 1200 epochs—longer than the 1000 epochs suggested by Dunn and Koes (2024) - the log-scaled validation loss curve for the endpoint vector field training shows a continued downward trend (see Figure A.1). This suggests that further training could potentially enhance performance. In high-dimensional generative modeling tasks, prolonged training beyond the point of plateauing loss is often required to achieve optimal results. Despite this, further training and hyperparameter optimization for the endpoint vector field, particularly for the QM9 and GEOM Drugs datasets, was beyond the scope and budget of this project.

The primary focus of this study has been optimizing the transport paths for "straightness" with OT loss with the assumption that this leads to lower integration error. However, other

approaches which rely on the Liptschitsness (smoothness) of the vector field could also be considered.

We discuss the implications of these potential sources of error in greater detail, along with proposed mitigation strategies, in Section 4.1.

Summary

In this section, we applied the proposed methodology to generate molecular conformations. Our results strongly support the hypothesis that optimizing the paths leads to a reduction in OT loss, implying straighter sample trajectories. However, this reduction did not translate into significant improvements in sampling efficiency or sample quality measured by Strain Energy distributions and Posebusters (PB) validity evaluated at different numbers of ODE integration steps, raising concerns about hypotheses 3 and 4. A comparison of the quality of samples generated using linear, cosine, and learned schedules revealed no clear advantage for the learned schedules despite their lower OT loss. These findings suggest that the connection between OT loss reduction and sample quality warrants further investigation, focusing on two key areas:

1. Refining the learned vector field approximation to the true vector field generating the transport process.

2. A closer inspection of into the relationship between improving the path efficiency and integration error. Particularly by considering not only straighter paths but also those that optimize for smoothness in the vector field.

3.4 Related Work

As discussed in Section 1.1, the choice of interpolation paths in Flow Matching and noising schedules in diffusion models is critical for the quality of generated samples (Yim et al., 2023a, Karras et al., 2022). However, most approaches still rely on heuristics.

Several studies have explored automated path optimization within the Flow Matching framework. One such study is MultiFlow (Campbell et al., 2024), which shares similarities with our approach and is also derived from (Pass, 2014). Campbell et al. (2024) applied this method to protein design, working with two modalities: continuous structure and categorical sequence of amino acids. They use a rate matrix which generates the probability paths in discrete space, as opposed to the vector field approach used in our method.

Another study by Shaul et al. (2023) delves into the theoretical foundations for devising an optimal path. They use the notion of kinetic energy to guide the design of the interpolation

schedule. Rather than optimizing for straightness using the OT loss, they focus on minimizing kinetic energy of the paths. Their approach also follows a two-stage process, but with a reversed order: they first optimize the path and then train the vector field based on the derived path. Notably, they do not investigate setting with multiple modalities. Nevertheless, their neural network design for optimizing the path has informed the neural network architecture in our method, which we tested on the settings with multiple modalities.

Chapter 4

Concluding Remarks

4.1 Discussion and Future Work

The previous chapter revealed both the strengths and limitations of the proposed optimal path flow methodology. While the approach effectively reduced OT loss and resulted in straighter transport trajectories, particularly in the simpler 2D Gaussian case, it did not consistently lead to improvements in sampling efficiency or sample quality for molecular conformation generation. This suggests that several aspects of the method require further refinement along the following sources of error:

Vector Field Approximation. The first source of error may arise if the learned vector field $u^\theta(t,x)$[1] is an imperfect approximation of the true vector field $u(t,x)$, which generates the true processes whose flow transport samples from source p_0 to target p_1 as described in Section 2.1.1. In this scenario, the learned endpoint vector field does not accurately correspond to the true process $\{p_t\}_{t=0}^1$ that interpolates between the Gaussian source and the target distribution of valid molecular conformations. Consequently, even if the OT loss—defined as the expected squared norm of the vector field—is minimized, the transport process induced by the learned vector field during inference might deviate from the target process, resulting in suboptimal sample quality and reduced efficiency. Several strategies could address this issue:

- *Calculate Actual Intermediate Paths*: One possible approach is to calculate the actual intermediate paths during the training of the vector field by transporting the points p_0 along the learned vector field in a fine-grained manner. This would match the process induced by the learned vector field more closely to the true flow. However, this method is computationally prohibitive as it requires numerous integration steps at every

[1] In this case, we consider the vector field parametrized with the endpoint, but the same reasoning applies.

training iteration, negating the advantages of simulation-free training. Nonetheless, this approach could still be informative when tested on a simplified case to inform further improvements.

- *Improve the Vector Field Network*: A more practical solution is to enhance the vector field network to achieve a closer approximation of the true process. We propose the following improvements:
  - Network Architecture Enhancements: Introducing architectural improvements tailored to better capture the molecular conformation process may enhance the accuracy of the vector field.
  - Longer Training and Hyperparameters Optimization: In this study, the vector field was trained for 1200 epochs on the QM9 dataset, but an inspection of the validation loss at the end of the training indicated the potential for further reduction. Given the complexity of high-dimensional datasets like small molecules (QM9), more extensive training may be necessary to fully converge to an accurate vector field, especially in the case of larger molecules (GEOM Drugs). Careful hyperparameter tuning, improved regularization, and batch normalization techniques could further enhance performance.

Integration Error. The second source of error relates to the integration process itself. Even though OT loss reduction suggests a more efficient path, optimizing for "straight lines" may not always lead to improved integration performance. Euler integration can be more challenging for piecewise linear functions (e.g., integrating over a triangle) compared to smooth curves (Butcher, 2016). While a straighter path is shorter, a longer curved path might result in lower integration error if the flow is smoother. To address integration error, we suggest the following directions:

- Use an explicit measure of integration error to inform further experiments based on the direct relation between the OT loss based on the norm of the vector field and the integration error.
- Instead of directly optimizing the norm of the vector field, an alternative approach could involve optimizing the norm of the Jacobian of the vector field, which measures the smoothness of the vector field and optimizing for it could lead to greater reductions in the error term for the Euler updates. This approach could lead to more accurate integration and better sampling efficiency.

Although Dunn and Koes (2024) has found that when generating small molecules with Flow Matching, keeping all modalities in continuous space with a simple Gaussian source

performs better than moving the process to a simplex, recent advances in discrete flow matching by Gat et al. (2024) may offer a promising alternative inviting further exploration.

While the strain energy in the generated small molecules closely matched the actual QM9 data, the evaluation only looked at the global structure. Local imperfections, potentially due to poor position modeling, may still exist. For de novo molecule design, we recommend calculating strain energy with a focus on local strain, which is particularly sensitive to distortions in atomic positions.

4.2 Conclusions

In this thesis, we proposed a methodology for learning an optimal transport path to improve the efficiency and quality of sample generation in multi-modalities settings. Motivated by the challenges in selecting effective schedules for multiple intertwined modalities, we sought to automate the process by leveraging neural networks. Our approach, based on the stochastic interpolant framework proposed by Albergo and Vanden-Eijnden (2023) and theoretical basis proposed in Albergo et al. (2023a), decouples the path from the vector field, enabling the training of a single vector field that can sample across all potential paths for multiple modalities. This avoids the need to train separate vector fields for each combination of modalities and leveraging the advantage of training one vector field network and then optimizing the path instead.

Guided by the efficiency implications of utilizing dynamic optimal transport in flow matching framework (Tong et al., 2024), we defined an optimization objective to minimize OT loss, focusing on achieving straighter trajectories in Euclidean space. This objective was then applied to train the interpolation path $\alpha^{\phi}(t)$, parameterized by a neural network, ensuring continuous, monotonic interpolation schedules for $\alpha_1(t)$

We validated this methodology first on a simple case of 2D Gaussian distributions, where theoretical OT loss was available as a benchmark. We found that optimizing the path does reduce the OT loss and produces straighter sample trajectories. We also observed that the modality that aligns with its targets faster is denoised more quickly.

Building on these results, we extended our approach to a more complex scenario: generating small molecules. We found that our approach produces consistent paths regardless of initialization or individual impact of modalities on joint optimization of the path. While path optimization did lead to lower OT loss in this setting, it did not translate into significant improvements in sampling efficiency or sample quality, as measured by Posebusters (PB) validity and Strain Energy (SE) distributions. This suggested that further refinement is

needed, particularly in the vector field approximation and the understanding of integration error.

We hypothesize that simply optimizing the path is insufficient for generating high-quality samples; the accuracy of the learned vector field in approximating the true vector field governing the transport process is crucial. We proposed several strategies to improve vector field such as network architecture enhancements and refining the training of the vector field network. We also suggest exploring recent advancements from discrete flow matching (Gat et al., 2024, Campbell et al., 2024) in the multimodal setting. Another aspect of the proposed future work is considering alternative norms for path optimization and exploring the smoothness of the vector field through the norm of the Jacobian of the vector field, which could potentially lead to greater reductions in integration error.

Lastly, we highlighted the importance of local strain energy in molecular design, as positional inconsistencies may affect the quality of generated structures.

As a final remark, it is important to note that the current leading model for drug design, AlphaFold 3, uses a simplified diffusion process (Abramson et al., 2024), even discarding some equivariance constraints from the previous versions of Alphafold version, yet it outperforms other approaches by leveraging the scale of transformers and vast amounts of data. However, since such extensive resources are not accessible to most research institutions working on drug discovery, it remains crucial to focus on simulation-free methods that can efficiently model multiple modalities jointly while maintaining high sample quality.

References

J. Abramson, J. Adler, J. Dunger, R. Evans, T. Green, A. Pritzel, O. Ronneberger, L. Willmore, A. J. Ballard, J. Bambrick, S. W. Bodenstein, D. A. Evans, C.-C. Hung, M. O'Neill, D. Reiman, K. Tunyasuvunakool, Z. Wu, A. Žemgulytė, E. Arvaniti, C. Beattie, O. Bertolli, A. Bridgland, A. Cherepanov, M. Congreve, A. I. Cowen-Rivers, A. Cowie, M. Figurnov, F. B. Fuchs, H. Gladman, R. Jain, Y. A. Khan, C. M. R. Low, K. Perlin, A. Potapenko, P. Savy, S. Singh, A. Stecula, A. Thillaisundaram, C. Tong, S. Yakneen, E. D. Zhong, M. Zielinski, A. Žídek, V. Bapst, P. Kohli, M. Jaderberg, D. Hassabis, and J. M. Jumper. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630 (8016):493–500, 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07487-w. https://doi.org/10.1038/s41586-024-07487-w.

M. S. Albergo and E. Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/pdf?id=li7qeBbCR1t.

M. S. Albergo, N. M. Boffi, M. Lindsey, and E. Vanden-Eijnden. Multimarginal generative modeling with stochastic interpolants, 2023a. https://doi.org/10.48550/arXiv.2310.03695.

M. S. Albergo, N. M. Boffi, and E. Vanden-Eijnden. Stochastic Interpolants: A Unifying Framework for Flows and Diffusions, Mar. 2023b. URL http://arxiv.org/abs/2303.08797. arXiv:2303.08797 [cond-mat].

J.-D. Benamou and Y. Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 84:375–393, 01 2000. doi: 10.1007/s002110050002.

J. D. Boom, M. Greenig, P. Sormanni, and P. Lio. Score-based generative models for designing binding peptide backbones, 9 2023. https://doi.org/10.48550/arXiv.2310.05764.

J. Bose, T. Akhound-Sadegh, K. FATRAS, G. Huguet, J. Rector-Brooks, C.-H. Liu, A. C. Nica, M. Korablyov, M. M. Bronstein, and A. Tong. SE(3)-stochastic flow matching for protein backbone generation. In *The Twelfth International Conference on Learning Representations*, 2024. https://openreview.net/forum?id=kJFIH23hXb.

M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021. https://arxiv.org/abs/2104.13478.

J. C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016.

M. Buttenschoen, G. M. Morris, and C. M. Deane. Posebusters: Ai-based docking methods fail to generate physically valid poses or generalise to novel sequences, 2023. https://arxiv.org/abs/2308.05777.

A. Campbell, J. Yim, R. Barzilay, T. Rainforth, and T. Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design, 2024. https://doi.org/10.48550/arXiv.2402.04997.

R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations, 2019. https://arxiv.org/abs/1806.07366.

G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. S. Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking, 2023. https://doi.org/10.48550/arXiv.2210.01776.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transportation distances, 2013. https://arxiv.org/abs/1306.0895.

B. N. e. a. Dauparas J, Anishchenko I. Robust deep learning-based protein sequence design using proteinmpnn. *Science*, 2022. 378(6615), 49–56. https://doi.org/10.1126/science.add2187.

K. Didi, F. Vargas, S. V. Mathis, V. Dutordoir, E. Mathieu, U. J. Komorowska, and P. Lio. A framework for conditional diffusion modelling with applications in motif scaffolding for protein design, 2024. https://doi.org/10.48550/arXiv.2312.09236.

I. Dunn and D. R. Koes. Mixed continuous and categorical flow matching for 3d de novo molecule generation, 2024. https://doi.org/10.48550/arXiv.2402.04997.

I. Gat, T. Remez, N. Shaul, F. Kreuk, R. T. Q. Chen, G. Synnaeve, Y. Adi, and Y. Lipman. Discrete flow matching, 2024. https://arxiv.org/abs/2407.15595.

C. Harris, K. Didi, A. R. Jamasb, C. K. Joshi, S. V. Mathis, P. Lio, and T. Blundell. Benchmarking generated poses: How rational is structure-based drug design with generative models?, 2023. https://arxiv.org/abs/2308.07413.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

H. Huang, L. Sun, B. Du, and W. Lv. Learning joint 2d 3d diffusion models for complete molecule generation, 2023. https://arxiv.org/abs/2305.12347.

B. Jing, S. Eismann, P. Suriana, R. J. L. Townshend, and R. Dror. Learning from protein structure with geometric vector perceptrons, 2021. https://arxiv.org/abs/2009.01411.

K. Kapusniak, P. Potaptchik, T. Reu, L. Zhang, A. Tong, M. Bronstein, A. J. Bose, and F. D. Giovanni. Metric flow matching for smooth interpolations on the data manifold, 2024. https://doi.org/10.48550/arXiv.2405.14780.

T. Karras, M. Aittala, T. Aila, and S. Laine. Elucidating the design space of diffusion-based generative models, 2022. https://arxiv.org/abs/2206.00364.

Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. https://openreview.net/forum?id=PqvMRDCJT9t.

Q. Liu. Rectified flow: A marginal preserving approach to optimal transport, 2022. https://arxiv.org/abs/2209.14577.

K. Martinkus, J. Ludwiczak, K. Cho, W.-C. Liang, J. Lafrance-Vanasse, I. Hotzel, A. Rajpal, Y. Wu, R. Bonneau, V. Gligorijevic, and A. Loukas. Abdiffuser: Full-atom generation of in-vitro functioning antibodies, 2023. https://doi.org/10.48550/arXiv.2308.05027.

A. Morehead and J. Cheng. Geometry-complete diffusion for 3d molecule generation and optimization. *Communications Chemistry*, 7, 07 2024. doi: 10.1038/s42004-024-01233-z.

NASA. Astrobiology at nasa, 2018. https://astrobiology.nasa.gov/news/chiral-molecules-may-have-hitched-rides-to-planets/.

B. Pass. Multi-marginal optimal transport: theory and applications, 2014.

X. Peng, J. Guan, Q. Liu, and J. Ma. Moldiff: Addressing the atom-bond inconsistency problem in 3d molecule diffusion generation, 2023. https://arxiv.org/abs/2305.07508.

G. Peyré and M. Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019. ISSN 1935-8237. doi: 10.1561/2200000073. http://dx.doi.org/10.1561/2200000073.

V. G. Satorras, E. Hoogeboom, and M. Welling. E(n) equivariant graph neural networks, 2022. https://arxiv.org/abs/2102.09844.

A. Schneuing, Y. Du, C. Harris, A. Jamasb, I. Igashov, W. Du, T. Blundell, P. Lió, C. Gomes, M. Welling, M. Bronstein, and B. Correia. Structure-based drug design with equivariant diffusion models, 2023. URL https://arxiv.org/abs/2210.13695.

N. Shaul, R. T. Q. Chen, M. Nickel, M. Le, and Y. Lipman. On kinetic optimal probability paths for generative models, 2023. https://doi.org/10.48550/arXiv.2306.06626.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

A. Tong, K. Fatras, N. Malkin, G. Huguet, Y. Zhang, J. Rector-Brooks, G. Wolf, and Y. Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. https://openreview.net/forum?id=CD9Snc73AW.

J. Torge, C. Harris, S. V. Mathis, and P. Lio. Diffhopp: A graph diffusion model for novel drug design via scaffold hopping, 2023. https://doi.org/10.48550/arXiv.2308.07416.

C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023a. https://openreview.net/forum?id=UaAD-Nu86WX.

C. Vignac, N. Osman, L. Toni, and P. Frossard. Midi: Mixed graph and 3d denoising diffusion for molecule generation, 2023b. https://arxiv.org/abs/2302.09048.

C. Villani. *Optimal transport – Old and new*, volume 338, pages xxii+973. 01 2008. doi: 10.1007/978-3-540-71050-9.

Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande. Moleculenet: A benchmark for molecular machine learning, 2018. https://arxiv.org/abs/1703.00564.

J. Yim, A. Campbell, A. Y. K. Foong, M. Gastegger, J. Jiménez-Luna, S. Lewis, V. G. Satorras, B. S. Veeling, R. Barzilay, T. Jaakkola, and F. Noé. Fast protein backbone generation with se(3) flow matching, 2023a.

J. Yim, B. L. Trippe, V. D. Bortoli, E. Mathieu, A. Doucet, R. Barzilay, and T. Jaakkola. Se(3) diffusion model with application to protein backbone generation, 2023b.

Appendix A

Additional Information

A.1 Marginal and Conditional Vector Field

Theorem: Given the conditional vector fields $u_t(x \mid z)$, which generate the conditional probability paths $p_t(x \mid z)$ for any conditioning distribution q, the marginal vector field $u_t(x)$ obtained via equation 2.6 generates the marginal probability path $p_t(x)$ obtained via equation 2.5.

Proof: Our aim is to show that the marginal vector field $u_t(x)$ derived from conditional vector field $(u_t(x \mid z)$ satisfies the Transport Equation:

$$\frac{\partial p_t(x)}{\partial t} = -\nabla \cdot (u_t(x) p_t(x))$$

$$\begin{aligned}
\frac{\partial p_t(x)}{\partial t} &= \frac{\partial}{\partial t} \int p_t(x \mid z) q(z) dz \\
&= \int \frac{\partial}{\partial t} p_t(x \mid z) q(z) dz \\
&= -\int \nabla \cdot (u_t(x \mid z) p_t(x \mid z)) q(z) dz \\
&= -\int \nabla \cdot (u_t(x \mid z) p_t(x \mid z) q(z)) dz \\
&= -\nabla \cdot \int (u_t(x \mid z) p_t(x \mid z) q(z)) dz \\
&= -\nabla \cdot \left(\int u_t(x \mid z) \frac{p_t(x \mid z) q(z)}{p_t(x)} p_t(x) dz \right) \\
&= -\nabla \cdot \left(\int u_t(x \mid z) \frac{p_t(x \mid z) q(z)}{p_t(x)} dz p_t(x) \right) \\
&= -\nabla \cdot (u_t(x) p_t(x))
\end{aligned}$$

This is a sufficient and necessary condition for $u_t(x)$ to generate $p_t(x)$.

A.2 Validation Loss From the Vector Field Training

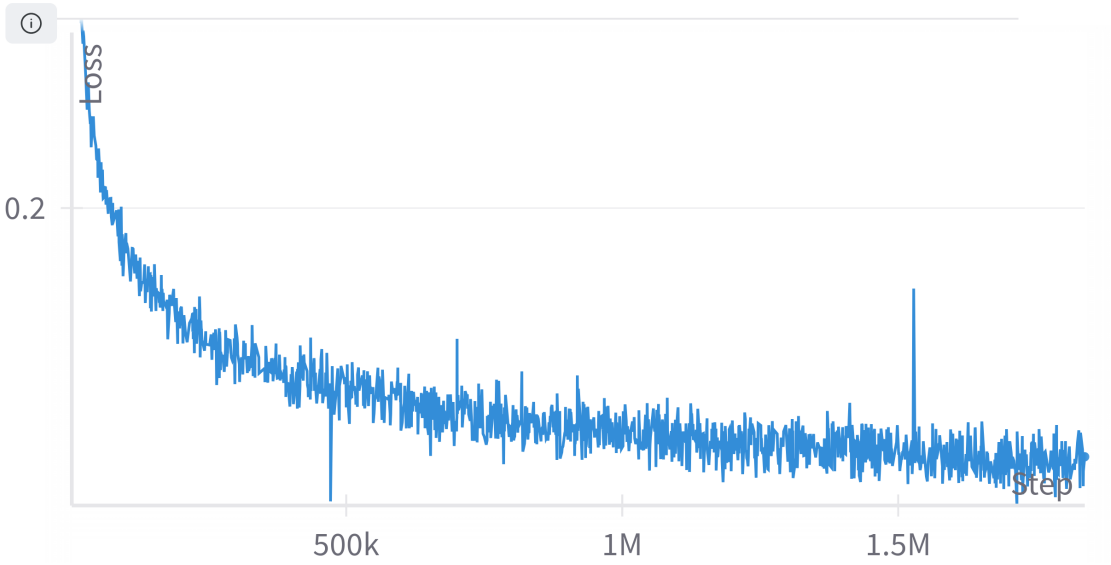


Fig. A.1 **Validation loss at the end of the endpoint vector field training for QM9 dataset** (y-axis is on the log scale). The loss keeps decreasing, suggesting potential room for further improvement.