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

Anonymous Author(s)
Affiliation
Address
email

Abstract

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

1 Introduction

TODO

2 Preliminaries

2.1 Flow Matching

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

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

To train v_θ, a supervised loss is used where the ground truth velocity is known analytically:

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

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

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

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

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

2.2 Minibatch Optimal Transport Flow Matching (OT-FM)

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

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

where $c(\cdot, \cdot)$ is typically Euclidean distance. This plan yields better-aligned pairs that shorten transport 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:

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

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

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

In parallel, the interpolation path is converted from a linear interpolant $x_t = (1-t)x_0 + tx_1$ to a VP 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 the original velocity field into one compatible with the new interpolant Kim et al. (2025):

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

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 $\rho(t) = \alpha_t/\sigma_t$, $\bar{\rho}(s) = \bar{\alpha}_s/\bar{\sigma}_s$.

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

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

3.2 Noise Injection while Preserving the Continuity Equation

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

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

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

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

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

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

3.3 Inference-Time Scaling for Diffusion Models

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

In addition to this search in noise space, the method proposes a "search over paths" strategy, in 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 π_{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 λ, 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:

$$\partial_t p_t(x) \;+\; \nabla_x \cdot \big(p_t(x)\, u_t(x)\big) \;=\; 0, \tag{CE}$$

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

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

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:

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

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

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

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

Remark (local criterion). Using the identity

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

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

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

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

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

which projects Gaussian noise onto the subspace orthogonal to the score direction. This guarantees 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 of ε with coefficients that are independent of the spatial variable x. Thus, $\nabla_x w_t = 0$ and the continuity equation is preserved.

4.3 Empirical Demonstration of Diversity Without Reducing Quality

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

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

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

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

  where λ controls the noise scale.

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

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

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

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

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

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

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

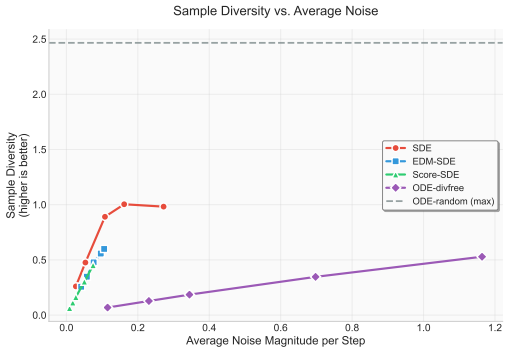

Figure 1: Sample diversity across increasing noise levels. Higher is better.

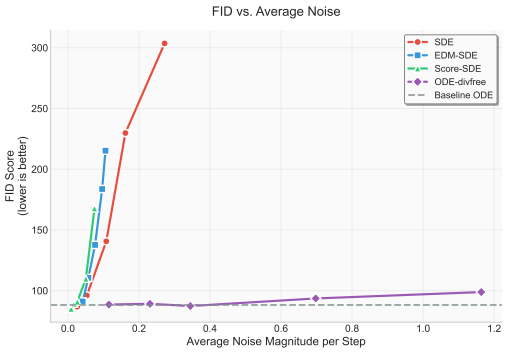

Figure 2: FID across increasing noise levels. Lower is better.

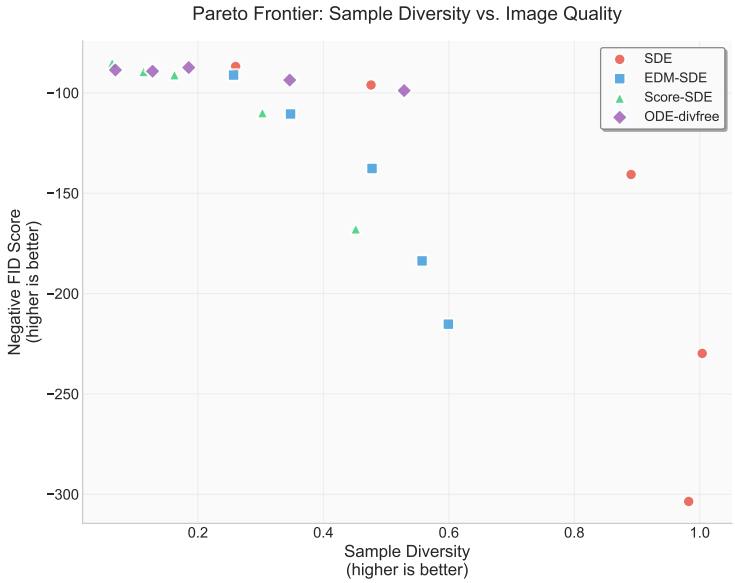

Figure 3: Pareto frontier analysis showing the trade-off between sample diversity and image quality (negative FID, where higher is better for both axes). Our divergence-free method achieves favorable positions on the frontier, maintaining high image quality while enabling substantial diversity gains 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_θ, initial noise $x_0 \sim \pi_{\text{ref}}$, noise scale λ, number of time steps N, step size $h = 1/N$
Ensure: Generated sample x_1

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

4.5 Two-Stage Random Search and Divergence-Free Exploration

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

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

Require: Flow matching model v_θ, verifier function $r(\cdot)$, compute budget B, number of top candidates K
Ensure: Top-K generated samples

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

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

5 Experiments

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

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

We demonstrate the effectiveness of our inference-time scaling approach on ImageNet 256×256 using 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.

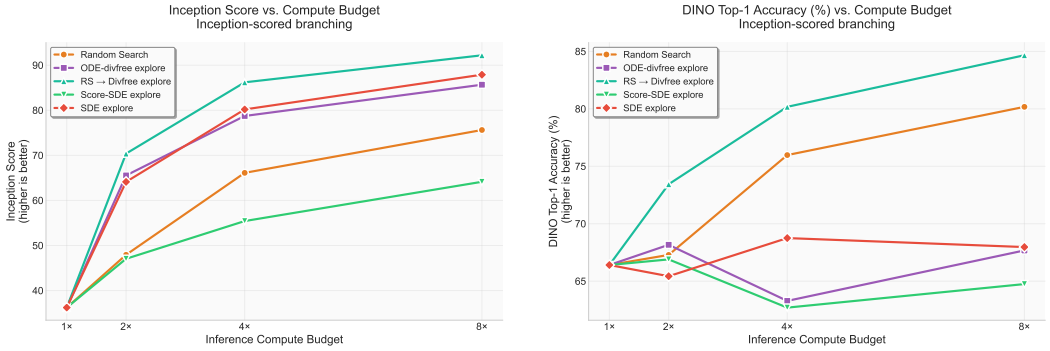

Figure 4: Inference-time scaling results using Inception Score-guided selection. Left: Inception Score vs. compute budget. Right: DINO Top-1 accuracy vs. compute budget.

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.

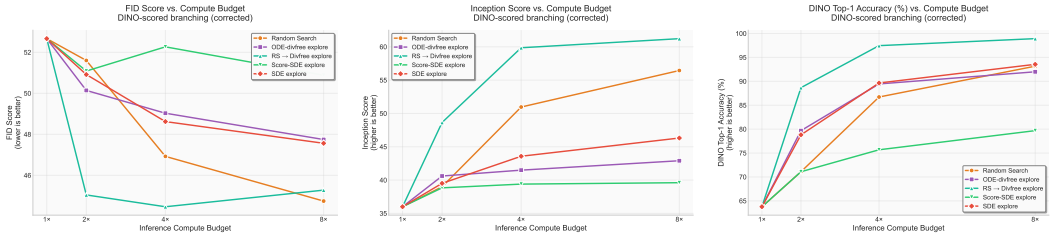

Figure 5: Inference-time scaling results using DINO-guided selection. Left: FID vs. compute budget. Center: Inception Score vs. compute budget. Right: DINO Top-1 accuracy vs. compute budget.

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 1×, 2×, 4×, and 8× using 64 protein samples per configuration. All methods are compared against the standard deterministic ODE sampling baseline (1× 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.

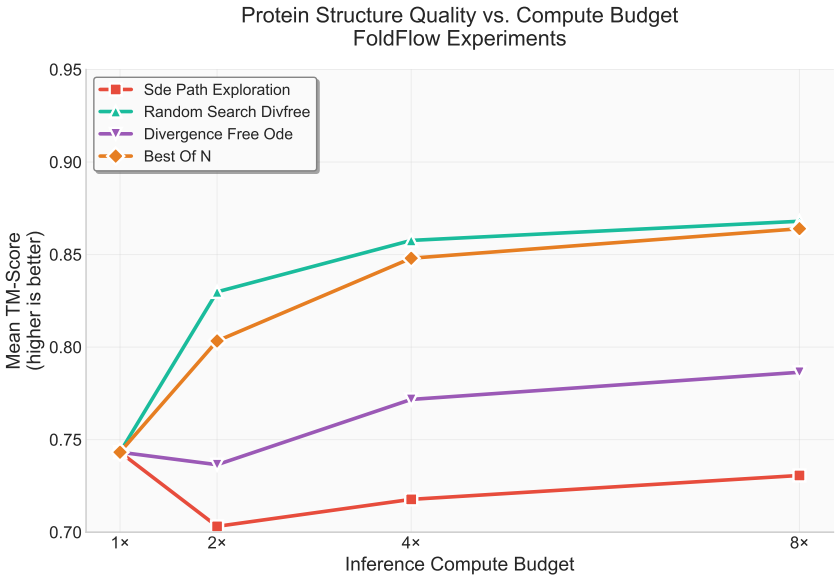

Figure 6: Protein structure quality (TM-score) vs. compute budget for FoldFlow experiments. All methods start from the same baseline (standard ODE sampling at 1× compute) and show how additional inference compute improves protein designability.

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

TODO: Add conclusion

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*.

Lightman, H. et al. (2023). Let's verify step by step. *arXiv:2305.20050*.

Lipman, Y. et al. (2023). Flow matching for generative modeling. *arXiv:2210.02747*.

Ma, N. et al. (2024). SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. *arXiv:2401.08740*.

Ma, N. et al. (2025). Inference-time scaling for diffusion models beyond scaling denoising steps. *arXiv:2501.09732*.

OpenAI (2024). The OpenAI o1 technical report. `https://openai.com/research/o1`. accessed May 2025.

Tong, A., Fatras, K., Malkin, N., et al. (2024). Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*.

Xu, Y. et al. (2024). Deepseek r1: Test-time compute scaling by tree search. *arXiv:2402.12345*.

Zhang, C. et al. (2023). Second-order solvers for probability-flow odes. *arXiv:2311.12345*.

A Additional Experimental Results

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

A.1 Complete Noise and Diversity Study Results

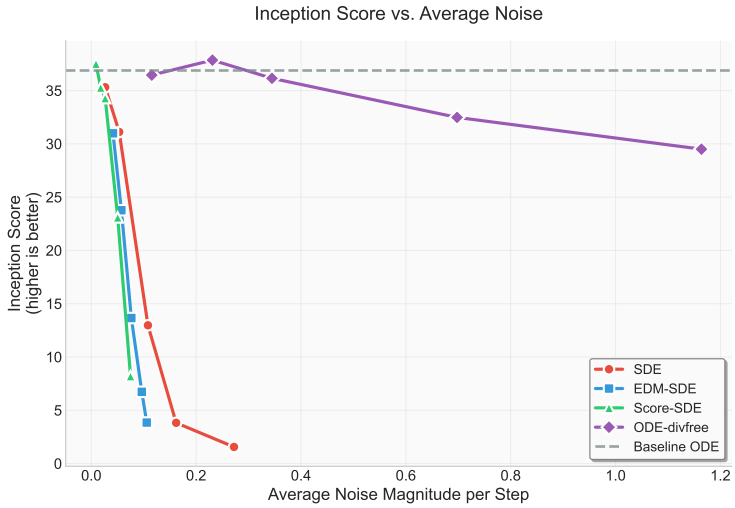

Figure 7: Inception Scores across increasing noise levels. Higher is better. This complements the diversity and FID analysis shown in the main text, demonstrating similar robustness patterns for our divergence-free method.

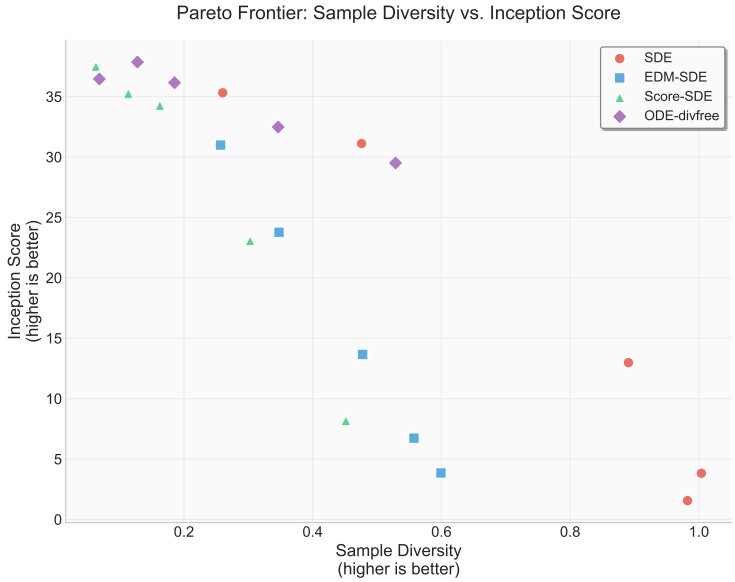

Figure 8: Pareto frontier analysis showing the trade-off between sample diversity and Inception Score (higher is better for both axes). This complements the FID-based Pareto analysis in the main text.

A.2 Complete Inference-Time Scaling Results

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

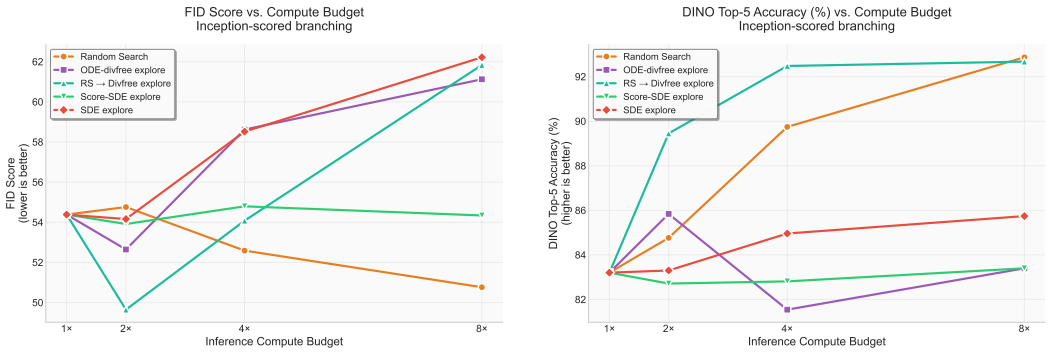

Figure 9: Additional Inception Score-guided scaling results. Left: FID vs. compute budget. Right: DINO Top-5 accuracy vs. compute budget.

A.2.2 DINO-Guided Scaling (All Metrics)

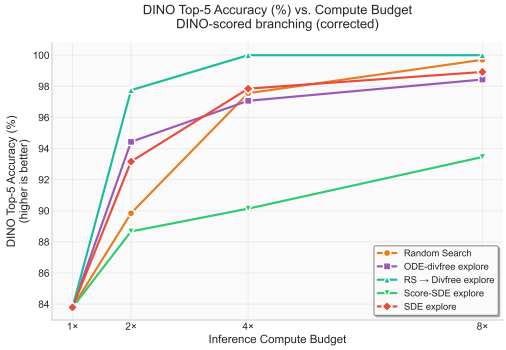

Figure 10: DINO Top-5 accuracy vs. compute budget for DINO-guided scaling experiments.

B Implementation Details

TODO: Add implementation details, hyperparameters, and computational requirements.

C Additional Theoretical Analysis

TODO: Add extended theoretical analysis and proofs.