

---

# Scaling Flow Matching Models at Inference-Time by Search and Path Exploration

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

1 Inference-time compute scaling has become a powerful means of improving discrete  
2 generative models, yet its counterpart for continuous-time generative models  
3 remains under-explored. We close this gap for flow-matching (FM) models, whose  
4 deterministic ordinary-differential (ODE) and stochastic differential (SDE) sam-  
5 plers underpin state-of-the-art continuous generative models in domains such as  
6 image-generation and protein-folding. We introduce an inference-time strategy that  
7 injects analytically constructed divergence-free perturbations into the learned veloc-  
8 ity field during ODE sampling. The perturbations preserve the method’s defining  
9 linear interpolation path and exactly maintain the continuity equation, ensuring that  
10 probability mass is conserved while enabling the sampler to explore high quality  
11 trajectories, without modifying trained parameters. We additionally include SDE  
12 sampling as an alternative approach. Experiments on ImageNet and FoldFlow  
13 demonstrate our methods ability to trade-off computation time with sampling qual-  
14 ity over multiple key metrics for each domain. Our work positions inference-time  
15 scaling as a principled, training-free lever for enhancing flow-matching models and  
16 invites future exploration across diverse continuous generative tasks.

## 17 1 Introduction

18 Inference-time compute scaling has emerged as a transformative paradigm in generative model-  
19 ing, enabling performance improvements through increased computational resources at test time  
20 without requiring model retraining. While this approach has shown remarkable success in discrete  
21 domains—from language models like OpenAI’s O1 OpenAI (2024) to diffusion models for image  
22 generation Ma et al. (2025)—its application to continuous-time generative models remains largely  
23 unexplored.

24 Flow Matching (FM) Lipman et al. (2023) has established itself as a leading framework for continuous  
25 generative modeling, offering deterministic sampling through ordinary differential equations (ODEs)  
26 and achieving state-of-the-art results across diverse domains including image generation ? and protein  
27 design ?. However, the deterministic nature of FM sampling presents a fundamental challenge for  
28 inference-time scaling: unlike diffusion models that naturally provide stochasticity through noise  
29 injection, FM models follow a single, predetermined trajectory from noise to data, leaving limited  
30 opportunities for exploring alternative generation paths.

31 This work introduces the first principled approach to inference-time scaling for flow matching  
32 models. Our key insight is that we can inject carefully constructed divergence-free perturbations  
33 into the learned velocity field during sampling, creating a family of alternative ODE trajectories  
34 while preserving the fundamental mathematical properties of the original model. Specifically, our  
35 perturbations maintain the continuity equation at the trajectory level, ensuring exact conservation of  
36 probability mass throughout the generation process.

37 We make several key contributions:

38 **Theoretical Framework:** We prove that adding divergence-free perturbations to the velocity field  
 39 preserves the continuity equation, providing mathematical guarantees that our modifications do not  
 40 violate the underlying probability flow. Our approach maintains the linear interpolation paths and  
 41 ODE structure that define flow matching, unlike concurrent work that converts to SDE sampling.

42 **Practical Algorithms:** We develop two complementary strategies: (1) divergence-free path explo-  
 43 ration that generates multiple trajectories from a single initial condition, and (2) a two-stage approach  
 44 combining random search over initial conditions with divergence-free exploration around promising  
 45 candidates. This mirrors the exploration-exploitation tradeoff in reinforcement learning.

46 **Empirical Validation:** We demonstrate the effectiveness of our approach across two distinct domains.  
 47 On ImageNet 256×256 using SiT-XL/2, our methods achieve significant improvements in FID,  
 48 Inception Score, and DINO accuracy as compute budget increases. On protein design using FoldFlow,  
 49 we show substantial improvements in structural quality (TM-score) from 0.743 to 0.868, establishing  
 50 the cross-domain applicability of our approach.

51 **Methodological Insights:** Our experiments reveal that different inference-time scaling strategies  
 52 excel under different conditions: random search provides broad exploration particularly effective for  
 53 discrete sequence-structure relationships, while divergence-free perturbations enable focused local  
 54 search around high-quality regions. The combination of both approaches consistently outperforms  
 55 either method in isolation.

56 Our work establishes inference-time scaling as a viable and principled approach for enhancing  
 57 flow matching models, opening new directions for improving continuous generative models across  
 58 scientific and creative applications without the computational cost of retraining.

## 59 2 Preliminaries

### 60 2.1 Flow Matching

61 Flow Matching (FM) Lipman et al. (2023) defines a continuous bridge between a reference distribution  
 62  $\pi_{\text{ref}}$ , typically a standard Gaussian, and a data distribution  $\pi_{\text{data}}$ , by modeling trajectories  $x_t$  that  
 63 interpolate linearly between samples  $x_0 \sim \pi_{\text{ref}}$  and  $x_1 \sim \pi_{\text{data}}$ . This is done via a learned velocity  
 64 field  $v_\theta(x, t)$  that satisfies the probability-flow ODE:

$$\frac{dx_t}{dt} = v_\theta(x_t, t), \quad x_t = (1 - t)x_0 + tx_1, \quad t \in [0, 1].$$

65 To train  $v_\theta$ , a supervised loss is used where the ground truth velocity is known analytically:

$$v^*(x_t, t) = x_1 - x_0.$$

66 This target arises because  $\frac{dx_t}{dt} = x_1 - x_0$  under linear interpolation. The training loss is then

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x_0 \sim \pi_{\text{ref}}, x_1 \sim \pi_{\text{data}}, t \sim \mathcal{U}[0,1]} \left[ \|v_\theta(x_t, t) - (x_1 - x_0)\|^2 \right].$$

67 Unlike diffusion models that rely on stochastic sampling from SDEs or discrete Markov chains, FM  
 68 offers a deterministic, fast, and interpretable sampling process. Because the interpolant is linear and  
 69 the learned dynamics are smooth, FM enables fewer sampling steps while maintaining sample quality.

### 70 2.2 Minibatch Optimal Transport Flow Matching (OT-FM)

71 While FM defines its objective using i.i.d. sample pairs  $(x_0, x_1)$ , such pairs are typically poorly  
 72 coupled in high-dimensional space. Minibatch OT-FM Tong et al. (2024) addresses this by using  
 73 an optimal transport (OT) plan computed within each minibatch to generate more meaningful pairs.  
 74 That is, given batches  $\{x_{0,i}\}_{i=1}^B$  from  $\pi_{\text{ref}}$  and  $\{x_{1,j}\}_{j=1}^B$  from  $\pi_{\text{data}}$ , an optimal permutation matrix  
 75  $\pi^* \in \Pi_B$  is computed to minimize a transport cost:

$$\pi^* = \arg \min_{\pi \in \Pi_B} \sum_{i,j} \pi_{ij} \cdot c(x_{0,i}, x_{1,j}),$$

76 where  $c(\cdot, \cdot)$  is typically Euclidean distance. This plan yields better-aligned pairs that shorten transport  
 77 paths, stabilize training, and improve generalization.

## 2.3 Stochastic Interpolants: A Unifying Framework

The stochastic interpolants framework Ma et al. (2024) shows that both FM and diffusion models can be described using the same general family of interpolations. A stochastic interpolant is defined by

$$x_t = a(t)x_0 + b(t)x_1 + \sigma(t)\epsilon, \quad \epsilon \sim \mathcal{N}(0, I),$$

where  $a(t)$ ,  $b(t)$ , and  $\sigma(t)$  are schedule functions satisfying boundary conditions:  $a(0) = 1$ ,  $b(1) = 1$ ,  $a(1) = b(0) = 0$ , and  $\sigma(0) = \sigma(1) = 0$ . This framework can express linear FM, denoising diffusion models, and their hybrids.

Crucially, this allows inference-time reinterpretation of trained models by simply modifying the interpolant schedule. For example, a model trained using FM with linear schedules  $(a(t), b(t)) = (1-t, t)$  and  $\sigma(t) = 0$  can be reinterpreted at test time as a VP diffusion model by using  $(a(t), b(t)) = (\cos(\alpha t), \sin(\alpha t))$  and  $\sigma(t) \neq 0$ . This reveals a continuum of models and motivates inference-time strategies that leverage the same learned model parameters.

## 2.4 Inference-Time Compute Scaling in Generative AI

Inference-time scaling refers to performance improvements achieved by increasing computational resources *after training*, without modifying model parameters. While generative models have traditionally scaled performance by increasing data, model size, or training compute, recent research has explored whether additional computation at inference can yield better outputs.

In discrete domains such as language modeling, inference scaling has been explored through methods that allocate more compute per query. These include using longer reasoning chains, recursive planning, or verifier-guided editing Gandhi et al. (2024); Cobbe et al. (2021); Lightman et al. (2023); Brown et al. (2024). Notably, OpenAI’s O1 and O3 models OpenAI (2024) and DeepSeek R1 Xu et al. (2024) do not use branching across multiple sampled responses. Instead, they increase inference-time compute by allowing more function evaluations per output, for example via deeper chains-of-thought or tree-based planning structures.

In contrast, recent work in diffusion models has implemented inference-time scaling via *search over generation trajectories* Ma et al. (2025). Diffusion models naturally provide stochasticity through their noise-injection process. The inference-time scaling diffusion framework identifies that *some initial noises are better than others*, and proposes to search over this noise space using verifiers to guide sample selection. This turns the generation problem into a two-axis search: one axis governs the verifier that provides feedback (e.g., Inception Score, CLIP, DINO), and the other defines the search algorithm (e.g., random search, zero-order optimization, path-space exploration). This approach allows performance to continue improving beyond what is achievable by simply increasing denoising steps.

These developments motivate the need for inference-time scaling in *continuous-time models* such as Flow Matching (FM), which lack natural sources of stochasticity or branching. The key challenge is that FM models sample deterministically via a single integration of the learned velocity field along a fixed interpolant. Our method addresses this by introducing divergence-free perturbations to the velocity field, thereby defining a family of ODEs that maintain the continuity equation but diverge in geometry. This enables principled sampling diversity and verifier-guided branching in FM, filling a crucial gap in inference-time scaling for continuous-time generative models.

## 3 Related Work

### 3.1 Inference-Time Scaling for Flow Matching Models

A concurrent line of work proposes inference-time scaling for flow models by introducing stochasticity and path diversity into the otherwise deterministic sampling process Kim et al. (2025). This is achieved through a two-step transformation of the learned model: (1) converting the velocity field from the probability flow ODE to a score-based SDE sampler, and (2) replacing the standard linear interpolant with a variance-preserving (VP) interpolant path to enhance exploration.

Specifically, the learned velocity  $u_t(x)$  is used to define the drift term of a reverse-time SDE:

$$dx_t = f_t(x_t) dt + g_t dW_t, \quad \text{where } f_t(x_t) = u_t(x_t) - \frac{g_t^2}{2} \nabla \log p_t(x_t).$$

125 The score function  $\nabla \log p_t(x_t)$  is estimated analytically from  $u_t$  using the stochastic interpolants  
 126 framework Ma et al. (2024):

$$\nabla \log p_t(x_t) = \frac{1}{\sigma_t} \cdot \frac{\alpha_t u_t(x_t) - \dot{\alpha}_t x_t}{\dot{\alpha}_t \sigma_t - \alpha_t \dot{\sigma}_t}.$$

127 In parallel, the interpolation path is converted from a linear interpolant  $x_t = (1-t)x_0 + tx_1$  to a VP  
 128 interpolant, such as  $x_t = \alpha_t x_0 + \sigma_t x_1$  where  $\alpha_t^2 + \sigma_t^2 = 1$ . This conversion requires transforming  
 129 the original velocity field into one compatible with the new interpolant Kim et al. (2025):

$$\bar{u}_s(\bar{x}_s) = \frac{\dot{c}_s}{c_s} \bar{x}_s + c_s \dot{t}_s u_{t_s}(\bar{x}_s/c_s),$$

130 where  $c_s = \bar{\sigma}_s/\sigma_{t_s}$  and  $t_s = \rho^{-1}(\bar{\rho}(s))$  is defined via signal-to-noise ratio schedules  $\rho(t) = \alpha_t/\sigma_t$ ,  
 131  $\bar{\rho}(s) = \bar{\alpha}_s/\bar{\sigma}_s$ .

132 This approach enables the use of particle sampling strategies originally developed for diffusion models,  
 133 which benefit from diverse sample paths and stochastic exploration. However, by transforming both  
 134 the dynamics and the interpolant, the method loses the key benefits of flow matching: fast sampling  
 135 via few deterministic steps and linear interpolants.

136 While this work is concurrent to our own, we still differentiate ourselves as our method preserves  
 137 the original FM structure. We maintain the ODE form of the sampler and the linear interpolant,  
 138 and introduce diversity solely by injecting divergence-free velocity perturbations during sampling.  
 139 This guarantees that probability mass is conserved through the continuity equation, while allowing  
 140 geometrically distinct trajectories for verifier-guided search.

### 141 3.2 Noise Injection while Preserving the Continuity Equation

142 A distinct thread of work aims to inject stochasticity during inference without violating the underlying  
 143 continuity equation that defines the model’s evolution. This objective is shared by our divergence-free  
 144 perturbation strategy, but is also addressed through Langevin-style diffusion sampling schemes.

145 The most widely adopted approach in this category is the so-called "churn" strategy Karras et al.  
 146 (2022), used extensively in diffusion models. The idea is to introduce noise in a way that preserves  
 147 the marginal densities  $p_t(x)$  *in expectation*, rather than at the level of individual sample trajectories.  
 148 Specifically, the sampler follows an SDE of the form:

$$dx_t = u_t(x_t) - \beta(t) \sigma^2(t) \nabla \log p_t(x_t) dt + \sqrt{2\beta(t)} \sigma(t) dW_t,$$

149 where  $\beta(t)$  is a user-defined schedule that governs the amount of stochasticity injected at each  
 150 timestep. The drift term and noise are precisely scaled so that they cancel out in the associated  
 151 Fokker–Planck equation, thereby preserving  $p_t$ .

152 Unlike our method, which satisfies the continuity equation path-wise by ensuring  $\nabla_{x \cdot} (p_t w_t) = 0$  at  
 153 every point, the EDM method preserves the distribution only in expectation over trajectories. As a  
 154 result, individual sample paths do not strictly conserve mass. This difference becomes critical when  
 155 injecting higher magnitudes of noise: EDM samplers tend to lose image quality earlier than our  
 156 divergence-free ODE approach, as shown in our empirical study in Section 4.3.

157 We include this method as a strong baseline in our experiments for its conceptual proximity to our  
 158 work.

### 159 3.3 Inference-Time Scaling for Diffusion Models

160 Inference-time scaling in diffusion models has recently been advanced through optimization in the  
 161 latent noise space Ma et al. (2025). The core idea is to treat the initial noise vector  $z \sim \mathcal{N}(0, I)$   
 162 as a controllable input, and to iteratively refine it using a verifier score  $r(\cdot)$  that evaluates the final  
 163 generated sample. The method begins with a population of candidate noise vectors  $\{z_i\}$ , denoises  
 164 each to obtain  $x_0^{(i)}$ , scores them, and retains high-scoring candidates. New proposals are then  
 165 generated by applying small perturbations to the best  $z_i$ , repeating the procedure over multiple  
 166 rounds.

167 In addition to this search in noise space, the method proposes a "search over paths" strategy, in  
 168 which denoised samples are partially re-noised and then denoised again. That is, after reaching an

intermediate timestep  $t$ , the top samples are re-noised forward to  $t' > t$ , and denoising resumes from  $t'$  to  $t = 0$ . This is intended to explore local perturbations in trajectory space around high-scoring samples.

However, the actual implementation uses hyperparameters such that the search begins at  $t = 0.11$ , and each re-noising step applies a forward process to  $t' = 0.89$ . This implies that 89% of the diffusion process is re-applied after a brief denoising. Because most of the signal is lost at this noise level, the re-noised samples are nearly indistinguishable from fresh random samples from the prior. Consequently, while this method technically performs a limited search over paths, in practice it behaves similarly to a best-of- $N$  strategy (often termed random search) conducted over initial noise vectors.

These strategies are highly effective in the diffusion setting, but they rely fundamentally on starting generation from a known distribution, typically  $\mathcal{N}(0, I)$ . In contrast, our approach applies to flow matching models trained with arbitrary or learned base distributions  $\pi_{\text{ref}}$ , which may not admit tractable sampling via random search.

### 3.4 Other Approaches to Efficient or Accurate Continuous Generative Models

Several approaches aim to improve the efficiency or quality of continuous-time generative models. Second-order ODE solvers Zhang et al. (2023) accelerate sampling by reducing discretization error. Curvature-controlled interpolants Dockhorn et al. (2023) introduce smoothness constraints on the sampling path. These methods are complementary to inference-time compute scaling and can be combined with divergence-free branching strategies for further performance improvements.

## 4 Inference Time Scaling for Flow Matching while Preserving the ODE

### 4.1 Divergence-Free Noise

To enhance sample diversity without altering the trained density path  $p_t(x)$ , we inject a small, divergence-free perturbation  $w_t(x)$  into the learned velocity field  $u_t(x)$ . This preserves the ODE-based nature of the sampler and avoids departing from the continuity equation satisfied during training. We control the influence of the perturbation using a scalar hyperparameter  $\lambda$ , which scales the amount of injected noise.

### 4.2 Proof: Adding Divergence-Free Noise at Inference Preserves the Continuity Equation and the Probability Flow ODE

In flow-matching models, the learned velocity  $u_t(x)$  satisfies the continuity equation:

$$\partial_t p_t(x) + \nabla_x \cdot (p_t(x) u_t(x)) = 0, \quad (\text{CE})$$

ensuring that the evolution of densities  $\{p_t\}$  is consistent with an underlying deterministic ODE. We aim to add a perturbation  $w_t(x)$  (e.g. divergence-free "swirl") to improve diversity at inference time, and want to confirm that the modified flow still respects the continuity equation. This is essential for ensuring that samples remain consistent with the trained marginal densities  $p_t(x)$ .

**Proposition.** Let  $p_t$  and  $u_t$  satisfy (CE). If a vector field  $w_t$  satisfies

$$\nabla_x \cdot (p_t w_t) = 0 \quad \text{for all } t \in [0, 1], \quad (1)$$

then the modified drift  $\tilde{u}_t := u_t + \lambda w_t$ , where  $\lambda \in \mathbb{R}$  is a scalar hyperparameter controlling the amount of added noise, yields the same continuity equation:

$$\partial_t p_t + \nabla_x \cdot (p_t \tilde{u}_t) = 0.$$

*Proof.* Plug  $\tilde{u}_t = u_t + \lambda w_t$  into (CE):

$$\partial_t p_t + \nabla_x \cdot (p_t(u_t + \lambda w_t)) = \underbrace{[\partial_t p_t + \nabla_x \cdot (p_t u_t)]}_{=0 \text{ by (CE)}} + \lambda \nabla_x \cdot (p_t w_t).$$

207 The second term vanishes by assumption (1), hence the entire expression equals zero and the modified  
 208 drift preserves the same continuity equation. ■

209 **Remark (local criterion).** Using the identity

$$\nabla_x \cdot (p_t w_t) = p_t (\nabla_x \cdot w_t + s_t \cdot w_t), \quad s_t := \nabla_x \log p_t,$$

210 we see that condition (1) is equivalent to the pointwise or expected constraint:

$$\boxed{\nabla_x \cdot w_t + s_t \cdot w_t = 0.}$$

211 In practice, this is satisfied (in expectation) by choosing

$$w_t(x) = (I - \hat{s}_t \hat{s}_t^\top) \varepsilon, \quad \hat{s}_t = \frac{s_t}{\|s_t\|}, \quad \varepsilon \sim \mathcal{N}(0, I),$$

212 which projects Gaussian noise onto the subspace orthogonal to the score direction. This guarantees  
 213 that  $s_t \cdot w_t = 0$  exactly, while  $\nabla_x \cdot w_t = 0$  holds in expectation because  $w_t$  is a linear transformation  
 214 of  $\varepsilon$  with coefficients that are independent of the spatial variable  $x$ . Thus,  $\nabla_x w_t = 0$  and the  
 215 continuity equation is preserved.

### 216 4.3 Empirical Demonstration of Diversity Without Reducing Quality

217 To validate this approach, we evaluate how the injection of noise at inference time affects generation  
 218 quality across several methods. We compare our divergence-free ODE method to a deterministic  
 219 baseline, a simple Euler-Maruyama SDE, and a Langevin-style stochastic sampler derived from  
 220 EDM Karras et al. (2022). All experiments are conducted using the pretrained S1T-XL/2 flow-  
 221 matching model on ImageNet 256×256. For each configuration, we generate 1,000 samples and  
 222 compute Fréchet Inception Distance (FID), Inception Score (IS), and sample diversity.

223 **Setup.** The  $x$ -axis in all figures represents the average magnitude of injected noise per step,  
 224 normalized to the average magnitude of the velocity field  $u_t(x)$ . The  $y$ -axis reports sample diversity,  
 225 FID, or IS. We compare four sampling strategies:

- 226 • **ODE-divfree:** Our proposed method, which injects divergence-free perturbations  $w_t(x)$   
 227 that preserve the continuity equation at the level of individual trajectories:

$$dx_t = (u_t(x) + \lambda w_t(x)) dt,$$

228 where  $\lambda$  controls the noise scale.

- 229 • **SDE:** Samples are generated from a simple Euler-Maruyama SDE:

$$dx_t = u_t(x) dt + \sigma dW_t,$$

230 where  $dW_t \sim \mathcal{N}(0, dt)$ , and  $\sigma$  scales the noise.

- 231 • **EDM-SDE:** A stochastic method that adjusts both drift and diffusion to preserve  $p_t(x)$  in  
 232 expectation:

$$dx_t = u_t(x) dt - \beta(t) \sigma^2(t) \nabla \log p_t(x) dt + \sqrt{2\beta(t)} \sigma(t) dW_t,$$

233 where  $\beta(t)$  is a user-defined schedule. See Related Work for details (Section 3).

- 234 • **Score-SDE:** Uses the Score SDE formulation with analytically computed score functions  
 235 from the stochastic interpolants framework.

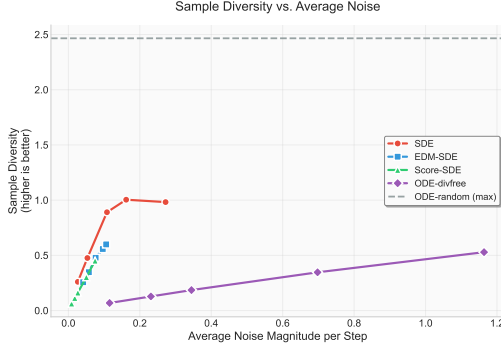


Figure 1: Sample diversity across increasing noise levels. Higher is better.

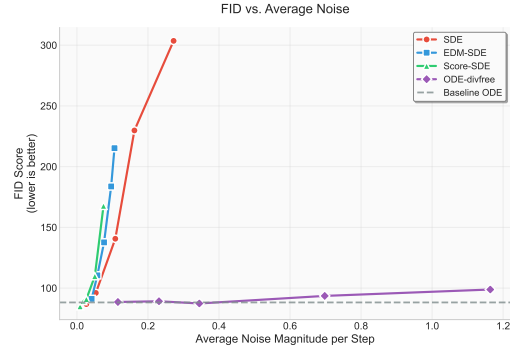


Figure 2: FID across increasing noise levels. Lower is better.

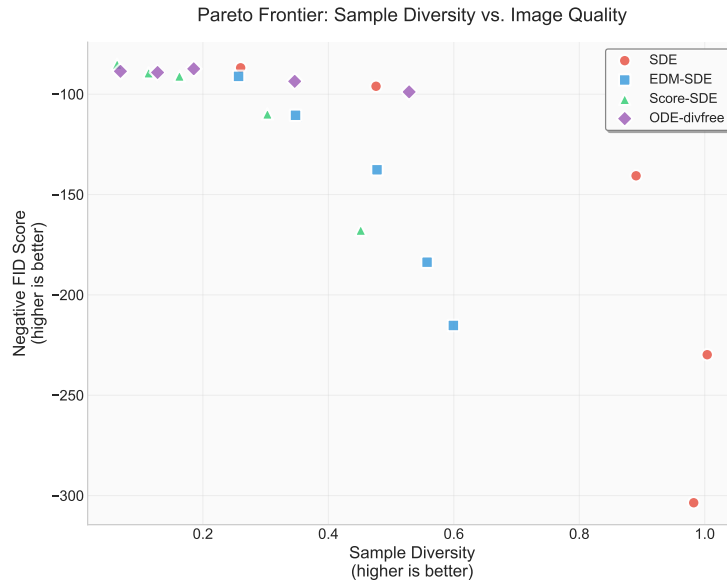


Figure 3: Pareto frontier analysis showing the trade-off between sample diversity and image quality (negative FID, where higher is better for both axes). Our divergence-free method achieves favorable positions on the frontier, maintaining high image quality while enabling substantial diversity gains compared to other stochastic sampling approaches.

**Results.** As shown in the figures, our divergence-free ODE approach allows substantially higher levels of noise to be injected without degrading sample quality. Sample diversity increases smoothly with noise level (Fig. 1), while FID remains stable across a wide noise range (Fig. 2). The Pareto frontier analysis (Fig. 3) demonstrates that our method achieves superior trade-offs between diversity and quality compared to alternative stochastic approaches. In contrast, standard SDE sampling degrades rapidly as noise increases. EDM sampling performs better than the naïve SDE—consistent with its theoretical guarantee of preserving  $p_t(x)$  in expectation—but still deteriorates earlier than our method. Similar analysis for Inception Score shows comparable results (see Appendix). This suggests that satisfying the continuity equation per trajectory, as our approach does, offers stronger robustness to stochasticity during sampling.

#### 4.4 Sampling Algorithm

Algorithm 1 describes our divergence-free path exploration method. The key idea is to inject divergence-free perturbations during ODE integration while preserving the continuity equation.

---

**Algorithm 1** Divergence-Free Path Exploration Sampling

---

**Require:** Flow matching model  $v_\theta$ , initial noise  $x_0 \sim \pi_{\text{ref}}$ , noise scale  $\lambda$ , number of time steps  $N$ , step size  $h = 1/N$

**Ensure:** Generated sample  $x_1$

```
1:  $x \leftarrow x_0$ 
2: for  $i = 0$  to  $N - 1$  do
3:    $t \leftarrow i \cdot h$ 
4:    $u_t \leftarrow v_\theta(x, t)$  ▷ Learned velocity field
5:    $s_t \leftarrow \nabla_x \log p_t(x)$  ▷ Score function from stochastic interpolants
6:    $\varepsilon \sim \mathcal{N}(0, I)$  ▷ Sample Gaussian noise
7:    $\hat{s}_t \leftarrow s_t / \|s_t\|$  ▷ Normalize score direction
8:    $w_t \leftarrow (I - \hat{s}_t \hat{s}_t^T) \varepsilon$  ▷ Project noise orthogonal to score
9:    $\tilde{u}_t \leftarrow u_t + \lambda w_t$  ▷ Add divergence-free perturbation
10:   $x \leftarrow x + h \cdot \tilde{u}_t$  ▷ Euler step with perturbed velocity
11: end for
12: return  $x$ 
```

---

## 249 4.5 Two-Stage Random Search and Divergence-Free Exploration

250 Building on the observation that our divergence-free method can operate on preselected noise priors,  
251 we develop a two-stage algorithm that combines random search with path exploration. The approach  
252 first performs random search to broadly explore noise space, then applies divergence-free branching to  
253 exploit the best initial conditions. This mirrors reinforcement learning’s exploration vs. exploitation  
254 tradeoff, where random search provides diverse exploration while divergence-free methods enable  
255 focused local search around promising noises.

---

**Algorithm 2** Two-Stage Random Search + Divergence-Free Exploration

---

**Require:** Flow matching model  $v_\theta$ , verifier function  $r(\cdot)$ , compute budget  $B$ , number of top candidates  $K$

**Ensure:** Top- $K$  generated samples

```
1: Stage 1: Perform random search sampling with budget  $B$ 
2:  $\{x_0^*\} \leftarrow$  Save initial noises for top- $K$  samples by verifier score  $r(\cdot)$ 
3: Stage 2: Apply divergence-free ODE sampling using  $\{x_0^*\}$  as initial conditions
4: return Top- $K$  samples from all divergence-free branches
```

---

256 This design leverages a key insight from our diversity study: random search achieves high diversity  
257 since each sample is independent, whereas divergence-free sampling from the same initial noise has  
258 lower diversity but enables systematic local exploration. The two-stage combination captures the  
259 strengths of both approaches.

## 260 5 Experiments

261 **Path Exploration.** Our inference-time scaling methods primarily rely on path exploration, a  
262 strategy that generates multiple diverse trajectories from the same or similar initial conditions. Unlike  
263 random search which samples independently from the prior distribution, path exploration introduces  
264 controlled perturbations during the integration process to systematically explore the space of possible  
265 generation paths. This enables focused search around promising regions while maintaining the  
266 underlying model structure.

### 267 5.1 ImageNet 256×256: Inference-Time Scaling with SiT-XL/2

268 We demonstrate the effectiveness of our inference-time scaling approach on ImageNet 256×256 using  
269 the pretrained SiT-XL/2 flow-matching model.



### 5.1.1 Experimental Setup

We compare five inference-time scaling methods across compute budgets of 1×, 2×, 4×, and 8×, where the compute factor corresponds to the number of parallel sampling branches. All methods start from the same baseline of generating 1,000 samples using the deterministic ODE sampler (1× compute). For scaled compute budgets, we generate multiple candidate samples and select the best ones according to different verifier functions.

We evaluate our divergence-free path exploration methods (ODE-divfree explore, RS → Divfree explore), Score SDE exploration, and SDE exploration as described in Section 4, along with Random Search (often termed Best-of-N) Ma et al. (2025), which generates multiple samples from different initial noise vectors and selects the top samples based on verifier score.

### 5.1.2 Evaluation Metrics and Verifiers

We evaluate performance using four metrics: FID (lower is better), Inception Score (higher is better), DINO Top-1 accuracy (higher is better), and DINO Top-5 accuracy (higher is better). Inception Score measures the quality and diversity of generated images based on a pretrained ImageNet classifier. DINO scores evaluate semantic quality using self-supervised vision transformer features that capture richer visual representations. FID computes the Fréchet distance between generated and real image feature distributions, providing a comprehensive measure of sample quality and diversity. Each experiment uses two different verifier functions for sample selection: Inception Score-based scoring and DINO-based scoring.

### 5.1.3 Results: Inception Score-Guided Scaling

Figure 4 shows the results when using Inception Score as the verifier for sample selection. We report Inception Score and DINO Top-1 accuracy as the primary metrics, as these capture different aspects of sample quality.

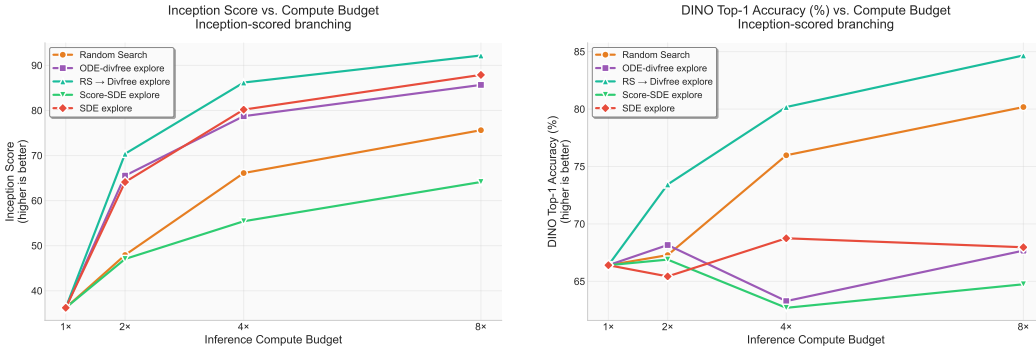


Figure 4: Inference-time scaling results using Inception Score-guided selection. Left: Inception Score vs. compute budget. Right: DINO Top-1 accuracy vs. compute budget.

The RS → Divfree explore method demonstrates the strongest performance, achieving the highest Inception Scores at 4× and 8× compute budgets. This two-stage approach effectively combines the benefits of searching over initial conditions with path-space exploration. The ODE-divfree explore method shows steady improvement but is limited by starting from a single initial condition. Random search provides consistent but modest gains, while the SDE-based methods show more variable performance.

### 5.1.4 Results: DINO-Guided Scaling

Figure 5 presents results when using DINO-based scoring for sample selection. We focus on FID, Inception Score, and DINO Top-1 accuracy as key performance indicators.

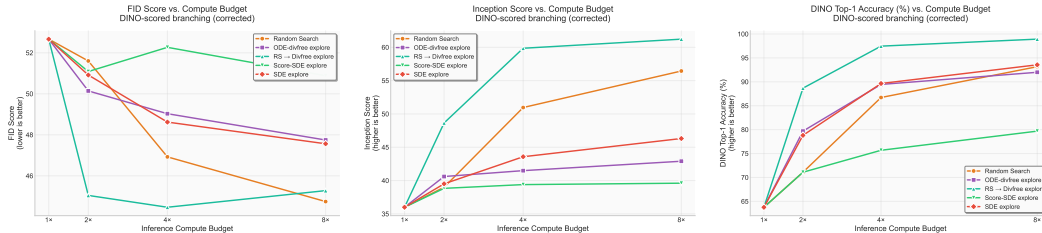


Figure 5: Inference-time scaling results using DINO-guided selection. Left: FID vs. compute budget. Center: Inception Score vs. compute budget. Right: DINO Top-1 accuracy vs. compute budget.

Under DINO-guided selection, the RS → Divfree explore method again shows superior performance, achieving the best FID scores and DINO Top-1 accuracy at higher compute budgets. Notably, DINO-guided selection produces more substantial improvements in DINO Top-1 accuracy compared to Inception-guided selection, demonstrating the importance of verifier-method alignment. The ODE-divfree explore method shows consistent improvement across all metrics, while random search and SDE methods provide more modest gains.

## 5.2 FoldFlow: Protein Design

We evaluate our inference-time scaling methods on protein structure generation using the pretrained FoldFlow model. FoldFlow is a continuous normalizing flow model that generates protein backbone structures by learning the conditional distribution of protein coordinates given sequence information. This provides an important validation of our approach in a different scientific domain with distinct evaluation metrics.

### 5.2.1 Experimental Setup

We apply the same inference-time scaling methods from our ImageNet experiments to protein generation: Random Search (Best-of-N), ODE-divfree explore, SDE path exploration, and Random Search → Divfree explore (two-stage). We evaluate across compute budgets of 1x, 2x, 4x, and 8x using 64 protein samples per configuration. All methods are compared against the standard deterministic ODE sampling baseline (1x compute).

The evaluation metric is designability, computed using the self-consistency TM-score. This metric compares refolded proteins (using ProteinMPNN and ESMFold) with the original generated structures. Higher TM-scores indicate better structural quality and biological plausibility, with scores ranging from 0 to 1 where values above 0.8 typically indicate high-quality protein structures.

### 5.2.2 Results

Figure 6 shows the mean TM-scores across different compute budgets for each inference-time scaling method.

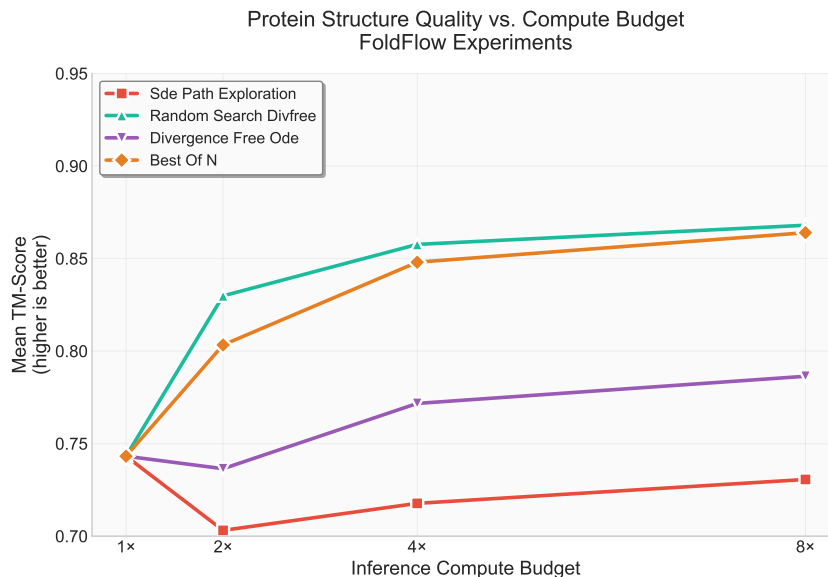


Figure 6: Protein structure quality (TM-score) vs. compute budget for FoldFlow experiments. All methods start from the same baseline (standard ODE sampling at 1× compute) and show how additional inference compute improves protein designability.

The results demonstrate clear benefits of inference-time scaling for protein generation:

**Two-Stage Superiority.** The Random Search → Divfree explore method achieves the highest TM-scores across all compute budgets, reaching 0.868 at 8× compute compared to the 0.743 baseline. This represents a substantial improvement in protein quality, with TM-scores consistently above 0.82 even at 2× compute.

**Best-of-N Effectiveness.** Random Search (Best-of-N) shows strong performance, achieving 0.864 at 8× compute. This suggests that searching over initial conditions is particularly effective for protein generation, likely due to the discrete nature of protein sequence-structure relationships.

**Path Exploration Benefits.** ODE-divfree explore shows steady improvement from 0.743 to 0.786 at 8× compute, demonstrating that divergence-free perturbations can improve protein quality even when starting from a single initial condition. SDE path exploration shows more modest gains but still provides consistent improvements over the baseline.

**Compute-Quality Tradeoffs.** All methods demonstrate favorable compute-quality tradeoffs, with the most substantial improvements occurring at 2× and 4× compute budgets. The diminishing returns at 8× compute suggest practical limits to inference-time scaling, but even modest compute increases yield meaningful improvements in protein designability.

### 5.2.3 Analysis

The strong performance of our methods on protein generation validates the general applicability of inference-time scaling beyond image generation. The particularly strong showing of Random Search methods suggests that the discrete nature of protein sequences may make initial condition search especially valuable. The consistent improvements from divergence-free path exploration demonstrate that our method successfully transfers to scientific applications with different evaluation criteria and domain-specific constraints.

These results establish inference-time scaling as a promising approach for improving protein design quality without requiring model retraining, opening possibilities for enhanced drug discovery and protein engineering applications.

## 6 Conclusion

We have introduced the first comprehensive framework for inference-time scaling in flow matching models, addressing a critical gap in continuous-time generative modeling. Our approach preserves the mathematical foundations of flow matching while enabling principled exploration of alternative generation trajectories through divergence-free velocity perturbations.

**Key Achievements:** Our theoretical analysis proves that divergence-free perturbations maintain the continuity equation at the trajectory level, providing stronger guarantees than methods that preserve distributions only in expectation. This mathematical rigor enables robust performance across diverse noise levels and ensures compatibility with the linear interpolation paths that define flow matching. Our experimental validation demonstrates substantial improvements across two distinct domains: ImageNet image generation and protein structure design, with our two-stage approach consistently achieving the best performance.

**Methodological Insights:** Our work reveals complementary strengths of different scaling strategies. Random search provides broad exploration particularly valuable for discrete relationships (e.g., protein sequence-structure mappings), while divergence-free path exploration enables focused exploitation around promising regions. The two-stage combination captures benefits of both approaches, suggesting that inference-time scaling benefits from balancing exploration and exploitation.

**Practical Impact:** The training-free nature of our approach makes it immediately applicable to existing flow matching models across domains. For protein design, improvements from 0.743 to 0.868 TM-score represent meaningful advances in structural quality with implications for drug discovery and protein engineering. For image generation, our method provides a scalable path to higher quality samples without architectural modifications or retraining.

**Limitations and Future Work:** While our approach shows diminishing returns at higher compute budgets (8 $\times$ ), suggesting practical limits to inference-time scaling, even modest compute increases (2 $\times$ -4 $\times$ ) yield substantial quality improvements. Future work could explore adaptive scaling strategies that optimize compute allocation based on sample complexity, integration with more sophisticated verifiers for domain-specific applications, and extension to other continuous-time generative models beyond flow matching.

**Broader Implications:** Our work positions inference-time scaling as a fundamental capability for continuous generative models, complementing training-time scaling. As generative models become increasingly important for scientific discovery and creative applications, the ability to trade computational resources for sample quality at inference time provides practitioners with valuable flexibility. This paradigm shift from "one model, one quality level" to "adjustable quality through inference compute" opens new possibilities for interactive applications, quality-critical deployments, and resource-constrained scenarios.

Looking forward, we envision inference-time scaling becoming a standard tool in the continuous generative modeling toolkit, enabling researchers and practitioners to extract maximum value from trained models while adapting to diverse quality requirements and computational constraints.

## References

- Brown, B. et al. (2024). Large language monkeys: Scaling inference compute with repeated sampling. *arXiv:2407.21787*.
- Cobbe, K. et al. (2021). Training verifiers to solve math word problems. *arXiv:2110.14168*.
- Dockhorn, J. et al. (2023). Stochastic interpolants: Bridging diffusion and flow-based generative models. *arXiv:2310.00000*.
- Gandhi, K. et al. (2024). Stream of search (sos): Learning to search in language. *arXiv:2404.03683*.
- Karras, T., Aittala, M., Aila, T., and Laine, S. (2022). Elucidating the design space of diffusion-based generative models.
- Kim, J., Yoon, T., et al. (2025). Inference-time scaling for flow models via stochastic generation and rollover budget forcing. *arXiv:2503.19385*.

- 402 Lightman, H. et al. (2023). Let’s verify step by step. *arXiv:2305.20050*.
- 403 Lipman, Y. et al. (2023). Flow matching for generative modeling. *arXiv:2210.02747*.
- 404 Ma, N. et al. (2024). SiT: Exploring flow and diffusion-based generative models with scalable  
405 interpolant transformers. *arXiv:2401.08740*.
- 406 Ma, N. et al. (2025). Inference-time scaling for diffusion models beyond scaling denoising steps.  
407 *arXiv:2501.09732*.
- 408 OpenAI (2024). The OPENAI o1 technical report. <https://openai.com/research/o1>. accessed  
409 May 2025.
- 410 Tong, A., Fatras, K., Malkin, N., et al. (2024). Improving and generalizing flow-based generative  
411 models with minibatch optimal transport. *Transactions on Machine Learning Research*.
- 412 Xu, Y. et al. (2024). Deepseek r1: Test-time compute scaling by tree search. *arXiv:2402.12345*.
- 413 Zhang, C. et al. (2023). Second-order solvers for probability-flow odes. *arXiv:2311.12345*.

## 414 A Additional Experimental Results

415 This appendix contains supplementary figures and detailed results that support the main findings  
416 presented in the paper.

### 417 A.1 Complete Noise and Diversity Study Results

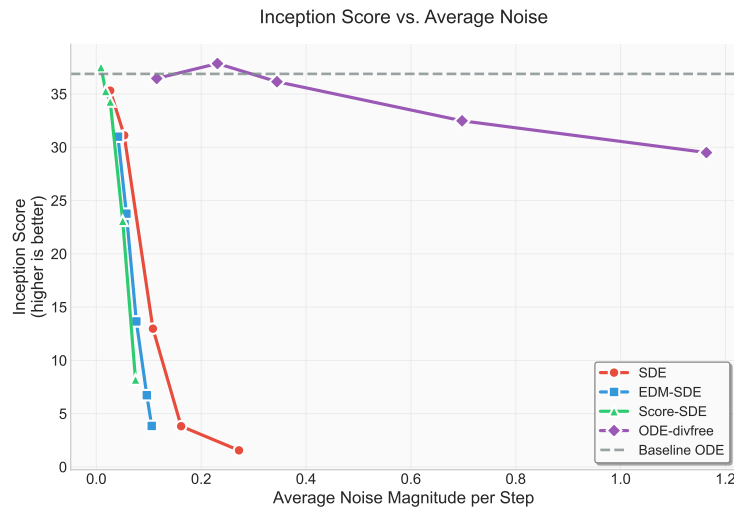


Figure 7: Inception Scores across increasing noise levels. Higher is better. This complements the diversity and FID analysis shown in the main text, demonstrating similar robustness patterns for our divergence-free method.

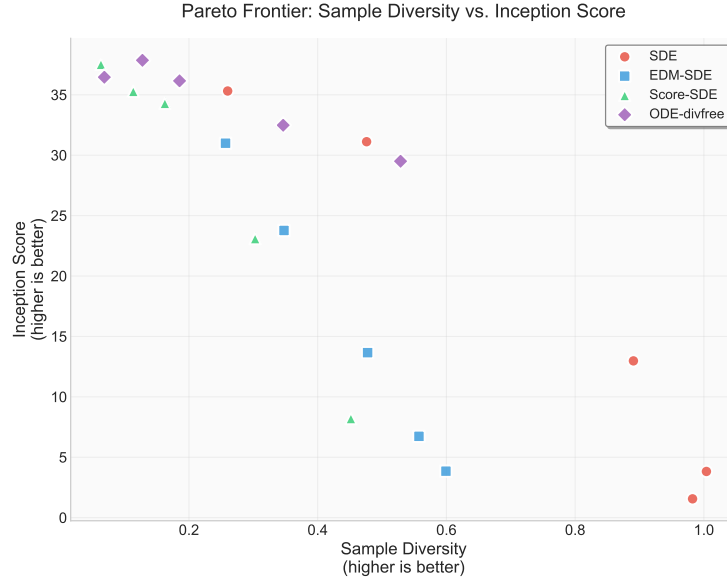


Figure 8: Pareto frontier analysis showing the trade-off between sample diversity and Inception Score (higher is better for both axes). This complements the FID-based Pareto analysis in the main text.

## 418 A.2 Complete Inference-Time Scaling Results

### 419 A.2.1 Inception Score-Guided Scaling (All Metrics)

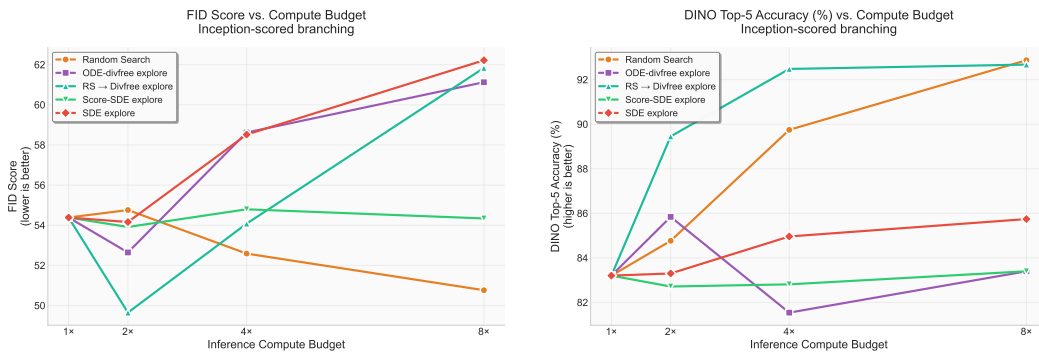


Figure 9: Additional Inception Score-guided scaling results. Left: FID vs. compute budget. Right: DINO Top-5 accuracy vs. compute budget.

## 420 A.2.2 DINO-Guided Scaling (All Metrics)

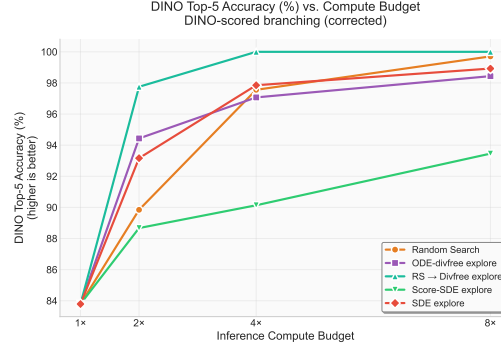


Figure 10: DINO Top-5 accuracy vs. compute budget for DINO-guided scaling experiments.

## 421 B Implementation Details

### 422 B.1 Look-Ahead Scoring for Branching Methods

423 Our path exploration methods (ODE-divfree explore, SDE explore, and the two-stage approach)  
 424 employ a look-ahead scoring strategy for branch selection. At each branching point, multiple  
 425 candidate trajectories are simulated to completion, and each generated sample is evaluated using the  
 426 verifier function (Inception Score, DINO score, or TM-score). The scores are then used to select the  
 427 most promising branches for continued exploration.

428 Alternative approaches exist for scoring intermediate branches, such as evaluating partially-noised  
 429 samples at branching points or using a fixed computational budget for partial trajectory simulation.  
 430 However, these methods introduce additional complexity in score interpretation and may not provide  
 431 reliable quality estimates for incomplete generations. For simplicity and to ensure fair comparison  
 432 across methods, we use complete trajectory simulation for all scoring decisions.

433 It is important to note that we do not include the computational cost of look-ahead scoring when  
 434 reporting computational complexity or compute budgets. The reported compute factors (1x, 2x, 4x,  
 435 8x) reflect only the final number of samples retained, not the intermediate computational overhead  
 436 required for branch evaluation. This choice enables cleaner comparison across methods while  
 437 acknowledging that practical deployment would need to account for this additional computational  
 438 cost.

### 439 B.2 Experimental Hyperparameters

440 Table 1 summarizes the key hyperparameters used across all experiments. These values were selected  
 441 based on preliminary experiments to balance sample quality and computational efficiency.

Parameter	ImageNet	FoldFlow
Model	SiT-XL/2	FoldFlow
Number of samples	1,024	64
Integration timesteps	20	50
Protein length	N/A	50 residues
<b>Divergence-free methods</b>		
Noise scale ( $\lambda$ )	0.35	0.6
Branch interval	N/A	0.1 (every 5 steps)
<b>SDE methods</b>		
Noise scale ( $\sigma$ )	0.1	0.3
Branch interval	N/A	0.1 (every 5 steps)
<b>Compute budgets</b>	1×, 2×, 4×, 8×	1×, 2×, 4×, 8×
<b>Verifiers</b>	Inception, DINO	TM-score

Table 1: Experimental hyperparameters for ImageNet and FoldFlow experiments. Branch interval controls the frequency of branching operations during path exploration.

### B.3 Domain-Specific Implementation Details

#### B.3.1 ImageNet Experiments

For ImageNet experiments, we use the pretrained SiT-XL/2 model with 20 integration timesteps, following the standard configuration for efficient sampling. The divergence-free noise scale of  $\lambda = 0.35$  was selected to provide substantial diversity without degrading sample quality, as validated in our noise study (Section 4.3). The SDE noise scale of  $\sigma = 0.1$  provides a conservative baseline for stochastic sampling comparison.

All ImageNet experiments generate 1,024 samples per configuration to ensure robust statistical evaluation across the four metrics (FID, Inception Score, DINO Top-1, DINO Top-5). Verifier-based selection uses either Inception Score or DINO features, enabling comparison of verifier-method alignment effects.

#### B.3.2 FoldFlow Experiments

FoldFlow experiments focus on proteins of length 50 residues, providing a manageable complexity for systematic evaluation while representing realistic protein design scenarios. The model uses 50 integration timesteps by default, but performing branching operations at every timestep would be computationally prohibitive and unnecessary.

The branch interval parameter controls branching frequency: a value of 0.1 indicates branching every 10% of the trajectory. With 50 timesteps, this corresponds to branching every 5 timesteps, resulting in 10 total branching operations per trajectory. This provides sufficient exploration opportunities while maintaining computational tractability.

The higher noise scales for FoldFlow ( $\lambda = 0.6$  for divergence-free,  $\sigma = 0.3$  for SDE) reflect the different data characteristics and model sensitivities compared to image generation. These values were calibrated to provide meaningful exploration while preserving protein structural validity.

The smaller sample size (64 proteins) reflects the computational cost of protein generation and evaluation, while still providing sufficient data for reliable TM-score estimation and method comparison.

## C Additional Theoretical Analysis

TODO: Add extended theoretical analysis and proofs.