ACTION-MINIMIZATION MEETS GENERATIVE MODELING: EFFICIENT TRANSITION PATH SAMPLING WITH THE ONSAGER-MACHLUP FUNCTIONAL

Sanjeev Raja[*], **Martin Sipka**[*], **Michael Psenka**[*], **Tobias Kreiman**, **Michal Pavelka**, **and Aditi S. Krishnapriyan**
University of California, Berkeley, Faculty of Mathematics and Physics, Charles University

ABSTRACT

Transition path sampling (TPS), which involves finding probable paths connecting two points on an energy landscape, remains a challenge due to the complexity of real-world atomistic systems. Current machine learning approaches rely on expensive training procedures and under-utilize growing quantities of atomistic data, limiting scalability and generalization. In this work, we address TPS by interpreting candidate paths as trajectories sampled from stochastic dynamics induced by the learned score function of pretrained generative models, namely denoising diffusion and flow matching. Under these dynamics, finding high-likelihood transition paths becomes equivalent to minimizing the Onsager-Machlup (OM) action functional, enabling us to repurpose pre-trained generative models for TPS in a zero-shot fashion. We demonstrate our approach on several fast-folding proteins, where we obtain diverse, physically realistic transition pathways, as well as tetrapeptides, where we demonstrate successful TPS on systems not seen by the generative model during training. Our method can be easily incorporated into new generative models, making it practically relevant as models continue to scale and improve.

1 INTRODUCTION

Efficiently sampling the configurational distribution of high-dimensional molecular systems is a grand challenge in statistical mechanics and computational chemistry Tuckerman (2023); Frenkel & Smit (2023). A key area of interest is the sampling of rare, transition events between two stable configurations, such as chemical reactions or protein unfolding Bolhuis et al. (2002); Dellago et al. (2002). This task, broadly known as transition path sampling (TPS), is challenging due to the presence of large energy barriers between states. Machine learning (ML) based methods have gained popularity for accelerating TPS by learning a control drift term to bias a system towards a desired target state Sipka et al. (2023); Holdijk et al. (2024); Du et al. (2024); Seong et al. (2024). However, these approaches rely on highly specialized training procedures and fail to exploit the growing quantity of atomistic simulation and structural data Bank (1971); Vander Meersche et al. (2024); Lewis et al. (2024), limiting their ability to scale and generalize to new systems. We provide a complete survey of the related literature in Appendix A.

Generative models can produce unbiased, independent samples from atomistic conformational ensembles Noé et al. (2019); Zheng et al. (2024); Jing et al. (2024a) and have shown the potential to generalize across chemical space Klein & Noé (2024); Lewis et al. (2024). However, they have not been directly used for TPS due to the use of uncorrelated states during training. In this work, we propose a conceptually simple post-training method to repurpose generative models to perform TPS in a zero-shot fashion. Our core idea exploits the fact that generative models based on denoising diffusion Ho et al. (2020) and flow matching Lipman et al. (2022) induce a set of stochastic Langevin dynamics on the data manifold, governed by their learned *score* function. Drawing inspiration from statistical mechanics, the probability of paths sampled from these dynamics can be characterized by a quantity known as the Onsager-Machlup (OM) action functional Onsager & Machlup (1953). As a result, it is possible to find, from first principles, high-probability paths between arbitrary points on the data manifold by minimizing the OM action with gradient-based optimizers. In the specific case where the data are Boltzmann-distributed, the learned score approximates the underlying atomistic force field Arts et al. (2023), and our approach has the direct, physical interpretation of TPS on a potential energy surface at a finite temperature.

[*]Equal Contribution.

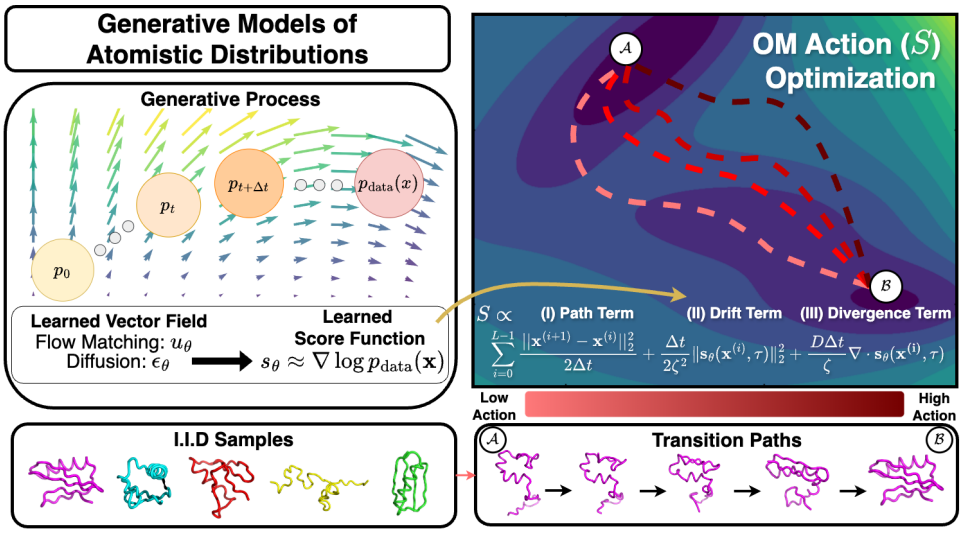

Figure 1: **Proposed Onsager-Machlup Action Optimization Schematic. (Left)** Atomistic generative models produce statistically independent samples via integration along a learned vector field, from which a score estimate, $s_\theta \approx \nabla \log p_{\text{data}}(\mathbf{x})$, can be extracted. The score can be interpreted as a drift term in the stochastic dynamics induced by the generative model. **(Right)** This connection can be leveraged to repurpose atomistic generative models to find high-probability transition paths between samples on the data manifold by minimizing the OM action functional, Eq. (5). The OM action has three terms which prioritize (I) low distance between adjacent points on the discretized path, (II) low-norm drifts, and (III) convexity of the underlying energy landscape. The variables are as follows: path point i ($\mathbf{x}^{(i)}$), trajectory length (L), timestep (Δt), friction (ζ), diffusion (D), latent time (τ). Although valid for any data distribution, in the special case of Boltzmann-distributed data, our approach has the natural interpretation of transition path sampling on a potential energy surface with an atomistic force field.

Our approach has a number of key advantages, namely

1. **Efficiency:** We do not require any complex training procedure specific to TPS, instead using pre-trained generative models off-the-shelf.
2. **Scalability:** We exploit the representations learned by generative models trained on large dataset, making approach scalable and generalizeable as models and datasets continue to grow.
3. **Flexibility:** We do not rely on system-specific collective variables (CVs), and can easily incorporate physical parameters such as temperature without retraining.

We demonstrate our method on complex, fast-folding proteins in Section 4.1 using both diffusion and flow-matching models. We find that OM action minimization yields transition path ensembles with a high degree of agreement with reference simulations, and at a fraction of the computational cost compared to unbiased MD or i.i.d. sampling from the generative models. Overall, our work points to the promise of large-scale generative modeling for enabling general-purpose, system-agnostic, and efficient transition path sampling for atomistic systems.

2 Preliminaries and Theory

We now introduce the main mathematical formalisms used in this work. First, in Section 2.1 we introduce the Onsager-Machlup action as a general way to compute path probabilities under a particular stochastic differential equation (SDE). We then show the applicability of this method to two core settings: 1) transition path sampling (Section 2.2) and 2) score-based generative modeling (Section 2.3). In the special case where the data used to train the generative models are Boltzmann-distributed, settings 1) and 2) become equivalent. This provides a unified way to find high-likelihood transition paths with generative models.

2.1 Probability of paths under stochastic dynamics

We introduce the following constant variance SDE which will underpin our proposed TPS framework:

$$\dot{\mathbf{x}} = \frac{1}{\zeta}\mathbf{\Phi}(\mathbf{x})\mathrm{d}t + \sqrt{2D}\mathrm{d}\mathbf{W}_\mathbf{t}, \tag{1}$$

where $\mathbf{\Phi}(\mathbf{x}):\mathbb{R}^k\to\mathbb{R}^k$ is a drift function in Euclidean space of dimension k, $\mathbf{W}_t$ is a standard Weiner process, and $D,\zeta>0$ are scalar constants governing diffusion noise levels and damping respectively. We consider drifts which can be written as the gradient of a scalar: $\mathbf{\Phi}(\mathbf{x})=-\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{x}}$, where $\phi(\mathbf{x}):\mathbb{R}^k\to\mathbb{R}$. By solving a Fokker-Planck equation for the time evolution of the state distribution $p(\mathbf{x},t)$, we can obtain an expression for the probability of a path $\mathbf{x}(\cdot)=\{\mathbf{x}(t)\}_{t=0}^{1}$ sampled from the SDE in Eq. (1).

$$P(\mathbf{x}(\cdot))\propto e^{(-S[\mathbf{x}(\cdot)])}. \tag{2}$$

To maximize this probability with respect to a path, we can equivalently minimize the negative log probability S, which is called the Onsager-Machlup action functional. Since we only consider discretized paths in this work, we focus on the discretized form of the OM action (see Appendix B and E for complete details and proof):

Definition 2.1. For a discrete path $\mathbf{X}=\{\mathbf{x}^{(0)},\ldots,\mathbf{x}^{(L)}\}$ generated by the SDE in Eq. (1) with drift $\mathbf{\Phi}$, the discretized form of the **Onsager-Machlup action functional** at timestep size Δt is given by:

$$\begin{aligned} S[\mathbf{X}]&=\frac{1}{2D}\left(A[\mathbf{X}]+B[\mathbf{X}]+C[\mathbf{X}]\right),\quad \text{where } A[\mathbf{X}]=\frac{1}{2\Delta t}\sum_{i=0}^{L-1}\left\|\mathbf{x}^{(i+1)}-\mathbf{x}^{(i)}\right\|_2^2,\\ B[\mathbf{X}]&=\frac{\Delta t}{2\zeta^2}\sum_{i=1}^{L-1}\left\|\mathbf{\Phi}(\mathbf{x}^{(i)})\right\|_2^2,\quad C[\mathbf{X}]=\frac{D\Delta t}{\zeta}\sum_{i=1}^{L-1}\nabla\cdot\mathbf{\Phi}(\mathbf{x}^{(i)}). \end{aligned} \tag{3}$$

2.2 Transition path sampling in molecular systems

Formally, in TPS we consider d-dimensional molecular systems with N_p-many particles interacting under a potential energy function $U(\mathbf{x}):\mathbb{R}^{N_p\times d}\to\mathbb{R}$, where $\mathbf{x}\in\Omega$ is a configuration of the system, and $\Omega\in\mathbb{R}^{N_p\times d}$ is the configuration space. The goal of TPS is to, for the given system and temperature, find most likely paths $\left\{\mathbf{x}^{(t)}\right\}_{0\le t\le L}$ over a time horizon L between two endpoints $\mathbf{x}^{(0)}\in\mathcal{A}\subset\Omega$, $\mathbf{x}^{(L)}\in\mathcal{B}\subset\Omega$. The underlying particle dynamics are governed by the SDE in Eq. (1), known in this context as overdamped Langevin dynamics. ϕ and $\mathbf{\Phi}$ have a clear physical interpretation as the potential energy and forces, respectively, i.e., $\phi(\mathbf{x}):=U(\mathbf{x})$ and $\mathbf{\Phi}(\mathbf{x}):=\mathbf{F}(\mathbf{x})=-\nabla U(\mathbf{x})$. Thus, if the forces $\mathbf{F}(\mathbf{x})$ are known, the OM action in Eq. (3) can be used to compute the probability of paths connecting endpoints $\mathbf{x}^{(0)},\mathbf{x}^{(L)}$ under the governing stochastic dynamics.

2.3 Score-based generative modeling

We now describe the connection between the SDE in Eq. (1) and generative models, namely denoising diffusion and flow matching. Specifically, we show that these models induce a set of stochastic dynamics whose drift is given by their learned *score* function. This provides a powerful framework to reason about high-probability transition paths on a broad class of data manifolds.

2.3.1 Stochastic dynamics under denoising diffusion models.

Denoising diffusion probabilistic models (DDPM) Ho et al. (2020) are a class of score-based generative models that de-noise corrupted samples. One hypothesis for constructing an SDE from DDPMs that remains in high-likelihood states is to leverage the process of iterative one-step denoising and noising at a fixed time marginal τ of the DDPM process . In the continuum limit, this process becomes an SDE:

$$d\mathbf{x}=\frac{1}{\sqrt{1-\bar{\alpha}_{\tau-1}}}\mathbf{s}_\theta(\mathbf{x},\tau)\mathrm{d}t+\sqrt{2}\mathrm{d}\mathbf{W}_\mathbf{t}, \tag{4}$$

with the following discretized OM action:

$$S_\theta(\mathbf{X})=\frac{1}{2D}\left(\sum_{i=0}^{L-1}\frac{1}{2\Delta t}\left\|\mathbf{x}^{(i+1)}-\mathbf{x}^{(i)}\right\|_2^2+\frac{\Delta t}{2\zeta^2}\left\|\mathbf{s}_\theta(\mathbf{x}^{(i)},\tau)\right\|_2^2+\frac{D\Delta t}{\zeta}\,\nabla\cdot\mathbf{s}_\theta(\mathbf{x}^{(i)},\tau)\right), \tag{5}$$

where $\zeta=\sqrt{1-\bar{\alpha}_{\tau-1}}$ and $D=1$. Please see Appendix C for further details and a complete proof.

Extension to Flow Matching. We show in Appendix D.2 that it is possible to extract a learned score s_θ from flow matching models, meaning that our OM action framework readily applies beyond DDPM.

2.3.2 PHYSICAL INTERPRETATION FOR BOLTZMANN-DISTRIBUTED DATA

By assuming underlying dynamics of the form Eq. (4), our framework combining generative models and the OM action can be used to compute the log-probabilities of paths between samples from *any* data distribution with a well-defined score. However, in the special case where the data are Boltzmann distributed, i.e., $p(\mathbf{x}) \propto \exp(-\frac{U(\mathbf{x})}{k_B T})$, where k_B is Boltzmann's constant and T is the temperature, the learned score $\mathbf{s}_\theta(\mathbf{x}, \tau = 0) : \mathbb{R}^{N_p \times d} \to \mathbb{R}^{N_p \times d}$ is interpretable as a physical, atomistic force field:

$$\mathbf{s}_\theta(\mathbf{x}, \tau = 0) \approx \nabla \log p(\mathbf{x}) \propto -\nabla U(\mathbf{x}) = \mathbf{F}(\mathbf{x}). \tag{6}$$

The constructed dynamics in Eq. (4) at $\tau = 0$ reduce to physical, overdamped Langevin dynamics governing particles on a potential energy surface. Thus, under our OM framework, finding high-probability paths between points on a Boltzmann-distributed data manifold directly aligns with the conventional notion of TPS for atomistic systems.

3 REPURPOSING GENERATIVE MODELS VIA ACTION MINIMIZATION

We now introduce our OM optimization approach to produce high probability transition paths between atomistic structures using pre-trained generative models. Given two atomistic configurations, $\mathbf{x}^{(0)}, \mathbf{x}^{(L)} \in \mathbb{R}^{N_p \times d}$ where $\mathbf{x}^{(0)} \in \mathcal{A}, \mathbf{x}^{(L)} \in \mathcal{B}$, we aim to sample a transition path $\mathbf{X} = \left\{\mathbf{x}^{(i)}\right\}_{i \in [0, L]}$ comprised of L discrete steps. Our core inductive bias is to interpret candidate paths as realizations of the denoise-noise SDE in Eq. (4), enabling tractable computation and optimization of path log-likelihoods via the OM action. Our approach proceeds in three primary steps: 1) computing an initial guess path, 2) performing OM optimization, and (in some cases) 3) decoding the optimized path back to the configurational space Ω. See Algorithm 1 for a precise description.

Computing an initial guess path. We opt to linearly interpolate at a *latent* level τ_{initial} of our pretrained generative model, which is known to produce samples close to the data manifold (Ho et al., 2020). We can then either decode the initial guess path back to the configurational space, in which case the subsequent OM optimization would occur in configurational space, or defer decoding, in which case optimization occurs at the latent level τ_{initial}. Both are justified, since the proposed dynamics in Eq. (4) are valid for any τ. See Appendix H for a complete description of the initial path guess method.

Optimization of the OM action. Starting from the initial guess, we find paths which have high probability (low action) under the SDE in Eq. (4). The optimization problem becomes,

$$\mathbf{X}^* = \underset{\mathbf{X} = \left\{\mathbf{x}^{(i)}\right\}_{i \in [0, L]}}{\operatorname{argmin}} \quad S_\theta[\mathbf{X}], \tag{7}$$

where S_θ is the generative model action in Eq. (5), and $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(L)}$ are kept fixed. We approximate this minimum via gradient descent on the path until the action is converged.

Decoding back to configurational space. If OM-optimization was performed at a nonzero latent time τ_{initial}, we decode the final path, obtained after K iterations of gradient descent, back to the configurational space. If optimization was performed in configurational space (i.e., decoding was already done in the first step), then this step is skipped.

4 RESULTS

We now present the results of our OM optimization approach for TPS with pre-trained generative models. In Appendix J, we further show that generative models trained on tetrapeptide configurations can be used for zero-shot generation of transition pathways on held-out sequences. Finally, in Appendix K, we demonstrate an additional use case beyond generative modeling, namely using a classical force field with OM optimization to generate all-atom transition paths for Chignolin and Trp-Cage.

4.1 FAST-FOLDING COARSE-GRAINED PROTEINS

We consider proteins exhibiting fast dynamical transitions, for which millisecond-scale, reference MD simulations were performed in Lindorff-Larsen et al. (2011).

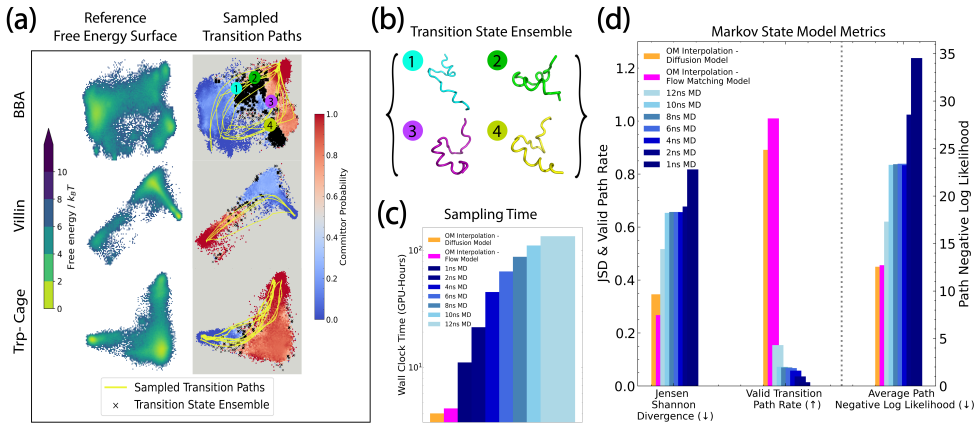

Figure 2: **OM optimization with diffusion and flow matching models trained on coarse-grained, fast-folding proteins. (a)** Reference free energy surfaces of fast-folding proteins, alongside transition paths produced by OM optimization (yellow) overlaid against the landscape of the empirically computed committor function $q(\mathbf{x})$. The transition state ensemble (black) is the set $\{\mathbf{x}: 0.45 \leq q(\mathbf{x}) \leq 0.55\}$. **(b)** Samples from the predicted transition state ensemble of BBA. **(c)** Runtime of OM optimization and varying lengths of MD simulation of BBA using the diffusion model's approximate force field. **(d)** MSM results averaged across 5 fast-folding proteins. Plotted are the Jensen-Shannon Divergence (JSD) between the sampled path and reference MSM state distributions, fraction of sampled paths which are valid under the reference MSM, and average path negative log likelihood.

Results. As shown in Fig. 2a, OM optimization yields diverse transition paths which intuitively pass through high density regions of the free energy landscape, projected onto the two slowest Time Independent Component (TIC) Noé et al. (2013) axes. For the BBA protein, the paths robustly sample the transition state ensemble (Fig. 2b), defined by the level set $\{\mathbf{x}: 0.45 \leq q(\mathbf{x}) \leq 0.55\}$ of the committor function, which is defined as the probability that a trajectory initiated at $\mathbf{x}^{(0)} = \mathbf{x}$ reaches the target state $\mathcal{B}$ before the starting state $\mathcal{A}$ (see Appendix L for details and theory). We also find that sampling transition paths of the BBA protein with OM optimization requires considerably less wall-clock time than performing an equivalent number of parallel MD simulations using the diffusion model's learned score as a force field (Fig. 2c). Across all proteins and both diffusion and flow matching models, OM optimization yields a higher percentage of valid paths and lower negative log likelihood under the Markov State Model Prinz et al. (2011); Noé et al. (2013) of the reference MD simulations (see Appendix L for details), compared with MD simulations of any of the considered lengths up to 12 ns (Fig. 2d). The Jensen-Shannon divergence (JSD) with respect to the reference MSM state distribution is also the lowest, indicating that the sampled paths traverse a similar distribution of MSM states as the reference simulations.

Robustness to sparse data in transition regions. To simulate the scenario in which transition states are not well-represented in the training data, we retrain diffusion models on datasets from which 99% of configurations with committor probability between 0.1 and 0.9 are removed. We find that our OM optimization approach is still able to sample plausible transition paths using this data-starved model (see Appendix L for more details), suggesting that our approach can be useful even if the underlying data distribution is far from an exhaustively sampled Boltzmann distribution.

5 Conclusion

Limitations. Our approach does not provably sample the full posterior distribution over paths, as in traditional shooting methods and recent ML approaches Du et al. (2024). However, we sample diverse paths by exploiting the stochastic generative model encoding and decoding process.

Future work. Incorporating OM optimization into larger generative models trained on more diverse data Lewis et al. (2024); Jing et al. (2024a) is a natural area for future development. Given the success of large-scale, co-evolutionary modeling of proteins Jumper et al. (2021), it would be interesting to investigate the extent to which pre-training generative models on large structural databases enables TPS on unseen systems. More broadly, OM action-minimization could be a powerful framework to generate interpolation paths in a variety of data modalities, including images, videos, and audio.

REFERENCES

Artur B. Adib. Stochastic actions for diffusive dynamics: Reweighting, sampling, and minimization. *The Journal of Physical Chemistry B*, 112(19):5910–5916, 05 2008. ISSN 1520-6106. doi: 10.1021/jp0751458.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. In *Proceedings of the International Conference on Machine Learning*, pp. 214–223, 2017.

Marloes Arts, Victor Garcia Satorras, Chin-Wei Huang, Daniel Zugner, Marco Federici, Cecilia Clementi, Frank Noé, Robert Pinsler, and Rianne van den Berg. Two for one: Diffusion models and force fields for coarse-grained molecular dynamics. *Journal of Chemical Theory and Computation*, 19(18):6151–6159, 2023.

Georgios Arvanitidis, L. Kai Hansen, and Søren Hauberg. Latent space oddity: On the curvature of deep generative models. In *Proceedings of the International Conference on Learning Representations*, 2018.

Protein Data Bank. Protein data bank. *Nature New Biol*, 233(223):10–1038, 1971.

Peter G Bolhuis and David WH Swenson. Transition path sampling as markov chain monte carlo of trajectories: Recent algorithms, software, applications, and future outlook. *Advanced Theory and Simulations*, 4(4):2000237, 2021.

Peter G Bolhuis, David Chandler, Christoph Dellago, and Phillip L Geissler. Transition path sampling: Throwing ropes over rough mountain passes, in the dark. *Annual review of physical chemistry*, 53 (1):291–318, 2002.

Ernesto E Borrero and C Dellago. Avoiding traps in trajectory space: Metadynamics enhanced transition path sampling. *The European Physical Journal Special Topics*, 225:1609–1620, 2016.

Gabriele Corso, Yilun Xu, Valentin De Bortoli, Regina Barzilay, and Tommi Jaakkola. Particle guidance: non-iid diverse sampling with diffusion models. *arXiv preprint arXiv:2310.13102*, 2023.

Eric Darve and Andrew Pohorille. Calculating free energies using average force. 2001. doi: 10.1063/1.1410978.

Avishek Das, Dominic C Rose, Juan P Garrahan, and David T Limmer. Reinforcement learning of rare diffusive dynamics. *The Journal of Chemical Physics*, 155(13), 2021.

Christoph Dellago, Peter G. Bolhuis, Félix S. Csajka, and David Chandler. Transition path sampling and the calculation of rate constants. *The Journal of Chemical Physics*, 108(5):1964–1977, 02 1998. ISSN 0021-9606. doi: 10.1063/1.475562.

Christoph Dellago, Peter G Bolhuis, and Phillip L Geissler. Transition path sampling. *Advances in chemical physics*, 123:1–78, 2002.

Stefan Doerr, Maciej Majewsk, Adrià Pérez, Andreas Krämer, Cecilia Clementi, Frank Noe, Toni Giorgino, and Gianni De Fabritiis. Torchmd: A deep learning framework for molecular simulations, 2020.

Yuanqi Du, Michael Plainer, Rob Brekelmans, Chenru Duan, Frank Noé, Carla P Gomes, Alán Aspuru-Guzik, and Kirill Neklyudov. Doob's lagrangian: A sample-efficient variational approach to transition path sampling. *arXiv preprint arXiv:2410.07974*, 2024.

Sebastian Falkner, Alessandro Coretti, Salvatore Romano, Phillip L Geissler, and Christoph Dellago. Conditioning boltzmann generators for rare event sampling. *Machine Learning: Science and Technology*, 4(3):035050, 2023.

Daan Frenkel and Berend Smit. *Understanding molecular simulation: from algorithms to applications*. Elsevier, 2023.

Hiroshi Fujisaki, Motoyuki Shiga, and Akinori Kidera. Onsager–Machlup action-based path sampling and its combination with replica exchange for diffusive and multiple pathways. *The Journal of Chemical Physics*, 132(13):134101, 04 2010. ISSN 0021-9606. doi: 10.1063/1.3372802.

I. M. Gel'fand and A. M. Yaglom. Integration in functional spaces and its applications in quantum physics. *Journal of Mathematical Physics*, 1(1):48–69, 01 1960. ISSN 0022-2488. doi: 10.1063/1.1703636.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Muhammad R Hasyim, Clay H Batton, and Kranthi K Mandadapu. Supervised learning and the finite-temperature string method for computing committor functions and reaction rates. *The Journal of chemical physics*, 157(18), 2022.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023. URL `https://arxiv.org/abs/1606.08415`.

Graeme Henkelman, Blas P. Uberuaga, and Hannes Jónsson. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *The Journal of Chemical Physics*, 113(22):9901–9904, 12 2000. ISSN 0021-9606. doi: 10.1063/1.1329672.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Lars Holdijk, Yuanqi Du, Ferry Hooft, Priyank Jaini, Berend Ensing, and Max Welling. Stochastic optimal control for collective variable free sampling of molecular transition paths. *Advances in Neural Information Processing Systems*, 36, 2024.

Lei Huang, Tingyang Xu, Yang Yu, Peilin Zhao, Xingjian Chen, Jing Han, Zhi Xie, Hailong Li, Wenge Zhong, Ka-Chun Wong, et al. A dual diffusion model enables 3d molecule generation and lead optimization based on target pockets. *Nature Communications*, 15(1):2657, 2024.

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

Ilia Igashov, Hannes Stärk, Clément Vignac, Arne Schneuing, Victor Garcia Satorras, Pascal Frossard, Max Welling, Michael Bronstein, and Bruno Correia. Equivariant 3d-conditional diffusion model for molecular linker design. *Nature Machine Intelligence*, pp. 1–11, 2024.

Bowen Jing, Bonnie Berger, and Tommi Jaakkola. Alphafold meets flow matching for generating protein ensembles, 2024a.

Bowen Jing, Hannes Stärk, Tommi Jaakkola, and Bonnie Berger. Generative modeling of molecular dynamics trajectories. *arXiv preprint arXiv:2409.17808*, 2024b.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Hendrik Jung, Kei-ichi Okazaki, and Gerhard Hummer. Transition path sampling of rare events by shooting from the top. *The Journal of chemical physics*, 147(15), 2017.

Julian Kappler and Ronojoy Adhikari. Stochastic action for tubes: Connecting path probabilities to measurement. *Phys. Rev. Res.*, 2:023407, Jun 2020. doi: 10.1103/PhysRevResearch.2.023407.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations*, 2013.

Leon Klein and Frank Noé. Transferable boltzmann generators. *arXiv preprint arXiv:2406.14426*, 2024.

Alessandro Laio and Michele Parrinello. Escaping free-energy minima. *Proceedings of the National Academy of Sciences of the United States of America*, 99(20):12562–12566, 10 2002.

Juyong Lee, In-Ho Lee, InSuk Joung, Jooyoung Lee, and Bernard R. Brooks. Finding multiple reaction pathways via global optimization of action. *Nature Communications*, 8(1):15443, 2017. ISSN 2041-1723. doi: 10.1038/ncomms15443.

Damian Leśniak, Igor Sieradzki, and Igor Podolak. Distribution-interpolation trade off in generative models. In *International Conference on Learning Representations*, 2018.

Sarah Lewis, Tim Hempel, José Jiménez Luna, Michael Gastegger, Yu Xie, Andrew YK Foong, Victor García Satorras, Osama Abdin, Bastiaan S Veeling, Iryna Zaporozhets, et al. Scalable emulation of protein equilibrium ensembles with generative deep learning. *bioRxiv*, pp. 2024–12, 2024.

Kresten Lindorff-Larsen, Stefano Piana, Ron O Dror, and David E Shaw. How fast-folding proteins fold. *Science*, 334(6055):517–520, 2011.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

S. Machlup and L. Onsager. Fluctuations and irreversible process. ii. systems with kinetic energy. *Phys. Rev.*, 91:1512–1515, Sep 1953. doi: 10.1103/PhysRev.91.1512.

James A. Maier, Carmenza Martinez, Koushik Kasavajhala, Lauren Wickstrom, Kevin E. Hauser, and Carlos Simmerling. ff14sb: Improving the accuracy of protein side chain and backbone parameters from ff99sb. *Journal of Chemical Theory and Computation*, 11(8):3696–3713, 2015. doi: 10.1021/acs.jctc.5b00255.

R. Mauri. *Non-Equilibrium Thermodynamics in Multiphase Flows*. Soft and Biological Matter. Springer Netherlands, 2012. ISBN 9789400754614. URL `https://books.google.cz/books?id=sD4G1azPEqcC`.

Mike Yan Michelis and Quentin Becker. On linear interpolation in the latent space of deep generative models. *arXiv preprint arXiv:2105.03663*, 2021.

Ryan Gotchy Mullen, Joan-Emma Shea, and Baron Peters. Easy transition path sampling methods: Flexible-length aimless shooting and permutation shooting. *Journal of chemical theory and computation*, 11(6):2421–2428, 2015.

Klaus Müller and Leo D Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theoretica chimica acta*, 53:75–93, 1979.

Frank Noé, Hao Wu, Jan-Hendrik Prinz, and Nuria Plattner. Projected and hidden markov models for calculating kinetics and metastable states of complex molecules. *The Journal of chemical physics*, 139(18), 2013.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

L. Onsager and S. Machlup. Fluctuations and irreversible processes. *Phys. Rev.*, 91:1505–1512, Sep 1953. doi: 10.1103/PhysRev.91.1505.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

Jan-Hendrik Prinz, Hao Wu, Marco Sarich, Bettina Keller, Martin Senne, Martin Held, John D Chodera, Christof Schütte, and Frank Noé. Markov models of molecular kinetics: Generation and validation. *The Journal of chemical physics*, 134(17), 2011.

Michael Psenka, Druv Pai, Vishal Raman, Shankar Sastry, and Yi Ma. Representation learning via manifold flattening and reconstruction. *Journal of Machine Learning Research*, 25(132):1–47, 2024.

Dominic C Rose, Jamie F Mair, and Juan P Garrahan. A reinforcement learning approach to rare trajectory sampling. *New Journal of Physics*, 23(1):013013, 2021.

Arne Schneuing, Charles Harris, Yuanqi Du, Kieran Didi, Arian Jamasb, Ilia Igashov, Weitao Du, Carla Gomes, Tom L Blundell, Pietro Lio, et al. Structure-based drug design with equivariant diffusion models. *Nature Computational Science*, 4(12):899–909, 2024.

Kiyoung Seong, Seonghyun Park, Seonghwan Kim, Woo Youn Kim, and Sungsoo Ahn. Transition path sampling with improved off-policy training of diffusion path samplers, 2024.

Aditya N Singh and David T Limmer. Variational deep learning of equilibrium transition path ensembles. *The Journal of Chemical Physics*, 159(2), 2023.

Martin Sipka, Johannes CB Dietschreit, Lukáš Grajciar, and Rafael Gómez-Bombarelli. Differentiable simulations for enhanced sampling of rare events. In *International Conference on Machine Learning*, pp. 31990–32007. PMLR, 2023.

Marcin Sobieraj and Piotr Setny. Granger causality analysis of chignolin folding. *Journal of Chemical Theory and Computation*, 18(3):1936–1944, 2022. doi: 10.1021/acs.jctc.1c00945.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Łukasz Struski, Michał Sadowski, Tomasz Danel, Jacek Tabor, and Igor T Podolak. Feature-based interpolation and geodesics in the latent spaces of generative models. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Mohammad M Sultan and Vijay S Pande. Automated design of collective variables using supervised machine learning. *The Journal of Chemical Physics*, 149(9):94106, 2018. doi: 10.1063/1.5029972.

Glenn M Torrie and John P Valleau. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *Journal of computational physics*, 23(2):187–199, 1977.

Mark E Tuckerman. *Statistical mechanics: theory and molecular simulation*. Oxford university press, 2023.

Eric Vanden-Eijnden and Matthias Heymann. The geometric minimum action method for computing minimum energy paths. *The Journal of chemical physics*, 128(6), 2008.

Eric Vanden-Eijnden et al. Transition-path theory and path-finding algorithms for the study of rare events. *Annual review of physical chemistry*, 61:391–420, 2010.

Yann Vander Meersche, Gabriel Cretin, Aria Gheeraert, Jean-Christophe Gelly, and Tatiana Galochkina. Atlas: protein flexibility description from atomistic molecular dynamics simulations. *Nucleic acids research*, 52(D1):D384–D392, 2024.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Claudio Zeni, Robert Pinsler, Daniel Zügner, Andrew Fowler, Matthew Horton, Xiang Fu, Sasha Shysheya, Jonathan Crabbé, Lixin Sun, Jake Smith, et al. Mattergen: a generative model for inorganic materials design. *arXiv preprint arXiv:2312.03687*, 2023.

Shuxin Zheng, Jiyan He, Chang Liu, Yu Shi, Ziheng Lu, Weitao Feng, Fusong Ju, Jiaxi Wang, Jianwei Zhu, Yaosen Min, et al. Predicting equilibrium distributions for molecular systems with deep learning. *Nature Machine Intelligence*, pp. 1–10, 2024.

Martin Šípka, Andreas Erlebach, and Lukáš Grajciar. Constructing collective variables using invariant learned representations. *Journal of Chemical Theory and Computation*, 19(3):887–901, 2023. ISSN 1549-9618. doi: 10.1021/acs.jctc.2c00729.

A Related Work

Transition path sampling. TPS is challenging due to the presence of large energy barriers, which create a substantial difference in timescales between rare events and the fastest dynamical motions of the system (e.g., bond vibrations). Many tools have been developed to tackle this challenge, and we refer to existing surveys for a more exhaustive description Bolhuis et al. (2002); Dellago et al. (2002); Vanden-Eijnden et al. (2010). Traditional shooting methods perturb the initial or intermediate states of a known trajectory to generate new trajectories via a Metropolis-Hasting criterion Mullen et al. (2015); Borrero & Dellago (2016); Jung et al. (2017); Bolhuis & Swenson (2021). These often suffer from high rejection rates, correlated samples, and the need for expensive molecular dynamics (MD) simulations during sampling. Another class of methods is based on adding an adjustable biasing potential to enhance the sampling of slow events, which includes umbrella sampling Torrie & Valleau (1977), metadynamics Laio & Parrinello (2002), and more advanced techniques such as eABF Darve & Pohorille (2001). These approaches require a carefully constructed, low-dimensional mapping of the problem via collective variables (CVs), which can be challenging, particularly when the characterization of the system around the transition state is uncertain. Attempts were made to design CVs with ML methods Sultan & Pande (2018); Šípka et al. (2023), yet they remain a challenge for many-atom systems. ML approaches have also been used to learn the biasing potential directly, including approaches based on stochastic optimal control Holdijk et al. (2024), differentiable simulation Sipka et al. (2023), reinforcement learning Das et al. (2021); Rose et al. (2021); Singh & Limmer (2023); Seong et al. (2024), and h-transform learning Singh & Limmer (2023); Du et al. (2024). In all of these approaches, unlimited access to the underlying potential energy and force field are assumed, but samples from the underlying data distribution are not available. As a result, the methods must be retrained from scratch for every new system of interest. Additionally, expensive, simulation-based training procedures are often employed, limited scalability to larger systems. Interpolation-based methods, such as the Nudged Elastic Band (NEB) method Henkelman et al. (2000) and the spring method Dellago et al. (1998), introduce springs between images to construct transition paths (spring method) or to directly locate saddle points (NEB). However, a significant challenge for both approaches lies in generating an initial guess, which is inherently unknown a priori. Among interpolation-like approaches, the Onsager-Machlup (OM) action has also been explored for TPS. Due to the limited availability of automatic differentiation techniques at the time, Laplacian operators were consistently avoided. This restriction limited its application to very low-dimensional problems, as in Vanden-Eijnden & Heymann (2008); Fujisaki et al. (2010), or led to the development of Laplace-free action formulations, as in Lee et al. (2017).

Atomistic generative models. Generative models can produce unbiased, independent samples from the configurational ensemble of molecular systems, pioneered by Boltzmann generators Noé et al. (2019) and since further developed for proteins Arts et al. (2023); Zheng et al. (2024); Jing et al. (2024a); Lewis et al. (2024), small molecules Huang et al. (2024); Schneuing et al. (2024); Igashov et al. (2024), and materials Zeni et al. (2023); Zheng et al. (2024). Generative models are typically trained to match the distribution of atomistic configurations from large-scale datasets, including structural databases Bank (1971) and long-timescale MD simulations Lindorff-Larsen et al. (2011); Vander Meersche et al. (2024). The latest models are typically trained with the denoising diffusion Ho et al. (2020) or flow matching Lipman et al. (2022) generative frameworks. Since the models generally reproduce the training distribution and thus cannot natively be used for studying dynamical or rare events, recent works adapt generative models to produce more diverse samples Corso et al. (2023), perform rare event sampling Falkner et al. (2023), perform MD simulations using the connection between diffusion models and force fields Arts et al. (2023), and learn generative models directly over trajectories Jing et al. (2024b).

Interpolations in generative models. Analagous to TPS, interpolation has been used to evaluate the smoothness and continuity of learned data manifolds and to generate realistic transitions between data points using generative models. While linear interpolation in model latent spaces is known to capture some continuity Kingma & Welling (2013); Goodfellow et al. (2014), geometric techniques such as geodesic interpolation and optimal transport Arjovsky et al. (2017); Arvanitidis et al. (2018); Leśniak et al. (2018); Michelis & Becker (2021); Struski et al. (2023); Psenka et al. (2024) better align with the intrinsic manifold structure of the data. Our proposed OM optimization approach can be seen as a novel interpolation mechanism which leverages the inductive bias of stochastic dynamical systems to generate high-probability transition paths.

B Interpretation and Variations on the Onsager-Machlup Action

Recall the discretized form of the OM action:

$$S[\mathbf{X}] = \frac{1}{2D}\left(A[\mathbf{X}] + B[\mathbf{X}] + C[\mathbf{X}]\right), \quad \text{where } A[\mathbf{X}] = \frac{1}{2\Delta t}\sum_{i=0}^{L-1}\left\|\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right\|_2^2,$$
$$B[\mathbf{X}] = \frac{\Delta t}{2\zeta^2}\sum_{i=1}^{L-1}\left\|\mathbf{\Phi}(\mathbf{x}^{(i)})\right\|_2^2, \quad C[\mathbf{X}] = \frac{D\Delta t}{\zeta}\sum_{i=1}^{L-1}\nabla\cdot\mathbf{\Phi}(\mathbf{x}^{(i)}). \tag{8}$$

The three summands of the OM action each have an intuitive interpretation. Term A encourages smooth transitions along the discretized path. Term B encourages paths remaining in regions with low-norm drifts, which are equilibria or saddle points of the underlying dynamics. Finally, term C encourages paths to remain in regions with low divergence of the drift, which can be interpreted as entropically favoring regions of convexity in the landscape of ϕ, where dynamics are more stable. The parameters $\zeta, \Delta t$, and D control the relative contribution of these three terms in a physically intuitive manner. For instance, at larger values of Δt, the contribution of term A is diminished, consistent with the intuition that larger "jumps" are more probable with larger timesteps. In the limiting case of negligible diffusivity D (analogous to temperature, see Appendix E), the divergence term can be omitted, yielding the Truncated OM Action:

Definition B.1. The **truncated OM action** of a discretized path $\mathbf{X}$ is given by:

$$S[\mathbf{X}] = \frac{1}{2D}\big(A[\mathbf{X}] + B[\mathbf{X}]\big). \tag{9}$$

In this work, we use both the truncated and full OM actions depending on the system considered.

C Onsager-Machlup Action for Iterated Noising/Denoising From a DDPM

We now show precisely how to define a set of stochastic dynamics induced by DDPM generative models, as introduced in Section 2.3.1. In order to use the OM action to compute path probabilities with a DDPM, we must construct a surrogate SDE in the form of Eq. (1), such that paths under this SDE have high likelihood under the data distribution used to train the model. The denoising (i.e., sampling) process of a DDPM may appear to be a natural candidate:

$$\mathbf{x}^{(T_d)} \sim \mathcal{N}(0, I), \tag{10}$$
$$\mathbf{x}^{(i-1)} = \frac{1}{\sqrt{1-\beta_\tau}}\left(\mathbf{x}^{(i)} + \frac{\beta_\tau}{\sqrt{1-\bar{\alpha}_\tau}}\mathbf{s}_\theta(\mathbf{x}^{(i)}, i)\right) + \sqrt{\beta_\tau}z. \tag{11}$$

While this process is definable as an Euler-Maruyama discretization of an SDE, it is not well suited for optimization over trajectories over the data distribution, since the vector field $\mathbf{s}_\theta(\mathbf{x}^{(i)}, i)$ is changing throughout the trajectory, and iterates near the noise $i = T_d$ will not necessarily follow dynamics determined by the data distribution. A large portion of the denoising trajectory thus has low likelihood under the data distribution. Therefore, we need to consider an alternative approach.

We consider the following iterated **denoise-noise** updates, writing $\mathbf{s}_\theta(\mathbf{x}, \tau) = -\epsilon_\theta(\mathbf{x}, \tau)$, where $\epsilon_\theta(\mathbf{x}, \tau)$ is the trained denoising model from the DDPM:

$$\mathbf{x}^{(i,\text{mid})} = \frac{1}{\sqrt{1-\beta_\tau}}\left(\mathbf{x}^{(i)} + \frac{\beta_\tau}{\sqrt{1-\bar{\alpha}_\tau}}\mathbf{s}_\theta(\mathbf{x}^{(i)}, \tau)\right) + \sqrt{\beta_\tau}z, \tag{12}$$
$$\mathbf{x}^{(i+1)} = \sqrt{1-\beta_\tau}\mathbf{x}^{(i,\text{mid})} + \sqrt{\beta_\tau}z', \tag{13}$$

where $z, z' \sim \mathcal{N}(0, I)$, α, β denote the usual diffusion model noise schedule variables, and $\bar{\alpha}_\tau = \prod_{i=1}^{\tau}\alpha_i$.

Combining the two updates yields a single update equivalent in distribution:

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \frac{\beta_\tau}{\sqrt{1-\bar{\alpha}_\tau}}\mathbf{s}_\theta(\mathbf{x}^{(i)}, \tau) + \sqrt{2\beta_\tau - \beta_\tau^2}z, \tag{14}$$

where $z \sim \mathcal{N}(0,I)$. Taking the continuum limit $\beta_\tau \to 0$, and noting that $2\beta_\tau - \beta_\tau^2 \approx 2\beta_\tau$ and $\bar{\alpha}_\tau \to \bar{\alpha}_{\tau-1}$ at this limit, we see that Eq. (14) is an Euler-Maruyama discretization, with timestep β_τ, of the following SDE:

$$d\mathbf{x} = \frac{1}{\sqrt{1-\bar{\alpha}_{\tau-1}}}\mathbf{s}_\theta(\mathbf{x},\tau)\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{W}_\mathbf{t}. \tag{15}$$

Note that the above construction holds for any $\tau \in \{1,...,T_d\}$ where T_d is the maximum diffusion time. A similar derivation can be found in Arts et al. (2023) for $\tau = 0$.

Eq. (15) is the same equation as Eq. (1) with the equivalence (up to constants),

$$\phi(\mathbf{x}) \propto -\log p_\tau(\mathbf{x}), \tag{16}$$

$$\mathbf{\Phi}(\mathbf{x}) \propto \mathbf{s}_\theta(\mathbf{x},\tau) \approx \nabla \log p_\tau(\mathbf{x}). \tag{17}$$

This means that, as introduced in Eq. (2) and Eq. (3), the likelihood of discrete paths $\mathbf{X} = \left(\mathbf{x}^{(t)}\right)_{0 \le t \le L}$ under the constructed SDE Eq. (15) can be evaluated via the OM action defined by the DDPM score $\mathbf{s}_\theta(\mathbf{x},\tau)$:

$$S_\theta(\mathbf{X}) = \frac{1}{2D}\left(\sum_{i=0}^{L-1}\frac{1}{2\Delta t}\left\|\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right\|_2^2 + \frac{\Delta t}{2\zeta^2}\left\|\mathbf{s}_\theta(\mathbf{x}^{(i)},\tau)\right\|_2^2 + \frac{D\Delta t}{\zeta}\nabla\cdot\mathbf{s}_\theta(\mathbf{x}^{(i)},\tau)\right), \tag{18}$$

where $\zeta = \sqrt{1-\bar{\alpha}_{\tau-1}}$ and $D = 1$. While we define ζ here based off of the DDPM variance parameters α_τ, in practice Δt, ζ, and D can be treated as hyperparameters to balance the relative contributions of the action's summands.

D Proofs for Score-Related Generative Model Objectives

For the sake of readability, we replicate proofs showing that both the training objectives for DDPM and flow matching models are equivalent to training against the score of a noised version of the data distribution (or in the case of flow matching, an invertible polynomial transformation of this score). See Vincent (2011) and Lipman et al. (2024) for example proofs for DDPM and flow matching respectively.

D.1 Relation between DDPM and the score

The DDPM objective Eq. (21) is closely linked to the score-matching objective Vincent (2011); Song & Ermon (2019) for training a score model $s_\theta(\mathbf{x},\tau) : \mathbb{R}^k \times \mathbb{R}^+ \to \mathbb{R}^k$ parametrized by θ:

$$\mathcal{L}_{\text{SM}}(\theta) = \mathbb{E}_{\tau,\mathbf{x}\sim p_\tau}\left[\|s_\theta(\mathbf{x},\tau) - \nabla\log(p_\tau(\mathbf{x}))\|_2^2\right], \tag{19}$$

where $\nabla \log p_\tau(\mathbf{x})$ is the score of the noised marginal distribution p_τ at time τ. This establishes the connection between the score and the optimal noise model: $\epsilon_{\theta^*}(\mathbf{x},\tau) \propto -\nabla \log p_\tau(\mathbf{x})$.

Below is a proof for the equivalence of standard DDPM training to score matching.

Theorem D.1 (DDPM-Score Matching Equivalence)**.** *Let $p_{data}(x_0)$ be the data distribution, and let x_τ be the noised variable defined through the forward process:*

$$x_\tau = \sqrt{\bar{\alpha}_\tau}x_0 + \sqrt{1-\bar{\alpha}_\tau}\epsilon, \quad \tau \sim \mathrm{Unif}(\{1,...,T_d\}), \quad x_0 \sim p_{data}, \quad \epsilon \sim \mathcal{N}(0,I), \tag{20}$$

where $\bar{\alpha}_\tau \in (0,1)$. Let $p_\tau(x_\tau) = \int p_{data}(x_0)\mathcal{N}(x_\tau; \sqrt{\bar{\alpha}_\tau}x_0, (1-\bar{\alpha}_\tau)I)dx_0$ be the marginal distribution of x_τ. Then the DDPM objective, defined as the following:

$$\mathcal{L}_{DDPM}(\theta) \overset{\text{def}}{=} \mathbb{E}_{\tau,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_\tau,\tau)\|_2^2\right], \tag{21}$$

satisfies the following equality:

$$\mathcal{L}_{DDPM}(\theta) = \mathbb{E}_{\tau,x_\tau}\left[(1-\bar{\alpha}_\tau)\|\nabla_{x_\tau}\log p_\tau(x_\tau) - s_\theta(x_\tau,\tau)\|_2^2\right] + C, \tag{22}$$

where $s_\theta(x_\tau,\tau) \overset{\text{def}}{=} -\epsilon_\theta(x_\tau,\tau)/\sqrt{1-\bar{\alpha}_\tau}$, and C is a constant independent of θ.

Proof. We begin with the DDPM training objective:

$$\mathcal{L}_{\text{DDPM}}(\theta) = \mathbb{E}_{\tau,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_\tau,\tau)\|_2^2\right]. \tag{23}$$

The core of the desired result is Tweedie's formula, which relates Gaussian-based denoising to the score of the noised distribution. For any random variable z generated as $z = \mu + \sigma\eta$ where μ is an arbitrary random vector, $\eta \sim \mathcal{N}(0,I)$, and $\sigma > 0$, Tweedie's formula gives the following posterior expectation:

$$\mathbb{E}[\mu|z] = z + \sigma^2 \nabla_z \log p(z), \tag{24}$$

where $p(z) = \int \mathcal{N}(z;\mu,\sigma^2 I)p(\mu)d\mu$ is the full marginal distribution of z. In the forward process, x_τ is generated via $x_\tau = \sqrt{\bar{\alpha}_\tau}x_0 + \sqrt{1-\bar{\alpha}_\tau}\epsilon$, which corresponds to:

$$\mu = \sqrt{\bar{\alpha}_\tau}x_0, \quad \sigma = \sqrt{1-\bar{\alpha}_\tau}, \quad z = x_\tau, \quad \eta = \epsilon. \tag{25}$$

Here, μ is a random variable (dependent on x_0), not a fixed parameter. Applying Tweedie's formula to the marginal distribution $p_\tau(x_\tau)$, we obtain:

$$\mathbb{E}\left[\sqrt{\bar{\alpha}_\tau}x_0 \middle| x_\tau\right] = x_\tau + (1-\bar{\alpha}_\tau)\nabla_{x_\tau}\log p_\tau(x_\tau). \tag{26}$$

Dividing through by $\sqrt{\bar{\alpha}_\tau}$ gets the posterior of the original sample x_0:

$$\mathbb{E}[x_0|x_\tau] = \frac{x_\tau}{\sqrt{\bar{\alpha}_\tau}} + \frac{1-\bar{\alpha}_\tau}{\sqrt{\bar{\alpha}_\tau}}\nabla_{x_\tau}\log p_\tau(x_\tau). \tag{27}$$

From the forward process definition, we rewrite in terms of ϵ:

$$\epsilon = \frac{x_\tau - \sqrt{\bar{\alpha}_\tau}x_0}{\sqrt{1-\bar{\alpha}_\tau}}, \tag{28}$$

and take conditional expectations given x_τ to get the following:

$$\mathbb{E}[\epsilon|x_\tau] = \frac{x_\tau - \sqrt{\bar{\alpha}_\tau}\mathbb{E}[x_0|x_\tau]}{\sqrt{1-\bar{\alpha}_\tau}}, \tag{29}$$

$$= \frac{x_\tau - \sqrt{\bar{\alpha}_\tau}\left(\frac{x_\tau}{\sqrt{\bar{\alpha}_\tau}} + \frac{1-\bar{\alpha}_\tau}{\sqrt{\bar{\alpha}_\tau}}\nabla_{x_\tau}\log p_\tau(x_\tau)\right)}{\sqrt{1-\bar{\alpha}_\tau}}, \tag{30}$$

$$= \frac{x_\tau - x_\tau - (1-\bar{\alpha}_\tau)\nabla_{x_\tau}\log p_\tau(x_\tau)}{\sqrt{1-\bar{\alpha}_\tau}}, \tag{31}$$

$$= -\sqrt{1-\bar{\alpha}_\tau}\nabla_{x_\tau}\log p_\tau(x_\tau). \tag{32}$$

Using the law of total expectation, we can expand the DDPM loss conditioned on x_τ,τ:

$$\mathcal{L}_{\text{DDPM}}(\theta) = \mathbb{E}_{\tau,x_\tau}\left[\mathbb{E}_\epsilon\left[\|\epsilon - \epsilon_\theta(x_\tau,\tau)\|_2^2 \,|\, x_\tau,\tau\right]\right]. \tag{33}$$

For any random vector ξ, $\mathbb{E}[\|\xi - c\|^2]$ is minimized when $c = \mathbb{E}[\xi]$. We can then make the following bias-variance decomposition:

$$\mathbb{E}_\epsilon\left[\|\epsilon - \epsilon_\theta(x_\tau,\tau)\|_2^2 \,|\, x_\tau,\tau\right] = \|\mathbb{E}[\epsilon \,|\, x_\tau,\tau] - \epsilon_\theta(x_\tau,\tau)\|_2^2 + \mathbb{E}\left[\|\epsilon - \mathbb{E}[\epsilon \,|\, x_\tau,\tau]\|_2^2 \,|\, x_\tau,\tau\right]. \tag{34}$$

Since the variance term is independent of θ, substituting $\mathbb{E}[\epsilon \,|\, x_\tau,\tau]$ and factoring out $-\sqrt{1-\bar{\alpha}_\tau}$ leads to:

$$\mathcal{L}_{\text{DDPM}}(\theta) = \mathbb{E}_{\tau,x_\tau}\left[(1-\bar{\alpha}_\tau)\left\|\nabla_{x_\tau}\log p_\tau(x_\tau) - \left(-\frac{\epsilon_\theta(x_\tau,\tau)}{\sqrt{1-\bar{\alpha}_\tau}}\right)\right\|_2^2\right] + C. \tag{35}$$

By defining $s_\theta(x_\tau,\tau) \coloneqq -\epsilon_\theta(x_\tau,\tau)/\sqrt{1-\bar{\alpha}_\tau}$, we obtain the score matching objective. □

D.2 Relation between Flow matching and score matching

We link our OM action-minimization framework introduced in Section 2.3 with a broader class of generative models beyond DDPM. Flow matching models Lipman et al. (2022) are a natural choice due to their strong performance in generative modeling tasks across modalities Jing et al. (2024a); Polyak

et al. (2024). Similarly to DDPM, flow matching generates samples through a repeated integration process over a learned vector field. For affine flows considered in this work, the training objective for the learned velocity field $u_\theta(x,\tau)$ takes the form,

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{\tau,\mathbf{x}\sim p_\tau}\left[\|u_\theta(\mathbf{x},\tau) - u_\tau(\mathbf{x})\|_2^2\right], \tag{36}$$

$$u_\tau(\mathbf{x}) = \mathop{\mathbb{E}}_{\mathbf{x}_0\sim p_0,\mathbf{x}_1\sim p_1}[\dot{\alpha_\tau}\mathbf{x}_1 + \dot{\sigma_\tau}\mathbf{x}_0 \,|\, \mathbf{x} = \alpha_\tau\mathbf{x}_1 + \sigma_\tau\mathbf{x}_0], \tag{37}$$

where p_0 and p_1 are the source and target distributions, $\alpha_\tau,\sigma_\tau : [0,1] \to [0,1]$ define a curve from $\mathbf{x}_0$ to $\mathbf{x}_1$: $\alpha_0 = \sigma_1 = 0$, $\alpha_1 = \sigma_0 = 1$, and $\alpha_\tau, -\sigma_\tau$ are both strictly increasing functions.

While extracting the learned score $\mathbf{s}_\theta$ from DDPM is straightforward via $\mathbf{s}_\theta(\mathbf{x},\tau) = -\epsilon_\theta(\mathbf{x},\tau)$, it is less clear how to do so for flow matching. However, we note the following:

1. By Eq. (19), the targets for the denoising model $\epsilon_\theta(\mathbf{x},\tau)$ in DDPM are equivalently the negative scores of the noised distribution, $-\nabla \log p_\tau(\mathbf{x})$.
2. The targets $u_t(\mathbf{x})$ for the flow matching model $u_\theta(\mathbf{x},\tau)$ can be converted to scores of the marginal distribution $\nabla \log p_\tau(\mathbf{x})$ through the formula:

$$\nabla \log p_\tau(\mathbf{x}) = \frac{\alpha_\tau}{\dot{\sigma}_\tau\sigma_\tau\alpha_\tau - \dot{\alpha}_\tau\sigma_\tau^2}\left(\frac{\dot{\alpha}_\tau}{\alpha_\tau}\mathbf{x} - u_\tau(\mathbf{x})\right). \tag{38}$$

We can thus extract an approximate score $\mathbf{s}_\theta^{\text{FM}}(\mathbf{x},\tau)$ from a trained flow matching model $u_\theta(\mathbf{x},\tau)$ via,

$$\mathbf{s}_\theta^{\text{FM}}(\mathbf{x},\tau) = \frac{\alpha_\tau}{\dot{\sigma}_\tau\sigma_\tau\alpha_\tau - \dot{\alpha}_\tau\sigma_\tau^2}\left(\frac{\dot{\alpha}_\tau}{\alpha_\tau}\mathbf{x} - u_\theta(\mathbf{x},\tau)\right). \tag{39}$$

By inserting $\mathbf{s}_\theta^{\text{FM}}(\mathbf{x},\tau)$ into the denoise-noise process Eq. (15), we again obtain an SDE of the form of Eq. (1). Hence, we can use the OM action to compute the log-probabilities of paths between arbitrary datapoints.

We now provide proof for this flow matching setting, showing that the training objective is also similar to a score matching objective, with a simple transformation between the flow matching targets and the scores.

Theorem D.2 (Flow Matching – Score Matching Conversion)**.** *Let $p_{data}(x_0)$ be the data distribution, and let x_τ be the noised variable defined through the interpolation process:*

$$x_\tau = \alpha_\tau x_1 + \sigma_\tau x_0, \quad \tau \sim \text{Unif}([0,1]), \quad x_1 \sim p_{data}, \quad x_0 \sim \mathcal{N}(0,I), \tag{40}$$

where $\alpha_\tau,\sigma_\tau : [0,1] \to [0,1]$ are strictly increasing and decreasing functions respectively that satisfy $\alpha_0 = \sigma_1 = 0$, $\alpha_1 = \sigma_0 = 1$. Let $p_\tau(x_\tau) = \int p_{data}(x_0)\mathcal{N}(x_\tau; \alpha_\tau x_1, \sigma_\tau^2 I)dx_0$ be the marginal distribution of x_τ. Then the flow matching objective, defined as the following:

$$\mathcal{L}_{FM}(\theta) \overset{\text{def}}{=} \mathbb{E}_{\tau,x_0,x_1}\left[\|u_\theta(x_\tau,\tau) - v_\tau(x_0,x_1)\|_2^2\right], \tag{41}$$

where $v_\tau(x_0,x_1) = \dot{\alpha}_\tau x_1 + \dot{\sigma}_\tau x_0$ the instance-wise curve velocity. The flow matching objective then satisfies the following equalities:

1. *We can equivalently train against targets of the unconditional velocities $u_\tau(\mathbf{x}) = \mathbb{E}_{\mathbf{x}_0\sim p_0,\mathbf{x}_1\sim p_1}[\dot{\alpha}_t\mathbf{x}_1 + \dot{\sigma}_t\mathbf{x}_0 \,|\, \mathbf{x} = \alpha_t\mathbf{x}_1 + \sigma_t\mathbf{x}_0]$:*

$$\mathcal{L}_{FM}(\theta) = \mathbb{E}_{\tau,x_0,x_1}\left[\|u_\theta(x_\tau,\tau) - u_\tau(x_\tau)\|_2^2\right] + C, \tag{42}$$

   where C is some constant independent of θ, and

2. *The equality $\nabla_x \log p_\tau(x) = \frac{\dot{\alpha}_\tau}{\alpha_\tau}x - \frac{\dot{\sigma}_\tau\sigma_\tau\alpha_\tau - \dot{\alpha}_\tau\sigma_\tau^2}{\alpha_\tau}u_\tau(x)$ holds, allowing us to write the flow matching objective in terms of the score:*

$$\mathcal{L}_{FM}(\theta) = \mathbb{E}_{\tau,x_0,x_1}\left[\|u_\theta(x_\tau,\tau) - \left(\frac{\dot{\alpha}_\tau}{\alpha_\tau}x_\tau - \frac{\dot{\sigma}_\tau\sigma_\tau\alpha_\tau - \dot{\alpha}_\tau\sigma_\tau^2}{\alpha_\tau}\nabla_{x_\tau}\log p_\tau(x_\tau)\right)\|_2^2\right] + C. \tag{43}$$

Proof. For part 1, we can expand the flow matching loss integrand by telescoping with respect to $u_\tau(x_\tau)$:

$$\|u_\theta(x_\tau,\tau)-v_\tau(x_0,x_1)\|^2=\|u_\theta(x_\tau,\tau)-u_\tau(x_\tau)+u_\tau(x_\tau)-v_\tau(x_0,x_1)\|^2, \tag{44}$$

$$=\|u_\theta(x_\tau,\tau)-u_\tau(x_\tau)\|^2+2\langle u_\theta(x_\tau,\tau)-u_\tau(x_\tau),u_\tau(x_\tau)-v_\tau(x_0,x_1)\rangle+\|u_\tau(x_\tau)-v_\tau(x_0,x_1)\|^2. \tag{45}$$

Since $\mathbb{E}_{\tau,x_0,x_1}\|u_\tau(x_\tau)-v_\tau(x_0,x_1)\|^2$ is constant with respect to θ, it suffices to show the following:

$$\mathbb{E}_{\tau,x_0,x_1}\langle u_\theta(x_\tau,\tau)-u_\tau(x_\tau),u_\tau(x_\tau)-v_\tau(x_0,x_1)\rangle=0. \tag{46}$$

Note that by definition we have the following relation between u and v:

$$\mathop{\mathbb{E}}_{x_0,x_1}[v_\tau(x_0,x_1)\,|\,x_\tau,\tau]=\mathop{\mathbb{E}}_{x_0,x_1}[\dot\alpha_\tau x_1+\dot\sigma_\tau x_0\,|\,x_\tau,\tau]=u_\tau(x_\tau). \tag{47}$$

We can then write the following by expanding the expectation using the tower rule:

$$\mathbb{E}_{\tau,x_0,x_1}\langle u_\theta(x_\tau,\tau)-u_\tau(x_\tau),u_\tau(x_\tau)-v_\tau(x_0,x_1)\rangle=\mathbb{E}_{x_\tau,\tau}\left[\mathop{\mathbb{E}}_{x_0,x_1}[\langle u_\theta(x_\tau,\tau)-u_\tau(x_\tau),u_\tau(x_\tau)-v_\tau(x_0,x_1)\rangle\,|\,x_\tau,\tau]\right], \tag{48}$$

$$=\mathop{\mathbb{E}}_{x_\tau,\tau}[\langle u_\theta(x_\tau,\tau)-u_\tau(x_\tau),u_\tau(x_\tau)-\mathbb{E}_{x_0,x_1}[v_\tau(x_0,x_1)\,|\,x_\tau,\tau]\rangle], \tag{49}$$

$$=\mathop{\mathbb{E}}_{x_\tau,\tau}[\langle u_\theta(x_\tau,\tau)-u_\tau(x_\tau),\mathbf{0}\rangle], \tag{50}$$

$$=0. \tag{51}$$

This concludes part 1. Note that the interpolations $x_\tau=\alpha_\tau x_1+\sigma_\tau x_0$ also follow proper form for Tweedie's formula, allowing us to write the following:

$$\mathbb{E}[\alpha_\tau x_1|x_\tau,\tau]=x_\tau+\sigma_\tau^2\nabla_x\log p_\tau(x_\tau), \tag{52}$$

$$\mathbb{E}[x_1|x_\tau,\tau]=\frac{1}{\alpha_\tau}x_\tau+\frac{\sigma_\tau^2}{\alpha_\tau}\nabla_x\log p_\tau(x_\tau). \tag{53}$$

Noting that $x_0=\frac{x_\tau-\alpha_\tau x_1}{\sigma_\tau}$, we can write u_τ as the following:

$$u_\tau(x)=\mathop{\mathbb{E}}_{x_0,x_1}[\dot\alpha_t x_1+\dot\sigma_t x_0\,|\,x_\tau=x\tau], \tag{54}$$

$$=\dot\alpha_t\mathop{\mathbb{E}}_{x_0,x_1}[x_1\,|\,x_\tau=x,\tau]+\dot\sigma_t\mathop{\mathbb{E}}_{x_0,x_1}[x_0\,|\,x_\tau=x,\tau], \tag{55}$$

$$=\frac{\dot\sigma_\tau}{\sigma_\tau}x+\left(\dot\alpha_\tau-\frac{\alpha_\tau\dot\sigma_\tau}{\sigma_\tau}\right)\mathop{\mathbb{E}}_{x_0,x_1}[x_1\,|\,x_\tau=x,\tau], \tag{56}$$

$$=\frac{\dot\sigma_\tau}{\sigma_\tau}x+\left(\dot\alpha_\tau-\frac{\alpha_\tau\dot\sigma_\tau}{\sigma_\tau}\right)\left(\frac{1}{\alpha_\tau}x+\frac{\sigma_\tau^2}{\alpha_\tau}\nabla_x\log p_\tau(x)\right), \tag{57}$$

$$=\frac{\dot\alpha_\tau}{\alpha_\tau}x-\frac{\dot\sigma_\tau\sigma_\tau\alpha_\tau-\dot\alpha_\tau\sigma_\tau^2}{\alpha_\tau}\nabla_x\log p_\tau(x). \tag{58}$$

This proves the desired relation between u_τ and $\nabla\log p_\tau$, and plugging into part 1 achieves the desired flow matching loss equality.

□

E Derivation of the action

E.1 Overdamped Langevin dynamics

We shall start the description of our system by formulating well-known Hamilton equations. The variables we are solving are $\mathbf{x}_i(t):\mathbb{R}^+\to\mathbb{R}^d$ and the corresponding momenta $\mathbf{p}_i(t):\mathbb{R}^+\to\mathbb{R}^d$ with a constant vector m_i representing the mass of every particle i in the system. Hamiltonian equations are formulated as follows

$$\begin{aligned}\dot{\mathbf{x}}_i(t)&=\frac{\mathbf{p}_i(t)}{m_i},\\ \dot{\mathbf{p}}_i(t)&=-\frac{\partial U(\mathbf{x}(t))}{\partial\mathbf{x}_i}.\end{aligned} \tag{59}$$

While these equations maintain energy and contain no representation of temperature, a modified SDE, with the term $\mathbf{W}(t)$ representing a Wiener procees and a damping constant γ

$$\begin{aligned}\dot{\mathbf{x}}_i(t) &= \frac{\mathbf{p}_i(t)}{m_i},\\ \dot{\mathbf{p}}_i(t) &= -\frac{\partial U(\mathbf{x}(t))}{\partial \mathbf{x}_i} - \gamma \frac{\mathbf{p}_i(t)}{m_i} + \sqrt{2\gamma k_B T}\mathbf{W}(t),\end{aligned} \tag{60}$$

or equivalently in one second order equation:

$$m_i \ddot{\mathbf{x}}_i(t) = -\frac{\partial U(\mathbf{x}(t))}{\partial \mathbf{x}_i} - \gamma m_i \dot{\mathbf{x}}_i + \sqrt{2m_i\gamma k_B T}\mathbf{W}(t), \tag{61}$$

can now represent a system that experiences thermal fluctuation. While the original Hamiltonian system is trapped in an energy well forever, the one guided by Langevin dynamics may overcome barriers between wells in finite time.

A questions then arises. Of all the possible paths of fixed physical parameters and time that connect two minima, which is the most probable? How do we calculate probabilities and penalize high energy regions or path that are making too large steps? The answer is provided by Onsager and Machlup in their works Onsager & Machlup (1953); Machlup & Onsager (1953). The second reference handles the full equation Eq. (60) while the first one is a reduction to a so-called overdamped state where the term $\ddot{\mathbf{x}}(t)$ can be neglected. After introduction of two auxiliary vector quantities $\zeta_i = m_i\gamma$ and $D_i = \frac{k_b T}{\zeta_i}$ we get the form

$$\dot{\mathbf{x}}_i = -\frac{1}{\zeta_i}\frac{\partial U(\mathbf{x}(t))}{\partial \mathbf{x}_i} + \sqrt{2D_i}\mathbf{W}(t). \tag{62}$$

or equally just with $\mathbf{F}(\mathbf{x}(t)) = -\frac{\partial U(\mathbf{x}(t))}{\partial \mathbf{x}_i}$

$$\dot{\mathbf{x}}_i = \frac{1}{\zeta_i}\mathbf{F}(\mathbf{x}(t)) + \sqrt{2D_i}\mathbf{W}(t). \tag{63}$$

Further we will follow a more general setting of the langevin equation consistent with Eq. (1). To recall:

$$\dot{\mathbf{x}} = \frac{1}{\zeta}\mathbf{\Phi}(\mathbf{x})\mathrm{d}t + \sqrt{2D}\mathrm{d}\mathbf{W}, \tag{64}$$

and

$$\phi(\mathbf{x}) := U(\mathbf{x}) \tag{65}$$

$$\mathbf{\Phi}(\mathbf{x}) := \mathbf{F}(\mathbf{x}) = -\nabla U(\mathbf{x}) \tag{66}$$

E.2 Most probable path under Langevin dynamics

Considering a single particle (for more particle systems see e.g. Kappler & Adhikari (2020)), since Eq. (1) is a stochastic differential equation, we can also write a partial differential equation for the probability density of the particle guided by this equations. In this case it is a well-known Fokker-Planck equation (note $\frac{\partial}{\partial \mathbf{x}}$ of a vector will be understood as a divergence operator to save space)

$$\frac{\partial P}{\partial t} = -\frac{\partial\left(\frac{\mathbf{\Phi}}{\zeta}P\right)}{\partial \mathbf{x}} + \frac{\partial}{\partial \mathbf{x}}\left(D\frac{\partial P}{\partial \mathbf{x}}\right). \tag{67}$$

As we will consider only potential forces in this work, let us denote $\mathbf{\Phi}(\mathbf{x}) = -\frac{1}{\zeta}\frac{\partial \phi(\mathbf{x})}{\partial \mathbf{x}}$. Now we will split the derivation into two parts.

1. $\mathbf{\Phi} = 0$:

Let us consider the solution in the following form:

$$P(\mathbf{x}, t \,|\, \mathbf{x}_0) = (4\pi D t)^{-\frac{3}{2}} e^{\frac{-(\mathbf{x}-\mathbf{x}_0)^2}{4Dt}}. \tag{68}$$

Then for the sequence of points in space and time $(\mathbf{x}^1,t^1),(\mathbf{x}^2,t^2),...(\mathbf{x}^N,t^N)$ we can write the following probability:

$$P(\mathbf{x}^1,t^1\,|\,\mathbf{x}^2,t^2\,|\cdots|\,\mathbf{x}^N,t^N)=\prod_{j=1}^{N+1}P(\mathbf{x}^j,t^j-t^{j-1}\,|\,\mathbf{x}_0). \tag{69}$$

Let us denote $t^j-t^{j-1}=\epsilon$ as we will be passing into a continuum limit in time. The probability can be rewritten by plugging in a solution Eq. (68) into

$$\prod_{i=1}^{N+1}P(\mathbf{x}^j,t^j-t^{j-1}\,|\,\mathbf{x}_0)=(4\pi Dt)^{-\frac{3}{2}(N+1)}\exp\left(-\frac{1}{4D\epsilon}\sum_{j=1}^{N+1}(\mathbf{x}_j-\mathbf{x}_{j-1})^2\right). \tag{70}$$

To make sure we can pass into the limit let us rewrite

$$\epsilon^{-\frac{3}{2}(N+1)}=e^{\frac{3}{2}\ln\epsilon}. \tag{71}$$

Important will be the part in the argument of the $\exp$ function. We can modify it to the form

$$\frac{1}{4D}\sum_{j=1}^{N+1}\left(\frac{\mathbf{x}-\mathbf{x}_0}{\epsilon}\right)^2\epsilon. \tag{72}$$

By passing into the limit $N\to\infty$ and realizing epsilon can be rewritten by its definition to $\epsilon=\frac{t}{N}$ we get the definition of an integral

$$\frac{1}{4D}\int_0^t(\dot{\mathbf{x}})^2dt. \tag{73}$$

Regarding the first term, while it may appear that

$$\lim_{N\to\infty}\left(4\pi D\frac{t}{N}\right)^{-\frac{3}{2}(N+1)}=\infty, \tag{74}$$

evaluation of this limit directly would be too hasty. One must consider the probability derived in the broader context of integration across the path. In that case, the constant will serve to normalize the probability. The fact that it does not depend on $\mathbf{x}$ also mean the probability of the path does not, relatively with respect to other paths, depend on this prefactor and only the exponential part is important. To find out more about precise mathematical justification we refer the reader to Gel'fand & Yaglom (1960). We shall denote the constant before exponential as C from now on because, as it is not dependent on $\mathbf{x}$ it will not influence our calculations. The final probability of a path is then given as follows:

$$P(\mathbf{x},t)=\int_{\mathbf{x}_0,t}^{\mathbf{x},t}\left[C\exp\left(-\frac{1}{4D}\int_0^t(\dot{\mathbf{x}}(s))^2ds\right)\right]d\mathbf{x}dt. \tag{75}$$

To maximize the likelihood of the path we clearly need to minimize the action

$$S_0(\mathbf{x}(t))=\frac{1}{4D}\int_0^t(\dot{\mathbf{x}}(s))^2ds. \tag{76}$$

Intuitively, the most probable path under no drift is the one that does not move from it's origin. The longer the trajectory, the least probable it is.

1. $\mathbf{\Phi}\neq 0$:

We will recall the assumption $\mathbf{\Phi}=-\frac{1}{\zeta}\nabla\phi(\mathbf{x})$ and shall use a transformation

$$P(\mathbf{x},t\,|\,\mathbf{x}_0)=G(\mathbf{x},t,|\,\mathbf{x}_0)\exp\left(\frac{1}{2D\zeta}\int_{\mathbf{x}(0)}^{\mathbf{x}(t)}\mathbf{\Phi}(\mathbf{s})d\mathbf{s}\right), \tag{77}$$

where from the properties of a potential function we can evaluate the integral to

$$-\overline{\phi}(\mathbf{x})\overset{\text{def}}{=}\frac{1}{2D\zeta}\int_{\mathbf{x}(0)}^{\mathbf{x}(t)}\mathbf{\Phi}(\mathbf{s})d\mathbf{s}=\frac{1}{2D\zeta}\int_{\mathbf{x}(0)}^{\mathbf{x}(t)}\mathbf{\Phi}(\mathbf{s})d\mathbf{s}=\frac{1}{2D\zeta}(-\phi(\mathbf{x})+\phi(\mathbf{x}_0)). \tag{78}$$

So for clarity:

$$P(\mathbf{x},t) = G(\mathbf{x},t)e^{-\overline{\phi}(\mathbf{x})}, \tag{79}$$

$$G(\mathbf{x},t) = P(\mathbf{x},t)e^{\overline{\phi}(\mathbf{x})}, \tag{80}$$

$$\nabla\overline{\phi}(\mathbf{x}) = \frac{1}{2D\zeta}\nabla\phi(\mathbf{x}). \tag{81}$$

And plug it in Eq. (67). We will now try to derive the equation that $G(\mathbf{x},t)$ has to fulfill. Let us evaluate left-hand side of the Eq. (67)

$$\frac{\partial P(\mathbf{x},t)}{\partial t} = \frac{\partial G(\mathbf{x},t)}{\partial t}e^{-\overline{\phi}(\mathbf{x})}. \tag{82}$$

For the right-hand side lets evaluate first the term

$$\begin{aligned}\frac{\partial\left(-\frac{1}{\zeta}\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{x}}P(\mathbf{x},t)\right)}{\partial\mathbf{x}} &= -P(\mathbf{x},t)\frac{1}{\zeta}\frac{\partial^2\phi(\mathbf{x})}{\partial\mathbf{x}^2} - \frac{1}{\zeta}\frac{\partial P(\mathbf{x},t)}{\partial\mathbf{x}}\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{x}}, \\ &= -2DP(\mathbf{x},t)\frac{\partial^2\overline{\phi}}{\partial\mathbf{x}^2} - 2D\frac{\partial P(\mathbf{x},t)}{\partial\mathbf{x}}\frac{\partial\overline{\phi}}{\partial\mathbf{x}}, \\ &= -2DGe^{-\overline{\phi}(\mathbf{x})}\frac{\partial^2\overline{\phi}}{\partial\mathbf{x}^2} - 2D\frac{\partial G}{\partial\mathbf{x}}e^{-\overline{\phi}(\mathbf{x})}\frac{\partial\overline{\phi}}{\partial\mathbf{x}} + 2DP\left(\frac{\partial\overline{\phi}}{\partial\mathbf{x}}\right)^2.\end{aligned} \tag{83}$$

While the other term

$$\begin{aligned}\frac{\partial P(\mathbf{x},t)}{\partial\mathbf{x}} &= \frac{\partial G(\mathbf{x},t)}{\partial\mathbf{x}}e^{-\overline{\phi}} - G(\mathbf{x},t)e^{-\overline{\phi}}\frac{\partial\overline{\phi}}{\partial\mathbf{x}}, \\ &= \frac{\partial G(\mathbf{x},t)}{\partial\mathbf{x}}e^{-\overline{\phi}} - P(\mathbf{x},t)\frac{\partial\overline{\phi}}{\partial\mathbf{x}}.\end{aligned} \tag{84}$$

The second derivative then with function arguments omitted for brevity, yet remaining the same

$$\begin{aligned}D\frac{\partial^2 P}{\partial\mathbf{x}^2} &= D\frac{\partial^2 G}{\partial\mathbf{x}^2}e^{-\overline{\phi}} - D\frac{\partial G}{\partial\mathbf{x}}e^{-\overline{\phi}}\frac{\partial\overline{\phi}}{\partial\mathbf{x}} - D\frac{\partial P}{\partial\mathbf{x}}\frac{\partial\overline{\phi}}{\partial\mathbf{x}} - DP\frac{\partial^2\overline{\phi}}{\partial\mathbf{x}^2}, \\ &= D\frac{\partial^2 G}{\partial\mathbf{x}^2}e^{-\overline{\phi}} - 2D\frac{\partial G}{\partial\mathbf{x}}e^{-\overline{\phi}}\frac{\partial\overline{\phi}}{\partial\mathbf{x}} + DP\left(\frac{\partial\overline{\phi}}{\partial\mathbf{x}}\right)^2 - DGe^{-\overline{\phi}}\frac{\partial^2\overline{\phi}}{\partial\mathbf{x}^2}.\end{aligned} \tag{85}$$

After subtracting the terms right-hand side:

$$\frac{\partial G}{\partial t}e^{-\overline{\phi}} = D\frac{\partial^2 G}{\partial\mathbf{x}^2}e^{-\overline{\phi}} - DGe^{-\overline{\phi}}\left(\frac{\partial\overline{\phi}}{\partial\mathbf{x}}\right)^2 + DGe^{-\overline{\phi}}\frac{\partial^2\overline{\phi}}{\partial\mathbf{x}^2}. \tag{86}$$

Or written nicely after exponential cancels and we return to $\phi(\mathbf{x})$ from $\overline{\phi}$

$$\frac{\partial G(\mathbf{x},t)}{\partial t} = D\frac{\partial^2 G(\mathbf{x},t)}{\partial\mathbf{x}^2} - G(\mathbf{x},t)\left(\frac{1}{4D}\left(\frac{1}{\zeta}\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{x}}\right)^2 - \frac{1}{2\zeta}\frac{\partial^2\phi(\mathbf{x})}{\partial\mathbf{x}^2}\right). \tag{87}$$

Which is a well studied diffusion-reaction equation

$$\frac{\partial u(\mathbf{x},t)}{\partial t} = D\frac{\partial^2 u(\mathbf{x},t)}{\partial\mathbf{x}^2} - ku(\mathbf{x},t). \tag{88}$$

Notice for $\phi(\mathbf{x}) \equiv 0$ we already solved this equation as it is identical to Fokker-Planck where $F = 0$. Lets call this solution u_0 Another observation is that for this equation we can formulate a solution in the form

$$\begin{aligned}u(\mathbf{x},t) &= u_0(\mathbf{x},t)e^{-\int_0^t k(s)ds}, \\ u_0(\mathbf{x},t) &= Ce^{\frac{-(\mathbf{x}-\mathbf{x}_0)^2}{4Dt}},\end{aligned} \tag{89}$$

where C is some normalization constant which we will not be careful about as it would not matter by the same argument as in the previous solution:

$$G(\mathbf{x},t) = C\exp\left(\frac{-(\mathbf{x}-\mathbf{x}_0)^2}{4D\epsilon} - \int_0^t k(s)ds\right), \tag{90}$$

and the original $P(\mathbf{x},t)$ using Eq. (77):

$$P(\mathbf{x},t) = C\exp\left(\frac{-(\mathbf{x}-\mathbf{x}_0)^2}{4D\epsilon} + \int_0^t \left(-k(s) + \frac{1}{2D\zeta}\mathbf{\Phi}(\mathbf{s})\right) ds\right). \tag{91}$$

Now we would repeat same multiplication of probabilities for small time increments. However, this time, situation is easier as integrals would simply extend in the sum. Therefore the only limit would be in the first term exactly as done before. The final probability of the path is then

$$P(\mathbf{x},t) = C\int_{\mathbf{x}_0,0}^{\mathbf{x},t} \exp\Big[-\frac{1}{4D}\int_0^t (\dot{\mathbf{x}})^2 + \left(\frac{1}{\zeta}\frac{\partial\phi}{\partial\mathbf{x}}\right)^2 - \frac{2D}{\zeta}\frac{\partial^2\phi}{\partial\mathbf{x}^2} - 2\mathbf{\Phi} d\mathbf{s}\Big] d\mathbf{x}dt. \tag{92}$$

The negative argument of the exponential will yet again be an action to minimize

$$S(\mathbf{x}(t)) = \frac{1}{4D}\int_0^t (\dot{\mathbf{x}})^2 + \left(\frac{1}{\zeta}\frac{\partial\phi}{\partial\mathbf{x}}\right)^2 - \frac{2D}{\zeta}\frac{\partial^2\phi}{\partial\mathbf{x}^2} - 2\mathbf{\Phi}(\mathbf{s})d\mathbf{s}. \tag{93}$$

Which can be further, by integrating forces along the path and using forces instead of a potential, modified to the more common form

$$S(\mathbf{x}(t)) = \frac{1}{2D}(\phi(\mathbf{x}) - \phi(\mathbf{x}_0)) + \frac{1}{4D}\int_0^t \dot{\mathbf{x}}^2 + \left(\frac{1}{\zeta}\frac{\partial\phi}{\partial\mathbf{x}}\right)^2 - \frac{2D}{\zeta}\frac{\partial^2\phi}{\partial\mathbf{x}^2} d\mathbf{s}. \tag{94}$$

This procedure to derive the Onsager-Machlup action is similar to that in Mauri (2012).

F FIXED ENDPOINTS

In the whole course of the paper we will be operating with fixed endpoints. This means the actual minimized action will be reduced simply to the following:

$$S(x(t_e)) = \frac{1}{4D}\int_0^t \dot{\mathbf{x}}^2 + \left(\frac{1}{\zeta}\frac{\partial\phi}{\partial\mathbf{x}}\right)^2 - \frac{2D}{\zeta}\frac{\partial^2\phi}{\partial\mathbf{x}^2} d\mathbf{s}. \tag{95}$$

The first and simplest strategy to keep endpoint constant is to include a penalty in the form

$$L_p = C_{spring}\left[(\mathbf{x}(0) - \overline{\mathbf{x}}_0)^2 + (\mathbf{x}(t) - \overline{\mathbf{x}}_T)^2\right]. \tag{96}$$

Interestingly, as verified experimentally the penalty term effectively works the same as using a more simple and straightforward method. The method of choice was to set gradients of the endpoint to 0 manually. Only a couple of points will be affected and the majority of the trajectory will the same for both approaches.

G NUMERICAL DISCRETIZATION

Another aspect to consider is the numerical discretization of the action. The work Adib (2008) discusses the aspect of different numerical evaluation of this actions stemming from stochastic nature of the Langevin equation. We shall follow the discretization S_2 as it is derived that the action can be used for direct probability maximization. The discretization we shall use in the code is

$$S(\mathbf{x}_0,\mathbf{x}_1 ... \mathbf{x}_n) = \frac{1}{4D}\sum_{j=1}^{N} -\frac{(\mathbf{x}_j - \mathbf{x}_{j-1})^2}{\Delta t} + \Delta t\left(\frac{\mathbf{\Phi}(\mathbf{x}_j)}{\zeta}\right)^2 + \frac{2D\Delta t}{\zeta}\nabla\cdot\mathbf{\Phi}. \tag{97}$$

To make also multidimensional, multiparticle system discretization clear we just extend the sum along spatial dimensions (index j) and we sum also particles (index k). The coefficient ζ is now a vector since it is originally $\zeta_k = \gamma/M_k$ where M_k is the vector of masses. The total action is then

$$S[\mathbf{x}^{(0)},...,\mathbf{x}^{(L)}] = \sum_{i=1}^{L-1}\sum_{j=1}^{N_p} \frac{1}{4D\Delta t}\left\|\mathbf{x}_j^{(i+1)} - \mathbf{x}_j^{(i)}\right\|^2 + \frac{\Delta t}{4D\zeta_j^2}\left\|\mathbf{\Phi}_j(\mathbf{x}^{(i)})\right\|^2 - \frac{\Delta t}{2\zeta_j}\nabla\cdot\mathbf{\Phi}_j(\mathbf{x}^{(i)}). \tag{98}$$

H ADDITIONAL DETAILS ON ONSAGER-MACHLUP ACTION MINIMIZATION METHOD

Algorithmic Description of OM Action Optimization with Generative Models We provide a full algorithmic description of our main OM action optimization method with generative models in Algorithm 1.

Algorithm 1 Onsager-Machlup Transition Path Optimization with Generative Models

Input:
Optimization time τ_{opt}
Generative model with learned time-conditional score $\mathbf{s}_\theta(\cdot,\tau_{\text{opt}})$
Two atomistic configurations $\mathbf{x}^{(0)} \in \mathcal{A} \subset \mathbb{R}^{N_p \times d}$, $\mathbf{x}^{(L)} \in \mathcal{B} \subset \mathbb{R}^{N_p \times d}$
Compute initial guess $\mathbf{X} = \{\mathbf{x}^{(0)},...,\mathbf{x}^{(L)}\} = \text{InitialGuess}(\mathbf{x}^{(0)},\mathbf{x}^{(L)},\tau_{\text{initial}})$ (Algorithm 2)
while not converged **do**
    Compute the OM action, $S_\theta[\mathbf{X}]$, with the learned vector field $\mathbf{s}_\theta(\cdot,\tau_{\text{opt}})$, using Eq. (5).
    Update $\mathbf{X} \leftarrow \text{optimizer}(\mathbf{X}, \nabla_{\mathbf{X}} S_\theta[\mathbf{X}])$ (keeping the endpoints $\mathbf{x}^{(0)},\mathbf{x}^{(L)}$ fixed)
end while

Note that since S_θ is a discretized integral, the entire trajectory must be optimized in parallel, which we achieve via vectorization over the path. In principle, when optimizing in configurational space, $\tau_{\text{opt}} \approx 0$ ensures that $\mathbf{s}_\theta(\mathbf{x},\tau)$ approximates the true, atomistic forces. However, in line with previous work Arts et al. (2023), we find that a small, nonzero value leads to better alignment with the true forces, and thus treat τ_{opt} as a hyperparameter. To accelerate computation of the divergence term, we use the Hutchinson estimator Hutchinson (1989), which enables computation of the trace without materializing the divergence matrix (see Appendix H for details on usage and convergence properties).

Initial Path Guess Methods. We provide a complete description and algorithmic formulation of the initial path guess method using a generative model, mentioned in Section 3. See Algorithm 2 for an algorithmic description.

The choice of initial path connecting the endpoints $\mathbf{x}^{(0)} \in \mathcal{A}$ and $\mathbf{x}^{(L)} \in \mathcal{B}$ is crucial in determining the quality of the subsequently optimized path. Naïve linear interpolations in configurational space are unlikely to generate physical paths, since plausible atomistic structures typically lie on a highly non-convex, low-dimensional manifold of Ω. We instead opt to linearly interpolate at a *latent* level τ_{initial} of our pretrained generative model, which is known to produce samples close to the data manifold (Ho et al., 2020).

Formally, consider a generative model with a non-parametric, probabilistic encoding (i.e corruption) process $q(\mathbf{x}_\tau|\mathbf{x}_0)$, and a corresponding, parametric decoding (i.e generative) process $p_\theta(\mathbf{x}_0|\mathbf{x}_\tau)$. We encode the endpoints of the path into the chosen latent level for the initial guess, τ_{initial} to produce two latent endpoints $\mathbf{z}^{(0)} \sim q(\mathbf{z}_{\tau_{\text{initial}}}|\mathbf{x}^{(0)})$ and $\mathbf{z}^{(L)} \sim q(\mathbf{z}_{\tau_{\text{initial}}}|\mathbf{x}^{(L)})$. We then interpolate linearly to generate a latent path $\mathbf{Z} = \{\mathbf{z}^{(i)} = (1-\frac{i}{L})\mathbf{z}^{(0)} + \frac{i}{L}\mathbf{z}^{(L)}\}_{i\in[0,L]}$. We can then either decode the path back to the configurational space via $p_\theta(\mathbf{x}|\mathbf{z}^{(i)})$ to obtain a path $\mathbf{X}$, in which case the subsequent OM optimization would occur in configurational space, or defer decoding, in which case optimization occurs at the latent level τ_{initial} starting from the latent path $\mathbf{Z}$. Generally, larger values of τ_{initial} yield more diverse initial paths at expense of decreased correspondence to the endpoints $\mathbf{x}^{(0)},\mathbf{x}^{(L)}$.

When using a classical FF, we do not have access to a generative model. Thus, we must use a different scheme than what is described in Section 3 to compute the initial guess path. We first start with a small number of replicas in each basin, creating a large gap in the middle of the path. We then optimize with an unphysically heavily weighted path term, creating short, but interpolating trajectory. After we are satisfied with the initial guess we multiply each replica twice, creating a path of double that we then optimize again. This simple procedure is repeated until we reach desired length of the path. See Algorithm 3 for an algorithmic description.

Hutchinson Trace Estimator The third term of the full OM action in Eq. (5) involves the divergence of the force, $\nabla \cdot F_\theta(\mathbf{x}^{(i)}, \tau_{\text{opt}})$, or equivalently for conservative forces, the trace of the Hessian

Algorithm 2 Initial Guess Path Generation with a Generative Model

Given a generative model with non-parametric encoder q, and decoder p_θ.
Function InitialGuess($\mathbf{x}^{(0)}$,$\mathbf{x}^{(L)}$,τ_{initial})
  Encode both samples into latent level τ_{initial} of the generative model:
    $\mathbf{z}^{(0)} \sim q(\mathbf{z}_{\tau_{\text{initial}}}|a)$, $\mathbf{z}^{(L)} \sim q(\mathbf{z}_{\tau_{\text{initial}}}|b)$
  Interpolate linearly (or spherically) in the latent space to generate an initial guess latent path:
    $\mathbf{Z} = \left\{\mathbf{z}^{(i)} = (1-\frac{i}{L})\mathbf{z}^{(0)} + \frac{i}{L}\mathbf{z}^{(L)}\right\}_{i\in[0,L]}$
  Decode each point on the initial latent path $\mathbf{Z}$ from τ_{initial} to $\tau = 0$ to produce a data path:
    $\mathbf{X} = \left\{\mathbf{x}^{(i)} \sim p_\theta(\mathbf{x}|\mathbf{z}^{(i)})\right\}_{i\in[0,L]}$
  Return $\mathbf{X}$

Algorithm 3 Initial Guess Path Generation with Iterative Unwrapping

Function InitialGuess($\mathbf{x}^{(0)}$,$\mathbf{x}^{(L)}$,L_1,N)
Initialize trajectory by copying boundary points $L_1/2$ times on both ends.
$\mathbf{X} = \left\{\mathbf{x}^{(0)},...,\mathbf{x}^{(0)},\mathbf{x}^{(L)},...,\mathbf{x}^{(L)}\right\}$
For m from 1 to N repeat:
  Duplicate every point along the path: $\longrightarrow$ $\mathbf{X} = \left\{\mathbf{x}^{(0)},\mathbf{x}^{(0)},\mathbf{x}^{(1)},\mathbf{x}^{(1)}...,\mathbf{x}^{(L)}\mathbf{x}^{(L)}\right\}$
  Optimize Truncated Onsager-Machlup action: $S_{trunc} \longrightarrow$ $\mathbf{X} = \text{argmin} S_\theta(\mathbf{X})$
Obtain initial guess $\mathbf{X} = \left\{\mathbf{x}^{(0)},\mathbf{x}^{(1)},...,\mathbf{x}^{(2^N * L_1 - 1)},\mathbf{x}^{(L)}\right\}$

(Laplacian) of a scalar energy. Naively computing gradients of this quantity can be prohibitively expensive. We thus employ the Hutchinson trace estimator Hutchinson (1989) to accelerate computation. Formally, let $\mathcal{H}(\mathbf{x}) = \nabla \cdot F_\theta(\mathbf{x},\tau_{\text{opt}}) \in \mathbb{R}^{N_p * d \times N_p * d}$. We approximate the trace of $\mathcal{H}$ as $\text{tr}(\mathcal{H}(\mathbf{x})) \approx \frac{1}{N}\sum_{i=1}^{N} \mathbf{v}^\intercal \mathcal{H}(\mathbf{x})\mathbf{v}$, where $\mathbf{v} \sim \mathcal{N}(0,I)$. By leveraging vector-Jacobian products (VJP), we can compute the trace without materializing $\mathcal{H}$ or its diagonal elements.

In practical terms, the estimator converges rather slowly. Let us denote the approximated trace by $\hat{\text{Tr}}$. One can derive the variance of the estimator as

$$Var(\hat{\text{Tr}}) = \frac{1}{N}\text{Var}(\mathbf{v}_i \cdot \mathcal{H}(\mathbf{x})\mathbf{v}_i), \tag{99}$$

which, when $\mathbf{v}_i$ are distributed identically means the error of the trace estimator decays as

$$|\hat{\text{Tr}} - \text{Tr}(\mathcal{H}(\mathbf{x}))| \leq \frac{C}{\sqrt{N}}. \tag{100}$$

Where C depends on the properties of the matrix. This convergence is rather slow and means that one requires many iterations to arrive to an accurate value of the trace. In practice, however, we found that $N = 15$ worked well and led to smooth OM optimization. This is likely due to the fact that our trajectories were composed of many neighboring points that likely had similar Laplacian values.

I MÜLLER-BROWN POTENTIAL EXPERIMENTS

We demonstrate our method on the 2D Müller-Brown potential Müller & Brown (1979), a canonical test system for TPS with a global minimum and two local minima, separated by saddle points.

Problem setup. Using a denoising diffusion model pretrained on samples from the underlying potential energy surface, we generate transitions between the two deepest energy minima using OM optimization. We generate the initial guess path via linear interpolation at $\tau_{\text{initial}} = 8$ (the maximum diffusion model time is $T_d = 1{,}000$). We perform 200 steps of OM optimization directly at $\tau_{\text{initial}} = 8$ using $\tau_{\text{opt}} = 8$, and finally decode the path back to the data distribution via the denoising process.

Results. As shown in Fig. 3, OM optimization with the truncated action yields a transition path (shown in blue) between the energy minima which passes through the lowest energy barrier. Due to the stochastic decoding process, the samples around the transition path exhibit natural diversity. Increasing D and

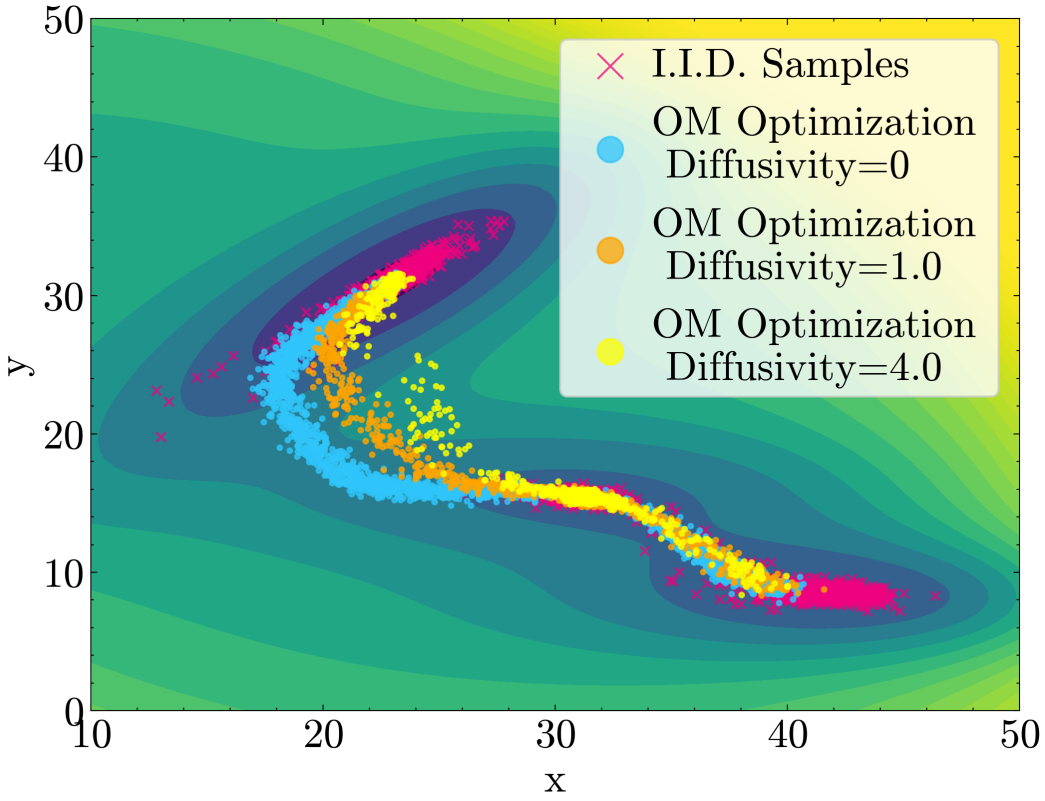

Figure 3: **OM optimization with a diffusion model on the 2D Müller-Brown potential.** Individual points along the OM-optimized paths are shown as dots. Increasing the diffusivity D causes the path to cross at a higher barrier. An equivalent number of I.I.D samples (red) fail to sample the transition region.

optimizing the full OM action results in qualitatively different transition paths (shown in orange and yellow). Samples are more concentrated in the three energy minima, and the path crosses a higher energy barrier, consistent with the larger scale of thermal fluctuations at increased temperatures. Meanwhile, 2500 i.i.d. samples (shown in red) from the diffusion model sample only the three energy minima, and fail to sample the transition region. This provides a proof-of-concept that our OM optimization procedure can be used to repurpose generative models to sample transition paths without specialized training.

We now provide further details on these Müller-Brown experiments. All experiments were performed on a single NVIDIA RTX A6000 GPU.

Potential Parameters. The exact form of the potential used is the following:

$$\begin{aligned} U(x,y) = &-17.3e^{-0.0039(x-48)^2-0.0391(y-8)^2} \\ &-8.7e^{-0.0039(x-32)^2-0.0391(y-16)^2} \\ &-14.7e^{-0.0254(x-24)^2+0.043(x-24)(y-32)-0.0254(y-32)^2} \\ &+1.3e^{0.00273(x-16)^2+0.0023(x-16)(y-24)+0.00273(y-24)^2} \end{aligned}$$

This generates the potential shown in Figure 1. Figure 4 shows OM optimization using the analytical potential as the force field. Increasing the diffusivity yields paths that cross higher energy barriers, aligning with physical intuition. The results with the diffusion model in Section **??** align with the paths derived from the analytical potential, confirming the validity of the diffusion model as an approximation of the forces.

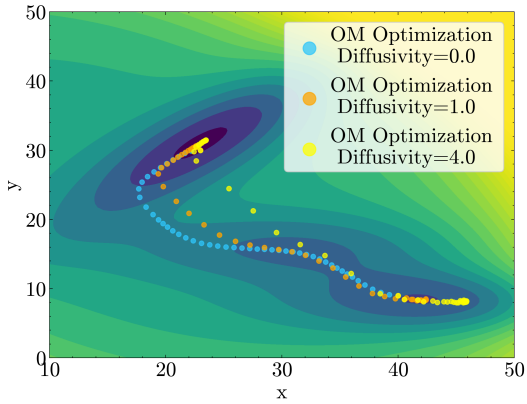

Figure 4: **OM optimization can be done with an analytical potential.** We show paths generated with OM optimization using the analytical potential with a timestep of 1, a friction of 1, and multiple diffusivities. Higher diffusivities, corresponding to higher temperatures in physical interpretation, can cross higher energy barriers, aligning with physical intuition.

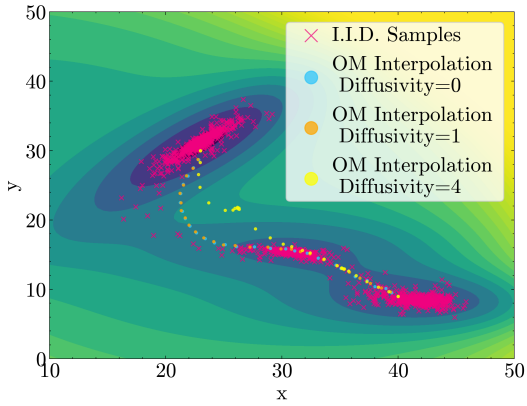

Figure 5: **OM optimization from a flow matching model**. The analogous experiment to Fig. 4, but using a flow matching model trained on Müller-Brown data, and using the extracted score via Eq. (38) for OM optimization. Since the stochastic encoding/decoding processes of flow matching models are deterministic and the Müller-Brown setting has a single minimal landscape, the encoding/decoding scheme in itself cannot generate diversity in this setting. Note that for higher diffusivities, OM optimization can find some erroneous minima left from the generative model (here at $D = 4$, OM optimization finds a local minima in the center, where a crease is formed).

Data generation. To ensure that the transition region is adequately represented with relatively short simulation times, we choose initial conditions for the simulations by uniformly sampling the transition path resulting from OM optimization under the true MB potential. We generate training data by running unbiased, constant-temperature simulations with the MB potential under Langevin dynamics. We run 1,000 parallel simulations for 1,000 steps, yielding a total of 1 million datapoints. Of these, 800,000 are used for training, and 200,000 are reserved for validation.

Training. We then train a standard denoising diffusion model on this dataset, with the denoising model parameterized by a 3-layer MLP with a GELU activation (Hendrycks & Gimpel, 2023) and a hidden dimension of 256. The model is trained for 10 epochs, with a batch size of 4096 and a learning rate of $1e-3$ using the Adam optimizer (Kingma & Ba, 2017).

Energy Laplacian Term. We estimate the Laplacian of the potential energy surface by using the Hutchinson Trace Estimator (see Section H). As shown in Figure 6, one random vector ($N = 1$) is enough to capture the local minima and the energy barrier using the Hutchinson Trace Estimator, so we

use $N = 1$ in our experiments. Using more random vectors gives a less noisy estimate of the Laplacian, trading off accuracy for computational expense.

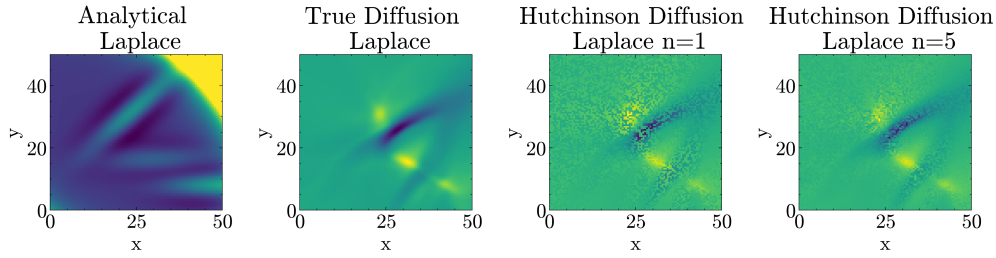

Figure 6: **Hutchinson Trace Estimator accurately estimates the Laplacian.** The diffusion model learns an estimate of the Laplacian that captures the Müller-Brown energy wells. The Hutchinson Trace Estimator efficiently approximates this Laplacian, and the estimate becomes less noisy when using more random samples.

OM Optimization Details. We pick two points on the potential energy surface (PES) at alternate ends of the transition barrier as target points for interpolation. The hyperparameters for the OM optimization are given in Table 1

Table 1: Hyperparameters used for OM action optimization on the Müller-Brown potential with diffusion.

Hyperparameter	**Value**
Number of Generated Paths	50
Action Type	Full
Initial Guess Time (τ_{initial})	8
Optimization Time (τ_{opt})	8
Optimization Steps	200
Optimizer	Adam
Learning Rate	0.2
Path Length (L)	50
Action Timestep (Δt)	0.01
Action Friction (γ)	0.01
Action Diffusivity (D)	0, 1.0, 4.0

J GENERALIZATION TO UNSEEN TETRAPEPTIDES

As another evaluation of our OM optimization approach, we consider tetrapeptide systems, which exhibit interesting dynamics and pose the challenge of generalization to held-out amino acid sequences.

Problem setup. We train denoising diffusion and flow matching generative models on a subset of the tetrapeptides simulated in Jing et al. (2024b), and apply our OM interpolation procedure to generate an ensemble of 100 transition paths between min-flux states for each of 59 held-out tetrapeptides that were *not seen* during training. We use the same MSM-based metrics as in Section 4.1 to evaluate the quality of generated transition paths. Following Jing et al. (2024b), we consider replicate MD simulations of varying lengths as baselines.

Results. As shown in Fig. 7, OM optimization achieves MSM metrics which are competitive with 50-100 ns MD simulations, which are considerably more computationally expensive to generate. This suggests the promise of OM optimization to generate transition paths on atomistic systems not explicitly seen during training.

We now provide additional details on these experiments. All experiments were performed on a single NVIDIA RTX A6000 GPU.

Coarse Graining. We coarse grain the tetrapeptides at the backbone level. That is, we represent each residue by three beads, representing the N, C_α, and C atoms. This yields a 12-bead representation for each tetrapeptide.

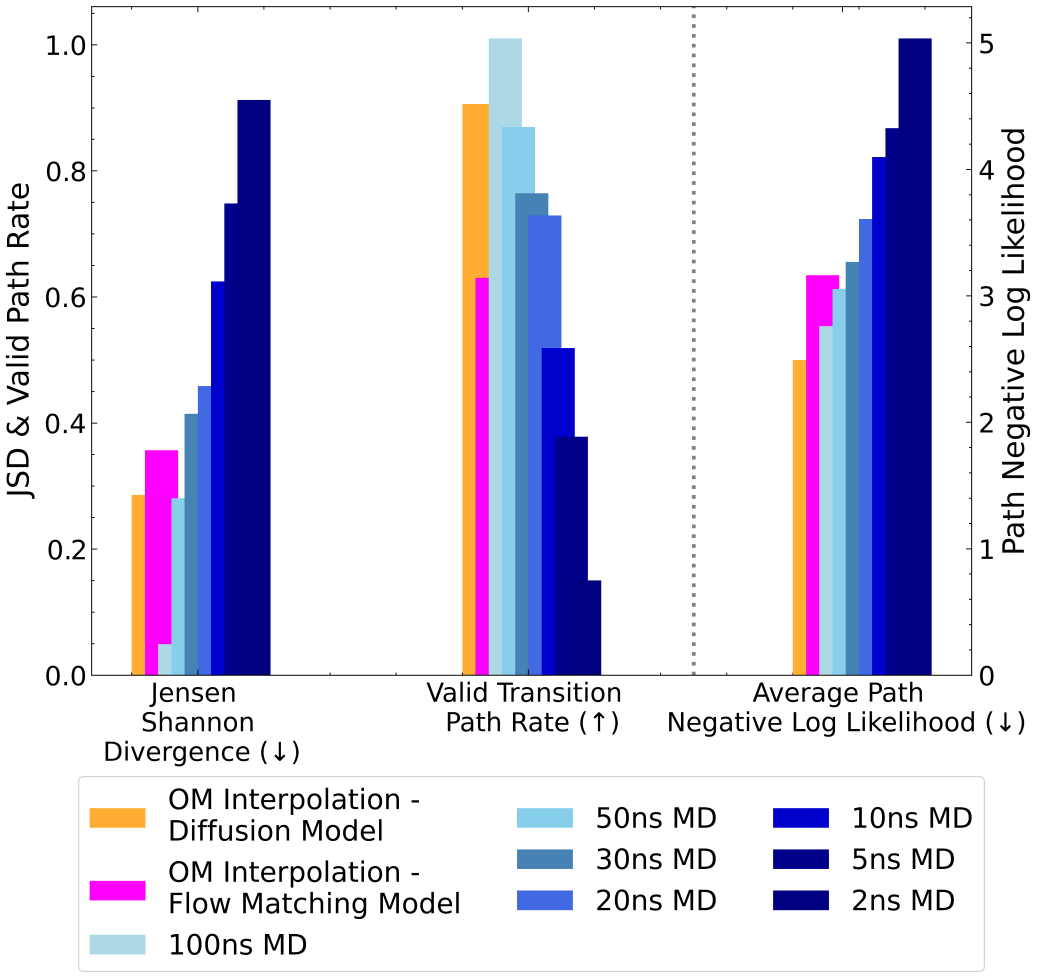

Figure 7: **OM optimization on held-out tetrapeptide sequences.** OM optimization with diffusion and flow models yields transition paths which compare strongly with variable-length MD simulations.

Training. We train diffusion and flow matching models, parameterized by a Graph Transformer with all the same architecture and training hyperparameters used for the fast-folding proteins (Section L), with the only difference being the inclusion of learnable bead embeddings for differentiating the amino acid backbone atoms. These are concatenated to the residue ordering and the diffusion/flow timestep to form the node features. We train on a subset of 700 tetrapeptides simulated in Jing et al. (2024b), taking 10,000 evenly spaced configurations from the simulations for each tetrapeptide.

OM Optimization on Held-Out Proteins. We perform OM optimization on 58 held-out tetrapeptide sequences not seen during training (again using the same splits as in Jing et al. (2024b)). The optimization hyperparameters are given in Table 2.

Evaluation. We use the same Markov State Model-based evaluation pipeline described in Section L for the fast-folding proteins. Following Jing et al. (2024b), the TIC dimensions are fit on the backbone torsion angles. The reported metrics are averaged over the 58 held-out test proteins.

Visualization of Sampled Paths. We provide TIC visualizations of sampled transition paths for selected tetrapeptides in Figure 8.

Table 2: Hyperparameters used for OM action optimization on tetrapeptides with diffusion models.

Hyperparameter	Value
Number of Generated Paths	100
Action Type	Truncated
Initial Guess Time	100
Optimization Time	0
Optimization Steps	250
Optimizer	Adam
Learning Rate	0.2
Path Length (L)	25
Action Timestep (Δt)	1
Action Friction (γ)	10
Action Diffusivity (D)	0

K CLASSICAL FORCE FIELDS ON ALL-ATOM PROTEINS

We also demonstrate that our OM action optimization framework is broadly useful for transition path sampling even beyond the setting of generative modeling. Specifically, we aim to find all-atom transition paths between the unfolded and folded states of the protein Chignolin and Trp-Cage, using a differentiable PyTorch implementation Doerr et al. (2020); Sipka et al. (2023) of the Amber *ff14SB*Maier et al. (2015) forcefield and the *TIP3P* implicit water model. We choose the physical parameters of the OM action to be consistent with commonly used values in molecular simulations (see Table 3). Since we do not have a generative model from which to obtain an initial path guess via latent interpolation as described in 3, we instead employ a **hierarchical unwrapping** warm-up procedure described in the Appendix H to obtain initial paths. As the classical force field is dominated by quadratic terms whose Laplacian is constant and thus uninformative for optimization, we use a zero-temperature approximation and optimize with the Truncated OM action. Using the Truncated action, we obtain a physical transition path of length 2.6ps (shown in Figure 9). This is much lower than previously reported transition path lengths for Chignolin Sobieraj & Setny (2022); Lindorff-Larsen et al. (2011), which can be explained by the fact that our trajectories proceed between the target states without fluctuations that would occur in unbiased simulations. The entire optimization took on the order of hours on one NVIDIA RTX A6000 GPU, including the generation of initial trajectory.

Table 3: Hyperparameters used for OM action optimization on all-atom proteins with a classical force field.

Hyperparameter	Chingolin - warmup	Chingolin	TRP Cage - warmup	TRP Cage
Number of points per path	40 - 2600	2600	40 - 2600	2600
Action Type	Truncated	Truncated	Truncated	Truncated
Optimizer	Adam	Adam	Adam	Adam
Learning Rate	10^{-4}	10^{-5}	10^{-4}	10^{-5}
Action Timestep (Δt)	1 fs	1 fs	1 fs	1 fs
Action Friction (γ)	10 ps^{-1}	10 ps^{-1}	10 ps^{-1}	10 ps^{-1}

L FAST-FOLDING PROTEIN EXPERIMENTS

We present additional details on the fast-folding protein results in Section 4.1. All experiments were performed on a single NVIDIA RTX A6000 GPU.

MD Simulations with Diffusion Model To assess the computational efficiency and accuracy of our OM optimization relative to alternative methods of sampling the configurational space (see Figure 2a and b), we run Langevin MD simulations for varying lengths of time (up to 12 ns) using the diffusion model's score function as an effective force field Arts et al. (2023), with a timestep of 2 fs. To ensure a fair comparison, we set the number of parallel MD trajectories to be the same as the number of transition paths sampled with OM optimization.

Markov State Model Construction. We provide further details on the Markov State Model analysis used to evaluate the quality of transition paths for the fast-folding protein experiments. We largely follow the procedure described in Jing et al. (2024b).

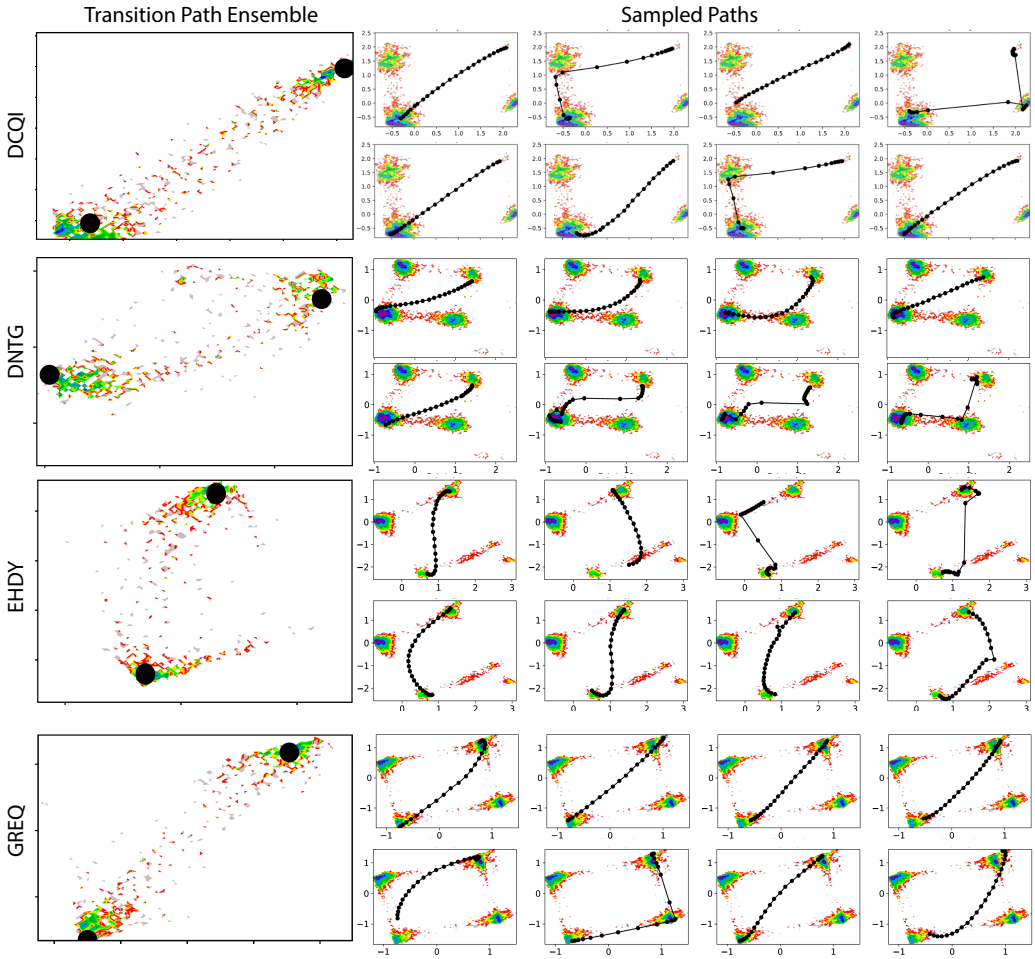

Figure 8: **Sampled transition paths from OM optimization on selected, held-out tetrapeptide sequences.** The sampled paths are diverse and intuitively pass through high density regions in the TIC free energy landscape.

We perform k-means clustering of the reference MD simulations into 20 clusters using the top 2 Time Independent Component (TIC) dimensions, which are fit on the pairwise distances and dihedral angles of the protein configurations. We then fit a Markov State Model (MSM) with a lagtime of 200ps (the frequency at which the simulations were saved) to obtain a transition probability matrix T between the 20 discrete states in the MSM (e.g, $T_{j,k} = p(s_{t+1} = k | s_t = j)$, where s_t and s_{t+1} are the states at time t and $t+1$). This constitutes the reference MSM.

To evaluate transition paths sampled from our OM optimization method, we first discretize them under the reference MSM (i.e represent them as a sequence of cluster indices between 1 and 20). We subsample the paths to be of length 10. From this, we compute the probability of the path under the reference MSM via:

We also sample 1,000 discrete, reference paths of length $L = 20$ (corresponding to a transition time of 200ps $\times$ 20 = 4ns) from the reference MSM, conditioned on the start and end states s_1 and s_L (these are the cluster indices of the transition endpoints $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(L)}$. This can be achieved by sampling states $s_2 \ldots s_{19}$ iteratively as

$$s_{t+1} \sim \frac{T^{(L-t-1)}_{:,s_L} T_{s_L,:}}{T^{(L-t)}_{s_t,s_L)}}, \tag{101}$$

where the superscipt denotes a matrix exponential. See Jing et al. (2024b) for precise details.

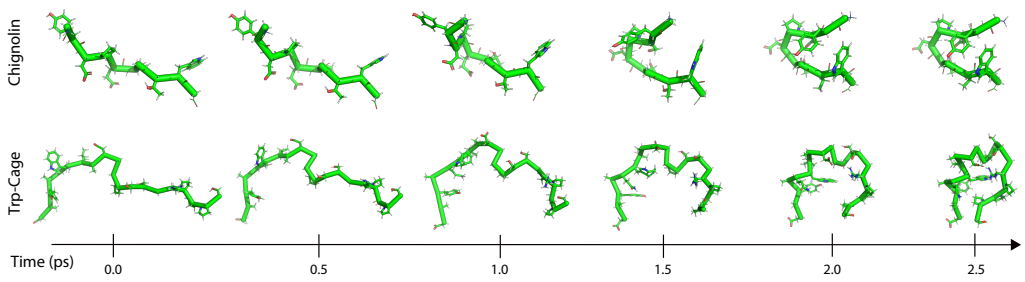

Figure 9: Transition paths from OM optimization of all-atom chignolin and trp-cage with a classical force field.

With both the reference and generated discretized paths, we compute the following metrics:

1. **Jensen-Shannon Divergence.** We compute the probability of each MSM state based on the frequency at which each state is visited in the discretized paths. We compute these probabilities for both the reference and generated paths, and compute the JSD between the categorical distributions.
2. **Path Negative Log Probability.** We compute the negative log probability of the generated paths (conditioned on the starting and ending states) under the reference MSM, averaged over all generated paths. Under the Markovian assumption, this is given by $-\log(P(s_1 \ldots s_L) = -\sum_{i=1}^{L-1} \log\left(\frac{T^{(L-t-1)}_{s_t,s_L} \cdot T_{s_t,s_{t+1}}}{T^{(L-t)}_{s_t,s_L}}\right)$.
3. **Fraction of Valid Paths.** We compute the fraction of generated paths with nonzero probability under the reference MSM.

When considering replicate MD simulations of different lengths (e.g 2 ns, 4 ns), we fit a MSM to the simulations using the same discretized clusters as were used to fit the reference MSM, and sample 1,000 paths in the same way described above.

Committor Function Analysis. For a transition event between endpoints $\mathcal{A}, \mathcal{B} \in \Omega$, the committor function $q(\mathbf{x})$, captures the probability that a trajectory initiated at $\mathbf{x}_0 = \mathbf{x}$ reaches $\mathcal{B}$ before $\mathcal{A}$:

$$q(\mathbf{x}) = \mathbb{E}[h_{\mathcal{B}}(\mathbf{x}_\tau) \,|\, \mathbf{x}_0 = \mathbf{x}]; \quad \tau = \arg\min_{t \in [0,+\infty)} \{\mathbf{x}_t \in \mathcal{A} \cup \mathcal{B} : \mathbf{x}_0 = \mathbf{x}\}, \tag{102}$$

where $h_{\mathcal{B}}$ is the indicator function for reaching state $\mathcal{B}$.

The transition state ensemble is formally defined as the level set $\{\mathbf{x} \in \Omega : q(\mathbf{x}) = 0.5\}$. The committor is obtainable as the solution to the steady-state backward Kolmogorov equation (BKE) Hasyim et al. (2022), which is generally infeasible to solve directly or numerically for high-dimensional systems. For the fast-folding proteins, we obtain an empirical estimate of the committor function by dividing the TIC configuration space of each protein into 100^2 discrete bins, and replacing the expectation in Equation 102 with an empirical average over trajectories starting from each bin in the reference MD simulations from Lindorff-Larsen et al. (2011). The resulting committor estimates for the fast-folding proteins are shown in Figure 10.

Model Architecture and Training. Our denoising diffusion and flow matching generative models are parameterized by a Graph Transformer architecture identical to what was used in Arts et al. (2023) (in the case of diffusion, we use the exact pretrained model from Arts et al. (2023)). To summarize, nodes are featurized by the ordering of each residue in the overall sequence, while edges are featurized by the pairwise C_α-C_α distances. Nodes and edges are then jointly treated as tokens for input to the Transformer, which updates the token representations. A scalar output is obtained by summing learned linear projections of the token representations. Both the denoising diffusion vector field ϵ_θ and the flow model velocity field v_θ are parameterized as the gradient of the final scalar output of the model with respect to the input C_α coordinates.

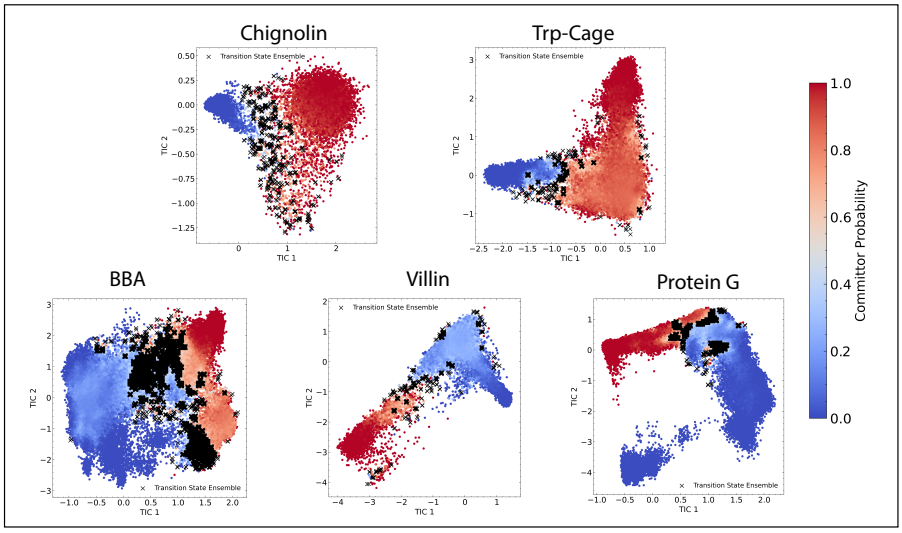

Figure 10: **Empirical committor landscapes for fast-folding proteins.** The committor is computed by binning the conformational space into 100^2 bins and measuring the frequency at which reference MD trajectories initiated in each bin reach the target state before the start state. The empirical transition ensemble is defined as the level set $\{\mathbf{x}: 0.45 \leq q(\mathbf{x}) \leq 0.55\}$.

For denoising diffusion, we use the pretrained models from Arts et al. (2023). For flow matching, we train our own models. We train models with 3 attention layers for 1 million iterations, using an Adam optimizer with a learning rate of 0.0004 and a cosine annealing schedule reducing to a minimum learning rate of 0.00001. We use an exponential moving average with $\alpha = 0.995$. The diffusion models use 1,000 integration steps at inference time, while the flow matching models use 10 steps.

Protein-specific training and architecture hyperparameters are given in Table 4.

Table 4: Architecture and training hyperparameters for diffusion and flow matching generative models on fast-folding proteins.

Hyperparameter	**Chignolin**	**Trp-cage**	**BBA**	**Villin**	**Protein G**
Batch size	512	512	512	512	256
Number of hidden features	64	128	96	128	128

OM Optimization Details. We list all the optimization hyperparameters used to perform OM optimization on the fast-folding proteins, for both diffusion and flow matching models, in Tables 5 and 6.

Table 5: Hyperparameters used for OM action optimization on fast-folding proteins with diffusion models.

Hyperparameter	**Chignolin**	**Trp-cage**	**BBA**	**Villin**	**Protein G**
Number of Generated Paths	8	8	32	4	4
Action Type	Truncated	Truncated	Full	Truncated	Truncated
Initial Guess Time (τ_{initial})	250	250	250	250	250
Optimization Time (τ_{opt})	20	15	20	10	10
Optimization Steps	2000	2000	2000	2000	2000
Optimizer	Adam	Adam	SGD	SGD	SGD
Learning Rate	0.2	0.2	0.001	0.001	0.001
Path Length (L)	200	200	200	200	200
Action Timestep (Δt)	0.1	0.1	0.1	1.0	0.1
Action Friction (γ)	10	10	10	10	10
Action Diffusivity (D)	0	0	0.01	0	0

Table 6: Hyperparameters used for OM action optimization on fast-folding proteins with flow matching models.

Hyperparameter	Chignolin	Trp-cage	BBA	Villin	Protein G
Number of Generated Paths	8	8	32	4	4
Action Type	Truncated	Truncated	Truncated	Truncated	Truncated
Initial Guess Time (τ_{initial})	7	7	7	7	7
Optimization Time (τ_{opt})	0.5	0.5	0.5	0.5	0.5
Optimization Steps	5000	5000	5000	5000	5000
Optimizer	SGD	SGD	SGD	SGD	SGD
Learning Rate	0.001	0.001	0.001	0.001	0.001
Path Length (L)	200	200	200	200	200
Action Timestep (Δt)	0.05	0.05	0.05	0.05	0.05
Action Friction (γ)	10	10	10	10	10
Action Diffusivity (D)	0	0	0	0	0

Complete Quantitative Results. In Table 7, we report Markov State Model metrics for all fast-folding proteins, showing results of OM optimization with diffusion and flow matching, and Langevin MD simulations of varying lengths, using the diffusion model's score function as the force field.

Table 7: Complete Markov State Model metrics for for fast-folding proteins. Metrics evaluated are: Fraction of Valid Paths (**FVP**), Path Negative Log Likelihood Mean (**PNLLM**), and Jensen-Shannon Divergence (**JSD**)

Protein	Sampling Method	FVP (↑)	PNLLM (↓)	JSD (↓)
Chignolin	Langevin MD (Diffusion) (2 ns)	0.005	34.5	0.40
	Langevin MD (Diffusion) (6 ns)	0.029	34.5	0.41
	Langevin MD (Diffusion) (12 ns)	0.035	34.5	**0.36**
	OM Optimization (Diffusion) (ours)	**1.0**	**20.8**	0.47
	OM Optimization (Flow Matching) (ours)	**1.0**	21.9	0.47
Trp Cage	Langevin MD (Diffusion) (2 ns)	0.046	27.3	0.40
	Langevin MD (Diffusion) (6 ns)	0.071	29.2	0.35
	Langevin MD (Diffusion) (12 ns)	0.078	23.9	0.37
	OM Optimization (Diffusion) (ours)	**1.0**	13.7	0.19
	OM Optimization (Flow Matching) (ours)	**1.0**	**12.6**	**0.18**
BBA	Langevin MD (Diffusion) (2 ns)	0.079	28.2	0.53
	Langevin MD (Diffusion) (6 ns)	0.19	21.8	0.48
	Langevin MD (Diffusion) (12 ns)	0.22	21.9	0.48
	OM Optimization (Diffusion) (ours)	0.91	17.4	0.42
	OM Optimization (Flow Matching) (ours)	**1.0**	**13.2**	**0.3**
Villin	Langevin MD (Diffusion) (2 ns)	0.0	34.5	1.0
	Langevin MD (Diffusion) (6 ns)	0.0	34.5	1.0
	Langevin MD (Diffusion) (12 ns)	0.387	15.7	0.32
	OM Optimization (Diffusion) (ours)	**1.0**	13.2	**0.18**
	OM Optimization (Flow Matching) (ours)	**1.0**	**12.1**	**0.18**
Protein G	Langevin MD (Diffusion) (2 ns)	0.0	34.5	1.0
	Langevin MD (Diffusion) (6 ns)	0.0	34.5	1.0
	Langevin MD (Diffusion) (12 ns)	0.0	34.5	1.0
	OM Optimization (Diffusion) (ours)	0.5	**11.1**	0.43
	OM Optimization (Flow Matching) (ours)	**1.0**	12.3	**0.23**

Visualization of Transition Paths. In Figures 11, 12, and 13, we provide additional visualizations of transition paths sampled by our diffusion and flow matching models for the fast folding proteins, both in TIC and atomic space.

Training without Transition Regions. We provide additional details on the data-starved experiment described in Section 4.1. Transition states are challenging to sample, and therefore may not be abundant in reference MD simulations or structural databases, which typically serve as training datasets for generative models. To simulate the scenario in which the underlying dataset is not exhaustive and under-represents the rare, transition regions, we remove 99% of the datapoints for which $0.1 \leq q(\mathbf{x}) \leq 0.9$, where $q(\mathbf{x})$ is the empirical committor value (described in **Committor Function Analysis**). Thus, most of the remaining datapoints have committor values close to 0 or 1, meaning they initiate trajectories which stay in their respective local energy minima without transitioning across the path. For Chignolin, Trp-Cage, and BBA, the subsampling procedure removes 1.4%, 68%, and

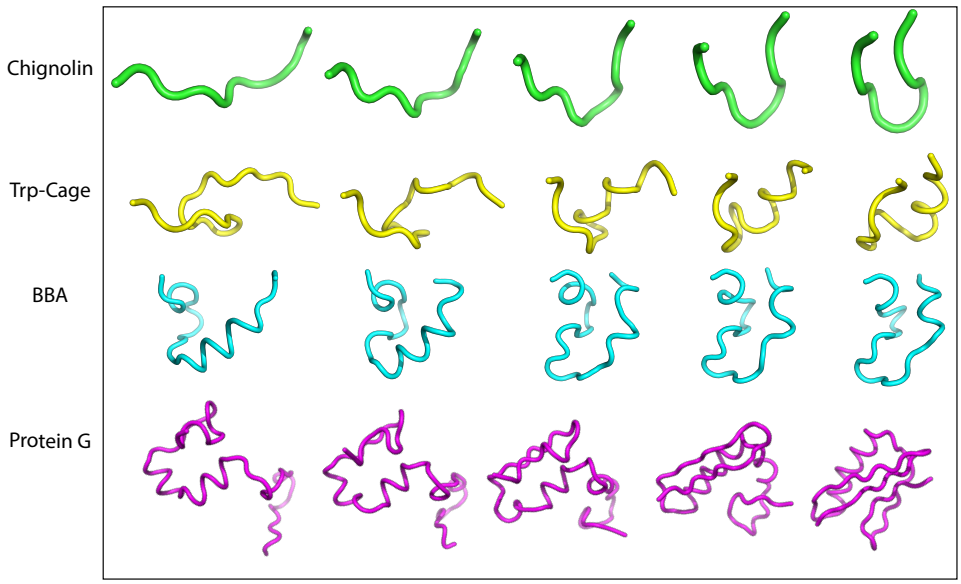

Figure 11: Visualization of sampled transition paths from OM optimization with pretrained diffusion models for fast-folding proteins.

85% of the datapoints, respectively. We train diffusion models on these subsampled datasets using the same hyperparameters described earlier, followed by OM optimization between the same endpoints, using the same hyperparameters as used before. As shown in Figure 14, the produced transition paths are similar to those shown in Figures 2a and 13. The paths still pass through the expected transition state regions (denoted in black), despite having seen them at a much lower frequency during training.

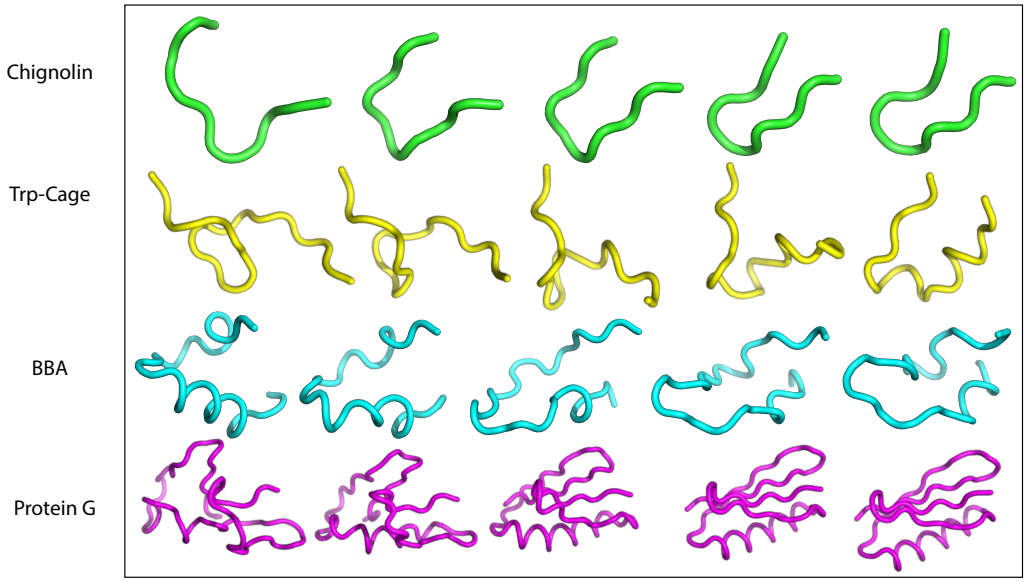

Figure 12: Visualization of sampled transition paths from OM optimization with pretrained flow matching models for fast-folding proteins.

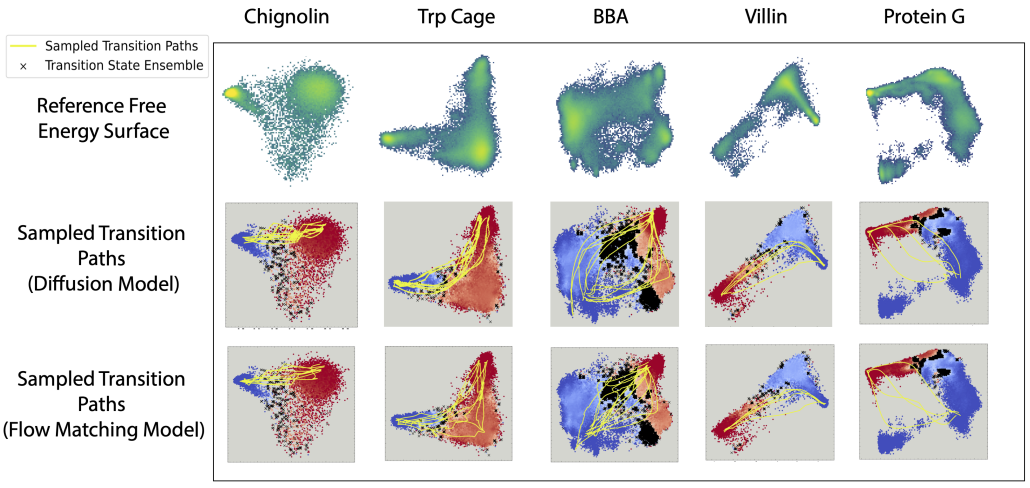

Figure 13: Visualization of sampled fast-folding protein transition paths in TIC space.

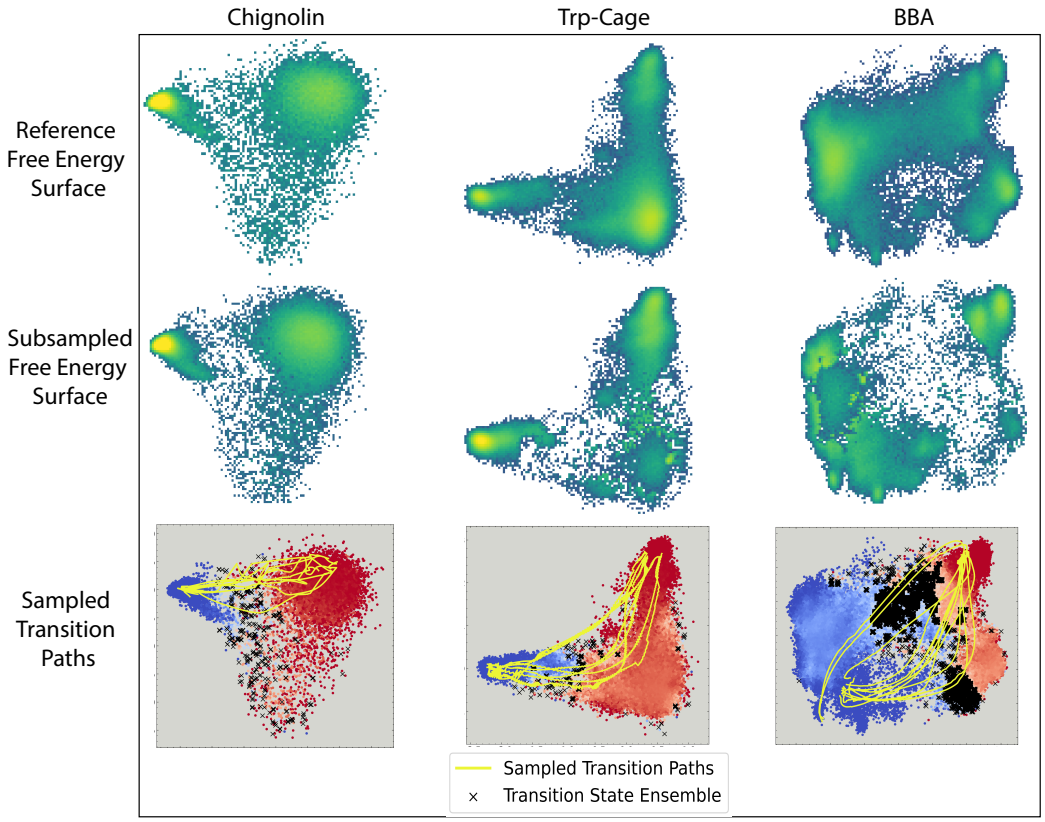

Figure 14: **Training datasets and sampled transition paths resulting from removing intermediate committor function values. (Top Row)** Original training datasets. **Middle Row** Datasets resulting from removing 99% of datapoints with committor values (obtained empirically) between 0.1 and 0.9. **Bottom row.** Transition paths resulting from OM optimization with a diffusion model trained on the subsampled datasets.