A Reproduction and Extension of "ChemFlow: Navigating Chemical Space with Latent Flows"

Apostolos Koukouvinis
National and Kapodistrian University of Athens
Athens, Greece
tolkoukouvinis@di.uoa.gr

Abstract

The discovery of novel molecules with specific functional properties is a significant challenge, primarily due to the vastness and complexity of the chemical space. Deep generative models offer a promising approach by mapping this space to a continuous and lower-dimensional latent representation, yet methods for efficient and targeted exploration of this latent space are crucial for practical applications. The paper "Navigating Chemical Space with Latent Flows" introduces `ChemFlow`, a framework that formulates molecular optimization as a dynamical system. It learns vector fields, or flows, within the latent space of a pre-trained generative model to transport an initial molecular distribution towards regions associated with desired properties or enhanced structural diversity. The framework unifies existing optimization methods and introduces novel traversal dynamics by incorporating physical priors, such as the Hamilton-Jacobi and wave equations.

In this technical report, we first detail the complete pipeline for the reproduction of the original paper's results, outlining the experimental setup, model implementation, and validation process. Furthermore, building upon the paper's foundation, we introduce and evaluate several algorithmic extensions. Inspired by advancements in machine learning models and the theory of differential equations, we propose modifications to the generative model and flow-based optimization process. Our evaluation demonstrates that these proposed algorithms achieve superior performance on several key molecular optimization and manipulation benchmarks, validating their efficacy as improvements to the original framework.

1 Introduction

The process of designing novel molecules with specific, optimized properties, known as *de novo* design, is a key task in drug discovery and materials science [3, 22]. A primary obstacle to this process is the scale of the chemical space, which consists of all theoretically possible molecules. The number of drug-like organic molecules is estimated to be in the range of 10^{23} to 10^{60} [1, 18]. The large size of this space makes exhaustive synthesis and experimental screening of all potential candidates practically impossible. This limitation necessitates the use of computational methods to efficiently search the chemical space and identify molecules with desired characteristics.

The integration of machine learning, particularly deep generative models, has provided a new set of tools for addressing this challenge [10, 26]. These models are trained on large datasets of existing molecules to learn an implicit representation of the underlying chemical distribution [4]. A common architectural pattern involves encoding discrete molecular structures into a continuous, low-dimensional vector space, often referred to as the latent space. By sampling points from this latent space and decoding them back into molecular structures, these models can generate novel chemical entities that were not present in the training data [20, 27, 30].

While the ability to generate valid and novel molecules is a prerequisite, the ultimate objective is goal-directed design: the creation of molecules that are optimized for one or more properties, such as high binding affinity to a protein target or favorable drug-like profiles [8]. This requirement transforms the problem from one of random sampling to one of targeted exploration and optimization within the learned latent space. The central technical challenge is therefore to develop methods that can effectively and efficiently navigate this continuous space to identify regions that correspond to molecules with superior properties [9].

The paper "Navigating Chemical Space with Latent Flows" [28] proposes a framework, `ChemFlow`, to address this navigation problem by drawing an analogy to dynamical systems. It models the optimization process as the evolution of a probability distribution of molecules over time, guided by a learned vector field, or flow. This flow is designed to transport the initial distribution towards a target region associated with desired molecular properties. This formulation provides a unified perspective on existing optimization techniques and enables the incorporation of traversal dynamics derived from physical principles, such as those governed by the Hamilton-Jacobi and wave equations.

This technical report presents two primary contributions. First, it documents the complete process of reproducing the results of the `ChemFlow` framework, including details on the model implementation, experimental setup, and validation. Second, building on the foundation of the original work, this report introduces and evaluates a set of algorithmic extensions. These extensions are designed to further improve the performance and efficiency of latent space navigation for goal-directed molecular design.

2 Background and Related Work

2.1 Molecular Representations

A foundational step in applying machine learning to molecular design is the conversion of chemical structures into a machine-readable format. Several representation schemes exist, each with different characteristics.

1D String Representations: Molecules can be represented as sequences of characters, with SMILES (Simplified Molecular-Input Line-Entry System) being a widely used format. Models based on recurrent neural networks (RNNs) can process these strings to learn their underlying grammar [9]. However, string representations do not explicitly encode the topological structure of a molecule, and adherence to strict syntactic rules is required to ensure chemical validity.

2D Graph Representations: A more natural representation treats molecules as graphs, where atoms are nodes and chemical bonds are edges. This captures the connectivity and topology of the molecule directly. Graph Neural Networks (GNNs) are a class of models designed specifically to operate on graph-structured data, making them suitable for learning from molecular graphs [4].

3D Geometric Representations: The most detailed representation includes the 3D coordinates of each atom in space. This captures the molecule's conformation and geometry, which is determinant for many biological activities. While models operating on 3D structures exist [27], they introduce additional complexity. The work discussed in this report focuses on models that utilize 2D graph or 1D string representations.

2.2 Generative Models for Molecular Structures

A variety of deep generative models have been adapted for the task of generating molecular structures.

Variational Autoencoders (VAEs): The VAE is a generative model consisting of an encoder, which maps input data to a latent distribution, and a decoder, which reconstructs the data from a latent sample [15]. VAEs have been applied to generate SMILES strings [9] and, more commonly, molecular graphs[13, 19].

Diffusion Models: This class of models learns to reverse a gradual noising process. Starting from random noise, a diffusion model iteratively denoises the data to produce a sample from the learned distribution. Diffusion models have demonstrated strong performance in generating high-quality samples in various domains, including molecular and materials design [27, 30].

2.3 Approaches to Latent Space Optimization

Given a generative model with a continuous latent space, several strategies have been developed to perform property optimization.

Gradient-Based Optimization: This approach involves training an auxiliary property prediction model that maps latent vectors to a specific chemical property. Optimization is then performed by using the gradients of this predictor to update a latent vector in the direction that improves the property score [9]. LIMO is a framework that utilizes this technique for targeted molecule generation [8]. A known issue with this method is that unconstrained optimization can move the latent vector to out-of-distribution regions, from which the decoder may generate invalid or unrealistic molecular structures. This has led to the development of constrained optimization methods that penalize deviation from the learned data manifold [11, 19].

Interpretable Traversal and Disentanglement: This line of work focuses on understanding the inherent structure of the latent space to enable more controlled navigation. The ChemSpacE framework, for example, identifies linear hyperplanes that separate molecules based on a given property and uses the normal to this plane as an interpretable direction for traversal [7]. A related objective is to learn a disentangled representation, where individual axes of the latent space correspond to distinct and independent factors of variation in the data, such as specific molecular properties [5, 6]. This concept of identifying interpretable directions has also been explored extensively in the context of image generation with GANs [12, 17, 23, 24].

2.4 Optimal Transport and Wasserstein Gradient Flows

The `ChemFlow` framework is built upon concepts from optimal transport theory. Optimal transport provides a mathematical formalism for finding the most efficient way to transform one probability distribution into another, where efficiency is measured by a transportation cost. The Wasserstein distance is a metric derived from this theory that quantifies the distance between two probability distributions.

A key theoretical result connects optimal transport to the dynamics of certain physical systems. It has been shown that some partial differential equations (PDEs), such as the Fokker-Planck equation, can be formulated as a gradient flow of an energy functional with respect to the Wasserstein metric [14]. In this context, a gradient flow describes a trajectory that follows the path of steepest descent for the functional. By viewing the optimization of molecules in a latent space as the transport of a probability distribution, the `ChemFlow` framework leverages this connection. It treats standard gradient-based optimization as one type of flow and proposes to learn vector fields that satisfy other physically motivated PDEs, so it is a method for regularizing and controlling the optimization process.

2.5 Neural Ordinary Differential Equations

A standard deep neural network, such as a Residual Network (ResNet), processes information through a sequence of discrete layers. Each layer takes an input, applies a transformation, and produces an output that becomes the input for the next layer. This can be viewed as taking a series of distinct steps to transform an initial input into a final output.

Neural Ordinary Differential Equations (Neural ODEs) propose a different perspective: what if instead of a fixed number of discrete layers, we could define a continuous transformation? [2]. This approach does not define the layers themselves. Instead, it uses neural network, denoted as f, to define the *derivative* of a hidden state $h(t)$ with respect to a continuous variable t (which can be thought of as a continuous notion of network "depth"). This relationship is captured by the equation:

$$\frac{dh(t)}{dt} = f(h(t), t, \theta) \tag{1}$$

In this formulation, the neural network f learns the rules that govern how the hidden state $h(t)$ changes continuously over time. This allows the model to map an input to an output via a continuous trajectory, rather than through a series of discrete layer-wise transformations.

The `ChemFlow` framework uses fixed mathematical rules, specifically Partial Differential Equations (PDEs), to guide navigation in the latent space. The core concept of a Neural ODE—that the dynamics of a transformation can be learned—provides an alternative. In our work, we extend this framework by replacing the pre-defined PDE dynamics with a learned Neural ODE. This allows the model to determine the optimal traversal paths for a given optimization task directly from data, instead of using a fixed physical model.

2.6 Transformer-Based Molecular Language Models

Language models, particularly those based on the transformer architecture, have been adapted for molecular generation by treating molecules as sequences of tokens [21, 25]. These models process molecular representations, such as SMILES or SELFIES strings, to learn the underlying patterns of chemical structure. The MolGen-Transformer is one such generative model, designed to create a well-structured latent space for molecular exploration [29]. It uses a transformer-based autoencoder to map a molecule to a latent vector and then decode that vector back to the original structure.

This model has two properties relevant to its use in latent space navigation. First, it operates on the SELFIES (SELF-referencIng Embedded Strings) molecular representation. The SELFIES format has a defined grammar that ensures any sequence of tokens corresponds to a chemically valid molecule, which removes the need for post-generation validity filtering. Second, the model is trained to achieve high reconstruction accuracy, meaning that a molecule encoded into the latent space can be decoded back to its original structure with minimal information loss. This property is important for ensuring that points in the latent space correspond to specific, recoverable molecules.

In our work, we replace the Variational Autoencoder (VAE) used in the original `ChemFlow` framework with the pre-trained `MolGen-Transformer` as our encoder-decoder system. We do not use any internal dynamics from the model itself. Instead, we use it to establish a static latent space and then apply external navigation algorithms, including random walks and local neighborhood searches.

3 Methodology

This section details the algorithmic foundations for navigating the chemical latent space. We first provide an intuitive overview of the core methods presented in the original `ChemFlow` paper. The goal here is not to delve into the mathematical foundations but to build a conceptual understanding of how these algorithms function. We then proceed to introduce our proposed extensions in the subsequent subsection.

3.1 Analysis of Algorithms in the Original Paper

The central idea of the paper is to find effective paths within the high-dimensional latent space learned by a generative model. This model, a Variational Autoencoder (VAE), encodes molecules into continuous vectors and decodes those vectors back into molecules. The challenge is to define a way to move from the vector of an initial molecule to a new vector corresponding to a molecule with improved properties. The paper explores several strategies for this traversal.

Linear Traversal. This is the most direct approach to latent space traversal. It first identifies a single, fixed direction (a vector) in the latent space that is associated with a desired property. The optimization process then consists of moving a molecule's latent representation along this pre-determined straight-line path. While computationally simple, this method operates on the strong assumption that the optimal path for property improvement is linear.

Gradient-Based Optimization (Gradient Flow). This method takes a more adaptive approach. It relies on a surrogate model trained to predict a specific molecular property given a latent vector. To optimize a molecule, the algorithm computes the gradient of this property with respect to the molecule's latent vector. This gradient points in the direction of the steepest increase for that property. The latent vector is then updated by taking a small step along this gradient direction. It is a local optimization technique that can be effective but may converge to suboptimal solutions if the property landscape is complex.

Flow-Based Traversal. The main proposal of `ChemFlow`[28] is to model the traversal paths as continuous trajectories, or "flows," which are governed by specific dynamical equations derived from physics. Instead of a simple straight line or a series of greedy gradient steps, these flows can trace more complex and structured paths through the latent space. The paper introduces several variants:

- **Hamilton-Jacobi (HJ) Flow:** This flow is derived from the principles of optimal transport. It aims to find the most efficient way to move an entire distribution of molecules towards a target distribution. It learns a velocity field that minimizes the kinetic energy of the traversal, effectively finding the "shortest" or most direct paths in the latent space.
- **Wave Flow:** This flow is governed by the second-order wave equation. This formulation induces oscillatory dynamics into the traversal path. Instead of moving monotonically towards the target, the path exhibits wave-like behavior. This characteristic can encourage exploration of regions around the primary optimization trajectory, potentially discovering a more diverse set of valid molecules.
- **Langevin Dynamics:** This method enhances the standard gradient-based approach by introducing a stochastic component. At each optimization step, the latent vector is updated not only by moving along the property gradient but also by adding a small, random vector sampled from a Gaussian distribution. This injection of noise helps the optimization process escape from local optima and perform a more global search of the latent space.

Guidance Mechanisms. To direct the flows, the framework requires a guidance signal. The dynamical equations ensure the smoothness of the flow, while the guidance will define the direction of the flow. The paper proposes two distinct mechanisms for generating this signal:

- **Supervised Guidance:** In this setting, the direction of the flow is determined by an external, pre-trained surrogate model that explicitly predicts the target molecular property. The gradient of this predictor provides the signal that guides the traversal towards regions of higher (or lower) property values.
- **Unsupervised Guidance:** When a specific property predictor is not available, the guidance signal is derived directly from the generative model's decoder. The flow is optimized to move in directions that cause the maximum change in

the output molecular structure, as measured by the Jacobian of the decoder network. The underlying hypothesis-heuristic is that directions of maximal structural variation are correlated with meaningful changes in latent molecular properties.

3.2 Our Algorithmic Extensions

Building on the framework established by the original paper, we explored several approaches to latent space navigation. The goal of these extensions was to investigate alternative dynamics and model architectures that could potentially yield improved performance. In this section, we detail the methods we implemented and evaluated, some of them worked while others were disappointing.

3.2.1 Direct Step Prediction with a Latent Stepper MLP. The first approach diverges from the concept of simulating continuous, time-dependent flows. The methods in the original paper, such as Wave and Hamilton-Jacobi flows, learn a *velocity field* that defines the direction of movement at any given point and time. The final trajectory is then constructed by integrating this velocity field over multiple small time steps.

Our alternative, which we term the "Latent Stepper," simplifies this process by framing the problem as direct step prediction. Instead of learning a velocity field, we train a Multi-Layer Perceptron (MLP) to directly predict the optimal update vector, Δz, for a given latent vector, z_t. The entire optimization step is thus reduced to a single forward pass and an addition: $z_{t+1} = z_t + \text{MLP}(z_t)$. The training objective for this MLP is to produce a Δz that maximizes the desired outcome in a single step.It is a supervised method and the loss function is designed to maximize the property improvement between the initial and final states, i.e., to maximize $(\text{Property}(z_{t+1}) - \text{Property}(z_t))$. To ensure stability and prevent the model from learning excessively large steps, a regularization term is added to the loss function to penalize the norm of the predicted Δz.

This approach replaces the machinery of solving differential equations with a standard supervised learning problem. While it lacks the explicit physical priors of the original flows, it offers a potentially simpler and more direct method for learning an effective, state-dependent update rule for latent space optimization. This method actually did not work, confirming the main idea of the paper that the smoothness of the motion plays a critical role.

3.2.2 Hybrid Guidance via Vector Averaging. This extension combines the supervised and unsupervised guidance mechanisms into a single update rule. The motivation is to simultaneously leverage the targeted nature of supervised optimization and the exploratory potential of unsupervised structural change.

The method operates by generating two candidate update vectors at each optimization step, t:

(1) A supervised update vector, Δz_{sup}, derived from a flow guided by a property predictor.
(2) An unsupervised update vector, Δz_{unsup}, derived from a flow that maximizes structural change.

Instead of selecting one vector over the other, the final update vector, Δz_{final}, is computed as the element-wise mean of the two candidate vectors:

$$\Delta z_{\text{final}} = \frac{\Delta z_{\text{sup}} + \Delta z_{\text{unsup}}}{2}$$

The latent vector is then updated using this averaged direction: $z_{t+1} = z_t + \Delta z_{\text{final}}$.

Actually, this approach did not work as well.

3.2.3 Learning the Dynamics with a Neural Ordinary Differential Equation. The Wave and Hamilton-Jacobi flows from the original paper impose strong physical priors on the latent space trajectories by forcing them to obey pre-defined partial differential equations (PDEs). While these priors can be effective, they may also be overly restrictive. An alternative is to allow the model to *learn* the optimal dynamics directly from the data.

To this end, we implemented a Neural Ordinary Differential Equation (Neural ODE) to model the latent space flow[2]. A Neural ODE parameterizes the derivative of the state with respect to time using a neural network. Specifically, we define a function, represented by an MLP, that learns the velocity field $v(z, t)$:

$$\frac{dz}{dt} = v(z, t; \theta) = \text{MLP}(z, t)$$

Here, the MLP takes both the current latent state z and the current time t as input and outputs the velocity vector dz/dt. Unlike the fixed-form PDEs, this MLP can learn any arbitrary continuous dynamics that are optimal for the task.

The trajectory from a starting point z_0 is computed by integrating this learned function over time using a numerical ODE solver. The training objective is to tune the parameters of the MLP such that the resulting trajectories maximize property improvement (in the supervised case) or structural change (in the unsupervised case) between the start and end points of the integration.

This approach replaces the hand-crafted physical equations with a flexible, data-driven dynamical system. The hypothesis is that a learned velocity field can discover more complex and effective traversal paths than those constrained by fixed physical laws, potentially leading to better optimization performance.

3.2.4 Replacing the Generative Backbone with MolGen-Transformer. This extension experiments with the impact of the underlying generative model on the optimization task. We replaced the Variational Autoencoder (VAE) used in the original paper with **MolGen-Transformer**[29], a transformer-based generative model. This model differs from the original VAE in several ways: it uses a transformer architecture[25], operates on SELFIES [16] representations to ensure chemical validity, and produces a sequence of latent vectors as its representation.

Given the change in the generative model and its latent space structure, we evaluated traversal using methods provided within the MolGen-Transformer framework. The specific methods evaluated were:

- **Random Walk:** The latent representation is perturbed at each step by adding a random, normalized vector.
- **Neighboring Search:** A function that generates a set of molecular variants by applying minimal perturbations to a starting molecule's latent representation, similar to the unstructured approach of ChemFlow.

This approach evaluates the effect of a different latent space structure on molecular optimization when combined with basic traversal techniques.

4 Evaluation

4.1 Experimental Setup

To reproduce and extend the results, we worked with two main codebases: the code from the original paper (https://github.com/garywei944/ChemFlow) and the code for the transformer architecture we used (https://github.com/baskargroup/MolTransformer_repo). We also created new scripts to implement our methods and run the evaluation. This section provides an overview of the results and a discussion. The code implementation is not detailed here. Instead, a notebook is provided with all the scripts needed to reproduce the results.

The scripts from the original paper required the datasets to be in a specific format, but the processed data was not available. We therefore downloaded and prepared the following datasets:

- **MOSES**: https://github.com/molecularsets/moses
- **ZINC250k**: https://www.kaggle.com/datasets/basu369victor/zinc250k/data
- **ChEMBL**: https://www.ebi.ac.uk/chembl/

Hardware and Software Environment. All code was run with the following configuration:

- **OS**: Ubuntu 22.04.3 LTS
- **CPU**: Intel Xeon Gold 5318Y, 2 sockets × 24 cores/socket, hyperthreading enabled (96 logical cores), base 2.10 GHz.
- **Memory**: 377 GiB RAM.
- **GPUs**: 2 × NVIDIA RTX A6000 (48 GB VRAM each), CUDA 12.2.

Our reproduction focused on the main results presented in the body of the paper, as the code for the additional experiments in the appendix was not provided. We were also unable to replicate the original docking experiments due to a lack of setup instructions for the specified software. To address this, we performed our own docking analysis using different software and protein target.

In the following subsections, we introduce each of the experiments reproduced, along with a comparison of the results mentioned in the paper and our results.

4.2 Unconstrained Molecule Optimization

The first set of experiments evaluates the performance of different traversal methods on unconstrained molecule optimization. As described in the original paper [28], this task involves starting from a random set of molecules and attempting to find new molecules that maximize a specific property, without any other constraints. The evaluation is on two standard molecular properties: the penalized logP (plogP), which is related to a molecule's solubility, and the Quantitative Estimate of Drug-likeness (QED). For both properties, a higher score is better.

Table 1 presents the results for this task. The table shows the scores for the top three molecules found (1ST, 2ND, 3RD) after 10 optimization steps. For each method and rank, we show the value reported in the original paper (OG) alongside the value we obtained in our reproduction (Repro.). The '(SPV)' and '(UNSUP)' suffixes denote supervised and unsupervised guidance, respectively.

An interesting observation from the reproduced results is the performance of the unsupervised methods. The original paper's claim of strong performance from 'WAVE (UNSUP)' was a key finding, as it suggested that meaningful property optimization could be achieved without a dedicated property predictor. Our inability to replicate this result, coupled with the method's near-baseline performance in our runs, questions the power of this specific unsupervised approach. The other unsupervised method, 'HJ (UNSUP)', performed better in our plogP task than originally reported but did not stand out in the QED task. This variability suggests that while the concept of unsupervised guidance is promising, its effectiveness may be highly dependent on the specific combination of the flow dynamics (Wave vs. HJ) and the target property, making it less reliable than supervised methods.

4.3 Distribution Shift During Optimization

Beyond finding a few top-performing molecules, an effective optimization method should be able to shift the entire distribution of a population of molecules towards the desired property range. The authors track the distribution of plogP scores for a set of molecules over 1000 optimization steps.

Figure 1 visualizes this process. Each subplot corresponds to a different optimization method. The x-axis represents the plogP score (higher is better), and the y-axis is the density. The curves show the distribution of plogP scores at different time steps ('t'), evolving from the initial distribution ('t=0', lightest shade) to the final distribution ('t=999', darkest shade). The top row shows the plots from the original paper, and the bottom row shows our reproduction. A successful method should show a clear rightward shift of the distribution over time, indicating that the entire population of molecules is improving.

The distribution plots from our reproduction are largely consistent with those from the original paper. In both sets of results, the 'Random' method fails to shift the distribution, serving as a baseline. The 'Gradient Flow' and 'Wave eqn. (spv)' methods show a minor shift, but the final distribution still has significant overlap with the initial one. This indicates that while they can find some improved molecules, they do not effectively optimize the entire population.

The most effective method in both the original and our reproduced results is 'Langevin Dynamics'. The plots clearly show a significant and sustained rightward shift of the entire plogP distribution. The final distribution ('t=999') is almost entirely separated from the initial distribution, indicating a successful optimization of the whole set of molecules.

4.4 Similarity-Constrained Molecule Optimization

A practical challenge in molecule optimization is to improve a desired property while ensuring the new molecule remains structurally similar to the original. This is a common task in lead optimization, where chemists make small modifications to a known active compound. This experiment evaluates the methods on such a task: maximizing plogP while adhering to a structural similarity constraint. The constraint is defined by δ, which sets a minimum

METHOD	PLOGP ↑						QED ↑					
	1ST		2ND		3RD		1ST		2ND		3RD	
	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.
RANDOM	3.52	3.523	3.43	3.433	3.37	3.366	0.940	0.940	0.933	0.933	0.932	0.932
GRADIENT FLOW	4.06	4.396	3.69	4.274	3.54	3.780	0.944	0.929	0.941	0.928	0.941	0.928
WAVE (SPV)	4.76	4.393	3.78	4.128	3.71	4.051	**0.947**	0.935	0.934	0.930	0.932	0.928
WAVE (UNSUP)	**5.30**	3.541	**5.22**	3.185	**5.14**	3.091	0.905	0.941	0.902	0.936	**0.978**	0.935
HJ (SPV)	4.39	4.051	3.70	3.780	3.48	3.695	0.946	0.933	0.941	0.902	0.940	0.894
HJ (UNSUP)	4.26	4.527	4.10	4.267	4.07	3.841	0.930	0.932	0.928	0.929	0.927	0.928
LD	4.74	4.700	3.61	4.026	3.55	3.558	**0.947**	0.944	**0.947**	0.933	0.942	0.933

Table 1: Unconstrained optimization

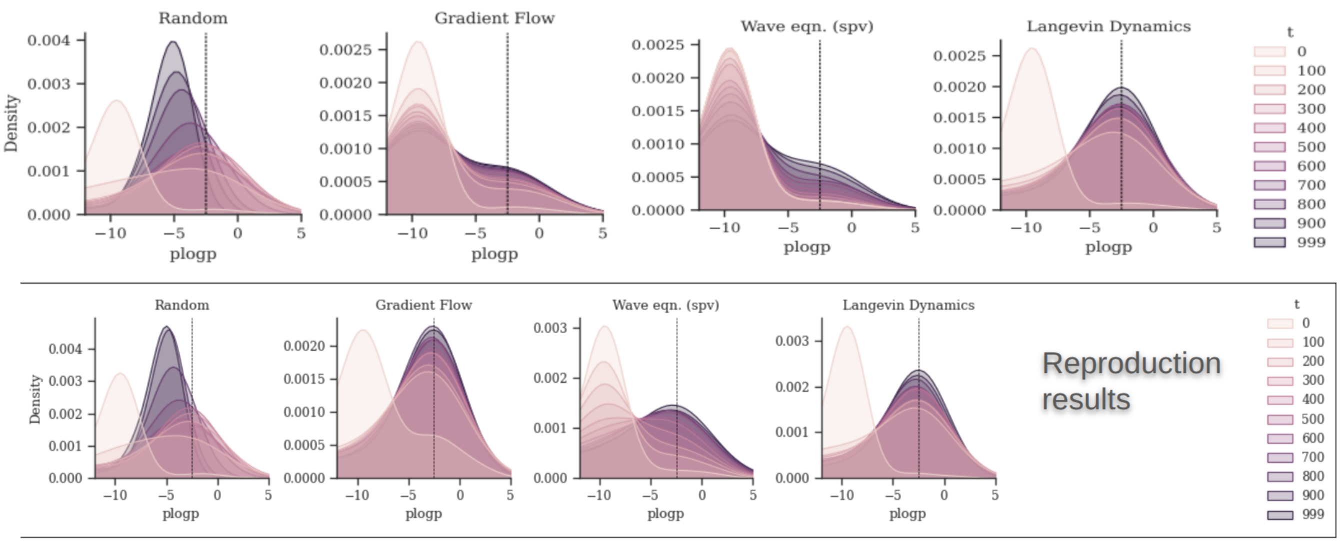

Figure 1: Molecular property plogP distribution shifts following the latent flow path. The top row shows the results from the original paper. The bottom row shows our reproduced results.

Tanimoto similarity to the starting molecule. A smaller δ means the optimization is less constrained and can produce more dissimilar molecules.

Table 2 shows the results of this experiment. The values are reported in the format: mean improvement ± standard deviation (success rate %). The success rate indicates the percentage of optimization runs that successfully found a molecule satisfying the similarity constraint. We compare the results from the original paper with our reproduced values for different similarity thresholds (δ).

In the least constrained case ($\delta = 0$), the results are generally consistent with the unconstrained optimization task. The original paper reported a very high improvement for ‘HJ (unsup)‘, which we were not able to fully replicate, although our reproduced value of 14.02 is still strong. Our reproduced results for the other methods in this column were often higher than originally reported, with higher success rates.

As the similarity constraint becomes stricter (increasing δ), the performance of all methods decreases, which is expected. ‘Langevin Dynamics (LD)‘ consistently performs well across all constraints in our reproduction, showing the best or near-best mean improvement and maintaining a high success rate even at $\delta = 0.2$. In contrast, the unsupervised methods (‘Wave (unsup)‘ and ‘HJ (unsup)‘) show a sharp drop in performance as the constraint tightens. Their effectiveness seems to rely on making large, unconstrained changes to the molecular structure, making them less suitable for fine-grained lead optimization. The supervised methods (‘Gradient Flow‘, ‘Wave (spv)‘, ‘HJ (spv)‘) show more stable but generally lower performance than ‘LD‘.

4.5 Multi-objective Molecule Optimization

Real-world drug design often requires optimizing multiple properties simultaneously. This experiment tests the methods on a multi-objective task: concurrently maximizing the QED score and the

Method	$\delta = 0$		$\delta = 0.2$		$\delta = 0.4$		$\delta = 0.6$	
	Original	Reproduced	Original	Reproduced	Original	Reproduced	Original	Reproduced
Random	11.76 ± 6.18 (99.0)	11.77 ± 6.22 (99.1)	7.64 ± 6.38 (80.0)	7.60 ± 6.33 (82.4)	5.03 ± 5.70 (52.1)	5.43 ± 6.05 (51.7)	2.37 ± 3.71 (21.1)	2.89 ± 4.60 (20.9)
Gradient Flow	7.88 ± 7.28 (60.4)	10.84 ± 7.34 (97.9)	7.20 ± 6.98 (56.5)	7.15 ± 6.48 (78.8)	5.45 ± 6.45 (41.9)	3.87 ± 4.80 (42.8)	3.60 ± 5.50 (18.4)	1.93 ± 3.42 (15.8)
Wave (spv)	6.83 ± 7.15 (59.6)	10.37 ± 7.22 (95.8)	5.62 ± 6.42 (54.9)	7.35 ± 6.50 (79.5)	4.31 ± 5.55 (41.9)	4.62 ± 5.74 (51.2)	2.47 ± 4.21 (20.6)	2.63 ± 4.55 (22.4)
Wave (unsup)	19.76 ± 13.62 (99.6)	13.17 ± 11.90 (97.8)	7.47 ± 9.62 (50.2)	5.31 ± 7.42 (64.1)	2.06 ± 4.37 (27.3)	1.82 ± 3.47 (37.0)	0.77 ± 2.21 (16.8)	0.86 ± 1.61 (18.4)
HJ (spv)	8.58 ± 8.08 (68.0)	10.47 ± 7.90 (95.8)	6.62 ± 7.44 (60.0)	7.35 ± 6.91 (83.8)	4.27 ± 5.40 (40.6)	4.36 ± 5.22 (57.6)	2.39 ± 4.10 (18.5)	1.94 ± 3.23 (29.9)
HJ (unsup)	**20.64 ± 12.93 (98.0)**	14.02 ± 12.31 (96.2)	8.57 ± 9.69 (50.1)	6.38 ± 8.34 (55.1)	2.12 ± 3.55 (19.5)	2.01 ± 3.57 (26.8)	0.67 ± 0.86 (8.6)	0.69 ± 1.03 (12.8)
LD	12.98 ± 6.23 (99.6)	13.10 ± 6.14 (99.6)	9.70 ± 6.21 (94.4)	**9.72 ± 6.56 (95.8)**	6.14 ± 5.99 (70.9)	5.74 ± 5.65 (73.0)	2.94 ± 4.34 (35.4)	3.17 ± 4.49 (38.0)

Table 2: Similarity-constrained plogP maximization

Synthetic Accessibility (SA) score. A higher QED indicates better drug-likeness, while a higher SA score indicates that the molecule is easier to synthesize. The optimization is performed under the same similarity constraints (δ) as the previous experiment.

Table 3 presents the results, separated into two parts for clarity: one showing the improvement in QED and the other showing the improvement in SA. The values are reported in the format: mean improvement ± standard deviation (success rate %). We compare the original paper's results with our reproduced values. The method with the highest improvement for each property and similarity level in our reproduction is bolded.

In this multi-objective setting, there are some discrepancies between the original and reproduced results, particularly for the SA improvement. Our reproduced SA improvements are much higher across the board for the less constrained cases ($\delta = 0, 0.2$). Focusing on our reproduced results, 'Langevin Dynamics (LD)' again demonstrates strong and balanced performance. For the $\delta = 0$ case, it achieves the highest improvement in both QED and SA. As the constraints tighten, its performance remains competitive. The unsupervised methods, particularly 'HJ (UNSUP)', show very high SA improvement at moderate constraints ($\delta = 0.2, 0.4$) but at the cost of very low success rates, making the results less reliable. The Wave-based methods ('Wave (SPV)' and 'Wave (UNSUP)') perform poorly in this task, often failing to find any valid improvements under tighter constraints (success rate of 0.0). Overall, 'Langevin Dynamics' appears to be the most reliable method for multi-objective optimization. The poor performance of the Wave-based flows suggests their oscillatory dynamics may not be well-suited for navigating the more complex landscape of a multi-property optimization problem.

4.6 Molecule Manipulation Success Rate

This experiment evaluates the ability of each method to perform "molecule manipulation," which is defined as finding a smooth traversal path in the latent space that monotonically improves a desired property. This tests not just the final outcome, but the quality of the optimization trajectory itself. The evaluation is performed across a variety of properties, including plogP, QED, SA, and predicted activities against three protein targets (DRD2, JNK3, GSK3B).

Table 4 reports the success rates for this task. The values are presented in the format: strict success rate / relaxed success rate (in %). A strict success requires the property to improve at every single step, while a relaxed success allows for small deviations. The 'RANKING' column shows the overall rank of the method based on its average performance, as reported in the original paper (OG) and our reproduction (Repro.).

In this experiment, the rankings from our reproduction align more closely with the original paper. 'Gradient Flow (GF)' is the top-ranked method in both, demonstrating the highest average strict success rate. This indicates that moving directly along the steepest ascent of the property predictor is a very effective strategy for achieving consistent, monotonic improvement, which is the definition of success in this task. 'Langevin Dynamics (LD)' also performs well, ranking third in our reproduction, showing that its stochastic nature does not prevent it from finding smooth, improving paths.

A key finding from our reproduction is the strong performance of 'WAVE (UNSUP)', which achieved the top rank. This result is surprising, as it performed poorly in the unconstrained optimization task. It suggests that while the unsupervised wave dynamics may not be the best for finding the absolute highest-scoring molecules, they are very effective at generating smooth, consistently improving trajectories. This could be because the oscillatory nature of the wave flow encourages small, stable changes. The 'RANDOM' method also shows a high relaxed success rate, particularly for the protein targets, suggesting that for some properties, even a random traversal direction in the latent space is likely to lead to improvement, a point noted in the original paper.

5 Algorithmic Extensions and Results

Building on the reproduced framework, we introduced and evaluated several of our own algorithmic extensions. The goal was to explore alternative approaches to latent space navigation that could potentially improve performance. This section details the results of these new methods on the unconstrained optimization task.

Table 5 compares the performance of our proposed methods against the reproduced results of the original paper's methods. The top portion of the table shows the reproduced baselines, while the bottom portion shows the results from our extensions. The 'ODE' method refers to our Neural ODE implementation. 'LATENT STEPPER' is our direct step prediction model. '*hybrid_ld_hj*' is the hybrid guidance method. The last two rows show the results of using simple traversal methods on the latent space of the 'MolGen-Transformer' model.

The results from our extensions provide several interesting insights. The 'LATENT STEPPER' and the hybrid guidance method ('*hybrid_ld_hj*') performed poorly, with scores close to or even

Multi-objective Maximization - QED Improvement								
Method	$\delta = 0$		$\delta = 0.2$		$\delta = 0.4$		$\delta = 0.6$	
	Original	Reproduced	Original	Reproduced	Original	Reproduced	Original	Reproduced
Random	45.5±13.3 (99.5)	34.7±16.4 (98.6)	20.7±14.4 (81.8)	19.0±14.6 (76.9)	12.8±10.7 (57.0)	13.0±11.1 (52.2)	8.0±8.3 (29.5)	12.2±12.1 (23.9)
ChemSpace	47.0±12.9 (99.6)	-	25.9±16.7 (88.2)	-	15.5±13.2 (63.4)	-	9.7±10.1 (31.6)	-
Gradient Flow	31.9±17.9 (89.9)	36.4±16.4 (99.4)	23.0±16.4 (80.0)	21.7±15.5 (79.1)	14.4±12.6 (59.4)	14.9±13.0 (46.1)	9.4±9.1 (32.2)	**13.6±12.8 (17.6)**
Wave (SPV)	14.6±14.4 (26.8)	35.1±16.7 (97.9)	12.3±11.7 (24.2)	12.3±2.6 (0.2)	9.8±10.0 (19.8)	0.0±0.0 (0.0)	6.8±6.5 (12.4)	0.0±0.0 (0.0)
Wave (UNSUP)	39.0±18.5 (96.1)	35.2±17.2 (94.5)	18.3±14.0 (69.4)	20.7±14.5 (3.9)	10.0±9.6 (43.2)	0.0±0.0 (0.0)	6.4±7.0 (24.9)	0.0±0.0 (0.0)
HJ (SPV)	45.2±13.7 (98.9)	35.0±15.7 (98.6)	22.7±15.6 (84.9)	20.0±15.3 (76.4)	13.9±12.0 (57.5)	11.7±10.8 (48.4)	9.3±9.7 (30.2)	8.5±7.6 (25.1)
HJ (UNSUP)	40.6±19.4 (96.8)	37.6±17.4 (96.5)	15.5±12.8 (71.4)	23.7±18.6 (4.4)	9.2±9.4 (46.2)	0.0±0.0 (0.0)	6.4±7.3 (26.9)	0.0±0.0 (0.0)
LD	47.0±13.1 (99.6)	**48.8±14.6 (100.0)**	**27.6±16.3 (92.1)**	26.1±15.4 (92.8)	15.4±12.5 (71.4)	**16.4±12.9 (71.6)**	9.6±9.4 (40.1)	12.7±11.9 (38.8)

SA Improvement (higher is better)								
Method	$\delta = 0$		$\delta = 0.2$		$\delta = 0.4$		$\delta = 0.6$	
	Original	Reproduced	Original	Reproduced	Original	Reproduced	Original	Reproduced
Random	8.34±8.06 (37.2)	31.11±16.66 (96.0)	6.70±7.11 (27.0)	13.27±10.97 (64.0)	4.80±5.73 (19.1)	9.26±8.80 (37.6)	2.61±3.21 (10.9)	5.37±5.70 (15.8)
Gradient Flow	9.56±8.49 (44.0)	39.07±17.99 (97.4)	7.19±6.66 (35.8)	15.68±12.54 (67.9)	5.27±5.30 (26.6)	9.12±8.40 (32.5)	3.04±3.62 (17.1)	6.00±6.82 (12.8)
Wave (SPV)	6.37±6.30 (12.9)	32.66±20.05 (95.5)	5.77±5.81 (12.4)	7.82±4.59 (0.2)	4.54±4.51 (10.5)	0.00±0.00 (0.0)	3.44±3.59 (7.1)	0.00±0.00 (0.0)
Wave (UNSUP)	15.21±10.17 (89.2)	35.28±17.09 (97.8)	7.69±6.51 (70.8)	18.77±16.98 (3.6)	3.92±3.86 (45.9)	0.79±nan (0.1)	2.22±1.97 (25.9)	0.00±0.00 (0.0)
HJ (SPV)	8.93±8.39 (52.1)	34.38±18.76 (98.4)	6.69±6.61 (38.8)	12.82±11.07 (64.9)	5.21±5.72 (29.2)	8.11±8.73 (34.9)	3.23±3.40 (18.6)	6.30±7.46 (14.6)
HJ (UNSUP)	16.03±10.31 (91.0)	38.69±17.70 (97.5)	7.35±6.34 (68.5)	**22.13±13.48 (5.9)**	4.22±4.09 (46.8)	**24.65±nan (0.1)**	2.73±2.85 (27.8)	0.00±0.00 (0.0)
LD	11.51±10.44 (69.2)	**51.73±18.43 (100.0)**	7.51±7.41 (45.4)	19.74±13.61 (86.9)	4.50±4.95 (33.6)	11.26±9.59 (57.8)	2.75±3.09 (20.1)	**7.36±7.24 (26.8)**

Table 3: Comparison of Original and Reproduced Results for Multi-objective (QED-SA) Maximization

Method	**RANKING**		**AVERAGE**		**PLOGP (↑)**		**QED (↑)**		**SA (↓)**		**DRD2 (↑)**		**JNK3 (↑)**		**GSK3B (↑)**	
	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.	OG	Repro.
RANDOM-1D	8	8	1.42 / 6.85	0.00 / 0.00	6.00 / 31.60	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	2.50 / 9.50	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00
RANDOM	5	4	0.57 / 42.3	**0.47** / 45.60	0.00 / 32.60	0.60 / 42.00	0.10 / 3.20	0.30 / 4.40	0.40 / 8.60	0.20 / 8.10	0.50 / 87.10	0.10 / **90.40**	1.50 / 81.40	0.70 / **82.80**	0.90 / 40.90	0.90 / 45.90
WAVE (UNSUP)	3	**1**	1.18 / 45.28	3.58 / 48.22	0.60 / 40.30	1.60 / **51.60**	0.60 / 6.20	0.50 / 2.30	1.90 / 16.50	2.90 / 13.80	0.40 / 86.40	3.20 / 83.50	1.80 / 78.20	6.00 / 79.90	1.80 / 44.10	**7.30** / **58.20**
WAVE (SPV)	6	5	1.85 / 8.08	2.47 / 9.73	0.00 / 0.20	4.30 / 21.20	3.40 / 12.10	6.50 / 16.60	4.00 / 18.60	3.10 / 16.90	3.50 / 17.10	0.70 / 3.20	0.00 / 0.20	0.10 / 0.20	0.20 / 0.30	0.10 / 0.30
HJ (UNSUP)	3	6	2.3 / 25.28	0.97 / 18.63	3.00 / 15.60	0.50 / 33.80	0.70 / 3.20	0.60 / 2.80	1.70 / 13.00	**1.00** / **12.40**	0.20 / 87.20	1.10 / 11.80	4.80 / 18.70	1.80 / 11.50	3.40 / 14.00	0.80 / 39.50
HJ (SPV)	6	7	1.97 / 7.4	2.40 / 9.10	3.00 / 13.20	4.60 / 19.80	3.00 / 7.20	5.90 / 15.00	3.70 / 15.00	2.60 / 13.40	1.90 / 8.50	1.30 / 6.40	0.20 / 0.50	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00
GF (SPV)	**1**	**1**	7.62 / 28.78	7.07 / 27.82	6.90 / 28.30	**7.00** / 29.80	**6.60** / 16.70	6.50 / 16.20	6.30 / 25.10	5.70 / 20.70	7.10 / 36.10	5.80 / 33.70	**11.70** / 35.50	10.10 / 36.50	7.10 / 31.00	**7.30** / 30.00
LD (SPV)	2	3	6.23 / 26.68	6.47 / 25.52	5.90 / 26.00	5.80 / 27.00	6.20 / 15.50	5.50 / 14.20	5.20 / 22.90	6.60 / 19.90	**6.00** / 33.30	4.80 / 31.60	8.40 / 33.80	9.20 / 32.40	5.70 / 28.60	6.90 / 28.00

Table 4: Success Rate of traversing latent molecule space to manipulate over a variety of molecular properties

METHOD	**plogP ↑**			**QED ↑**		
	1ST	2ND	3RD	1ST	2ND	3RD
RANDOM	3.52	3.43	3.37	0.940	0.933	0.932
GRADIENT FLOW	4.40	4.27	3.78	0.929	0.928	0.928
WAVE (SPV)	4.39	4.13	4.05	0.935	0.930	0.928
WAVE (UNSUP)	3.54	3.19	3.09	0.941	0.936	0.935
HJ (SPV)	4.05	3.78	3.70	0.933	0.902	0.894
HJ (UNSUP)	4.53	4.27	3.84	0.932	0.929	0.928
LD	4.70	4.03	3.56	0.944	0.933	0.933
ODE	4.23	3.79	3.71	0.936	0.928	0.927
LATENT STEPPER	3.65	3.57	3.54	nan	nan	nan
`hybrid_ld_hj`	2.95	2.77	2.64	nan	nan	nan
`random_walk_transformer`	4.52	4.40	4.38	**0.948**	**0.948**	**0.948**
`neighbor_search_transformer`	**5.35**	**5.32**	**5.14**	**0.948**	**0.948**	**0.948**

Table 5: Performance of Algorithmic Extensions on Unconstrained Optimization

below the ‘RANDOM‘ baseline. The failure of the ‘LATENT STEPPER‘ supports the original paper's premise that enforcing smooth, continuous dynamics is beneficial; predicting large, discrete steps appears to be an unstable approach. The hybrid method's failure suggests that simply averaging guidance vectors from different sources is not an effective way to combine them.

The Neural ODE[2] approach performed respectably, achieving scores comparable to ‘GRADIENT FLOW‘ and ‘WAVE (SPV)‘. While it did not outperform the top methods like ‘Langevin Dynamics‘, its solid performance indicates that learning the traversal dynamics is a viable alternative to using fixed physical equations. This suggests that with further tuning, a learned dynamical system could be a powerful tool for this task.

The most significant results came from replacing the generative backbone. Using the ‘MolGen-Transformer‘[29] and applying simple traversal methods yielded the best performance in our entire study. The ‘$neighbor\_search\_transformer$‘ method achieved a top plogP score of 5.35, outperforming all methods from the original paper and our other extensions. Both transformer-based methods also achieved the highest scores for QED maximization. This strongly suggests that the quality and structure of the latent space itself are critically important. A well-structured latent space, like that of the ‘MolGen-Transformer‘, can enable even simple navigation techniques to achieve state-of-the-art results, potentially reducing the need for complex, flow-based traversal dynamics.

5.1 Molecular Docking Simulation

To evaluate the practical relevance of the generated molecules, we performed a molecular docking simulation. Docking is a computational method that predicts the preferred orientation of one molecule (a ligand) when bound to a second (a protein receptor) to form a stable complex. The result is a score, typically in kcal/mol, that estimates the binding affinity. A lower (more negative) score indicates a stronger, more favorable binding interaction. This is a key task in drug discovery, as strong binding to a target protein is often a prerequisite for a drug's efficacy.

For this experiment, we used the estrogen receptor alpha (ESR1) as the protein target, a receptor implicated in breast cancer. In a drug discovery context, stronger binding is often a primary goal, as it can lead to a more potent therapeutic effect at lower concentrations. We used AutoDock Vina (https://github.com/ccsb-scripps/AutoDock-Vina) as the docking software. The process for each generated molecule involved converting its SMILES string to a 3D structure, preparing it for docking, and then running the Vina simulation to calculate its binding affinity to ESR1.

Table 6 shows the top five binding affinity scores obtained from molecules generated by each optimization method. The methods were run in two modes: one where the optimization was guided by plogP, and another where it was guided by QED. This tests whether optimizing for general chemical properties can indirectly lead to molecules with good binding affinity for a specific target.

Optimization Method	Score_1	Score_2	Score_3
Guided by plogP			
`plogp_fp`	-4.76	-5.15	-5.25
`plogp_hj_sup`	-4.65	-4.80	-4.86
`plogp_hj_unsup`	-5.55	-5.28	-4.87
`plogp_hybrid_sup_unsup`	-4.46	-4.80	-4.24
`plogp_latent_stepper`	nan	-4.64	-5.34
`plogp_limo`	-4.31	-5.45	-4.37
`plogp_neighboring_search`	-5.73	-5.85	-5.83
`plogp_random`	-5.85	-4.64	-5.76
`plogp_random_walk`	-5.73	-5.72	-5.73
`plogp_wave_sup`	-4.89	-4.98	-4.48
`plogp_wave_unsup`	-4.81	-4.66	-4.08
Guided by QED			
`qed_fp`	nan	-5.54	-4.16
`qed_hj_sup`	-4.41	-5.02	nan
`qed_hj_unsup`	-4.66	-5.24	-5.88
`qed_limo`	-4.74	-5.80	-5.43
`qed_neighboring_search`	**-6.09**	-4.98	-5.07
`qed_neural_ode`	**-6.22**	-5.82	-5.47
`qed_random_walk`	**-6.18**	**-5.92**	**-6.17**
`qed_wave_sup`	-5.72	-4.97	-5.23
`qed_wave_unsup`	-5.18	-5.58	-5.79

Table 6: Binding Affinity Scores (kcal/mol) for ESR1. Lower is better.

The docking results show a clear distinction between the two optimization modes. When guided by plogP, no single method consistently produced molecules with strong binding affinity. The 'random' and transformer-based methods found some molecules with good scores (around -5.8 kcal/mol), but the performance was not significantly better than other methods. This suggests that optimizing for plogP does not strongly correlate with finding good binders for ESR1.

In contrast, when the optimization was guided by QED, several methods produced molecules with much better binding affinities. Our 'Neural ODE' extension and the transformer-based methods ('$random\_walk\_transformer$' and '$neighbor\_search\_transformer$') achieved the best scores, with values below -6.0 kcal/mol. The '$random\_walk\_transformer$' was particularly effective, generating three molecules with scores of -6.18, -5.92, and -6.17 kcal/mol.

This outcome indicates that optimizing for QED, a general measure of drug-likeness, is a more effective surrogate task for finding potential ESR1 binders than optimizing for plogP. Furthermore, the strong performance of our extensions, especially the combination of the 'MolGen-Transformer' with a random walk, again highlights the importance of the underlying generative model's latent space. A well-structured space can guide the generation towards chemically relevant and structurally diverse molecules, which increases the chances of discovering potent binders for a specific protein target.

5.2 Limitations and Future Work

During our reproduction and extension of the framework, we identified a specific area that presents limitations and offer opportunities for future investigation. Specifically, it is the performance of the surrogate property predictors.

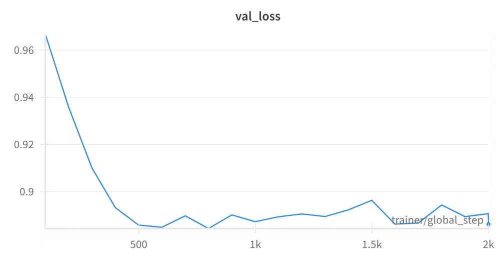

Figure 2: Validation loss of the surrogate predictor for the QED property. The x-axis represents the training steps, and the y-axis represents the loss value.

All supervised methods within the `ChemFlow` framework depend on the guidance provided by a surrogate model, which is a small neural network trained to predict a specific molecular property. The quality of this guidance is therefore critical to the success of the optimization. Figure 2 shows the validation loss for the QED surrogate predictor used in our experiments. At first glance, the curve appears to show a successful training process, with the loss dropping and then stabilizing.

However, a closer look at the y-axis scale reveals a potential issue. The validation loss only decreases from approximately 0.97 to a plateau around 0.88. This relatively small drop in loss suggests that the surrogate model does not learn to predict the QED property with high accuracy. This observation was consistent across the surrogate models for other properties as well. Since the supervised traversal methods rely entirely on the gradient from these predictors, their limited predictive power could act as a significant bottleneck, potentially explaining why methods like 'Gradient Flow' and 'Wave (spv)' did not consistently outperform the baselines.

A clear avenue for future work would be to address this limitation by replacing the simple surrogate models with more powerful and accurate property predictors. This could involve using more complex architectures, such as Graph Neural Networks (GNNs) that can learn directly from the molecular structure, or leveraging large, pre-trained models that have been specifically developed for property prediction.

6 Conclusion

This report detailed the reproduction and extension of the ChemFlow framework, a method that applies physically-motivated flows for latent space navigation in molecular optimization. Our reproduction effort successfully replicated the training pipelines and the key experimental results presented in the paper's main tables. Our evaluation, covering unconstrained, constrained, and multi-objective optimization, confirmed that methods like Langevin Dynamics are effective for improving molecular properties. One of our contributions was the evaluation of several algorithmic extensions. A key finding emerged from integrating a different generative model, the MolGen-Transformer, as the backbone. In this new latent space, we observed that simple unsupervised navigation techniques, including neighborhood search and random walks, achieved high scores in unconstrained optimization and molecular docking tasks. These results highlight the effectiveness of combining a well-structured latent space with direct traversal methods. Additionally, we experimented with learning the traversal dynamics directly using a Neural Ordinary Differential Equation (Neural ODE), replacing the fixed physical equations of the original flows. This demonstrated that a data-driven dynamical system is also a viable approach for guiding molecular optimization.

References

[1] Regine S. Bohacek, Colin McMartin, and Wayne C. Guida. 1996. The art and practice of structure-based drug design: A molecular modeling perspective. Medicinal Research Reviews 16, 1 (1996), 3–50. doi:10.1002/(SICI)1098-1128(199601)16:1<3::AID-MED1>3.0.CO;2-6

[2] Ricky T Q Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. Neural ordinary differential equations. ArXiv (2018). arXiv:1806.07366

[3] Wei Chen, Xuesong Liu, Sanyin Zhang, and Shilin Chen. 2023. Artificial intelligence for drug discovery: Resources, methods, and applications. Molecular Therapy - Nucleic Acids 31 (2023), 691–702. doi:10.1016/j.omtn.2023.02.019

[4] Yuanqi Du, Tianfan Fu, Jimeng Sun, and Shengchao Liu. 2022. MolGenSurvey: A systematic survey in machine learning models for molecule design. ArXiv (2022). arXiv:2203.14500

[5] Yuanqi Du, Xiaojie Guo, Amarda Shehu, and Liang Zhao. 2022. Interpretable molecular graph generation via monotonic constraints. ArXiv (2022). arXiv:2203.00412

[6] Yuanqi Du, Xiaojie Guo, Yinkai Wang, Amarda Shehu, and Liang Zhao. 2022. Small molecule generation via disentangled representation learning. Bioinformatics 38, 12 (2022), 3200–3208. doi:10.1093/bioinformatics/btac296

[7] Yuanqi Du, Xian Liu, Nilay Mahesh Shah, Shengchao Liu, Jieyu Zhang, and Bolei Zhou. 2023. ChemSpacE: Interpretable and Interactive Chemical Space Exploration. Transactions on Machine Learning Research (2023). https://openreview.net/forum?id=C1Xl8dYCBn

[8] Peter Eckmann, Kunyang Sun, Bo Zhao, Mudong Feng, Michael K Gilson, and Rose Yu. 2022. LIMO: Latent Inceptionism for targeted molecule generation. ArXiv (2022). arXiv:2206.09010

[9] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. 2016. Automatic chemical design using a data-driven continuous representation of molecules. ArXiv (2016). arXiv:1610.02415

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. Deep Learning. MIT Press.

[11] Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. 2017. Constrained Bayesian optimization for automatic Chemical Design. ArXiv (2017). arXiv:1709.05501

[12] Ali Jahanian, Lucy Chai, and Phillip Isola. 2019. On the "steerability" of generative adversarial networks. ArXiv (2019). arXiv:1907.07171

[13] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction tree variational autoencoder for molecular graph generation. ArXiv (2018). arXiv:1802.04364

[14] Richard Jordan, David Kinderlehrer, and Felix Otto. 1998. The Variational Formulation of the Fokker–Planck Equation. SIAM Journal on Mathematical Analysis 29, 1 (1998), 1–17. doi:10.1137/S0036141096303359

[15] Diederik P Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. ArXiv (2013). arXiv:1312.6114

[16] Mario Krenn, Florian Häse, Akshatkumar Nigam, Pascal Friederich, and Alán Aspuru-Guzik. 2019. Self-Referencing Embedded Strings (SELFIES): A 100% robust molecular string representation. (May 2019). arXiv:1905.13741 [cs.LG]

[17] Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. 2022. Diffusion Models already have a Semantic Latent Space. ArXiv (2022). arXiv:2210.10960

[18] Christopher A. Lipinski, Franco Lombardo, Beryl W. Dominy, and Paul J. Feeney. 1997. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. Advanced Drug Delivery Reviews 23, 1 (1997), 3–25. doi:10.1016/S0169-409X(96)00423-1

[19] Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander L Gaunt. 2018. Constrained graph variational autoencoders for molecule design. ArXiv (2018). arXiv:1805.09076

[20] Hannes H. Loeffler, Jiazhen He, Alessandro Tibo, Jon Paul Janet, Alexey Voronov, Lewis H. Mervin, and Ola Engkvist. 2024. Reinvent 4: Modern AI-driven generative molecule design. J. Cheminformatics 16, 1 (2024), 12. doi:10.1186/s13321-024-00812-5

[21] Łukasz Maziarka, Tomasz Danel, Sławomir Mucha, Krzysztof Rataj, Jacek Tabor, and Stanisław Jastrzębski. 2020. Molecule Attention Transformer. ArXiv (2020). arXiv:2002.08264

[22] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. 2018. Inverse molecular design using machine learning: Generative models for matter engineering. Science 361, 6400 (2018), 360–365. doi:10.1126/science.aat2663

[23] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. 2020. InterFaceGAN: Interpreting the disentangled face representation learned by GANs. ArXiv (2020). arXiv:2005.09635

[24] Yujun Shen and Bolei Zhou. 2020. Closed-form factorization of latent semantics in GANs. ArXiv (2020). arXiv:2007.06600

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. ArXiv (2017). arXiv:1706.03762

[26] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Peter Van Katwyk, Andreea Deac, Anima Anandkumar, Carla P. Gomes, Shirley Ho, Pushmeet Kohli, Joan Lasenby, Tie Yan Liu, Arjun Manrai, Debora Marks, Bharath Ramsundar, Le Song, Jimeng Sun, Jian Tang, Petar Veličković, Max Welling, Linfeng Zhang, Connor W. Coley, Yoshua Bengio, and Marinka Zitnik. 2023. Scientific discovery in the age of artificial intelligence. Nature 620, 7972 (Aug. 2023), 47–60. doi:10.1038/s41586-023-06221-2

[27] Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, Helen E. Eisenach, Woody Ahern, Andrew J. Borst, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Nikita Hanikel, Samuel J. Pellock, Alexis Courbet, William Sheffler, Jue Wang, Preetham Venkatesh, Isaac Sappington, Susana Vázquez Torres, Anna Lauko, Valentin De Bortoli, Emile Mathieu, Sergey Ovchinnikov, Regina Barzilay, Tommi S. Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. 2023. De novo design of protein structure and function with RFdiffusion. Nature 620, 7976 (2023), 1089–1100. doi:10.1038/s41586-023-06415-8

[28] Guanghao Wei, Yining Huang, Chenru Duan, Yue Song, and Yuanqi Du. 2024. Navigating Chemical Space with Latent Flows. In The Thirty-eighth Annual Conference on Neural Information Processing Systems. https://openreview.net/forum?id=aAaV4ZbQ9j

[29] Chih-Hsuan Yang, Rebekah Duke, Parker Delaney Sornberger, Moses Ogbaje, Chad Risko, and Baskar Ganapathysubramanian. 2024. MolGen-Transformer: An open-source self-supervised model for Molecular Generation and Latent Space Exploration. In AI for Accelerated Materials Design - NeurIPS 2024. https://openreview.net/forum?id=wvoZnYhUsR

[30] Claudio Zeni, Robert Pinsler, Daniel Zügner, Andrew Fowler, Matthew Horton, Xiang Fu, Sasha Shysheya, Jonathan Crabbé, Lixin Sun, Jake Smith, Ryota Tomioka, and Tian Xie. 2023. MatterGen: a generative model for inorganic materials design. ArXiv (2023). arXiv:2312.03687