UC Berkeley
UC Berkeley Electronic Theses and Dissertations

Title

Variational Approaches to Path Problems in Deep Learning

Permalink

https://escholarship.org/uc/item/04b4p8gz

ISBN

9798173785923

Author

Psenka, Michael

Publication Date

2026-08-10

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Variational Approaches to Path Problems in Deep Learning

by

Michael Peter Psenka

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Aditi Krishnapriyan, Chair
Professor Pieter Abbeel
Professor Angjoo Kanazawa
Professor Yann LeCun, New York University

Summer 2026

Variational Approaches to Path Problems in Deep Learning

Copyright 2026
by
Michael Peter Psenka

Abstract

Variational Approaches to Path Problems in Deep Learning

by

Michael Peter Psenka

Doctor of Philosophy in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Aditi Krishnapriyan, Chair

Many problems in modern deep learning can be framed as path problems: starting from an initial state and given a learned model of the domain, find a path to a target state. This framing recurs across reinforcement learning (an agent transitioning to a high-reward state under a learned critic), molecular modeling (a protein transitioning between conformations under a learned generative model), and long-horizon planning and control (state-action sequences under a learned world model). These problems are typically under-determined: there are infinitely many candidate paths, and the learned model has varying levels of confidence in-between start and target states. A well-posed treatment requires a principled criterion for which paths are best.

This dissertation argues that variational reformulations are a useful unifying tool across these problems, and identifies two complementary sub-programs within the broader variational program. The first applies when the data manifold is smooth, low-dimensional, and recoverable: a geometric flow reshapes the manifold toward convexity, which we read as descent on a convexification energy, until the learned representation becomes amenable to classical linear analysis. The second, where the bulk of the dissertation sits, applies when the data manifold (and the models that fit it) can be more rough. Here, we frame path problems as the minimization of a functional on path space induced directly by the learned model itself (its score, its critic, or its dynamics), drawing on classical literature in the calculus of variations and optimal control. Three instances are developed: Q-score matching for diffusion reinforcement learning, Onsager–Machlup interpolation for molecular and image-latent dynamics, and parallel long-horizon planning over world models. For each of these examples, we will see various ways of smoothing out optimization problems over deep learning-based dynamics, whether it be simply through the implicit smoothing-by-splitting of the variational approach or by directly reshaping the gradient structure.

To my family and friends, without whom none of this would be possible.

Contents

Contents ii

List of Figures vi

List of Tables viii

I Foundations 1

1 Introduction 2
- 1.1 Path problems in modern deep learning . . . 2
- 1.2 Variational reformulations . . . 3
- 1.3 Roughness of deep learned models . . . 4
- 1.4 Contributions and roadmap . . . 4

2 The Variational Principle, from Physics to Learned Models 6
- 2.1 Calculus of variations, in brief . . . 6
- 2.2 A few classical examples, briefly . . . 7
- 2.3 Learned dynamical objects . . . 7
- 2.4 Variational principle for the deep learning era . . . 8

II Variational Reshaping of the Representation 10

3 Manifold Flattening as a Variational Program on Representations 11
- 3.1 Setup: minimal autoencoding . . . 12
- 3.2 Manifold flattening creates minimal representations . . . 14
- 3.3 Geometric flows and the convexification flow . . . 16
- 3.4 Flattening Networks (FlatNet) . . . 19
- 3.5 Algorithm intuition and convergence analysis . . . 25
- 3.6 Experiments . . . 29
- 3.7 Related work . . . 32
- 3.8 What this sub-program does and does not assume . . . 33

III Variational Optimization of Paths over Learned Representations **41**

4 Interlude: The Roughness of Modern Learned Representations **42**
4.1 The smooth-manifold reflex, and where it breaks . . . 42
4.2 Concrete evidence for roughness . . . 43
4.3 Why we cannot always flatten our way out . . . 45
4.4 The dissertation's response: accept the roughness, reshape the functional . . . 45
4.5 Two complementary tools, one program . . . 46

5 Q-Score Matching: Variational Policy Improvement for Diffusion Reinforcement Learning **47**
5.1 Setup: diffusion policies and the policy-improvement problem . . . 48
5.2 Continuous-time formulation . . . 49
5.3 Policy gradient for diffusion policies . . . 51
5.4 The variational framing: Q-score matching . . . 52
5.5 Theory: optimality conditions and the Boltzmann fixed point . . . 54
5.6 Multimodality preservation . . . 57
5.7 Algorithmic instantiation . . . 58
5.8 Experiments . . . 59
5.9 Related work . . . 61
5.10 Reflection . . . 63

6 Onsager–Machlup Interpolation for Learned Generative Dynamics **64**
6.1 From a generative model to a stochastic dynamical system . . . 65
6.2 The Onsager–Machlup functional . . . 67
6.3 Domain A: molecular conformational transitions . . . 68
6.4 Domain B: image-latent interpolation . . . 76
6.5 Formalizing the diffusion-time knob . . . 81
6.6 Related work . . . 83
6.7 Reflection: OM as a score-induced functional . . . 85

7 Parallel Long-Horizon Planning over Learned World Models **86**
7.1 Problem formulation . . . 87
7.2 Decoupling dynamics for gradient-based planning . . . 89
7.3 Adversarial-robustness reading and the visual world-model wall . . . 93
7.4 Theory . . . 94
7.5 Experiments . . . 97
7.6 Related work . . . 101
7.7 Limitations and future work . . . 102

IV Closing **103**

8 Conclusion **104**
8.1 The two sub-programs, side by side . . . 104
8.2 Smoothing by splitting . . . 105
8.3 The physics inspiration, in retrospect . . . 105
8.4 Open problems . . . 106
8.5 Closing . . . 107

V Appendices **108**

A Appendix to Chapter 3 (Manifold Flattening) **108**
A.1 Differential-geometric background . . . 108
A.2 Proofs . . . 110
A.3 Convergence of the convexification flow: status and proof program . . . 112
A.4 Time complexity . . . 113
A.5 Additional experiments . . . 114

B Appendix to Chapter 5 (Q-Score Matching) **116**
B.1 Proof of the policy-gradient expansion . . . 116
B.2 Proof of Theorem 5.2 (deterministic optimality) . . . 118
B.3 Proof of Theorem 5.4 (stochastic optimality) . . . 122
B.4 The Boltzmann fixed-point reading . . . 123
B.5 The algorithm in full . . . 124
B.6 Additional experiments . . . 124
B.7 Hyperparameters and benchmark details . . . 125

C Appendix to Chapter 6 (Onsager–Machlup Interpolation) **127**
C.1 Score-based training objectives . . . 127
C.2 Derivation of the OM action . . . 130
C.3 Geodesics and energy-minimizing paths . . . 133
C.4 Gaussian-blur norm . . . 133
C.5 Implementation details: Müller-Brown experiments . . . 134
C.6 Implementation details: alanine dipeptide . . . 136
C.7 Implementation details: fast-folding proteins . . . 137
C.8 Implementation details: tetrapeptides . . . 139
C.9 All-atom proteins with a classical force field . . . 140
C.10 Hutchinson trace estimator and τ_{opt} selection . . . 141
C.11 Implementation details: image interpolation . . . 142

D Appendix to Chapter 7 (GRASP) **155**
D.1 Linear-convergence theory in the convex setting . . . 155

D.2 Gaussian noise regularization 157
D.3 Connection to Boltzmann sampling 158
D.4 State-gradient-free dynamics 158
D.5 Additional ablations 162
D.6 Additional figures 163
D.7 Hyperparameters and benchmark details 163

Bibliography **165**

List of Figures

3.1 Idealized interpolation through manifold flattening. 15
3.2 Convex hull vs. manifold. 17
3.3 FlatNet layers. 19
3.4 One step of the FlatNet construction. 20
3.5 FlatNet iteration snapshots on a 3/4-circle curve. 27
3.6 FlatNet on a noisy sine wave. 30
3.7 FlatNet vs. VAE family on a 1D GP manifold in $\mathbb{R}^2$. 35
3.8 FlatNet vs. VAE family on a 2D GP manifold in $\mathbb{R}^3$. 36
3.9 Reconstruction error: FlatNet vs. FactorVAE on high-D GP manifolds. 37
3.10 Intrinsic dimension estimation. 37
3.11 FlatNet on a synthetic 2D MNIST manifold. 38
3.12 FlatNet reconstruction on MNIST. 38
3.13 MNIST interpolation: FlatNet vs. linear. 39
3.14 MNIST generation via FlatNet. 39
3.15 Counterexample: noisy closed circle. 39
3.16 Counterexample: lifted Swiss roll. 40

4.1 (TODO concrete experiment) The rough-landscape mental picture. 43

5.1 The diffusion policy as a path. 49
5.2 Conceptual illustration of QSM training. 53
5.3 Gridworld pedagogical reduction. 56
5.4 Multimodality preservation in the quadruped task. 58
5.5 Toy cartpole multimodality. 60
5.6 QSM vs. SAC and TD3 on DM Control. 60
5.7 Policy gradient applied directly to a diffusion-model policy. 61
5.8 QSM vs. Diffusion-QL on DM Control. 62

6.1 OM action optimization schematic. 69
6.2 OM optimization on the 2D Müller-Brown potential. 72
6.3 OM optimization on alanine dipeptide. 73
6.4 OM optimization on fast-folding proteins. 75
6.5 OM optimization on unseen tetrapeptide sequences. 76

6.6 OM image interpolations on CelebA-HQ. 82
6.7 OM vs. baselines on CelebA-HQ. 83
6.8 Effect of λ on MNIST interpolations. 83

7.1 Difficulty of the planning problem. 88
7.2 Serial vs. parallel planning over a learned world model. 90
7.3 Sensitivity of state gradients in a visual world model. 91
7.4 Virtual states learned through planning. 93
7.5 Cumulative success rates over time at fixed horizon. 98

A.1 Intrinsic distortion vs. FactorVAE. 114
A.2 FactorVAE intrinsic distortion across epochs. 115

B.1 Policy-gradient baseline on additional environments. 125
B.2 Diffusion depth scaling. 125

C.1 Müller-Brown OM optimization with analytical potential and flow matching. . . 135
C.2 Hutchinson trace estimator for the Laplacian. 135
C.3 Committor and rate estimation on Müller-Brown. 144
C.4 Empirical committor landscapes for fast-folding proteins. 145
C.5 OM transition paths vs. reference MD. 146
C.6 Diffusion model transition paths for fast-folding proteins. 147
C.7 Flow matching transition paths for fast-folding proteins. 148
C.8 Additional TIC-space transition paths. 148
C.9 Varying time horizons. 149
C.10 Training data and paths after removing intermediate committor values. 150
C.11 Sampled transition paths on held-out tetrapeptides. 151
C.12 All-atom transition paths with classical force fields. 152
C.13 τ_{opt} selection via cosine similarity to true forces. 152
C.14 Random MNIST pairs and their OM interpolations. 153
C.15 Latent vs. image-space Gaussian blur. 153
C.16 OM image-edit interpolation comparison. 154

D.1 Converged GRASP plans at Push-T $H = 80$. 164
D.2 Local-minimum failure under naïve dynamics relaxation. 164

List of Tables

6.1 OM vs. enhanced-sampling baselines on alanine dipeptide. 74

7.1 Open-loop planning results on long-range Push-T. Success rate (%) and median success time (seconds; successful trials only) across planning horizons. 500 trials per setting. Each cell reports *Success / Time*. 98
7.2 Short-term planning. Success rate (%) for Push-T, PointMaze, and WallSingle. 500 trials per setting. GRASP has a comparable success rate with consistently low completion time (Table 7.3). 99
7.3 Median completion times (seconds) for short-term experiments. 500 trials per setting. GRASP has consistently low completion times alongside comparable success rates (Table 7.2). 99
7.4 Ablations over GD sync steps, noise level for Langevin dynamics, and the detached-gradient approach. GD sync happens every 100 stochastic steps. Push-T, $H = 40$. Time reported is median over successful trials. 100
7.5 GRASP performance on latent-encoder and JEPA-style world models. Success rates (%). Horizon shows the executed action horizon (and the corresponding multi-step rollout length in parentheses, e.g. "15 (75)" means 15 executed actions with 75-step total rollout). Asterisk (*) indicates statistical significance over baselines; † uses CEM sync step (GRASP-CEM); ‡ anneals the goal-loss coefficient and noise variance between sync steps (GRASP-dagger). 100

C.1 Hyperparameters for OM action optimization on the Müller-Brown potential with diffusion. 136
C.2 Architecture and training hyperparameters for diffusion and flow-matching models on fast-folding proteins. 138
C.3 Hyperparameters for OM action optimization on fast-folding proteins with diffusion models. 138
C.4 Hyperparameters for OM action optimization on fast-folding proteins with flow-matching models. 139
C.5 Hyperparameters for OM action optimization on tetrapeptides with flow-matching models. 140
C.6 Hyperparameters for OM action optimization on all-atom proteins with a classical force field. 141

D.1 Stochastic GD baselines ablation, showing alternatives from our method for a stochastic gradient-based planner. None beat GRASP's performance (Table 7.4). Reported time is average over experiments; intervals are 95% CI. 163

Acknowledgments

This dissertation is the product of conversations, collaborations, and patience extended by many people over the years. I am especially grateful to my advisor, Aditi Krishnapriyan, whose technical taste and intellectual generosity shaped not only what I worked on but how I approach research. I am also grateful to my collaborators on the papers that became these chapters, for the discussions and shared frustrations that turn raw ideas into finished work. To the broader Berkeley AI community: I could not be happier with the community and support that I have.

Part I

Foundations

Chapter 1

Introduction

1.1 Path problems in modern deep learning

Many problems in deep learning are *path problems* rather than classification, regression, or sampling problems: given a starting configuration and a learned model of how configurations evolve, find a route to a target configuration.

A reinforcement-learning agent stands in a low-reward state and needs to reach a high-reward state under a learned critic. For a protein, the task is to understand the dynamical transition from one folded conformation to another. A latent code for an image lies near one mode of a generative model and we want to interpolate to another while respecting the model's learned geometry. A robot has a long-horizon plan in mind and must commit to a sequence of actions consistent with a learned world-model rollout.

Each of these problems shares a structure. We are given:

1. an initial state A and a target state B in some space $\mathcal{X}$;
2. a learned object, a score, a critic, a learned dynamics map, a representation, that encodes some relevant aspect of how trajectories in $\mathcal{X}$ should behave;
3. an implicit but vague goal of producing a trajectory $\gamma : [0, T] \to \mathcal{X}$ with $\gamma(0) = A$ and $\gamma(T) = B$ that is "good."

The problem as stated is under-determined. There are infinitely many candidate trajectories. The learned object does not directly specify which is correct: it gives gradient information, transition information, or smoothness information, but it does not on its own select a unique path. Worse, the regions of $\mathcal{X}$ that lie between the well-explored training neighborhoods of A and B are where the learned object is least trustworthy. A well-posed path problem demands a principled criterion, supplied by the learned object itself, for deciding which paths are good.

1.2 Variational reformulations

The thesis of this dissertation is that such a criterion is most naturally supplied by a variational reformulation. We frame the path problem as a minimization over a functional on path space:

$$\gamma^\star \;=\; \arg\min_{\gamma}\; \mathcal{L}[\gamma] \qquad \text{subject to } \gamma(0) = A,\; \gamma(T) = B. \tag{1.1}$$

The functional $\mathcal{L}$ is induced by the learned object: in some chapters it is induced by a learned critic, in others by a learned score, in others by a learned dynamics map. In every case it captures, in mathematical form, what we want from the path.

This is a familiar move in classical applied mathematics. Calculus of variations and optimal control take exactly this form. What is new is the source of the functional: it comes from a learned model, not from a Lagrangian written down by a physicist or an engineer. And the learned model brings with it complications, particularly roughness, that the classical theory does not assume.

Two sub-programs

There are two complementary ways to engage variational thinking with a learned model.

The first, which we call *sub-program A: variational reshaping of the representation*, treats the underlying space $\mathcal{X}$ as the variable. One writes down a functional whose argument is the embedded data manifold (or the encoder/decoder pair that parametrizes it), and descends that functional via a geometric flow. The data manifold is flattened toward convexity, so that downstream operations like interpolation become well-posed. Chapter 3 presents the dissertation's contribution to this sub-program: a convexification flow that drives an embedded data manifold $\mathcal{M}$ toward its convex hull, with the resulting flattened manifold supporting a minimal autoencoding pair.

The second, which we call *sub-program B: variational optimization of paths over a fixed (rough) representation*, accepts the underlying space as given, including all of its roughness, and brings the variational machinery to bear on paths through it. The functional is of the form (1.1), with $\mathcal{L}$ assembled from the learned model's outputs. The path is the variable; the space is fixed. Chapters 5, 6, and 7 present three instances of sub-program B, in which $\mathcal{L}$ is induced respectively by a learned critic (Q-score matching for diffusion reinforcement learning), a learned score (Onsager–Machlup interpolation for molecular and image-latent dynamics), and a learned dynamics map (parallel long-horizon planning over world models). Chapter 4 is a short interlude between the two sub-programs, arguing on empirical grounds that modern learned representations are rough enough to motivate sub-program B as a complement to sub-program A.

The two sub-programs are not in competition. They differ in what they optimize over and in what they take as fixed. In sub-program A, the path problem becomes a representation-learning problem: making the space well behaved makes the path problem easy. Sub-program B accepts the representation produced by an upstream training procedure and solves the

path problem on that space. Both responses are variational, which places them within a single, larger program.

1.3 Roughness of deep learned models

The central technical fact this dissertation contends with is the roughness of learned models. Classical optimal control assumes its dynamics maps are smooth, well-conditioned, and analytic enough that gradient-based optimization can be reasoned about in closed form. Classical manifold learning assumes its target manifolds are smooth, compact, low-dimensional embeddings. Modern learned models, particularly those produced by self-supervised training on high-dimensional data, routinely violate these assumptions in ways that have substantive algorithmic consequences.

A self-supervised image-feature model produces representations whose trajectories under smooth video are not smooth in feature space. Composed learned dynamics models accumulate jacobian singular values exponentially with rollout length (Bengio, Simard, et al., 1994). Image classifiers are notoriously sensitive to small input perturbations (Szegedy et al., 2013; I. J. Goodfellow et al., 2015); the adversarial-robustness literature is largely a documentation of how rough the gradients of large vision models are in their input spaces. These observations are the empirical character of the representations modern deep learning produces, and the rest of the dissertation takes them as given rather than trying to train them away.

Chapter 4 treats this in detail. For now, what matters is the structural response. Sub-program A handles roughness by removing it when possible: the convexification flow turns a rough manifold into a flat one, after which the resulting representation supports clean linear analysis. Sub-program B handles roughness by accommodating it: the functional $\mathcal{L}$ is constructed so that the optimization in (1.1) remains well-conditioned even when the learned model's local geometry is poor. The strategies for the second response form a recurring technical theme of Chapters 5–7, which we will call *smoothing-by-splitting*: in each case, a quantity that would otherwise need to be evaluated by back-propagating through a long composition of rough operations is replaced by a variational surrogate that depends only on local, well-conditioned ingredients.

1.4 Contributions and roadmap

The contributions of this dissertation can be summarized as follows.

- **A taxonomic clarification.** The path-problem framing, the distinction between sub-programs A and B, and the diagnosis of roughness as the technical obstacle that unifies both sub-programs.

- **A representation-reshaping algorithm.** Manifold flattening via a convexification flow, with a constructive realization as a layer-wise learned encoder-decoder pair, supported by a theorem identifying minimal autoencoding with flattening.
- **A path-space algorithm for diffusion reinforcement learning.** Q-score matching: a score-matching surrogate that aligns the denoising direction of a diffusion policy with the action gradient of a learned critic, yielding stable policy improvement that preserves multimodality.
- **A path-space algorithm for extracting dynamics from generative models.** Onsager–Machlup interpolation: a path-likelihood functional under an SDE derived from a diffusion model, used to produce interpolations through molecular conformational landscapes and image-latent spaces, with the diffusion time providing a continuous smoothness-vs-accuracy knob.
- **A path-space algorithm for long-horizon planning.** Lagrangian-relaxed parallel planning over learned world models, with a gradient-cut variant (GRASP) that is specifically designed to remain well-conditioned in the presence of rough visual representations, and an adversarial-robustness reading of the failure modes that motivate the variant.

Roadmap. Chapter 2 develops the variational vocabulary used by the rest of the dissertation. Chapter 3 presents the manifold-flattening sub-program in full. Chapter 4 is a short interlude that makes the empirical case for roughness as the technical obstacle motivating the rest of the dissertation. Chapters 5, 6, and 7 present the three instances of sub-program B. Chapter 8 synthesizes the dissertation's central claims and discusses open problems.

Each technical chapter is based on a published or in-submission paper. Chapter 3 is based on Psenka, Pai, et al., 2024. Chapter 5 is based on Psenka and Abbeel, 2024. Chapter 6 is based on two papers, one on molecular dynamics Psenka et al., 2025a and one on image-latent interpolation Psenka et al., 2025b. Chapter 7 is based on Psenka, LeCun, et al., 2026. Where the chapter's content closely follows the paper, this is noted at the chapter opening; in every case the chapter has been re-framed, slimmed, or re-derived to fit the dissertation's argument.

Chapter 2

The Variational Principle, from Physics to Learned Models

This chapter develops the mathematical vocabulary that the rest of the dissertation uses. It is not a textbook treatment of calculus of variations or of optimal control; competent treatments of both subjects exist, and the reader looking for one should consult I. M. Gelfand and Fomin, 1963; Kirk, 2004. The goal here is to set out the minimum machinery needed to read the technical chapters. One fact to keep in mind throughout: the learned objects we will plug into these variational problems are not the smooth analytic vector fields of classical mechanics, and their roughness will need to be confronted. The chapter ends by cataloging exactly what kinds of learned objects appear later and what classical assumptions each one violates.

2.1 Calculus of variations, in brief

The basic object of the calculus of variations is a functional, a real-valued map whose input is itself a function. The most familiar form is

$$\mathcal{S}[\gamma] \;=\; \int_0^T L\big(t, \gamma(t), \dot{\gamma}(t)\big)\,\mathrm{d}t, \tag{2.1}$$

where $\gamma : [0, T] \to \mathbb{R}^n$ is a curve in some configuration space and $L : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a Lagrangian. A typical variational problem is to find the curve γ that minimizes (or stationarizes) $\mathcal{S}$ subject to boundary conditions $\gamma(0) = A$ and $\gamma(T) = B$.

The first variation of $\mathcal{S}$ with respect to a perturbation $\eta : [0, T] \to \mathbb{R}^n$ vanishing at the endpoints is computed by Taylor expansion:

$$\delta\mathcal{S}[\gamma; \eta] \;=\; \int_0^T \Big[\partial_x L\big(t, \gamma, \dot{\gamma}\big) \cdot \eta + \partial_{\dot{x}} L\big(t, \gamma, \dot{\gamma}\big) \cdot \dot{\eta}\Big]\,\mathrm{d}t. \tag{2.2}$$

Integrating the second term by parts and using $\eta(0) = \eta(T) = 0$, we obtain the necessary condition for γ to be a stationary point of $\mathcal{S}$: the *Euler–Lagrange equation*,

$$\frac{\partial L}{\partial x}\big(t, \gamma, \dot{\gamma}\big) - \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{x}}\big(t, \gamma, \dot{\gamma}\big) \;=\; 0. \tag{2.3}$$

A path $\gamma^\star$ minimizing $\mathcal{S}$ subject to fixed endpoints must satisfy (2.3); conversely, every solution of (2.3) with the given boundary conditions is a candidate minimizer.

This is the analytical framework. For the variational problems in this dissertation it will not generally be useful to solve the Euler–Lagrange equation in closed form: the functionals we encounter are induced by learned models, so L is itself the output of a neural network, and (2.3) is not analytically tractable. What we will use the framework for is structural: to recognize that what we are doing is variational; to identify the appropriate functional; and to understand the consequences of perturbing the functional or its inputs.

2.2 A few classical examples, briefly

A few short examples ground the vocabulary in physically familiar settings.

Fermat's principle. Geometric optics is reproduced by demanding that light propagating between two points in a medium of refractive index $n(x)$ extremize the optical path length $\mathcal{S}[\gamma] = \int_0^T n(\gamma(t))\,\|\dot{\gamma}(t)\|\,\mathrm{d}t$. The Euler–Lagrange equation produces Snell's law at interfaces.

Classical mechanics. A mechanical system with kinetic energy $T(\dot{\gamma})$ and potential energy $V(\gamma)$ has Lagrangian $L = T - V$, and the corresponding Euler–Lagrange equation is Newton's second law.

Geodesics. On a Riemannian manifold with metric g, the curve of shortest length between two points minimizes $\mathcal{S}[\gamma] = \int_0^T \sqrt{g_{ij}(\gamma)\dot{\gamma}^i\dot{\gamma}^j}\,\mathrm{d}t$. The Euler–Lagrange equation gives the geodesic equation, and solutions are the manifold's geodesics.

These examples are deliberately brief. The point is not to survey the calculus of variations but to recognize that the same template—write a functional, extremize it subject to boundary conditions—recovers radically different physical theories. The dissertation borrows the template, not the specific examples.

2.3 Learned dynamical objects

The variational problems in this dissertation are built around learned objects rather than analytic ones. For reference, here is the catalog of objects we will encounter in the technical chapters, with notes on what they are mathematically and what classical assumptions each one fails to satisfy.

Score functions. A diffusion model (Ho et al., 2020; Song and Ermon, 2019) trained on a distribution $p(x)$ produces, for each noise level σ, a score model $s_\theta(x, \sigma) \approx \nabla_x \log p_\sigma(x)$, where p_σ is the distribution at noise level σ. Mathematically, s_θ is a vector field on $\mathbb{R}^D \times \mathbb{R}_{>0}$. It is generally not the gradient of an analytic potential, and even at fixed σ it is not Lipschitz on the scales relevant to high-dimensional data. We use score functions extensively in Chapter 6, where they provide the drift of an SDE from which an Onsager–Machlup action can be derived.

Critic / value functions. In reinforcement learning (Richard S. Sutton and Barto, 2018), a critic $Q_\eta(s, a)$ estimates the expected cumulative return of taking action a in state s and following a fixed policy thereafter. The relevant variational ingredient is not Q_η itself but its action gradient $\nabla_a Q_\eta(s, a)$, which we will treat as a vector field on the action space. Like scores, action gradients of learned critics are not generally smooth in any useful sense, but they do encode local information about which actions are improving. Chapter 5 uses critic action gradients as the right-hand side of a score-matching surrogate.

Learned dynamics maps. A learned world model (Hafner, T. Lillicrap, et al., 2019; Assran et al., 2025) produces a one-step map $f_\theta(s_t, a_t) \approx s_{t+1}$. The relevant variational complication is compositional: the iterated map f_θ^T has jacobian whose singular values blow up (or vanish) exponentially in T, so even when f_θ itself is well-behaved, its long-horizon composition is not. Chapter 7 confronts this directly by Lagrangian-relaxing the rollout.

Learned representations. Self-supervised and generative pretraining produce encoders $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^p$ that map data to feature vectors. The induced data manifold $\mathbf{f}(\mathcal{X})$ is in some cases approximately smooth and low-dimensional, but in many cases is not; Chapter 4 discusses evidence to this effect. Chapter 3 works with representations under the assumption that the underlying data does live near a smooth flattenable manifold, and constructs a representation that explicitly enforces this.

2.4 Variational principle for the deep learning era

This dissertation's overall goal is to take the variational template that has been a unifying lens in physics and mechanics, and to modernize and apply it to a class of path problems in modern deep learning. The technical chapters that follow each take one of the learned objects of Section 2.3, identify the right functional whose stationary points correspond to good trajectories (or, in Chapter 3, good representations) and develop the reshaping needed to make the resulting optimization tractable when the learned ingredient is rough. The unification will not be as clean as in physics; learned objects are rough where Lagrangians are analytic, and the resulting functionals must be reshaped rather than directly extremized. But the structural promise is the same: recognizing path problems as instances of a single

variational pattern is productive, both for finding new algorithms and for understanding what otherwise-disparate algorithms share.

Part II

Variational Reshaping of the Representation

Chapter 3

Manifold Flattening as a Variational Program on Representations

This chapter is based on Psenka, Pai, et al., 2024, joint work with Druv Pai, Vishal Raman, Shankar Sastry, and Yi Ma. The paper is reframed here as the dissertation's contribution to sub-program A, a variational program acting directly on the representation. The bulk of the chapter follows the original paper closely; full proofs and additional experimental material are collected in Appendix A.

Autoencoding (Kramer, 1991) remains a central framework in deep learning and representation learning (Bengio, Courville, et al., 2013), where one aims to both encode data into a more compact representation and decode it back into the original data format. The framework appears throughout modern deep-learning applications, both as a tool to ease downstream learning (Rombach et al., 2022; K. He et al., 2022; Hafner, T. Lillicrap, et al., 2019) and as a design principle for the network itself (Ronneberger et al., 2015; Devlin et al., 2018).

In typical practice, the encoder and decoder are black-box neural networks whose architectural choices—latent dimension, depth, width, and regularization—are made by trial and error or by precedent. When one is building autoencoder models in a novel domain, this dependence on engineering knowledge can substantially slow development. This chapter develops a geometric alternative: a framework for automatically constructing an autoencoder model from a *geometric* model of the dataset, namely the manifold hypothesis (Vidal et al., 2005; Fefferman et al., 2016; B. C. Brown et al., 2022; Yu, K. H. R. Chan, et al., 2020). Given this assumption, along with mild regularity and sampling conditions, we will greedily construct the layers of an encoder/decoder pair in a forward fashion, in the spirit of K. Chan et al., 2022; Hinton, 2022, such that the learned encoding is as low-dimensional as possible in as few layers as possible. The construction is automatic in the sense that there are only a few scalar hyperparameters, each mathematically interpretable, and in practice each insensitive to its setting.

Our approach is through *manifold flattening*, or *manifold linearization*: if we can globally deform the data manifold into a linear structure, then simpler linear methods, e.g., PCA

for compression, can be applied on top of the constructed features. Motivation comes from common practice. Any successfully trained deep model whose last layer is linear (e.g., a multi-layer perceptron) has implicitly "linearized" its problem, in the sense that the deep model has learned a map all but the last layer such that the target problem is solvable by a linear map (the last layer). The chapter's approach makes this implicit linearization explicit and constructive.

Concretely, our approach focuses on *extrinsic curvature*. By iteratively constructing maps that model local curvature, the algorithm flattens and reconstructs the manifold, and these local flattenings are arranged so that the aggregate converges to a global flattening, akin to a classical *geometric flow*. This second-order treatment goes beyond the piecewise-linear, first-order character of many deep architectures (Yu, Buchanan, et al., 2024) and suggests how one might build a deep-learning model that explicitly represents extrinsic curvature.

Within the dissertation's argument, this construction is the entire body of sub-program A. The functional has the embedded data manifold itself as its argument, so the variational reformulation operates directly on the geometry. In the next chapter (Chapter 4) we examine when the manifold hypothesis remains tenable for representations produced by modern deep-learning systems, and use that examination to motivate the path-space sub-program B of Chapters 5–7.

3.1 Setup: minimal autoencoding

We begin by formalizing the autoencoding problem. Consider a *dataset* $\mathcal{X} = \{x_1, \dots, x_N\} \subset \mathbb{R}^D$. We assume the data is drawn near a low-dimensional embedded submanifold $\mathcal{M} \subset \mathbb{R}^D$ of intrinsic dimension $d \leq D$; we call $\mathcal{M}$ the *data manifold*. We aim to construct an *encoder* $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^p$ and a *decoder* $\mathbf{g} : \mathbb{R}^p \to \mathbb{R}^D$ that together form a *minimal autoencoding pair* in the following sense.

Definition 3.1. Given an embedded submanifold $\mathcal{M} \subset \mathbb{R}^D$ and a tolerance $\varepsilon > 0$, a pair of continuous functions $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^p$ and $\mathbf{g} : \mathbb{R}^p \to \mathbb{R}^D$ is an *ε-minimal autoencoding pair* for $\mathcal{M}$ if both of the following hold:

1. (Autoencoding.) $\mathbf{f}$ is injective on $\mathcal{M}$, and $\|\mathbf{g}(\mathbf{f}(x)) - x\|_2 \leq \varepsilon$ for all $x \in \mathcal{M}$.
2. (Minimality.) p is the smallest integer such that there exist continuous $\mathbf{f}, \mathbf{g}$ for which the autoencoding condition holds.

Computational considerations. Beyond producing a minimal pair, we want an algorithm that scales to modern data. Ideally:

1. *Linear time in the number of samples N*: $O(N)$. Datasets routinely reach millions of points, so even quadratic time in N becomes infeasible.

2. *Linear time in the ambient dimension D*: $O(D)$. Likewise, ambient dimensions in image and feature data reach into the thousands or millions.

3. *Polynomial time in the intrinsic dimension d*: $O(d^k)$ for some $k \in \mathbb{N}$. The intrinsic dimension is typically small; for example, MNIST (L. Deng, 2012) has ambient dimension $D = 784$, but a single class of digits is estimated to have intrinsic dimension $12 \leq d \leq 14$ (Hein and Audibert, 2005; Costa and Hero, 2006; Facco et al., 2017). Even so, manifold-learning methods in general require time exponential in d, which is infeasible for even modest d.

Assumptions on the manifold $\mathcal{M}$. We assume:

1. $\mathcal{M}$ is smooth. Standard regularity assumption.

2. $\mathcal{M}$ is compact. Satisfied for closed and bounded embedded submanifolds, which arise routinely in applications, e.g., 28×28 greyscale images embed into the bounded box $[0, 1]^{784} \subset \mathbb{R}^{784}$.

3. $\mathcal{M}$ is connected. We focus on a single class of data; multi-class extensions are an important direction for future work.

4. $\mathcal{M}$ is flattenable. This is more substantive and is defined precisely in Section 3.2 below. It is the most restrictive of the four assumptions, and we will argue in Section 3.7 and in Chapter 4 where it does and does not apply in practice.

Assumption on the dataset $\mathcal{X}$. The main assumption on $\mathcal{X}$ is geometric, quantifying how well the dataset and the manifold represent each other.

Definition 3.2. Given an embedded submanifold $\mathcal{M} \subset \mathbb{R}^D$, a finite dataset $\mathcal{X} = \{x_1, \ldots, x_N\} \subset \mathbb{R}^D$, and scalars $\varepsilon, \delta > 0$, we say $\mathcal{X}$ is (ε, δ)-*faithful* to $\mathcal{M}$ if both of the following hold:

1. ($\mathcal{M}$ represents $\mathcal{X}$.) $\inf_{x \in \mathcal{M}} \|x - x_i\|_2 \leq \varepsilon$ for all $i \in \{1, \ldots, N\}$.

2. ($\mathcal{X}$ represents $\mathcal{M}$.) $\min_{i \in \{1,\ldots,N\}} \|x - x_i\|_2 \leq \delta$ for all $x \in \mathcal{M}$.

Equivalently, $\mathcal{M}$ is an ε-cover of $\mathcal{X}$ and $\mathcal{X}$ is a δ-cover of $\mathcal{M}$.

Because $\mathcal{M}$ is compact, for every $\varepsilon, \delta > 0$, there exists a finite (ε, δ)-faithful dataset for $\mathcal{M}$. Unlike many treatments in statistical learning theory, we do not make explicit distributional assumptions, e.g., i.i.d. sampling, on $\mathcal{X}$.

Putting it together. With these definitions, the algorithmic target is:

Definition 3.3. Given an embedded submanifold $\mathcal{M} \subset \mathbb{R}^D$ of intrinsic dimension d and a faithful dataset $\mathcal{X} \subset \mathbb{R}^D$, an algorithm *efficiently computes a minimal autoencoding pair for* $\mathcal{M}$ if it computes a minimal autoencoding pair in $O(DNd^k)$ flops for some $k \in \mathbb{N}$, with access only to $\mathcal{X}$ (and not to $\mathcal{M}$ or d).

The rest of the chapter develops such an algorithm, and views it as the descent of a geometric energy functional whose argument is the embedded data manifold.

3.2 Manifold flattening creates minimal representations

The high-level strategy is: first remove the nonlinearity of the data manifold $\mathcal{M}$, then apply a PCA-type construction on the flattened manifold to obtain the minimal autoencoding. As a warm-up, if $\mathcal{M}$ were already an affine subspace, PCA would suffice directly: estimate the dimension from the spectrum, take the encoder to be orthogonal projection onto the top d principal components, and take the decoder to be the adjoint. Motivated by this, we ask: when can the curved $\mathcal{M}$ be smoothly deformed to an affine subspace? And when it can, does that deformation, together with PCA, give the minimal autoencoding?

The classical notion of extrinsic flatness is captured by the second fundamental form $\mathrm{I\!I}$ (Definition A.3 in Appendix A.1).

Definition 3.4. Let $\mathcal{M} \subset \mathbb{R}^D$ be a smooth embedded submanifold of dimension d with second fundamental form $\mathrm{I\!I}$. We say $\mathcal{M}$ is *flat* if $\mathrm{I\!I}_{x_0}(u, v) = 0$ for all $x_0 \in \mathcal{M}$ and all $u, v \in T_{x_0}\mathcal{M}$.

A manifold is flat exactly when it has no extrinsic curvature. The manifold need not start flat; what matters is whether some smooth map $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^D$ flattens it, so we work with the image $\mathbf{f}(\mathcal{M}) = \{\mathbf{f}(x) \mid x \in \mathcal{M}\}$ and ask whether this image is flat.

Definition 3.5. Let $\mathcal{M} \subset \mathbb{R}^D$ be a compact, connected, smooth embedded submanifold of dimension d. We say $\mathcal{M}$ is *flattenable* if there exist smooth maps $\mathbf{f}, \mathbf{g} : \mathbb{R}^D \to \mathbb{R}^D$ such that:

1. $\mathbf{g}(\mathbf{f}(x)) = x$ for every $x \in \mathcal{M}$, and
2. $\mathbf{f}(\mathcal{M})$ is a convex subset of a d-dimensional affine subspace of $\mathbb{R}^D$.

In this case, we say $(\mathbf{f}, \mathbf{g})$ *flatten and reconstruct* $\mathcal{M}$. Condition 1 makes $\mathbf{f}$ injective on $\mathcal{M}$; condition 2 is the property the convexification flow of Section 3.3 promotes, and by Part 1 of Theorem 3.6 it implies flatness.

The chapter's main equivalence is the following.

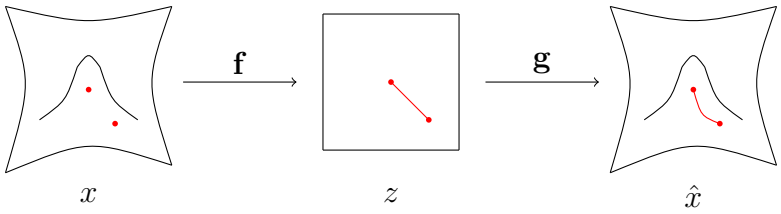


Figure 3.1: Idealized interpolation through manifold flattening on a 2D manifold in $\mathbb{R}^3$. To interpolate two points on the data manifold, map them through the flattening map $\mathbf{f}$ to the flattened space, take their convex interpolants, and map them back via the reconstruction map $\mathbf{g}$.

Theorem 3.6 (Autoencoding–flattening equivalence)**.** *Let $\mathcal{M} \subset \mathbb{R}^D$ be a flattenable submanifold of dimension d, flattened and reconstructed by smooth maps $\mathbf{f}, \mathbf{g} : \mathbb{R}^D \to \mathbb{R}^D$. Then:*

1. *$\mathbf{f}(\mathcal{M})$ is flat.*
2. *$\mathbf{f}(\mathcal{M})$ is contained in a unique affine subspace of dimension d.*
3. *Let $U \in \mathbb{R}^{D\times d}$ be an orthonormal basis for that subspace, stacked as a matrix, and let $z_0 \in \mathbf{f}(\mathcal{M})$. Then $\mathbf{f}'(x) = U^\top(\mathbf{f}(x) - z_0)$ and $\mathbf{g}'(z) = \mathbf{g}(Uz + z_0)$ form a minimal autoencoding pair (Definition 3.1) for $\mathcal{M}$ with $\varepsilon = 0$. If $\mathbf{f}, \mathbf{g}$ reconstruct only approximately, with $\max_{x\in\mathcal{M}} \|\mathbf{g}(\mathbf{f}(x)) - x\|_2 \le \varepsilon$, then $\mathbf{f}', \mathbf{g}'$ form an ε-minimal autoencoding pair.*

The proof relies on standard differential-geometric facts and is given in Appendix A.2. The content of the theorem is that *finding flattening and reconstruction maps is equivalent to finding a minimal autoencoding*: the algorithmic target reduces from an ill-defined optimization over abstract function classes to a concrete geometric problem of smoothly deforming $\mathbb{R}^D$ to flatten the data manifold. A first visualization of the resulting interpolation behavior is in Figure 3.1.

A note on the flattenable assumption. Under flattenability the converse of Theorem 3.6 also holds: any lossless smooth autoencoding pair generates a flattening. We argue this is a reasonable assumption for data manifolds in the following sense: for any continuous autoencoding pair $\mathbf{f}, \mathbf{g}$ (e.g., a neural-network encoder/decoder pair), one of the following typically occurs:

1. $\mathbf{f}(\mathcal{M})$ is a convex set, in which case $\mathcal{M}$ is flattenable; or
2. $\mathbf{f}(\mathcal{M})$ is not convex, in which case samples generated via $\mathbf{g}(z)$ for $z \sim \mathcal{N}(0, I)$ fall outside $\mathcal{M}$ with high probability.

The second outcome is incompatible with the standard variational-autoencoder sampling procedure: manifolds amenable to autoencoding via VAEs are essentially required to be

flattenable. The flattenable assumption is therefore already implicit whenever VAE-style sampling is used.

A variational reading. Theorem 3.6 identifies the goal — a convex, hence flat, $\mathbf{f}(\mathcal{M})$ — but does not give a procedure for achieving it. Consider the geometric energy functional

$$\mathcal{E}[\mathcal{M}] \;=\; \mathcal{H}^{d'}\big(\mathrm{conv}(\mathcal{M}) \setminus \mathcal{M}\big), \qquad d' := \dim \mathrm{aff}\big(\mathrm{conv}(\mathcal{M})\big), \tag{3.1}$$

the d'-dimensional Hausdorff measure, taken in the affine hull of $\mathrm{conv}(\mathcal{M})$, of the gap between $\mathcal{M}$ and its convex hull. This functional measures the *non-convexity* of $\mathcal{M}$: $\mathcal{E}[\mathcal{M}] = 0$ exactly when $\mathcal{M} = \mathrm{conv}(\mathcal{M})$, i.e. when $\mathcal{M}$ is convex, and $\mathcal{E}[\mathcal{M}] > 0$ otherwise. By Part 1 of Theorem 3.6, a convex d-dimensional set is flat, so driving $\mathcal{E}$ to zero flattens $\mathcal{M}$. A natural algorithmic strategy is therefore to descend $\mathcal{E}$: construct a one-parameter family $\mathcal{M}(t)$ of embedded submanifolds with $\mathcal{M}(0) = \mathcal{M}$ and $\mathcal{M}(t)$ evolving so as to decrease $\mathcal{E}[\mathcal{M}(t)]$. This is the setup of a geometric flow.

We caution the reader that the paper Psenka, Pai, et al., 2024 does not prove that the flow we construct below is the gradient flow of $\mathcal{E}$ in any technical sense; it defines the flow directly via its velocity field. The reading of that flow as a descent on $\mathcal{E}$ is the natural way to place the chapter inside the dissertation's overall variational framing. The primary mathematical object is the flow itself, defined in the next section; the language of "descent on $\mathcal{E}$" is a useful intuition.

3.3 Geometric flows and the convexification flow

A common practice in differential geometry for manifold manipulation is to define a differential equation and study the evolution of $\mathcal{M}$ under the corresponding dynamics. These equations are called *geometric evolution equations* or *geometric flows*; they evolve complicated manifolds into simpler, more uniform ones and underlie several major theorems, including the uniformization theorem.

The most well-known geometric flow is *Ricci flow* (Hamilton, 1982), which acts on the intrinsic metric. For our problem this is ill-suited: it is an open problem whether a realization of Ricci flow exists in the embedding space (Coll et al., 2020). There are also flows that minimize extrinsic curvature, such as *curve-shortening flow* (Abresch and Langer, 1986) and *mean curvature flow* (Huisken, 1984). Both have rich theory, but each requires restrictive settings: curve-shortening flow is only defined on curves ($d = 1$), and mean curvature flow on hypersurfaces ($d = D - 1$). Neither covers the high-codimension regime typical of data manifolds.

We introduce a geometric flow that is well-defined on general embedded submanifolds of Euclidean space and that flattens the input manifold $\mathcal{M}$.

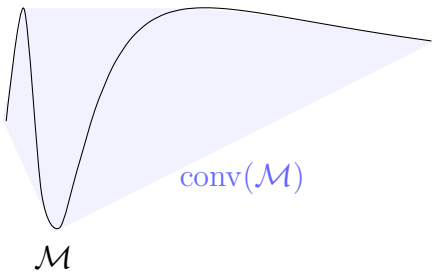


Figure 3.2: The convex hull of a manifold $\mathcal{M}$ (blue) versus the manifold itself. If there is no difference between $\mathrm{conv}(\mathcal{M})$ and $\mathcal{M}$, the manifold is convex and therefore flat. Local flattenings push $\mathcal{M}$ into its convex hull.

Convexification flow

Recall from Theorem 3.6 that flatness and convexity are intimately connected. Our flow accordingly focuses on minimizing the gap between the convex hull of the manifold and the manifold itself. We call it the *convexification flow*.

To compress $\mathcal{M}$ until it becomes convex, the flow should push every point $x_0 \in \mathcal{M}$ toward the boundary $\partial\,\mathrm{conv}(\mathcal{M})$. However, $\mathrm{conv}(\mathcal{M})$ is challenging to compute in high dimensions (Erickson, 1996; Gale, 1963, Ch. 3), so the flow should rely only on local information that gives a direction from x_0 toward $\partial\,\mathrm{conv}(\mathcal{M})$.

Fix a smooth integrable kernel $\varphi : \mathbb{R}^D \to [0,1]$ centered at the base point x_0 (for instance $\varphi(x) = e^{-\lambda\|x-x_0\|_2^2}$, as used in Section 3.4), so that the average below depends on x_0 through φ. The *local average* of $\mathcal{M}$ at x_0 is

$$A_{\mathcal{M}}(x_0) \;:=\; \frac{\int_{\mathcal{M}} x\,\varphi(x)\,\mathrm{d}x}{\int_{\mathcal{M}} \varphi(x)\,\mathrm{d}x}. \tag{3.2}$$

$A_{\mathcal{M}}(x_0)$ does not in general lie on $\mathcal{M}$; when there is nonzero local curvature, it lies in $\mathrm{conv}(\mathcal{M}) \setminus \mathcal{M}$. Local averages therefore serve as a proxy for the direction toward the boundary of the convex hull (Figure 3.2).

Definition 3.7 (Unnormalized convexification flow)**.** Let $x_0 \in \mathcal{M}$. The trajectory $x : [0,\infty) \to \mathbb{R}^D$ under the unnormalized convexification flow is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = \bar{x}(t) - x(t), \tag{3.3}$$

$$x(0) = x_0, \tag{3.4}$$

where $\bar{x}(t) = A_{\mathcal{M}(t)}(x(t))$ and $\mathcal{M}(t) = \{x(t) \mid x(0) \in \mathcal{M}\}$ is the time-t image of $\mathcal{M}$ under the flow.

Fix $t \geq 0$ and suppose $\mathcal{M}(t)$ is convex. Then for $x(t)$ in the interior of $\mathcal{M}(t)$ we have $\bar{x}(t) = x(t)$; the flow is stationary. However, $\bar{x}(t) - x(t)$ need not vanish on $\partial\mathcal{M}(t)$, so the flow

continues to act on boundary points after the interior has stabilized. Like other unnormalized geometric flows, this evolves $\mathcal{M}(t)$ into a singularity as $t \to \infty$. We resolve this by projecting onto the normal bundle.

Normalized convexification flow

Inspecting (3.3), the singular behavior arises when $\dot{x}(t)$ has a nonzero tangential component along $\mathcal{M}(t)$. Restricting the velocity to the normal space $N_{x(t)}\mathcal{M}(t)$ prevents motion within $\mathcal{M}$ itself.

Definition 3.8 (Normalized convexification flow)**.** Let $x_0 \in \mathcal{M}$. The trajectory $x : [0, \infty) \to \mathbb{R}^D$ under the normalized convexification flow is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = P_{N_{x(t)}\mathcal{M}(t)}\big(\bar{x}(t) - x(t)\big), \tag{3.5}$$

$$x(0) = x_0, \tag{3.6}$$

where $P_{N_{x(t)}\mathcal{M}(t)}$ is orthogonal projection onto the normal space at $x(t)$, and $\bar{x}(t)$ and $\mathcal{M}(t)$ are defined as before.

Two properties are worth noting. First, the velocity vanishes whenever $\mathcal{M}(t)$ is locally flat at $x(t)$: in that case $\bar{x}(t) - x(t)$ lies tangent to $\mathcal{M}(t)$ and is projected out. Second, motion is always normal to the current manifold, which avoids the purely tangential reparameterization collapse that drives the unnormalized flow toward a singularity. The normalized flow does not entirely avoid singularities either, but we will see that careful discretization handles those that remain.

Discretizing the normalized convexification flow

We now show that a first-order discretization of the normalized convexification flow recovers a local affine projection, which we will use directly as the building block of the FlatNet construction in Section 3.4. Fix $h > 0$ as the discretization interval and write $x[k] = x(kh)$, $\bar{x}[k] = \bar{x}(kh)$, $\mathcal{M}[k] = \mathcal{M}(kh)$. A first-order Taylor approximation of (3.5) gives

$$\begin{aligned} x[k+1] &\approx x[k] + h\, P_{N_{x[k]}\mathcal{M}[k]}\big(\bar{x}[k] - x[k]\big) \\ &= x[k] + h\,(\mathrm{id} - P_{T_{x[k]}\mathcal{M}[k]})\big(\bar{x}[k] - x[k]\big) \\ &= (1-h)x[k] + h\big(\bar{x}[k] + P_{T_{x[k]}\mathcal{M}[k]}(x[k] - \bar{x}[k])\big) \\ &= (1-h)x[k] + h\, P_{T_{x[k]}\mathcal{M}[k] + \bar{x}[k]}\big(x[k]\big). \end{aligned} \tag{3.7}$$

Reading (3.7), the discrete update is a convex combination of $x[k]$ and an affine projection of $x[k]$ onto the tangent space at $x[k]$ translated by the local average. Extending the update to a neighborhood of $x[k]$, with the projector held fixed at the central point, this is exactly the local flattening operation that FlatNet will perform layer by layer. The interpolation

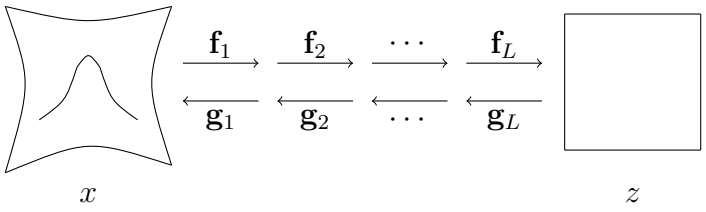


Figure 3.3: Construction of the encoder $\mathbf{f}_{\mathrm{CC}} = \mathbf{f}_L \circ \cdots \circ \mathbf{f}_1$ and decoder $\mathbf{g}_{\mathrm{CC}} = \mathbf{g}_1 \circ \cdots \circ \mathbf{g}_L$ as compositions of per-layer flattening and reconstruction maps.

weight h corresponds to a partition-of-unity weight $\varphi(x)$; both control how strongly the affine projection is applied at points near vs. far from the chosen base point.

The next section makes this construction explicit and discusses how the relevant geometric quantities, the tangent space and the second fundamental form, are estimated from finite samples of $\mathcal{M}$.

3.4 Flattening Networks (FlatNet)

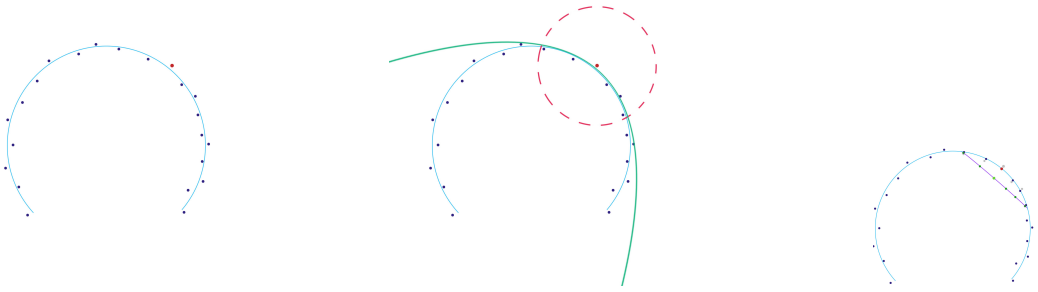
We now describe an algorithmic procedure for globally-defined flattening and reconstruction maps. Each pair is a composition of multiple simpler functions, akin to layers in a neural network (Figure 3.3). Accordingly we call the construction the *Flattening Network*, or *FlatNet*.

A large difference from a standard neural network is that the layers are constructed iteratively in a white-box process with no backpropagation. The design is qualitatively similar to other white-box constructions (K. Chan et al., 2022) but differs in the details.

We summarize the construction:

1. Initialize $\mathbf{f}_{\mathrm{CC}} = \mathbf{g}_{\mathrm{CC}} = \mathrm{id}_{\mathbb{R}^D}$.
2. For each layer $\ell \in \{1, \ldots, L\}$:
   a) Sample a point x_0 uniformly at random from $\mathcal{X}$.
   b) Fit a local quadratic model to a neighborhood of x_0, and compute a local flattening map $\mathbf{f}_{\mathrm{loc}} : \mathbb{R}^D \to \mathbb{R}^D$.
   c) Use a smooth partition of unity to glue $\mathbf{f}_{\mathrm{loc}}$ with the identity outside the neighborhood, giving a global flattening map $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^D$.
   d) Compute an analytic global approximate inverse $\mathbf{g} : \mathbb{R}^D \to \mathbb{R}^D$ with $\mathbf{g} \circ \mathbf{f} \approx \mathrm{id}_{\mathbb{R}^D}$.
   e) Compose with the running composite: $\mathbf{f}_{\mathrm{CC}} \leftarrow \mathbf{f} \circ \mathbf{f}_{\mathrm{CC}}$, $\mathbf{g}_{\mathrm{CC}} \leftarrow \mathbf{g}_{\mathrm{CC}} \circ \mathbf{g}$.
3. Return $(\mathbf{f}_{\mathrm{CC}}, \mathbf{g}_{\mathrm{CC}})$.

The following subsections develop each step.



(a) (b) (c)

Figure 3.4: One step of the FlatNet construction. **(a)** From the dataset (dots) drawn near a manifold, randomly choose a base point x_0 (red). **(b)** Find the largest radius around x_0 where a local quadratic model (green) fits the data well. **(c)** Use the model to flatten the patch (purple) and to reconstruct onto the local model. The process is repeated on the resulting manifold until it is flattened.

Local quadratic models for data manifolds

Since we aim to manipulate and reconstruct a dense manifold $\mathcal{M}$ from finite samples $\mathcal{X}$, we need a model that fits $\mathcal{M}$ locally. Most classical manifold-learning methods rely on the fact that, near a fixed point $x_0 \in \mathcal{M}$, the manifold is approximately linear for a small enough ball.

A purely linear model would suffice for flattening, but it discards information needed to reconstruct the nonlinearity afterward. So we go one order higher. The exponential map at x_0, $\exp_{x_0} : T_{x_0}\mathcal{M} \to \mathcal{M}$, is a local bijection from the d-dimensional tangent space to the full manifold. Hence, $\exp_{x_0}$ perfectly models the local structure of $\mathcal{M}$. Unfortunately $\exp_{x_0}$ is typically not analytically known and is hard to estimate from finite data. We replace it with its second-order Taylor expansion (Monera et al., 2014):

$$\exp_{x_0}(v) \;\approx\; x_0 + v + \tfrac{1}{2}\mathrm{I\!I}_{x_0}(v, v), \tag{3.8}$$

where $\mathrm{I\!I}_{x_0} : T_{x_0}\mathcal{M} \times T_{x_0}\mathcal{M} \to N_{x_0}\mathcal{M}$ is the second fundamental form of $\mathcal{M}$ at x_0. This expansion is the basis of all the local constructions below.

Proposed local flattening and reconstruction maps

We take the local flattening map $\mathbf{f}_{\text{loc}} : \mathbb{R}^D \to \mathbb{R}^D$ to be an affine projection:

$$\mathbf{f}_{\text{loc}}(x) \;=\; P_{S+x_c}\{x\} \;=\; P_S(x - x_c) + x_c, \tag{3.9}$$

where S is a linear subspace and $x_c \in \mathbb{R}^D$ is a fixed base point. There are two design choices: S and x_c. We pick S to keep $\mathbf{f}_{\text{loc}}$ locally invertible, and x_c to make the overall flattening process likely to converge.

Choosing S. Set $S = T_{x_0}\mathcal{M}$. If (3.8) holds, then for $v \in T_{x_0}\mathcal{M}$ we have

$$\begin{aligned}\mathbf{f}_{\text{loc}}(\exp_{x_0}(v)) &\approx P_{T_{x_0}\mathcal{M}+x_c}\{x_0 + v + \tfrac{1}{2}\mathrm{I\!I}_{x_0}(v,v)\} \\ &= \underbrace{P_{T_{x_0}\mathcal{M}}(v)}_{=v} + \tfrac{1}{2}\underbrace{P_{T_{x_0}\mathcal{M}}(\mathrm{I\!I}_{x_0}(v,v))}_{=0} + \underbrace{P_{T_{x_0}\mathcal{M}}(x_0 - x_c) + x_c}_{=P_{T_{x_0}\mathcal{M}+x_c}(x_0)} \\ &= v + \mathbf{f}_{\text{loc}}(x_0), \end{aligned} \tag{3.10}$$

using $P_{T_{x_0}\mathcal{M}}(v) = v$ for $v \in T_{x_0}\mathcal{M}$ and $P_{T_{x_0}\mathcal{M}}(\mathrm{I\!I}_{x_0}(v,v)) = 0$ because $\mathrm{I\!I}_{x_0}(v,v) \in N_{x_0}\mathcal{M}$. The local flattening map is therefore approximately invertible in a neighborhood $\mathcal{N}_{x_0}$ of x_0, with local reconstruction

$$\mathbf{g}_{\text{loc}}(z) \;=\; x_0 + (z - \mathbf{f}_{\text{loc}}(x_0)) + \tfrac{1}{2}\mathrm{I\!I}_{x_0}\big(z - \mathbf{f}_{\text{loc}}(x_0),\, z - \mathbf{f}_{\text{loc}}(x_0)\big). \tag{3.11}$$

Direct computation gives

$$\begin{aligned}\mathbf{g}_{\text{loc}}(\mathbf{f}_{\text{loc}}(\exp_{x_0}(v))) &\approx \mathbf{g}_{\text{loc}}(v + \mathbf{f}_{\text{loc}}(x_0)) \\ &= x_0 + v + \tfrac{1}{2}\mathrm{I\!I}_{x_0}(v,v) \\ &\approx \exp_{x_0}(v). \end{aligned} \tag{3.12}$$

Choosing $S = T_{x_0}\mathcal{M}$ is optimal in preserving the local metric at x_0, though intrinsic volume within the neighborhood can compress depending on the curvature at x_0. Flattening to the tangent space is the simplest verification that the local flattening is invertible.

Choosing x_c. The offset x_c induced in (3.10) is an important design choice: it sets the global direction of the iterated flattening process and therefore its convergence. We want x_c to (a) push the flattened manifold closer to a globally flat state and (b) lie close to the dataset $\mathcal{X}$ so that the resulting global flattening is reasonably smooth.

The certificate of convergence is the convex hull. By Theorem 3.6, a convex submanifold of Euclidean space is flat; conversely, any difference between $\text{conv}(\mathcal{M})$ and $\mathcal{M}$ indicates a direction in which to flatten toward the convex hull. For a probability measure P on $\mathcal{M}$, the extrinsic average $\int_{\mathcal{M}} x \,\mathrm{d}P(x)$ lies in $\text{conv}(\mathcal{M})$. Choosing the density of P with respect to the manifold measure to be proportional to a smooth function $\varphi : \mathbb{R}^D \to [0,1]$ with $\varphi(x) \approx 1$ on $\mathcal{N}_{x_0}$ and φ decaying rapidly outside, we set

$$x_c \;=\; \bar{x} \;:=\; \frac{\int_{\mathcal{M}} x\,\varphi(x)\,\mathrm{d}x}{\int_{\mathcal{M}} \varphi(x)\,\mathrm{d}x}. \tag{3.13}$$

By construction $x_c \in \text{conv}(\mathcal{M})$, while the localization of φ keeps x_c close to x_0.

To summarize: $S = T_{x_0}\mathcal{M}$ and $x_c = \bar{x}$ via (3.13), with φ to be specified next.

From local to global

To make $\mathbf{f}_{\text{loc}}$ globally well-defined, we glue it with the identity outside the neighborhood $\mathcal{N}_{x_0}$ via a smooth scalar partition-of-unity function $\varphi : \mathbb{R}^D \to [0,1]$ with $\varphi(x) \approx 1$ for x in (a closed subset of) $\mathcal{N}_{x_0}$ and $\varphi(x) \approx 0$ outside $\mathcal{N}_{x_0}$. Then the globally defined flattening map is

$$\mathbf{f}(x) \;\coloneqq\; \varphi(x)\mathbf{f}_{\text{loc}}(x) + (1-\varphi(x))x, \tag{3.14}$$

so that $\mathbf{f}(x) \approx \mathbf{f}_{\text{loc}}(x)$ on $\mathcal{N}_{x_0}$ and $\mathbf{f}(x) \approx x$ outside.

There are many candidates for φ. We require two practical properties:

1. φ should yield smooth global maps in (3.14). Sharp cusps (as in Figure 3.4c) would make further flattening and reconstruction unnecessarily difficult.
2. φ should allow computable inverses of (3.14). Local invertibility via $\mathbf{g}_{\text{loc}}$ does not automatically extend to a globally invertible $\mathbf{f}$; we need a closed-form (or efficiently computable) globally defined reconstruction.

To motivate the choice of φ, we attempt to compute an approximate inverse of $\mathbf{f}$. First, we simplify (3.14):

$$\begin{aligned}\mathbf{f}(x) &= \varphi(x)\big(P_{T_{x_0}\mathcal{M}}(x-x_c)+x_c\big) + (1-\varphi(x))x \\ &= P_{T_{x_0}\mathcal{M}}\{x\} + P_{N_{x_0}\mathcal{M}}\big\{\varphi(x)x_c + (1-\varphi(x))x\big\}.\end{aligned}$$

Geometrically: the tangential component of x is preserved, while the normal component is replaced by a convex combination of x_c and x with weight $\varphi(x)$. The following calculation suggests an approximate inverse:

$$\begin{aligned}&\mathbf{f}(x) + \varphi(x)\Big(x_0 - P_{T_{x_0}\mathcal{M}+x_c}(x_0) + \tfrac{1}{2}\mathrm{I\!I}_{x_0}\big(P_{T_{x_0}\mathcal{M}}(\mathbf{f}(x)-x_0), P_{T_{x_0}\mathcal{M}}(\mathbf{f}(x)-x_0)\big)\Big) \\ &= x + \varphi(x)\underbrace{\Big(-P_{N_{x_0}\mathcal{M}}(x-x_0) + \tfrac{1}{2}\mathrm{I\!I}_{x_0}\big(P_{T_{x_0}\mathcal{M}}(x-x_0), P_{T_{x_0}\mathcal{M}}(x-x_0)\big)\Big)}_{\approx 0} \qquad (3.15) \\ &\approx x, \qquad (3.16)\end{aligned}$$

where the approximation $P_{N_{x_0}\mathcal{M}}(x-x_0) \approx \frac{1}{2}\mathrm{I\!I}_{x_0}(P_{T_{x_0}\mathcal{M}}(x-x_0), P_{T_{x_0}\mathcal{M}}(x-x_0))$ follows from the definition of the second fundamental form.

We are therefore motivated to set $z = \mathbf{f}(x)$ and obtain the reconstruction

$$\mathbf{g}(z) \;\coloneqq\; z + \varphi(x)\Big(x_0 - P_{T_{x_0}\mathcal{M}+x_c}(x_0) + \tfrac{1}{2}\mathrm{I\!I}_{x_0}\big(P_{T_{x_0}\mathcal{M}}(z-x_0), P_{T_{x_0}\mathcal{M}}(z-x_0)\big)\Big). \tag{3.17}$$

But this is not a well-defined global map because the coefficient $\varphi(x)$ depends on $x \approx \mathbf{g}(z)$. The following lemma resolves this on the manifold.

Lemma 3.9 (Local invertibility of a flattening layer)**.** *Let $\lambda > 0$, let $\varphi(x) = e^{-\lambda\|x-x_0\|_2^2}$, and let $\mathbf{f}$ be as in (3.14). Suppose $\mathcal{M}$ is ε-quadratically modellable at x_0 within a radius $r \le r_{Q,\ell,\varepsilon}(x_0)$ (the quadratic-modellability radius of Section 3.5, for the layer ℓ under construction), so that within $B_r(x_0)$ the manifold is, up to error ε, the graph of $t \mapsto x_0 + t + \frac{1}{2}P_{N_{x_0}\mathcal{M}}\,\mathrm{I\!I}_{x_0}(t,t)$ over $t \in T_{x_0}\mathcal{M}$. Then $\mathbf{f}$ fixes the tangent coordinate, $\mathbf{f}$ restricted to $\mathcal{M} \cap B_r(x_0)$ is injective, and there is a smooth function φ^{-1}, defined on $\mathbf{f}(\mathcal{M} \cap B_r(x_0))$, such that $\varphi^{-1}(\mathbf{f}(x)) = \varphi(x)$ for all $x \in \mathcal{M} \cap B_r(x_0)$.*

The proof is in Appendix A.2; it is not apparent from the statement, but clear from the proof, that $\varphi^{-1}(z)$ is efficiently computable, requiring only the inversion of a smooth scalar function that is strictly monotone on the branch connected to the tangent plane (e.g., by the secant method initialized at $s = 0$).

With this choice of φ, the reconstruction is well defined on $\mathbf{f}(\mathcal{M})$ and extends smoothly to a neighborhood of it:

$$\mathbf{g}(z) \;\coloneqq\; z + \varphi^{-1}(z)\Big(x_0 - P_{T_{x_0}\mathcal{M}+x_c}(x_0) + \tfrac{1}{2}\mathrm{I\!I}_{x_0}\big(P_{T_{x_0}\mathcal{M}}(z-x_0), P_{T_{x_0}\mathcal{M}}(z-x_0)\big)\Big). \tag{3.18}$$

The Gaussian $\varphi(x) = e^{-\lambda\|x-x_0\|_2^2}$ is not strictly a partition of unity (it is positive everywhere rather than compactly supported), but in our use it serves the same purpose, with the added benefit of avoiding sharp cusps in the resulting manifold. The parameter λ controls the effective radius of φ: the relation $e^{-\lambda r^2} \le \varepsilon$ implies $r \ge O(\lambda^{-1/2})$, so we think of the radius of φ as $\lambda^{-1/2}$.

Finally, since φ already localizes around x_0, we use the same kernel for the local-mean weighting in (3.13): $x_c = \frac{\int_{\mathcal{M}} x\,\varphi(x)\,\mathrm{d}x}{\int_{\mathcal{M}} \varphi(x)\,\mathrm{d}x}$.

Estimation from finite samples

We now turn to estimation. Generic estimates are denoted with tildes ($\tilde{x}_c$, $\tilde{U}$); optimal estimates are denoted with hats ($\hat{x}_c$, $\hat{U}$).

Local average. The local average x_c is approximated by the sample average:

$$\hat{x}_c \;\coloneqq\; \frac{\sum_{i=1}^N x_i\,\varphi(x_i)}{\sum_{i=1}^N \varphi(x_i)}. \tag{3.19}$$

Tangent space and second fundamental form. The tangent space $T_{x_0}\mathcal{M}$ is harder to estimate from finite data. Local PCA (Hoppe et al., 1993; Oue, 1996) is a popular choice but relies on the manifold being approximately linear near x_0, which weakens when curvature is non-negligible. Improvements past local PCA exist (P. Zhang et al., 2011), but the joint estimation of tangent space and curvature is delicate.

We estimate $T_{x_0}\mathcal{M}$ by minimizing the autoencoding loss directly. By (3.15),

$$\|x_i - \mathbf{g}(\mathbf{f}(x_i))\|_2^2 = \varphi(x_i)^2 \left\| \tfrac{1}{2}\mathrm{I\!I}_{x_0}(P_{T_{x_0}\mathcal{M}}(x_i - x_0), P_{T_{x_0}\mathcal{M}}(x_i - x_0)) - P_{N_{x_0}\mathcal{M}}(x_i - x_0) \right\|_2^2.$$

Parameterize $T_{x_0}\mathcal{M}$ by a matrix $U \in \mathbb{R}^{D\times d}$ with orthonormal columns, so that $U^\top U = I_d$ and $UU^\top = P_{T_{x_0}\mathcal{M}}$, and parameterize $\frac{1}{2}\mathrm{I\!I}_{x_0}$ by its coordinates in the basis defined by the columns of U, stored as a 3-tensor $V \in \mathbb{R}^{D\times d\times d}$:

$$\tfrac{1}{2}\mathrm{I\!I}_{x_0}(w_1, w_2) \;=\; V(w_1, w_2) \;\coloneqq\; \sum_{j=1}^{d}\sum_{k=1}^{d} v_{jk}\, \langle u_j,\, w_1\rangle \langle u_k,\, w_2\rangle\,, \tag{3.20}$$

where $v_{jk} \in \mathbb{R}^D$ are the slices through the first coordinate of V. The autoencoding loss becomes

$$\mathcal{L}_\varphi(\tilde{U}, \tilde{V}) \;\coloneqq\; \frac{1}{N}\sum_{i=1}^{N} \varphi(x_i)^2 \left\| \tilde{V}\big(\tilde{U}\tilde{U}^\top(x_i - x_0), \tilde{U}\tilde{U}^\top(x_i - x_0)\big) - (I - \tilde{U}\tilde{U}^\top)(x_i - x_0) \right\|_2^2. \tag{3.21}$$

We define the global reconstruction problem and its optimal value as

$$\mathsf{Recon}(d, \varphi) \;\coloneqq\; \inf_{\tilde{U}\in\mathsf{O}(D,d),\, \tilde{V}\in\mathbb{R}^{D\times d\times d}} \mathcal{L}_\varphi(\tilde{U}, \tilde{V}). \tag{3.22}$$

The minimizers $(\hat{U}, \hat{V})$ translate into estimates $\hat{T}_{x_0}\mathcal{M}$ and $\hat{\mathrm{I\!I}}_{x_0}$.

For fixed $\tilde{U}$, the optimal $\hat{V}$ is a least-squares solution and is computed in closed form. The remaining optimization in $\hat{U}$ is over the Stiefel manifold $\mathsf{O}(D, d)$, an $O(Dd)$-dimensional non-convex problem with well-studied optimization (Fraikin et al., 2007; J. Li et al., 2020; Qu et al., n.d.; Zhai, Zitong Yang, et al., 2020; Zhai, Mehta, et al., n.d.). All associated complexities meet the $O(NDd^k)$ requirement of Definition 3.3.

One last subtlety: $\mathrm{I\!I}_{x_0}$ has codomain $N_{x_0}\mathcal{M}$ (a $(D-d)$-dimensional subspace), but the bilinear V has codomain $\mathbb{R}^D$. The following proposition (proof in Appendix A.2) shows that, under mild conditions, the least-squares solution $\hat{V}$ automatically has codomain orthogonal to the columns of $\tilde{U}$, recovering the correct normal-space structure.

Proposition 3.10. *Fix $\tilde{U} \in \mathsf{O}(D, d)$, and suppose $\varphi(x_i) > 0$ for all $x_i \in \mathcal{X}$. Then any minimizer $\hat{V}$ of $\inf_{\tilde{V}} \mathcal{L}_\varphi(\tilde{U}, \tilde{V})$ satisfies $\tilde{U}^\top\hat{V}(x_i - x_0, x_i - x_0) = 0$ for all $x_i \in \mathcal{X}$. Furthermore, if the matrix $A \in \mathbb{R}^{N\times\frac{1}{2}(d^2+d)}$, whose ith row contains the upper-triangular entries of the matrix $B_i \in \mathbb{R}^{d\times d}$ with $(B_i)_{jk} = \langle u_j,\, x_i - x_0\rangle \langle u_k,\, x_i - x_0\rangle$, has full column rank, then $\hat{V}$ is unique and $\tilde{U}^\top\hat{V}(w_1, w_2) = 0$ for all $w_1, w_2 \in \mathbb{R}^D$.*

Estimating the local dimension. The intrinsic dimension d is unknown a priori; rather than asking the user to specify it, FlatNet selects it adaptively. The principle is that d is the

smallest dimension such that the local quadratic model reconstructs $\mathcal{X}$ to within a desired precision $\varepsilon_{\text{dim}} > 0$:

$$\hat{d} \;=\; \min\{\tilde{d} \in \{1, \dots, D\} : \mathsf{Recon}(\tilde{d}, \varphi_0) \le \varepsilon_{\text{dim}}\}, \tag{3.23}$$

using a default kernel $\varphi_0(x) = e^{-\lambda_0 \|x - x_0\|_2^2}$ with hyperparameter $\lambda_0 > 0$. The interpretation of λ_0 is a default neighborhood radius, set so that we expect $\mathcal{X}$ to populate every intrinsic direction within that radius.

The optimization is in similar spirit to fixed-rank methods for low-rank matrix completion such as ALS (Takács and Tikk, 2012) and Riemannian methods (Vandereycken, 2013): we adaptively choose $\hat{d}$ by finding the smallest $\tilde{d}$ for which (3.22) reaches the threshold ε_{dim}. Since $d \ll D$ in practice, this can be solved by iteratively increasing $\tilde{d}$ from 1; further, since theoretically d should be the same at every $x_0 \in \mathcal{M}$, the search at the next iterate can be warm-started from the previous $\hat{d}$.

Learning a good partition of unity. Once $\hat{d}$ is fixed, the only remaining parameter is the scale λ in $\varphi_\lambda(x) = e^{-\lambda \|x - x_0\|_2^2}$. We choose λ to make each layer as "productive" as possible by maximizing the radius of the partition (and thus the spatial extent of each layer's effect):

$$\hat{\lambda} \;=\; \inf\{\tilde{\lambda} > 0 : \mathsf{Recon}(\hat{d}, \varphi_{\tilde{\lambda}}) \le \varepsilon_{\text{POU}}\}. \tag{3.24}$$

Defining $\ell(\tilde{\lambda}) \coloneqq \mathsf{Recon}(\hat{d}, \varphi_{\tilde{\lambda}})$ (a monotonic function of $\tilde{\lambda}$), the above reduces to a scalar root-find $\hat{\lambda} = \ell^{-1}(\varepsilon_{\text{POU}})$, efficiently computed by the secant method.

In practice, since $\mathcal{X}$ is finite, there is a smallest radius below which the autoencoding optimization in (3.22) becomes ill-conditioned. We therefore impose a cap $\lambda_{\max}$ and amend with a multiplicative coefficient:

$$\varphi(x) \;\coloneqq\; \min\Big(\alpha_{\max},\, \sqrt{\varepsilon_{\text{POU}} / \ell(\hat{\lambda})}\Big) \cdot e^{-\hat{\lambda} \|x - x_0\|_2^2}, \tag{3.25}$$

with $\hat{\lambda} = \min(\lambda_{\max}, \ell^{-1}(\varepsilon_{\text{POU}}))$ and $\alpha_{\max} \in (0, 1]$ a user-set parameter that prevents the formation of sharp cusps in the resulting manifold.

Full algorithm

The complete FlatNet construction is given in Algorithm 1.

A time-complexity analysis showing that Algorithm 1 fits the $O(NDd^k)$ requirement of Definition 3.3 is in Appendix A.4.

3.5 Algorithm intuition and convergence analysis

We collect here some intuition and an informal convergence analysis. A complete convergence theorem requires structure from the sample size and density, noise level, and curvature of

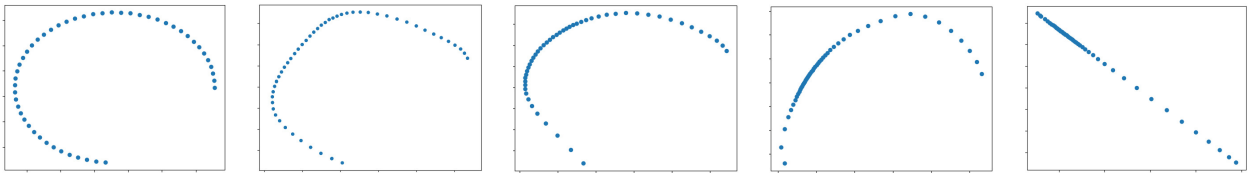
Algorithm 1 FlatNet construction.

function FLATNETCONSTRUCTION($\mathcal{X}, L, \varepsilon_{\text{dim}}, \lambda_0, \varepsilon_{\text{POU}}, \lambda_{\max}, \alpha_{\max}$)
  Initialize $\mathbf{f}_{\text{CC}}, \mathbf{g}_{\text{CC}} \leftarrow \text{id}_{\mathbb{R}^D}$
  for $\ell \in \{1, \dots, L\}$ **do**
    Sample a random point $x_0 \in \mathcal{X}$
    ▷ *Estimate the local dimension*
    Define $\varphi_0 : x \mapsto e^{-\lambda_0 \|x - x_0\|_2^2}$ ▷ (3.23)
    $\hat{d} \leftarrow \min\{\tilde{d} \in \{1, \dots, D\} : \mathsf{Recon}(\tilde{d}, \varphi_0) \le \varepsilon_{\text{dim}}\}$
    ▷ *Compute the partition of unity*
    For each $\tilde{\lambda}$, define $\varphi_{\tilde{\lambda}} : x \mapsto e^{-\tilde{\lambda}\|x - x_0\|_2^2}$
    Define $\ell : \tilde{\lambda} \mapsto \mathsf{Recon}(\hat{d}, \varphi_{\tilde{\lambda}})$
    $\hat{\lambda} \leftarrow \min(\lambda_{\max}, \ell^{-1}(\varepsilon_{\text{POU}}))$
    Define φ as in (3.25)
    ▷ *Estimate local average, tangent space, second fundamental form*
    $\hat{x}_c \leftarrow \sum_i x_i \varphi(x_i) / \sum_i \varphi(x_i)$
    $(\hat{U}, \hat{V}) \leftarrow$ minimizers of $\mathcal{L}_\varphi$ in (3.22)
    ▷ *Build the layer's flatten / reconstruct pair*
    Define $\mathbf{f}_{\text{loc}}(x) = \hat{U}\hat{U}^\top (x - \hat{x}_c) + \hat{x}_c$
    Define $\mathbf{f}(x) = \varphi(x)\mathbf{f}_{\text{loc}}(x) + (1 - \varphi(x))x$
    Compute φ^{-1} such that $\varphi^{-1}(\mathbf{f}(x)) = \varphi(x)$ ▷ Lemma 3.9
    Define $\mathbf{g}$ as in (3.18)
    ▷ *Compose with previous layers*
    $\mathbf{f}_{\text{CC}} \leftarrow \mathbf{f} \circ \mathbf{f}_{\text{CC}}$
    $\mathbf{g}_{\text{CC}} \leftarrow \mathbf{g}_{\text{CC}} \circ \mathbf{g}$
  end for
  return $(\mathbf{f}_{\text{CC}}, \mathbf{g}_{\text{CC}})$
end function

$\mathcal{M}$, and we leave such a theorem to future work. What we present here is enough to give a reader who works through the algorithm a sense of why it converges and what it looks like when a bad local minimum is hit.

Network diagram

Recall that FlatNet builds a flattening pair $\mathbf{f}_{\text{CC}}, \mathbf{g}_{\text{CC}}$ by repeatedly constructing and composing forward and backwards maps, $\mathbf{f}_{\text{CC}} = \mathbf{f}_L \circ \cdots \circ \mathbf{f}_1$ and $\mathbf{g}_{\text{CC}} = \mathbf{g}_1 \circ \cdots \circ \mathbf{g}_L$, with each layer pair



(a) $\mathcal{M} = \mathcal{M}_0$ (b) $\mathcal{M}_{20}$ (c) $\mathcal{M}_{40}$ (d) $\mathcal{M}_{60}$ (e) $\mathcal{M}_{80}$

Figure 3.5: Snapshots of $\mathcal{M}_\ell = \mathbf{f}_{1:\ell}(X)$ after ℓ layers have been constructed, for $\ell \in \{0, 20, 40, 60, 80\}$. The flattening is not isometric, so some intrinsic distortion is expected; the distortion is controlled by the requirement that each flattening be invertible.

$(\mathbf{f}_\ell, \mathbf{g}_\ell)$ taking the form

$$\begin{aligned}\mathbf{f}_\ell(x) &= \varphi(x)\big(UU^\top x + (I - UU^\top)x_c\big) + (1 - \varphi(x))x,\\ \mathbf{g}_\ell(z) &= z + \varphi^{-1}(z)\left((I - UU^\top)(x_0 - x_c) + \sum_{j,k} v_{jk}\,\langle u_j,\, z - x_0\rangle\,\langle u_k,\, z - x_0\rangle\right).\end{aligned} \tag{3.26}$$

The encoder layer takes the structural form of a multi-layer perceptron layer with a residual connection: a projection through $U^\top, U$, gated by the (Boltzmann-shaped) weight $\varphi(x)$, added back to the input. The decoder is the analogous second-order undo. Note that in generalizing this framework, the bilinear Π-style map can be replaced with any local interpretable model of the data, which suggests one route to white-box networks with explicitly second-order layers.

What does each iteration look like?

To build intuition we run FlatNet on data sampled from a three-quarters-circle curve and plot the output of $\mathbf{f}_{\mathrm{CC}}$ after $0, 20, 40, 60, 80$ layers (Figure 3.5). Let $\mathbf{f}_{1:\ell}$ denote the composition of the first ℓ layers; define $\mathcal{X}_\ell = \mathbf{f}_{1:\ell}(\mathcal{X})$ and $\mathcal{M}_\ell = \mathbf{f}_{1:\ell}(\mathcal{M})$. Note that the layers are built in a "closed-loop fashion" (Ma et al., 2022): the flattening direction at each layer (i.e., the estimated tangent space) is determined by repeatedly going back to the data through the corresponding reconstruction map and finding an optimal flattening direction.

Convergence analysis

A potential function for convergence. We measure how close $\mathcal{M}$ is to being flattened by the gap between $\mathcal{M}$ and its convex hull. The simplest scalar measure is the Hausdorff metric (Birsan and Tiba, 2005) between $\mathcal{M}$ and $\mathrm{conv}(\mathcal{M})$. Since $\mathcal{M} \subseteq \mathrm{conv}(\mathcal{M})$ and $\mathcal{M}$ is compact,

$$H(\mathcal{M}) \;\coloneqq\; \max_{x' \in \mathrm{conv}(\mathcal{M})} \min_{x \in \mathcal{M}} \|x - x'\|_2 \,. \tag{3.27}$$

Here $H(\mathcal{M}) = 0$ if and only if $\mathcal{M} = \text{conv}(\mathcal{M})$, i.e. $\mathcal{M}$ is convex; and a convex $\mathcal{M}$ is flat, with second fundamental form identically zero (Theorem 3.6). So H measures how non-convex $\mathcal{M}$ is.

However, H alone does not characterize convergence of Algorithm 1: there exist manifolds with $\mathcal{M} \subseteq T_x\mathcal{M}$ at all $x \in \mathcal{M}$ that nonetheless have nonzero extrinsic curvature. Such tangent-contained manifolds are fixed points of FlatNet in the sense that any constructed flattening map evaluates to the identity, but they are not convex or flat. We therefore shift to a local convexity measure: integrating H over neighborhoods of fixed radius $\eta > 0$,

$$\Omega_\eta(\mathcal{M}) \coloneqq \int_{\text{int}_\eta(\mathcal{M})} H(\mathcal{M} \cap B_\eta(x))\, \mathrm{d}x, \tag{3.28}$$

$$\text{int}_\eta(\mathcal{M}) \coloneqq \{x \in \mathcal{M} : \partial\mathcal{M} \cap B_\eta(x) = \emptyset\}. \tag{3.29}$$

If $H(\mathcal{M} \cap B_\eta(x)) = 0$ at each interior x, the tangent space cannot change locally, hence cannot change globally; but $\mathcal{M}$ is not required to be convex (the integral is restricted to the interior). A discretized version over $\mathcal{X}$ gives

$$\tilde{\Omega}_\eta(\mathcal{M}) = \sum_{i=1}^{N} H(\mathcal{M} \cap B_\eta(x_i)). \tag{3.30}$$

The role of η is clarified: if $\mathcal{X}$ is an η-cover of $\mathcal{M}$ (Definition 3.2), then $\tilde{\Omega}_\eta(\mathcal{M}) = 0$ implies $\mathcal{M} \subseteq T_x\mathcal{M}$ for all $x \in \mathcal{M}$, which implies a good convergence property for Algorithm 1.

Convergence via modelable radius. Let $r_{Q,\ell,\varepsilon}(x) \geq 0$ be the maximum radius r around x such that $\mathcal{M}_\ell$ is modelable by a quadratic of error at most ε inside the ball of radius r around x:

$$\max_{\substack{y \in \mathcal{M}_\ell \\ \|y-x\|_2 \leq r}} \max_{\substack{z \in \hat{Q} \\ \|z-x\|_2 \leq r}} \|y - z\|_2 \leq \varepsilon, \tag{3.31}$$

where $\hat{Q}$ is a local quadratic model (3.8) for some proposed tangent space and second fundamental form. We set $r_{Q,\ell,\varepsilon}(x) = \infty$ if no such upper bound exists.

The behavior of $r_{Q,\ell,\varepsilon}$ characterizes good versus bad fixed points of Algorithm 1:

1. Suppose $\mathcal{M}_\ell \subseteq T_x\mathcal{M}_\ell$ for some $\ell \in \mathbb{N}$, halting Algorithm 1 at a good fixed point. Then $r_{Q,\ell,\varepsilon}(\mathbf{f}_{1:\ell}(x_i)) = \infty$ for all i (setting $\hat{\mathrm{I\!I}}_{x_i} = 0$), so $(r_{Q,\ell,\varepsilon}(\mathbf{f}_{1:\ell}(x_i)))_\ell$ is unbounded.

2. Suppose on the contrary there exists $x_j \in \mathcal{X}$ for which $(r_{Q,\ell,\varepsilon}(\mathbf{f}_{1:\ell}(x_j)))_\ell$ is bounded. Then $\mathcal{M}_\ell \nsubseteq T_{x_j}\mathcal{M}_\ell$ for all ℓ, so if $\mathcal{M}_\ell$ reaches a fixed point at some ℓ, there must exist some x_j with $r_{Q,\ell,\varepsilon}(\mathbf{f}_{1:\ell}(x_j)) < \eta$, since otherwise there exists an atlas of $\mathcal{M}_\ell$ where each chart is completely flat, contradicting non-flatness.

A dichotomy on the trajectory $(\mathcal{M}_\ell)_\ell$, without assuming convergence:

1. If for some ℓ and $x_i \in \mathcal{X}$ we have $r_{Q,t,\varepsilon}(\mathbf{f}_{1:t}(x_i)) < \eta$ for all $t > \ell$, then Algorithm 1 either does not converge or converges to a bad local minimum with $\mathcal{M}_\ell \not\subseteq T_x\mathcal{M}_\ell$.

2. If for all ℓ and $x_i \in \mathcal{X}$ there exists $t > \ell$ with $r_{Q,t,\varepsilon}(\mathbf{f}_{1:t}(x_i)) \geq \eta$, then Algorithm 1 can always make some progress on every x_i.

Translating "progress" into a positive descent statement $\lim_{\ell\to\infty} \tilde{\Omega}(\mathcal{M}_\ell) = 0$ is the missing ingredient for a full convergence theorem. While a single iteration at x_i decreases the contribution from x_i to (3.30), the iteration may inadvertently increase $H(\mathcal{M}_\ell \cap B_\eta(x_j))$ for nearby points x_j; proving these cross-terms do not blow up requires careful global analysis. Appendix A.3 states, as conjectures, what a full convergence result would require for both the continuous flow and its FlatNet discretization, together with the proof program and the open step that currently blocks it.

3.6 Experiments

We evaluate FlatNet on synthetic low-dimensional manifold data, randomly generated high-dimensional Gaussian-process manifolds (Lahiri et al., 2016; Lawrence and Hyvärinen, 2005), and real-world image data. Code for FlatNet and the experiments is publicly available.[1] Three observations distinguish the method from existing alternatives:

1. *No need to select the intrinsic dimension d*: it is learned from the data via (3.23), in contrast to VAEs and many manifold-learning methods which require it as input.

2. *No need to select a stopping time*: the convexification flow empirically stabilizes once the learned representation is flat.

3. *Learned manifolds from noisy data are smooth*, as the figures below illustrate.

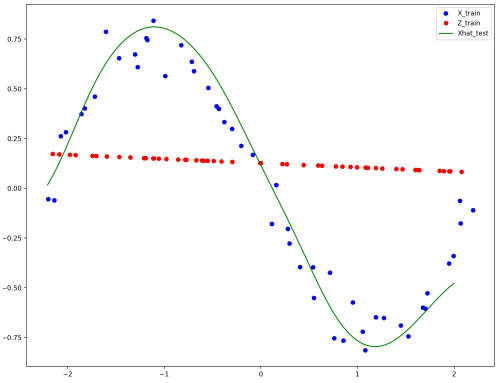
Low-dimensional manifold data

We test on three families:

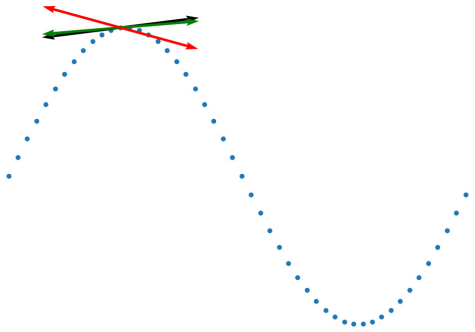
- Data sampled from a (noisy) sine wave.
- Data sampled from a Gaussian-process manifold of intrinsic dimension $d = 1$ in $\mathbb{R}^2$.
- Data sampled from a Gaussian-process manifold of $d = 2$ in $\mathbb{R}^3$.

Gaussian-process manifold generation. We lay out N vectors of intrinsic coordinates in a matrix $C \in \mathbb{R}^{d\times N}$ (uniformly drawn in our experiments). For each $i \in \{1, \dots, D\}$, we set up a correlation matrix $\Sigma_i \in \mathbb{R}^{N\times N}$ with entries $(\Sigma_i)_{pq} = \frac{L_i}{D} e^{-\rho_{pq}/2}$, where $L_i > 0$ is a

[1] https://github.com/michael-psenka/manifold-linearization



(a) Converged FlatNet result on noisy sine-wave data.



(b) Tangent-space estimation vs. local PCA on sine-wave data.

Figure 3.6: Results of FlatNet on data sampled from a noisy sine wave in $\mathbb{R}^2$. **(a)** Blue: training data X. Red: flattened data Z. Green: reconstruction $\hat{X}_{\text{test}}$ of interpolated test points. **(b)** Each arrow is a tangent space: black is ground truth, green is the FlatNet estimate, red is local PCA. FlatNet's joint estimation of tangent and curvature is robust to noise where local PCA degrades.

hyperparameter (set to 1) and $\rho_{pq} = \|c_p - c_q\|_2^2$. The ith row of X is sampled independently from $\mathcal{N}(0_N, \Sigma_i)$. Stacking, we obtain the data matrix $X \in \mathbb{R}^{D\times N}$.

Performance is evaluated as follows. Given data $X \in \mathbb{R}^{D\times N}$, plot features $Z = \mathbf{f}_{\text{CC}}(X)$ and reconstructions $\hat{X} = \mathbf{g}_{\text{CC}}(Z)$. Features should form an affine subspace; reconstructions should be close to X sample-wise. To probe generalization, linearly interpolate between pairs of features in Z to get $Z_{\text{test}} \in \mathbb{R}^{D\times M}$ for $M \gg N$, then reconstruct $\hat{X}_{\text{test}} = \mathbf{g}_{\text{CC}}(Z_{\text{test}})$. When the ground-truth tangent space is known (e.g., for the sine wave), we can also directly compare estimated tangent spaces.

In Figure 3.6, $N = 50$, $D = 2$, $d = 1$, $M = 5000$. Linearization, reconstruction, generalization, and tangent-space estimation are all demonstrated.

In Figure 3.7, we compare FlatNet against three VAE variants on a Gaussian-process manifold ($d = 1$, $D = 2$, $N = 50$): vanilla VAE (Kingma and Welling, 2013), β-VAE ($\beta = 4$) (Higgins et al., 2017), and FactorVAE ($\gamma = 30$) (Kim and Mnih, 2018). Each VAE encoder/decoder is a 5-layer MLP with hidden dimension 100, latent dimension $d = 1$, trained with Adam at learning rate 10^{-3} for 100 epochs. FlatNet is convincingly better at reconstruction and generalization on this low-dimensional regime.

Figure 3.8 shows the analogous comparison on a 2D GP manifold in $\mathbb{R}^3$ ($N = 50$, $D = 3$, $d = 2$, $M = 5000$). VAE latent dimensions are now $d = 2$. The qualitative observation persists: FlatNet recovers the manifold geometry where the VAE family does not.

High-dimensional manifold data

We test on random Gaussian-process manifolds in ambient $D = 100$ with various intrinsic dimensions d. Typically $N = 1000$. We measure two quantities: reconstruction quality and intrinsic-dimension-estimation accuracy. The latter is essentially trivial in low dimension but in high dimension serves as a rough certificate for the accuracy of each local flattening iteration.

In Figure 3.9 we evaluate average reconstruction error on $\mathcal{X} = \{x_1, \dots, x_N\}$,

$$\text{reconstruction error} \approx \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{x}_i\|_2^2, \quad \hat{x}_i \coloneqq \mathbf{g}_{\text{CC}}(\mathbf{f}_{\text{CC}}(x_i)), \tag{3.32}$$

across 3 trials, for each $d \in \{5, 10, 15, 20\}$. We compare against FactorVAE configured as in the low-dimensional experiments except with latent dimension d. (Vanilla VAE and β-VAE performed nearly identically to FactorVAE and are omitted for clarity.) FlatNet outperforms by a clear margin.

In Figure 3.10 we evaluate dimension estimation. Each iteration ℓ of the FlatNet construction estimates a local intrinsic dimension $\hat{d}_\ell$; we estimate the global intrinsic dimension as $\hat{d} \coloneqq \text{mode}(\hat{d}_\ell : \ell \in \{1, \dots, L\})$. We compare against MLE (Levina and Bickel, 2004) and TwoNN (Facco et al., 2017) on the same Gaussian-process manifolds. FlatNet is competitive with the state of the art, despite not being explicitly designed for dimension estimation.

Real-world imagery data

Real-world imagery such as MNIST tends to lie on high-curvature or non-differentiable pathological manifolds (Wakin et al., 2005). To make this tractable we use the (vectorized) Fourier transform of the image $\mathcal{F}\{x_i\}$ as the input to FlatNet rather than the image x_i itself, exploiting the easier smoothness of Fourier-domain transformations.

A first experiment uses a synthetic MNIST manifold: a single image rotated and translated to span a 2D parameter family embedded in pixel space ($D = 32 \times 32 = 1024$). Figure 3.11 shows the result. FlatNet automatically determines that the manifold is 2D and recovers features close to the intrinsic coordinates, in contrast to the top-2 PCA representation.

On the full MNIST data set, Figure 3.12 shows reconstruction, Figure 3.13 shows linear interpolation in the flattened feature space corresponding to semantic interpolation in image space, and Figure 3.14 shows random generation by sampling from the flattened latent affine subspace.

Low-dimensional counterexamples

We present two counterexamples to show the method's limitations and what failure looks like.

Closed circle. A noisy circle is a manifold that is not theoretically flattenable; any continuous autoencoder cannot flatten a closed loop. Figure 3.15 shows the behavior: FlatNet starts forming flat patches until they cusp, after which no further invertible flattening progress is possible and the algorithm halts.

Unaugmented Swiss roll. A standard Swiss roll is theoretically flattenable but is "crammed": it is a hypersurface ($d = D - 1$) with self-inward curling, so compression-based flattening folds it into itself. On the pure Swiss roll, FlatNet halts with little change.

However, appending a simple nonlinear feature $(x_1, x_2, x_3) \mapsto (x_1, x_2, x_3, x_1^2 + x_3^2)$ resolves the issue. FlatNet then both flattens efficiently and produces an approximate reconstruction (Figure 3.16). Reconstruction quality drops slightly — the extra coordinate is fit too — but the learned features correspond closely to the Swiss roll's intrinsic coordinates.

Since FlatNet is compressive in nature (derived from "compressing out" extrinsic curvature), these failure modes are intuitive: a manifold is more likely to "run into itself" when it lacks room to compress (e.g., hypersurfaces). We expect this to happen less for high-codimension data manifolds, and the Swiss-roll fix highlights a direction for future work: nonlinear lifting before flattening, to allow the manifold more room.

3.7 Related work

We survey the most direct alternatives to FlatNet on Definition 3.3, then methods that share spirit but solve different problems.

Variational autoencoders

The most well-known autoencoder family at present is the *variational autoencoder* (VAE) (Kingma and Welling, 2013), which has seen extensive empirical success in vision (Higgins et al., 2017; Van Den Oord, Vinyals, et al., 2017; Kim and Mnih, 2018) and, more recently, as a black-box backbone for nonlinear dimensionality reduction inside *diffusion* models (Vahdat et al., 2021; Rombach et al., 2022). Despite this success, VAEs have well-known shortcomings, including posterior collapse (Lucas et al., 2019). The most common issue across all VAE variants is that the encoder and decoder are black-box neural networks: important design choices are made via extensive trial and error or via prior knowledge from problem-class precedent, so applying VAEs in a novel domain can be expensive (Rezende and Viola, 2018). This chapter proposes a white-box alternative that performs nonlinear dimensionality reduction by deforming data structure to a lower-dimensional affine subspace, with fewer hyperparameters and greater robustness to their settings.

Manifold learning

Manifold-learning methods (Izenman, n.d.; N. Zheng et al., n.d.) and intrinsic manifold-learning methods (Schwartz and Talmon, n.d.; Lin and Hongbin Zha, n.d.; Qiao et al., n.d.; J. Zhang et al., n.d.) seek low-dimensional representations preserving geometric structure. Local-structure-preserving methods include LLE (Roweis and Saul, 2000) and variants, local tangent space alignment (Zhenyue Zhang and Hongyuan Zha, 2003), t-SNE (Van der Maaten and Hinton, 2008), and Laplacian eigenmaps (Belkin and Niyogi, 2003). Global-structure methods include Isomap (Tenenbaum et al., 2000) and UMAP (McInnes et al., 2018). These methods offer useful results on a broad class of manifolds but have practical limitations on real-world data (B. Li et al., 2019).

The first limitation is the lack of an explicit projection map between the original data and the low-dimensional representation. To project a new point one must re-run the algorithm on the full dataset plus the new point, an additional computational cost especially problematic for large datasets. RKHS extensions (J.-W. Xu et al., 2008; J.-B. Li et al., 2008) avoid this out-of-sample problem but require good kernel and parameter selection. Neural-network parameterizations (Jansen et al., 2017; Schmidt et al., 1992; C. P. Chen, 1996) reintroduce VAE-like training difficulties due to their black-box nature.

A second limitation is that these methods require the intrinsic dimension as input, which is hardly ever known a priori in practice. Intrinsic-dimension estimators (Campadelli et al., 2015; Levina and Bickel, 2004; Carter et al., 2009) can supply this at additional cost.

Distribution learners

GANs (I. Goodfellow et al., 2014), normalizing flows (Kobyzev et al., 2019), and diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song and Ermon, 2019) are effective at learning distributions (Arora et al., 2018) and at sampling from them (S. Chen et al., 2022), but they do not learn *representations* or *features* for individual data points in the sense of Definition 3.1. They solve a different problem.

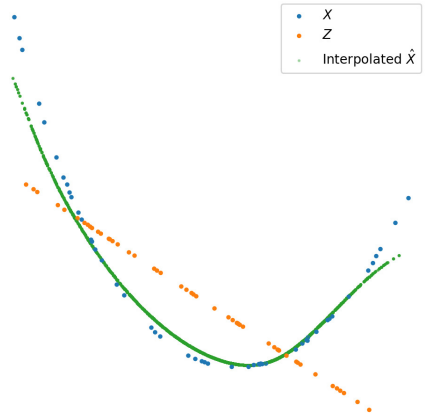
3.8 What this sub-program does and does not assume

The construction above is well defined and convergent under a specific set of assumptions on the data: that it lies near a smooth, compact, connected, flattenable embedded submanifold of $\mathbb{R}^D$. Smoothness, compactness, and connectedness are mild and routine. The flattenable assumption is more substantive. It says, roughly, that there exists a smooth deformation of $\mathbb{R}^D$ that turns the data manifold into a piece of an affine subspace. We argued in Section 3.2 that this assumption is reasonable for many naturally occurring data manifolds, since it is essentially implied by the existence of a smooth lossless autoencoding pair; any data set that admits an autoencoding solution in principle admits a flattening solution as well.

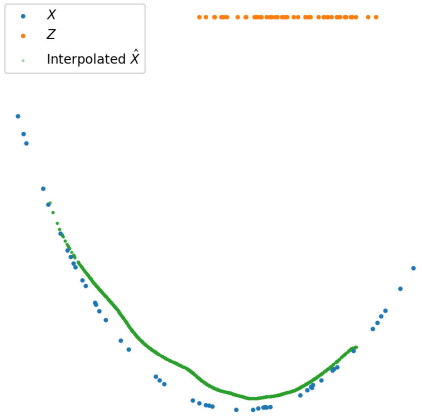
What this chapter *does not assume* is that the learned representation produced by some upstream training procedure satisfies any of these conditions. The chapter is about

constructing $(\mathbf{f}_{\mathrm{CC}}, \mathbf{g}_{\mathrm{CC}})$ from data under the manifold hypothesis; it is not about applying the convexification flow to, say, the latent space of a pretrained vision transformer or a generative diffusion model. For many such pretrained representations the assumptions of this chapter, in particular that the representation is a smooth, low-dimensional, flattenable submanifold, do not hold even approximately. That observation is the subject of the next chapter, the interlude that opens Part III. The interlude argues, with concrete examples, that the rough geometry of modern learned representations is real, and that the rest of the dissertation accordingly proceeds by accepting the rough geometry as a given and reshaping the functional instead of the space.

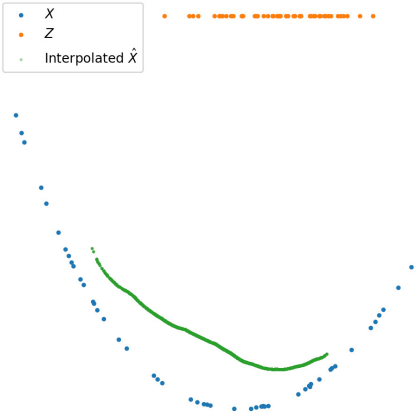
Sub-program A applies when the manifold hypothesis is recoverable, and the resulting flattened representation makes downstream tasks substantially easier. Representations produced by training procedures designed for other goals routinely violate that hypothesis. For those representations, the dissertation turns to the path-space program of Part III.



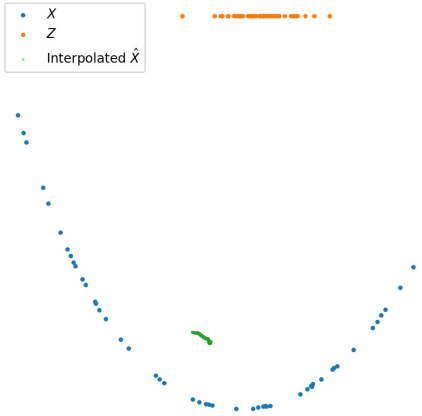
(a) FlatNet.



(b) VAE.



(c) β-VAE.



(d) FactorVAE.

Figure 3.7: FlatNet vs. VAE, β-VAE, and FactorVAE on $N = 50$ points sampled from a random 1D Gaussian-process manifold in $\mathbb{R}^2$. FlatNet is the only method that retains the smoothness of the original data manifold, an expected consequence of explicitly using the data geometry.



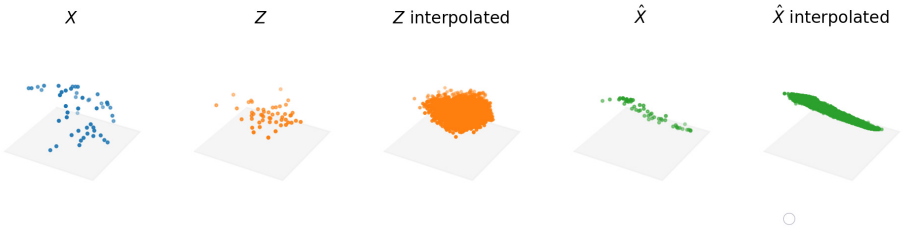
(a) FlatNet.



(b) VAE.



(c) β-VAE.



(d) FactorVAE.

Figure 3.8: FlatNet vs. VAE, β-VAE, and FactorVAE on $N = 50$ points sampled from a random 2D Gaussian-process manifold in $\mathbb{R}^3$. FlatNet is the only method here that recovers the underlying manifold geometry, and it generalizes accordingly. The FlatNet feature space is a 2D affine subspace, as desired; this would be visually apparent under rotation.

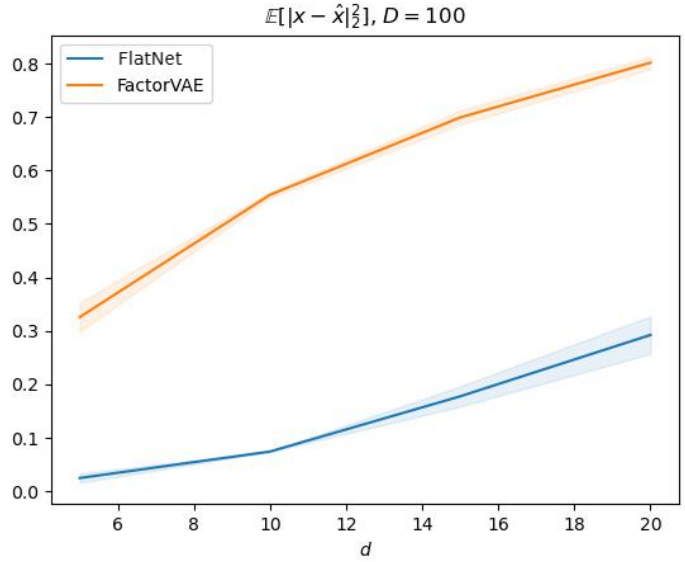


Figure 3.9: Reconstruction error of FlatNet and FactorVAE on $N = 1000$ points sampled from a random $d \in \{5, 10, 15, 20\}$-dimensional Gaussian-process manifold in $\mathbb{R}^{100}$. Error bars are ± 1 standard deviation across 3 trials.

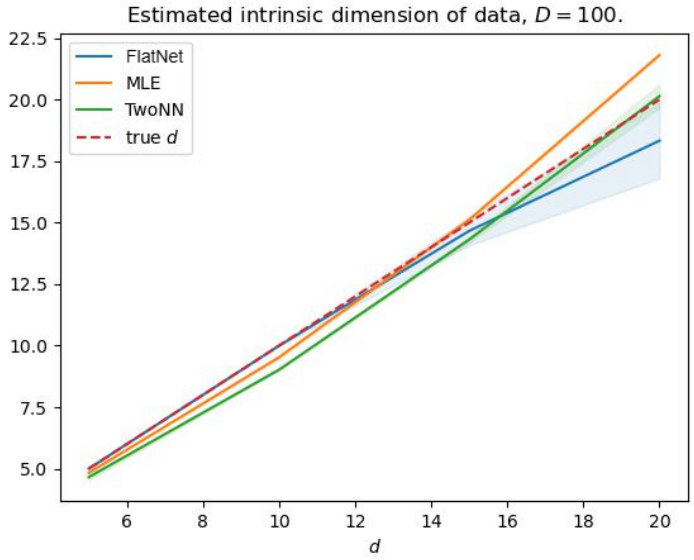
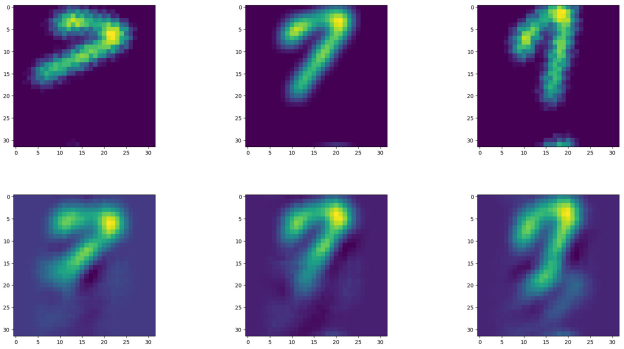
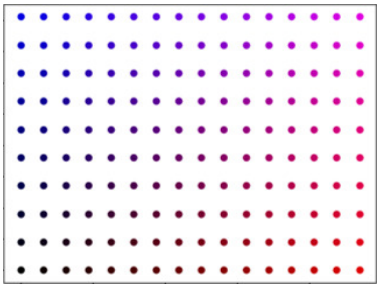


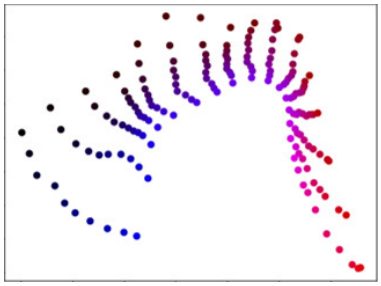
Figure 3.10: Global intrinsic-dimension estimation of FlatNet, MLE (Levina and Bickel, 2004), and TwoNN (Facco et al., 2017) on $N = 1000$ points sampled from a random $d \in \{5, 10, 15, 20\}$-dimensional Gaussian-process manifold in $\mathbb{R}^{100}$. Error bars are ± 1 standard deviation across 3 trials. Sample count is held fixed to mirror the relative data sparsity of real data manifolds; dimension estimates degrade more for larger d.



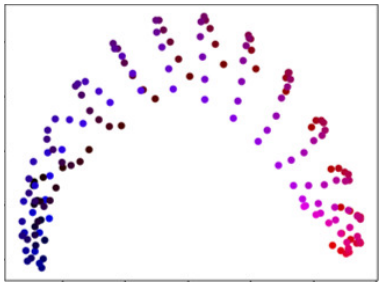
(a) Original rotated/translated images (top) vs. FlatNet reconstruction $\mathbf{g}_{\mathrm{CC}}(\mathbf{f}_{\mathrm{CC}}(x))$ (bottom).



(b) Intrinsic coordinates (y, θ).



(c) FlatNet representation.



(d) Top-2 PCA representation.

Figure 3.11: Results of FlatNet on a 2D manifold constructed from a single MNIST digit by rotation $\theta \in [-30°, 30°]$ and y-translation $\in [-7, 7]$ pixels. FlatNet automatically determines $\hat{d} = 2$. The learned representation is smoothly deformable into the intrinsic (y, θ) coordinates, while top-2 PCA produces a tangled representation that requires a more complex map to undo.

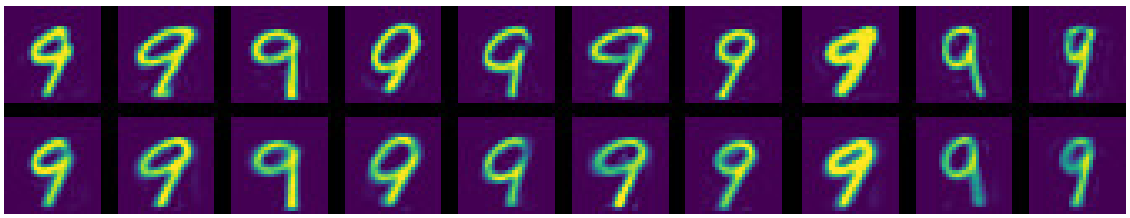


Figure 3.12: Original x_i (top) vs. reconstruction $\hat{x}_i := \mathcal{F}^{-1}\{\mathbf{g}_{\mathrm{CC}}(\mathbf{f}_{\mathrm{CC}}(\mathcal{F}\{x_i\}))\}$ (bottom) on MNIST. The reconstruction preserves digit identity and salient geometry; it is closer to an archetype than to a pixel-perfect copy. The encoder $\mathbf{f}_{\mathrm{CC}}$ appears to compress semantically inessential detail.

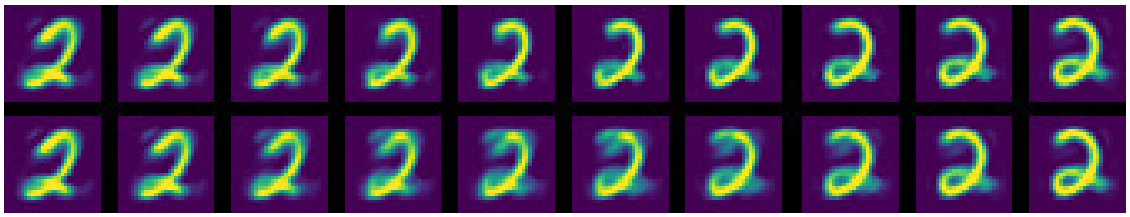


Figure 3.13: Interpolation in feature space: FlatNet (top), $\hat{x} = \mathcal{F}^{-1}\{\mathbf{g}_{\mathrm{CC}}(\theta \mathbf{f}_{\mathrm{CC}}(\mathcal{F}\{x_1\}) + (1-\theta)\mathbf{f}_{\mathrm{CC}}(\mathcal{F}\{x_2\}))\}$, vs. linear interpolation (bottom), $\hat{x} = \theta x_1 + (1-\theta)x_2$. FlatNet's feature space supports semantic interpolation in image space.

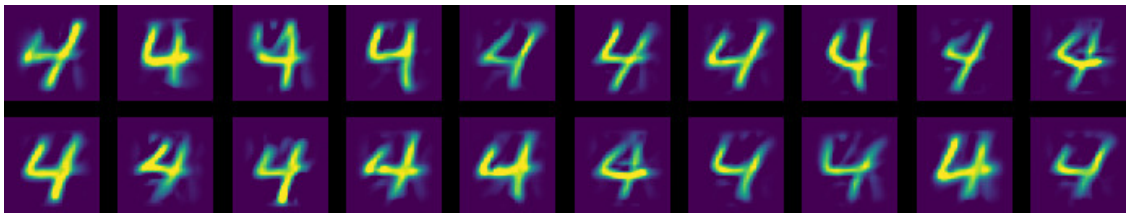
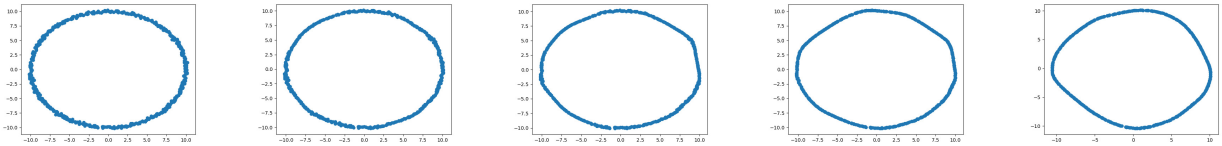
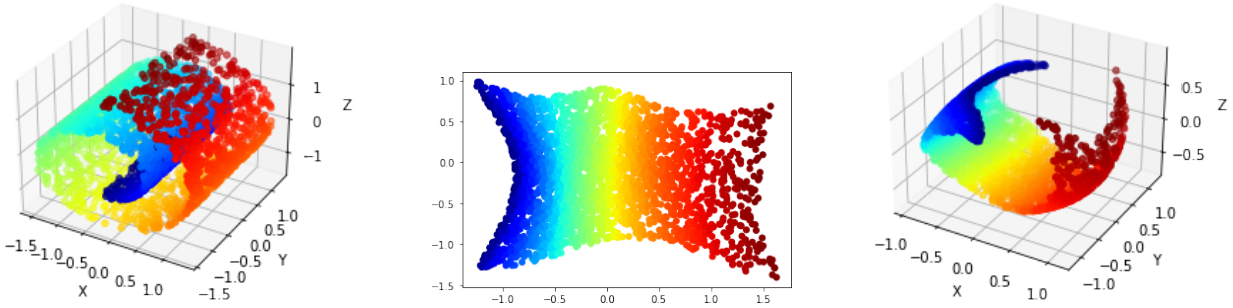


Figure 3.14: Sampling using FlatNet: $\hat{x} = \mathcal{F}^{-1}\{\mathbf{g}_{\mathrm{CC}}(z)\}$ where z is a Gaussian random variable supported on the affine subspace $\mathbf{f}_{\mathrm{CC}}(\mathcal{F}\{\mathcal{M}\})$.



(a) $\mathcal{M}_0$ (b) $\mathcal{M}_{20}$ (c) $\mathcal{M}_{40}$ (d) $\mathcal{M}_{60}$ (e) $\mathcal{M}_{80}$

Figure 3.15: Features $\mathcal{M}_\ell = \mathbf{f}_{1:\ell}(X)$ over $\ell \in \{0, 20, 40, 60, 80\}$, for data sampled from a noisy circle. The data is denoised onto a closed curve that oscillates but does not converge to a flat representation, reflecting the theoretical non-flattenability of closed loops.



(a) X. (b) $\mathbf{f}_{\text{CC}}(X)$. (c) $\mathbf{g}_{\text{CC}}(\mathbf{f}_{\text{CC}}(X))$.

Figure 3.16: Plots of (a) the Swiss-roll data ($N = 3000$), (b) the learned features after a converged FlatNet flattening on lifted features $(x_1, x_2, x_3, x_1^2 + x_3^2)$, and (c) the learned reconstruction. Reconstruction accuracy drops because of the added coordinate, but the learned features closely match the original generating coordinates.

Part III

Variational Optimization of Paths over Learned Representations

Chapter 4

Interlude: The Roughness of Modern Learned Representations

This is a short chapter. Its job is to bridge from sub-program A (Chapter 3) to sub-program B (Chapters 5–7), and to do so by making one empirical argument explicit: the rest of the dissertation cannot assume, as Chapter 3 did, that data lies near a smooth, compact, low-dimensional, flattenable embedded submanifold. The representations modern deep-learning systems actually produce are, in many cases, rough in ways that matter for downstream algorithms. The next three chapters respond by accepting this geometry and bringing the variational machinery to bear on paths through it.

4.1 The smooth-manifold reflex, and where it breaks

Classical manifold learning rests on a clean picture. Data points are samples from a probability distribution supported near a smooth low-dimensional embedded submanifold of some high-dimensional ambient space (Fefferman et al., 2016). The goal of representation learning is to recover that manifold structure, after which downstream operations, interpolation, density estimation, classification, can be carried out using the simpler intrinsic geometry. Methods built on this picture include Isomap (Tenenbaum et al., 2000), locally linear embedding (Roweis and Saul, 2000), Laplacian eigenmaps (Belkin and Niyogi, 2003), and the manifold-flattening approach of Chapter 3.

The picture is clean enough that, when one stays inside it, it produces real algorithms with provable guarantees. Many representations produced by modern deep learning, however, do not fit the picture. The mismatch is not a matter of small constants; it is a matter of the geometry being qualitatively different from what classical manifold learning describes.

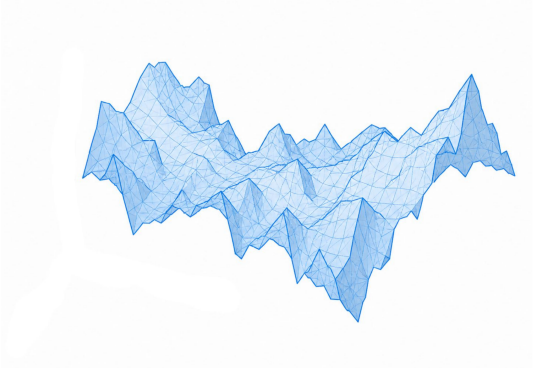


Figure 4.1: A schematic of the geometry of a typical learned representation, drawn to scale with the smoothness one might naively assume. The peaks and valleys are not artifacts of high curvature in a smooth landscape; they reflect Lipschitz constants and adversarial directions present at the relevant length scales.

4.2 Concrete evidence for roughness

We collect four vignettes here, each pointing at a different facet of the same underlying phenomenon. None of them is a formal theorem. They are evidence, of the kind that motivates a change of methodological commitments rather than the kind that proves a claim.

Pixel-space trajectories of smooth motion. Consider a synthetic video of a small white square translating from left to right across a black background at constant velocity. Intrinsically, the motion is a one-parameter family with derivative bounded above and below in a natural way: the square's center has constant velocity and the rest of its state is constant. Now consider the same motion as a trajectory in the ambient $H \times W$-dimensional pixel space (with H, W the image height and width). Each pixel's value as a function of time looks like a (slightly smoothed) Heaviside step: zero until the moving square's edge crosses that pixel, then briefly nonzero as it passes, then zero again. The trajectory's velocity in pixel space is near zero except during these brief edge-crossing transitions, where it spikes to a magnitude that grows with the resolution. Thus, motion that is smooth in its underlying parametrization can be highly non-smooth in the embedded ambient representation because the representation does not factor through the right intrinsic coordinates.

Self-supervised feature maps as superpixels. Modern self-supervised vision features, DINO (Caron et al., 2021) being one example, have spatial structure that resembles superpixel segmentation more than a low-dimensional embedding of image content. A typical DINO feature map has shape on the order of $14 \times 14 \times 384$: the spatial layout is preserved (most patch tokens correspond to a fixed location in the input image), and what changes from patch to patch is the contents of the 384-dimensional descriptor. As a result, feature-space trajectories under smooth visual change (a slow camera pan; a continuously deforming object)

tend to inherit the discrete spatial structure of the underlying tokenization. They are not smooth low-dimensional trajectories in the way a clean manifold-learning picture would predict. Empirically, one can verify this directly: pick two video frames a small time-step apart, compute their DINO features, and compare the norm of the feature-space displacement to a naive prediction based on a low-dimensional smooth-manifold model. The displacement is reliably larger than the smooth-manifold model predicts, and the gap grows with the spatial resolution of the feature map.

Adversarial robustness as a roughness symptom. The adversarial-robustness literature (Szegedy et al., 2013; I. J. Goodfellow et al., 2015) can be read, and is most charitably read, as a careful documentation of how rough learned representations actually are in their input space. A standard finding is that small, carefully chosen perturbations of an input image can drive a classifier's output to an arbitrary class with high confidence. This is a statement about the magnitude of the classifier's local gradients: in the directions an adversarial-attack algorithm finds, the classifier's response per unit input change is large enough that small input perturbations produce arbitrary output changes. Decades of attempts to train models that are simultaneously accurate and robust have produced a consistent empirical finding that the two objectives are in tension. We do not need a sharp theorem to draw the methodological implication: a representation whose gradients are large enough to enable adversarial attacks is not a representation whose downstream optimization problems are well-conditioned. Any algorithm that takes gradients through such a representation must contend with this. Chapter 7 returns to this observation in the most direct way, showing that the failure mode of long-horizon planning through visual world models is, structurally, an adversarial-robustness phenomenon.

Compositional dynamics gradients. A different but related observation arises whenever one back-propagates through a long composition of a learned dynamics map. Let $f_\theta : \mathbb{R}^n \to \mathbb{R}^n$ denote a one-step learned dynamics map, and consider its T-fold composition $f_\theta^T = f_\theta \circ \cdots \circ f_\theta$. The Jacobian of f_θ^T at a point is a product of T Jacobians of f_θ. By submultiplicativity of the operator norm, the largest singular value of this product is at most the product of the largest singular values of the constituent Jacobians, $\sigma_{\max}(\prod_t J_t) \leq \prod_t \sigma_{\max}(J_t)$, and generically grows or decays geometrically in T, since those factors are essentially never all exactly 1. In practice, the spectrum of ∇f_θ^T is dominated by either explosion or collapse: a few directions in which gradient signal grows exponentially in T, and many directions in which it vanishes exponentially in T. This is the standard exploding/vanishing-gradient phenomenon (Bengio, Simard, et al., 1994), well known in the RNN literature; what matters for our purposes is that it appears in any setting where a learned dynamics is rolled out for long horizons and one wants to use gradients of the resulting trajectory for control. Chapter 7 confronts this directly.

4.3 Why we cannot always flatten our way out

There are three reasons, each of which delineates the scope of sub-program A rather than rebutting it.

First, many learned representations are produced by training procedures whose objective is something other than manifold structure. A self-supervised image model is trained to make features predictive of one another, or of a held-out token, or of a contrastive partner; it is not trained to make the resulting feature space a smooth low-dimensional submanifold of $\mathbb{R}^D$. The resulting representation often does not satisfy even the precondition for the convexification flow (smooth low-dimensional flattenable embedded submanifold) to be applicable. We cannot "flatten" what is not, even approximately, a manifold in the relevant sense.

Second, even when the manifold hypothesis approximately holds, the flattening procedure depends on assumptions, in particular the flattenability assumption of Chapter 3, that are not easy to verify a priori and that empirically fail for some representations of interest. When the assumptions fail, the FlatNet construction does not produce a useful encoder; we would have no way of telling we were in that regime except by trying it and observing reconstruction failure.

Third, even when sub-program A would work in principle, applying it has costs. Constructing $(\mathbf{f}_{\mathrm{CC}}, \mathbf{g}_{\mathrm{CC}})$ requires a finite sample dense enough that local quadratic models are well-estimated, and a sufficient number of FlatNet layers; the result is a different representation from the one the upstream pretraining produced. For many downstream applications, the upstream representation has properties (semantic alignment, transfer behavior, downstream task performance) that are useful in their own right and that we have no guarantee the flattened representation would preserve. Re-flattening can be a strict downgrade for the downstream use case.

4.4 The dissertation's response: accept the roughness, reshape the functional

The rest of the dissertation accordingly takes a different approach. It takes the learned representation, whatever upstream training procedure produced it, in whatever rough form it comes, as a given. It then asks: what is the right variational machinery to bring to bear on *paths* through this representation, holding the geometry fixed?

Chapters 5, 6, and 7 show that the variational principle does not require manifold smoothness. It requires only that one can write down a meaningful functional $\mathcal{L}[\gamma]$ on paths γ, whose stationary points capture what we want from the path. The functional can be assembled from the learned object's local outputs (a critic action gradient, a score, a one-step dynamics prediction), without ever passing through the kind of global smooth-manifold assumption that sub-program A relies on.

The resulting functional can still be difficult to optimize because it depends on a learned object whose values, gradients, and curvature are rough. The technical contribution of each

of the next three chapters is to show how, within the variational framing, one can reshape the functional itself (rather than the underlying space) so that the optimization remains well-conditioned. The recurring technique is what we will call *smoothing-by-splitting*: a quantity that would otherwise require evaluating a long, rough composition is replaced by a variational surrogate that depends only on local, well-conditioned ingredients.

4.5 Two complementary tools, one program

The two sub-programs are not in competition. Chapter 3's convexification flow applies when the data manifold is approximately smooth, compact, low-dimensional, and flattenable. The flattened representation supports clean linear analysis and minimal autoencoding with provable guarantees. For a fixed, rough representation produced by an upstream training procedure, the next three chapters instead apply variational machinery to paths without requiring the representation to be manifold-smooth.

The dissertation's larger argument is that the variational principle places both choices within one program. Whether the problem is over embedded submanifolds (sub-program A) or paths in a fixed embedded space (sub-program B), the intellectual structure is the same: identify the right functional, descend it, and exploit the freedom available in choosing it. With that observation in hand, we turn to the three instances of sub-program B.

Chapter 5

Q-Score Matching: Variational Policy Improvement for Diffusion Reinforcement Learning

This chapter is based on Psenka and Abbeel, 2024, joint work with Alejandro Escontrela, Pieter Abbeel, and Yi Ma. The paper is reframed here as the dissertation's first instance of sub-program B: a functional on paths in a fixed (rough) learned-model landscape, where the functional is induced by a learned critic. The bulk of the chapter follows the original paper closely; full proofs are collected in Appendix B.

Reinforcement learning succeeds across domains ranging from discrete game environments like Go, Chess, and Poker (Silver, A. Huang, et al., 2016; Silver, Hubert, et al., 2017; N. Brown and Sandholm, 2019) to continuous environments like goal-oriented robotics (Kober et al., 2013; Sünderhauf et al., 2018; Ibarz et al., 2021; P. Wu et al., 2022). Robotics applications typically work in continuous vector spaces for both states and actions, which makes parameterizing the action distribution a difficult challenge: one must choose between easy sampling (e.g., Gaussian policies (Agostini and Celaya, 2010)) and expressiveness.

Diffusion models (Hyvärinen and Dayan, 2005; Ho et al., 2020) are a promising parametrization. By iteratively noising and denoising, they have shown empirical success in generative tasks (Rombach et al., 2022; Watson et al., 2022), and in the RL context they offer both expressiveness and easy sampling without explicit normalization constants. However, applying diffusion models to online RL is relatively underdeveloped, and the nuances of how to train them in this setting are still being worked out.

An unexplored approach is to align the score of the learned policy $\nabla_a \log \pi(a|s)$ with the score of an optimal policy $\nabla_a \log \pi^*(a|s)$. Traditional score matching is ill-posed here: we have no samples from $\nabla_a \log \pi^*(a|s)$, and not even samples from $\pi^*(a|s)$ itself. The chapter's central observation is that, in the RL context, one can match the score of π to that of π^* by iteratively matching the score of π to the action gradient of the state-action value function $\nabla_a Q^\pi(s,a)$. This gives a geometric reading of policy optimization: the focus shifts to iteratively pushing the vector field $\nabla_a \log \pi(a|s)$ toward the vector field $\nabla_a Q^\pi(s,a)$. We

call the resulting method *Q-score matching* (QSM). We then apply this to off-policy RL with diffusion-model policies, an important but largely unexplored area.

Within the dissertation's argument, this construction is the first instance of sub-program B: a variational functional defined on paths (in this case, reverse-diffusion paths through action space) induced by a learned critic, where the optimization is reshaped to remain well-conditioned over the rough learned-model landscape. The recurring smoothing-by-splitting principle of Chapter 4 appears here in its simplest form: rather than backpropagating through the K-deep denoising chain, the optimization is replaced by a local score-matching surrogate whose gradients depend only on local Jacobians of the denoising network.

The chapter is structured as follows. We first set up diffusion policies in the discrete DDPM picture used throughout the rest of the dissertation. We then introduce a continuous-time stochastic-differential-equation formulation that proves convenient for theoretical analysis, and show how time-discretization recovers the diffusion-model picture. We discuss why the standard policy gradient is problematic for diffusion policies. We then introduce QSM, give a theoretical analysis culminating in optimality-condition theorems and a Boltzmann fixed-point characterization, present an algorithmic instantiation, and report empirical results across a range of continuous-control tasks.

5.1 Setup: diffusion policies and the policy-improvement problem

A diffusion policy (Zhendong Wang et al., 2022) parametrizes an agent's distribution over actions, given the current state, as the marginal distribution of the final iterate of a reverse diffusion process (Ho et al., 2020; Song and Ermon, 2019). Concretely, given a state s, sampling an action proceeds by initializing $a^{(K)} \sim \mathcal{N}(0, I)$ in the action space and iterating

$$a^{(k-1)} \;=\; a^{(k)} + \alpha_k\, \varepsilon_\theta\big(a^{(k)}, s, k\big) + \sigma_k\, \xi_k, \qquad \xi_k \sim \mathcal{N}(0, I), \tag{5.1}$$

for $k = K, K-1, \ldots, 1$, with α_k, σ_k schedule parameters and ε_θ the learned denoising network. The deterministic limit ($\sigma_k = 0$) is a probability-flow ODE; the stochastic version is the standard reverse SDE. In either case, the final iterate $a^{(0)}$ is the sampled action.

This is an expressive policy class. Because it samples actions by following a learned noise-to-data trajectory, it can in principle represent multimodal action distributions of the form $\pi(a|s) = \sum_i w_i(s)\mathcal{N}(\mu_i(s), \Sigma_i(s))$, including modes that have no continuous parametric structure. This multimodality is useful. In many control problems the right thing to do at a given state is not unique, and an agent that can commit to multiple alternative behaviors at training time has a better chance of finding good ones than an agent restricted to unimodal Gaussians.

The difficulty is policy improvement. Suppose we have a learned critic $Q_\eta(s, a)$ estimating the expected return of action a in state s. How should we update the parameters θ of the diffusion policy so that it samples, with higher probability, actions with high $Q_\eta(s, a)$?

Two existing approaches both have known difficulties.

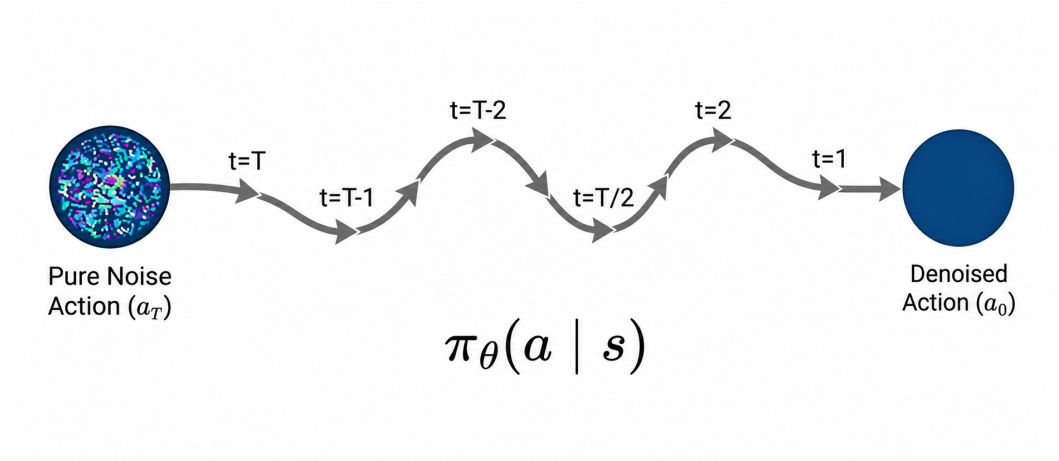


Figure 5.1: The diffusion policy $\pi_\theta(a \mid s)$ samples actions by initializing pure noise $a^{(K)}$ and iteratively denoising through K steps to produce $a^{(0)}$, an action used in the environment. The reverse-diffusion trajectory is the path the variational machinery of this chapter acts on.

Policy gradient. One can write a likelihood-based policy gradient by computing $\nabla_\theta \log \pi_\theta(a|s)$ for the diffusion policy. This requires either marginalizing over the noise samples ξ_k in (5.1) (intractable for nontrivial schedules) or using a score-function estimator that is empirically very high-variance. Section 5.3 works through the resulting formula and argues that the variance scales unfavorably with the number of diffusion steps K.

Backpropagation through the reverse process. The alternative, often called diffusion-QL in the literature (Zhendong Wang et al., 2022), is to backpropagate through the entire reverse-diffusion chain in (5.1). For fixed noise samples and a fixed initialization, the final action $a^{(0)}$ is a deterministic function of θ (through the dependence on ε_θ at each step), and one can compute $\nabla_\theta Q_\eta(s, a^{(0)})$ by the chain rule through all K iterations of (5.1). This is a well-defined gradient, but it suffers from two problems: the gradients pass through K compositions of ε_θ, producing a version of the exploding/vanishing-gradient phenomenon of Chapter 4; and the resulting update is biased toward making $a^{(0)}$ a deterministic function of s (equivalently, biased toward unimodal policies), because what the chain rule maximizes is the value at the *one* action that the reverse process produces given fixed noise.

The empirical consequence is that diffusion-QL works but collapses the multimodality of the policy in characteristic ways. The expressiveness of the diffusion policy class is forfeited in the process of improving it.

5.2 Continuous-time formulation

The DDPM picture in (5.1) is the natural setting for implementation and for connecting to broader diffusion literature. For theoretical analysis, however, it is convenient to work with

a continuous-time stochastic-differential-equation (SDE) formulation in which states and actions co-evolve as a joint dynamical system. We use the SDE picture for the optimality-condition theorems in Section 5.5, then bridge back to the DDPM picture for the algorithm and experiments.

Joint state-action SDE

We denote the state space as a Euclidean space $\mathcal{S} = \mathbb{R}^s$ and the action space as $\mathcal{A} = \mathbb{R}^a$. The continuous-time setting is:

$$\begin{aligned} \mathrm{d}s &= F(s,a)\,\mathrm{d}t + \Sigma_s(s,a)\,\mathrm{d}B_t^s, \\ \mathrm{d}a &= \Psi(s,a)\,\mathrm{d}t + \Sigma_a(s,a)\,\mathrm{d}B_t^a, \\ s(0) &= s_0, \qquad a(0) = a_0. \end{aligned} \tag{5.2}$$

The vector field $\Psi : \mathcal{S} \times \mathcal{A} \to \mathcal{A}$ is the "score" of the policy, and is the main parameter for policy optimization in this setting. $F : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the continuous state dynamics. Σ_s, Σ_a map $\mathcal{S} \times \mathcal{A}$ to positive semidefinite matrices in $\mathbb{R}^{s\times s}$ and $\mathbb{R}^{a\times a}$ respectively, representing the covariance structure of the dynamics; B^s, B^a are separate standard Brownian motions on $\mathcal{S}$ and $\mathcal{A}$.

A note on the term "score". We use Ψ throughout to highlight the analogy with the classical score of a distribution $\nabla_a \log \pi(a|s)$, most clearly visible in Langevin dynamics (Langevin, 1908; Papanicolaou, 1977; Welling and Teh, 2011). The vector field Ψ in (5.2) need not literally be the gradient of any log-density; the terminology is suggestive rather than literal.

While many works formulate RL in continuous state/action/time settings (Bradtke and Duff, 1994; Doya, 2000; Jia and X. Y. Zhou, 2022; Jia and X. Y. Zhou, 2023; Zhao et al., 2024), it is less common to formulate the action as a joint dynamical system with its own SDE. The joint formulation matters for the connection to diffusion models, made explicit below.

The objective is to maximize path integral loss functions of the form

$$Q^\Psi(s_0,a_0) \;=\; \mathbb{E}\int_0^\infty \gamma^t\, r(s(t,s_0,a_0))\,\mathrm{d}t, \tag{5.3}$$

where $r : \mathcal{S} \to [0,1]$ is the reward function, the expectation is over paths from (5.2), $s(t,s_0,a_0)$ is the state at time t starting from (s_0,a_0), and $\gamma \in (0,1)$ is a discount factor. Discretizing in time gives the familiar Q-function $\mathbb{E}\sum_{t=0}^\infty \gamma^t r(s(t))$, and truncating at a horizon T with $\sum_{T+1}^\infty \gamma^i \approx 0$ gives the finite-horizon version. We write the Q-function with superscript Ψ to emphasize the dependence on the score (rather than the initial conditions): we want to find a Ψ^* that maximizes $Q^\Psi(s_0,a_0)$ at fixed (s_0,a_0), or in expectation over initial conditions $(s,a) \sim \mathbb{P} \times \pi$:

$$J(\Psi) \;=\; \mathbb{E}_{\mathbb{P}\times\pi}\, Q^\Psi(s,a). \tag{5.4}$$

Time discretization recovers the diffusion-model picture

The action model in (5.2) is unusual in that actions $a(t)$ evolve as a smooth flow rather than as an explicit function of the state. The motivation appears when we discretize at different time scales for state and action dynamics. Apply Euler-Maruyama with K inner action-steps for each state-step:

$$\begin{aligned} s_{t+1} &= s_t + F(s_t, a_t) + z, \quad z \sim \mathcal{N}(\mathbf{0}, \Sigma_s(s_t, a_t)), \\ a_t^i &= a_t^{i-1} + \tfrac{1}{K}\Psi(s_t, a_t^{i-1}) + z_t^i, \\ z_t^i &\sim \mathcal{N}\big(\mathbf{0}, \tfrac{1}{K}\Sigma_a(s_t, a_t^{i-1})\big), \\ a_{t+1} &= a_t^K, \end{aligned} \tag{5.5}$$

with $a_t^i \coloneqq a_{t+i/K}$. Discretizing action dynamics at K times the fidelity of state dynamics recovers a score-based generative model: this is analogous to a time-invariant diffusion model of depth K, with denoising mean given by Ψ and variance by Σ_a. Concretely, identifying a_t^{i-1} in (5.5) with $a^{(k)}$ in (5.1) and identifying $\frac{1}{K}\Psi(s_t, a_t^{i-1})$ with $\alpha_k \varepsilon_\theta(a^{(k)}, s, k)$, we see that the SDE drift Ψ is the continuous-time analog of the denoising vector field ε_θ. The theory developed in the continuous-time setting therefore translates approximately to the diffusion-model setting up to time-discretization error. The time-scaling also reveals an intuition: as the number of sampling steps K increases, the initial condition a_{t-1} becomes less influential on a_{t+1} and the action becomes more strongly dependent on s_t through Ψ.

For the rest of this chapter, we work in the continuous-time setting for theoretical analysis (Sections 5.3–5.5) and translate to the DDPM picture for the algorithm and experiments (Sections 5.7–5.8).

5.3 Policy gradient for diffusion policies

Before introducing QSM, it is useful to make precise why the standard policy gradient is problematic for diffusion policies, since this motivates the design of the alternative.

The policy gradient (Richard S Sutton et al., 1999) for the global objective $J(\theta) = \mathbb{E}_{s,a}\, Q^\pi(s,a)$ is

$$\nabla_\theta J(\theta) \;=\; \mathbb{E}_{(s_0,a_0,s_1,a_1,\dots)} \sum_{t=1}^{\infty} Q(s_t, a_t)\, \nabla_\theta \log \pi_\theta(a_t \mid s_t), \tag{5.6}$$

where the expectation is over trajectories produced by π_θ and the environment dynamics. For a time-discretized diffusion-model policy, we do not have access to the global density $\pi(a|s)$, but only to the incremental conditional densities $\pi(a^\tau \mid a^{\tau-1}, s)$ at each diffusion step. The marginal density is

$$\pi(a^K \mid s) \;=\; \int_{a^{K-1}} \cdots \int_{a^1} \pi(a^1 \mid s) \prod_{\tau=2}^{K} \pi(a^\tau \mid a^{\tau-1}, s)\, \mathrm{d}a^1 \cdots \mathrm{d}a^{K-1}, \tag{5.7}$$

an integral that is infeasible to compute directly for nontrivial K.

Substituting (5.7) into (5.6) and simplifying via the standard log-gradient identity gives

$$\nabla_\theta J(\theta) \;=\; \mathbb{E} \sum_{t=1}^{\infty} Q(s_t, a_t^K) \left(\sum_{\tau=1}^{K} \nabla_\theta \log \pi_\theta(a_t^\tau \mid a_t^{\tau-1}, s_t) \right), \tag{5.8}$$

where the expectation is over all states $\{s_t\}_{t=1}^{\infty}$ and all internal action samples $\{\{a_t^\tau\}_{\tau=1}^{K}\}_{t=1}^{\infty}$. The proof is given in Appendix B.1.

The form (5.8) is theoretically clean but practically problematic. The expectation now depends explicitly on the internal action states $\{a_t^\tau\}_{\tau=1}^{K}$, adding an extra factor of dimensionality on the scale of $K \cdot |\mathcal{A}|$, where $|\mathcal{A}|$ is the action dimension. While the scores for the original objective $J(\theta) = \mathbb{E}_{s,a} Q(s,a)$ are invariant to the inner steps, the internal gradient estimates for $\nabla_\theta J(\theta)$ in (5.8) *do* depend on internal action sequences, so finite-sample estimates of $\nabla_\theta J(\theta)$ inherit this extra variance. The result, observed empirically in Section 5.8, is sample inefficiency: a policy gradient applied directly to a diffusion-model policy fails to learn optimal policies in the off-policy setting on standard benchmarks. Importantly, the form (5.8) also uses no explicit structure of diffusion models; it holds for any model with internal action stages. We are thus motivated to pursue an alternative that uses the score-based structure of diffusion models more directly.

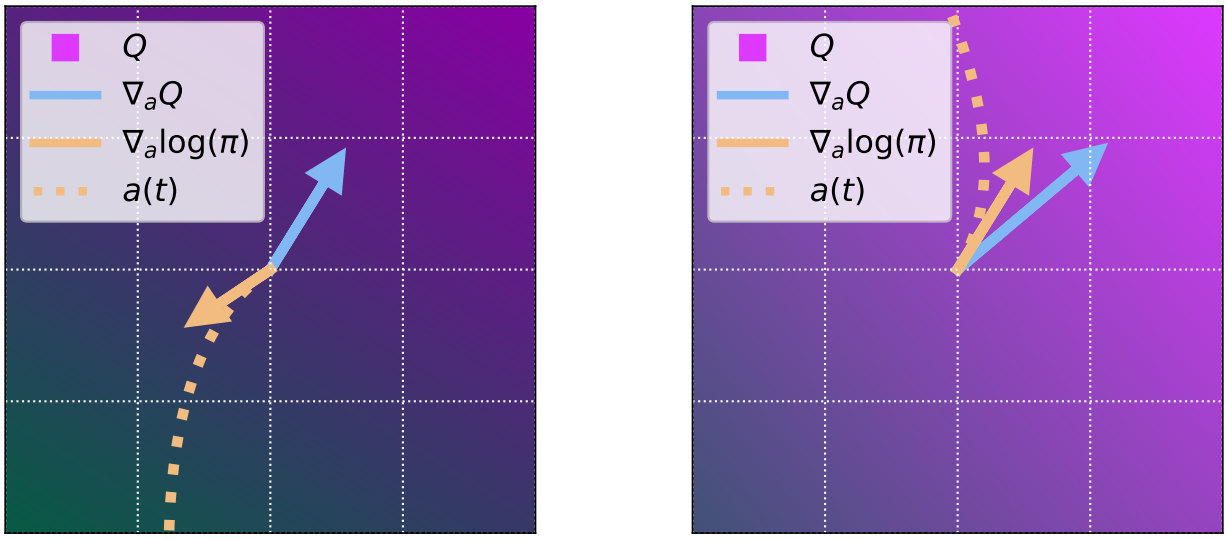
5.4 The variational framing: Q-score matching

The reverse-diffusion process can be read as a path through action space, parametrized by the denoising-time index k. The endpoint $a^{(K)}$ is fixed (pure noise); the endpoint $a^{(0)}$ is what we are trying to control. The question is what we want from the intermediate denoising directions $\varepsilon_\theta(a^{(k)}, s, k)$, or equivalently the SDE drift $\Psi(s,a)$ at intermediate (s,a).

In the path-space variational framing of this dissertation, we want a functional $\mathcal{L}[\Psi]$ whose stationary points correspond to good drift directions. The structural requirement is that $\mathcal{L}$ depend only on local information about the policy and the critic, not on a backprop through the full denoising chain, so the resulting optimization avoids the problems of Section 5.3.

Our goal is to match Ψ to that of an optimal distribution with respect to J, denoted Ψ^*. Unlike the standard score-matching literature, we have neither samples from Ψ^* nor from the action distribution π^* it generates. A surrogate Ψ^* approximator is required.

A natural starting point is dimensional analysis. The drift $\Psi(s,a)$ is a vector field $\mathcal{S} \times \mathcal{A} \to \mathcal{A}$. The action gradient of the Q-function, $\nabla_a Q^\Psi(s,a)$, is also a vector field $\mathcal{S} \times \mathcal{A} \to \mathcal{A}$. The two have the same shape. If the goal is to make the diffusion policy improve under the learned critic, the simplest variational object aligns these two vector fields. We accordingly set $\Psi^* \approx \nabla_a Q$ and define the actor update as iterative optimization of Ψ against $\nabla_a Q$.



(a) Iteration 0: random initialization of Ψ.

(b) Iteration 1: one step of QSM.

Figure 5.2: Conceptual illustration of QSM training. Orange arrows are the policy drift $\Psi(s, a)$ produced by the current policy; blue arrows are the action gradients $\nabla_a Q^\Psi(s, a)$ of the current critic. One step of QSM updates Ψ to align with $\nabla_a Q^\Psi$, scaled by λ. As training proceeds, the drift converges to the direction of steepest ascent of Q, and the diffusion policy samples preferentially from regions of high Q.

Definition 5.1 (Q-score matching loss)**.** Let $\varepsilon_\theta(a, s, k)$ be a learned denoising network and $Q_\eta(s, a)$ a learned critic. The *Q-score matching* (QSM) loss for the policy parameters θ is

$$\mathcal{L}_{\text{QSM}}(\theta) \;=\; \mathbb{E}_{s,k,a}\Big[\big\|\varepsilon_\theta(a, s, k) - \lambda\, \nabla_a Q_\eta(s, a)\big\|_2^2\Big], \tag{5.9}$$

where the expectation is over states s from the replay buffer, denoising-time indices k uniformly from $\{1, \ldots, K\}$, and action samples a from a chosen distribution (in practice the noised actions encountered during forward diffusion of the current policy's outputs), and $\lambda > 0$ is a scalar hyperparameter.

The loss (5.9) fits the variational template. It is a functional whose argument is the denoising network ε_θ, viewed as a vector field on action space; its stationary points are configurations in which ε_θ matches the direction of steepest ascent of Q_η at each action, up to scaling by λ. Computing the loss requires only local evaluations: ε_θ at a single action, $\nabla_a Q_\eta$ at the same single action. No backpropagation through the reverse-diffusion chain is required to evaluate or differentiate (5.9) with respect to θ.

This local construction is the chapter's instance of sub-program B and the smoothing-by-splitting principle introduced in Chapter 4.

5.5 Theory: optimality conditions and the Boltzmann fixed point

The variational form (5.9) can be derived from a dimensional-analysis argument alone, but does iterative matching of Ψ to $\nabla_a Q^\Psi$ provably converge to an optimal score? The next two theorems give such a result, in increasing levels of generality.

Deterministic setting

We begin in the deterministic setting where state and action both evolve as paired ordinary differential equations. To simplify, we optimize the Q-function at a fixed point, without loss of generality at the origin $Q(\mathbf{0}, \mathbf{0})$. Even at a fixed point, Q is still a path integral, so the setting is far from trivial. The following theorem relates the optimal policy's score to the action gradient of the Q-function.

Theorem 5.2 (Optimality condition, deterministic setting)**.** *Consider the joint deterministic dynamics governing $s(t) \in \mathbb{R}^s$ and $a(t) \in \mathbb{R}^a$:*

$$\begin{aligned} \dot{s}(t) &= F(s(t), a(t)), & \dot{a}(t) &= \Psi(s(t), a(t)), \\ s(0) &= \mathbf{0}, & a(0) &= \mathbf{0}, \end{aligned} \tag{5.10}$$

with $\|\Psi(s,a)\|_2 \le C$ for all (s,a) and Ψ Lipschitz. Denote by $s(t, s_0, a_0)$ the resulting state from initial conditions (s_0, a_0). Let $r : \mathbb{R}^s \to [0,1]$ be smooth. Fix $\gamma \in (0,1)$. Let $\Psi^* : \mathbb{R}^s \times \mathbb{R}^a \to \mathbb{R}^a$ be a vector field maximizing

$$J(\Psi) = Q^\Psi(\mathbf{0}, \mathbf{0}), \tag{5.11}$$

$$Q^\Psi(s_0, a_0) = \int_0^\infty \gamma^t r(s(t, s_0, a_0))\, \mathrm{d}t, \tag{5.12}$$

over smooth flows Ψ with norm bound $\|\Psi\|_2 \le C$ and finite Lipschitz constant (not necessarily equal to that of the reference flow). Then $\Psi^*(s,a) = \alpha_{s,a} \nabla_a Q^{\Psi^*}(s,a)$ for some $\alpha_{s,a} > 0$ for all (s,a) along the trajectory $(s(t), a(t))$ where $\nabla_a Q^{\Psi^*}(s,a) \neq \mathbf{0}$.

The proof is in Appendix B.2.

The theorem captures the central intuition: *if there is any point where $\Psi(s,a)$ and $\nabla_a Q^\Psi(s,a)$ are misaligned, then a vector field Ψ' obtained by aligning Ψ to $\nabla_a Q^\Psi$ at that point yields $Q^{\Psi'}(\mathbf{0},\mathbf{0}) > Q^\Psi(\mathbf{0},\mathbf{0})$.* This formalizes the intuition that the functional gradient with respect to the score Ψ is proportional to the action gradient $\nabla_a Q$.

Once the theorem extends from a fixed initial point to integrals over Q, the collinearity condition extends to the entire integrated space.

Corollary 5.3 (Distributional version)**.** *Consider the setting of Theorem 5.2 but with the modified loss*

$$J(\Psi) \;=\; \mathbb{E}_{\mathbb{P}\times\pi}\, Q^\Psi(s,a), \tag{5.13}$$

where $\mathbb{P} \times \pi$ is a probability measure on $\mathbb{R}^s \times \mathbb{R}^a$ absolutely continuous with respect to Lebesgue measure and with strictly positive density everywhere (full support). For any maximizer Ψ^*, $\Psi^*(s,a) = \alpha_{s,a} \nabla_a Q^{\Psi^*}(s,a)$ for all $(s,a) \in \mathbb{R}^s \times \mathbb{R}^a$ where $\nabla_a Q^{\Psi^*}(s,a) \neq \mathbf{0}$.

Stochastic setting

The result extends to the full stochastic setting of (5.2).

Theorem 5.4 (Optimality condition, stochastic setting)**.** *Consider the joint stochastic dynamics:*

$$\begin{aligned} \mathrm{d}s(t) &= F(s(t),a(t))\,\mathrm{d}t + \Sigma_s(s(t),a(t))\,\mathrm{d}B_t^s, \\ \mathrm{d}a(t) &= \Psi(s(t),a(t))\,\mathrm{d}t + \Sigma_a(s(t),a(t))\,\mathrm{d}B_t^a, \\ s(0) &= s_0, \qquad a(0) = a_0, \end{aligned} \tag{5.14}$$

with Ψ, F globally Lipschitz, and Σ_s, Σ_a globally Lipschitz into positive semidefinite matrices, and B^s, B^a independent standard Brownian motions. Consider the loss

$$J(\Psi) = \mathbb{E}_{\mathbb{P}\times\pi}\, Q^{\Psi}(s,a), \tag{5.15}$$

$$Q^{\Psi}(s_0,a_0) = \mathbb{E}\int_0^{\infty} \gamma^t\, r(s(t,s_0,a_0))\,\mathrm{d}t. \tag{5.16}$$

Then for any optimal Ψ^* with respect to J, $\Psi^*(s,a) = \alpha_{s,a} \nabla_a Q^{\Psi^*}(s,a)$ for some $\alpha_{s,a} > 0$ for all $(s,a) \in \mathbb{R}^s \times \mathbb{R}^a$ where $\nabla_a Q^{\Psi^*}(s,a) \neq \mathbf{0}$ in the support of the joint distribution induced by $\mathbb{P} \times \pi$ and the stochastic dynamics.

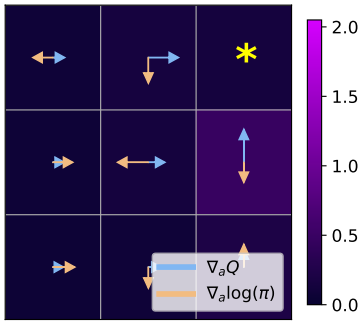
The proof is in Appendix B.3. While the theorem does not specify an algorithm, it justifies the family: iteratively matching $\Psi(s,a)$ to $\nabla_a Q(s,a)$ provides strict global increases of the Q-function.

Pedagogical reduction in gridworld

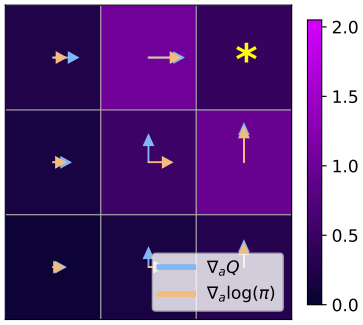
To build intuition for Theorem 5.4, we examine a reduction in a discrete-state gridworld, where $\mathcal{S} = \{1,\dots,M\} \times \{1,\dots,N\}$ and $\mathcal{A} = \{\text{LEFT}, \text{RIGHT}, \text{UP}, \text{DOWN}\}$. Diffusion models are not directly defined on discrete spaces, but there is a sensible reduction via the Langevin-dynamics link.

Suppose (for Euclidean $\mathcal{S}, \mathcal{A}$) we have completed the matching in Theorem 5.4 and obtained $\Psi^*(s,a) = \alpha \nabla_a Q(s,a)$ for fixed $\alpha > 0$. Take the dynamics of (5.2) at a fixed state $(F(s,a) = 0,\ \Sigma_s(s,a) = 0)$ with $\Sigma_a(s,a) = \sqrt{2}\, I_a$. Under standard conditions (Bakry et al., 2014), the stationary distribution of $a(t)$ as $t \to \infty$ for any fixed s is

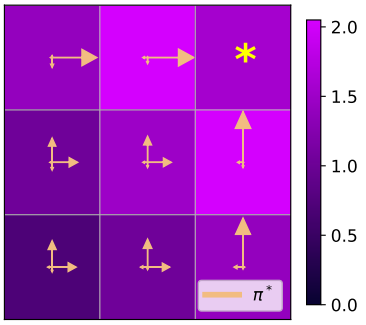
$$\pi(a|s) \;\propto\; e^{\alpha Q(s,a)}. \tag{5.17}$$



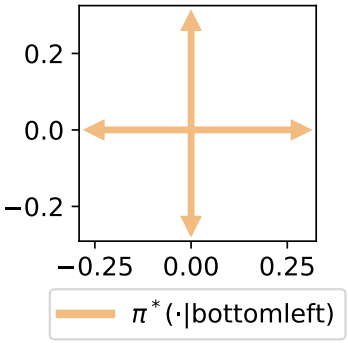
(a) Randomly initialized π^0.



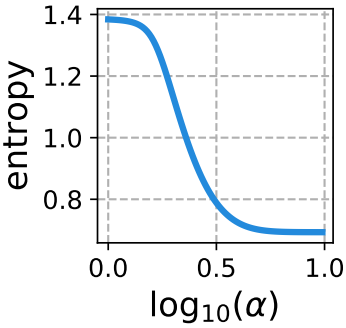
(b) Once-updated π^1.



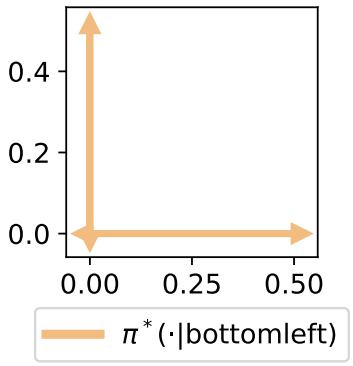
(c) Converged π^*.



(d) Converged policy at $\alpha = 1$.



(e) Entropy vs. α.



(f) Converged policy at $\alpha = 10$.

Figure 5.3: Pedagogical simulation of QSM's reduction to a single-goal gridworld. **Top row:** two iterations of $\pi(a|s) \leftarrow e^{\alpha Q^\pi(s,a)} / \sum_{a'} e^{\alpha Q^\pi(s,a')}$ for $\alpha = 2$. Square colors show expected reward; we use the local maximizing direction to define discrete gradients $\nabla_a Q(s,a) := a^* Q(s,a^*)$ where $a^* = \arg\max_{a'} Q(s,a')$, and analogously for $\nabla_a \log \pi(a|s)$. **Bottom row:** effect of α on the converged distribution. Left, $\alpha = 1$; right, $\alpha = 10$. Larger α produces lower entropy.

QSM can therefore be read as matching the full action distribution $\pi(a|s)$ to the Boltzmann distribution of the Q-function, $\frac{1}{Z} e^{\alpha Q(s,a)}$ with $Z = \int_{\mathcal{A}} e^{\alpha Q(s,a)} \, \mathrm{d}a$. Directly setting $\pi(a|s) \propto e^{\alpha Q(s,a)}$ is infeasible in continuous spaces, but in the finite case we can represent $\pi(a|s)$ explicitly as a matrix and use the update rule

$$\pi'(a|s) \;=\; \frac{e^{\alpha Q^\pi(s,a)}}{\sum_{a'} e^{\alpha Q^\pi(s,a')}}. \tag{5.18}$$

This is the same as soft policy iteration (Haarnoja et al., 2018), but using the standard Q-function rather than the soft Q-function. In simulated gridworld, α acts as an inverse-entropy regularization parameter: lower α gives higher-entropy converged distributions (see Figure 5.3).

The Boltzmann reading is exact in the finite-action setting and, in the continuous setting, holds as an idealized equilibrium limit.

Proposition 5.5 (Finite-action Boltzmann fixed point)**.** *Let $\mathcal{A}$ be finite. If the soft policy iteration* (5.18) *converges, its fixed point is the Boltzmann policy $\pi^{(\lambda)}(a|s) \propto \exp\big(\lambda\, Q^{\pi^{(\lambda)}}(s,a)\big)$.*

This is the standard soft-policy-iteration fixed point (Haarnoja et al., 2018), with the ordinary Q-function in place of the soft Q-function; it is what Figure 5.3 exhibits.

In the continuous setting the same reading holds as an idealization rather than a property of the finitely-many-step policy actually deployed. Viewing the reverse process (5.1) with drift $\lambda \nabla_a Q_\eta$ as an overdamped Langevin sampler run to equilibrium, its stationary law is $\pi^{(\lambda)} \propto \exp(\lambda Q_\eta)$, which tends to $\exp(\lambda Q^*)$ as $Q_\eta \to Q^*$. Two conditions matter here (Appendix B.4): this requires $\exp(\lambda Q_\eta)$ to be integrable over $\mathcal{A}$ (a confining Q_η, or compact $\mathcal{A}$; boundedness of Q^* on an unbounded $\mathcal{A}$ is not sufficient), and it is the $t \to \infty$ equilibrium, which the fixed-horizon K-step sampler realizes only approximately.

The interpretation: λ (or α) controls how greedy the policy is. Small λ produces a near-uniform distribution over actions; large λ concentrates probability mass on the argmax of Q^*. The policy is multimodal whenever Q^* has multiple near-optimal actions at s, and the diffusion policy preserves the multimodality. This contrasts sharply with the diffusion-QL update, which is biased toward unimodal collapse regardless of the structure of Q^*.

5.6 Multimodality preservation

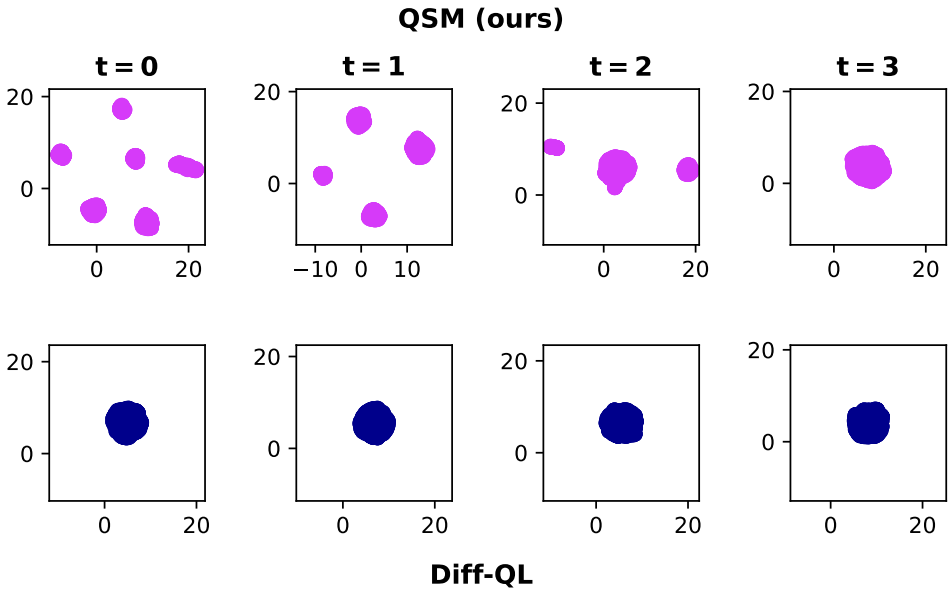
QSM preserves multimodality without explicit regularization, unlike alternative diffusion-RL update rules. We add no entropy term, no behavior-cloning penalty, and no other multimodality-preserving mechanism; the variational form of (5.9) alone keeps the policy from collapsing.

A diagnostic experiment makes this concrete. Take a quadruped-control task in which an agent must locomote from a randomized initial pose. At the beginning of an episode, the optimal action is ambiguous: there are multiple equally-good strategies for initiating the gait. As the agent commits to one, the optimal action becomes unimodal. A multimodal policy that respects this structure should produce broad action distributions early and narrow ones later.

A diffusion policy trained by QSM exhibits exactly this pattern (Figure 5.4). Visualizing the distribution of action samples at a fixed state, the early-denoising samples spread across multiple modes; the late-denoising samples concentrate. A diffusion policy trained by diffusion-QL on the same task and architecture exhibits unimodal collapse from the first epoch: across the ensemble, the policy commits to one strategy and discards the alternatives. The two policies are otherwise comparable in their final rewards; the difference is in the exploration behavior implied by the learned policy, and (relatedly) in the population of strategies the policy ends up representing.



(a) Initial state.



(b) Action samples across denoising time for QSM (top) and diffusion-QL (bottom).

Figure 5.4: Multimodal action distributions on the quadruped-walk task. From the initial state, 1,000 actions are sampled from each policy at successive denoising time-steps and projected to $\mathbb{R}^2$ via UMAP (McInnes et al., 2018). The QSM policy (top) retains multiple distinct modes through early denoising and only commits to a single mode once the trajectory becomes effectively unambiguous; the diffusion-QL policy (bottom) collapses to a unimodal distribution at every denoising step. The two policies share identical architectures and sampling procedure; only the policy-update rule differs. The diversity in QSM samples therefore comes from the training methodology, not from the architecture.

5.7 Algorithmic instantiation

A practical implementation of QSM combines the QSM policy update with a standard critic-update routine. We use N-step returns (Richard S. Sutton and Barto, 2018) and the Q-function optimization of DDPG (T. P. Lillicrap et al., 2019), with the twin-critic min-Q trick of TD3 (Fujimoto et al., 2018a; Fujimoto et al., 2018b). For each update step, actions from the replay buffer are noised according to a variance-preserving Markov chain (cf. Hansen-Estruch et al., 2023); we evaluate $\nabla_a Q$ at the noised actions and supply this as the target to the score model ε_θ. The update is computed for every environment step with batches of size 256. Both critic and score model are 2-layer MLPs with 256 units per layer.

Algorithm 2 summarizes the procedure. We add a small amount of Gaussian noise to the final sampled actions, and to the sampled minibatch actions used for training. Alternative exploration strategies (e.g., limiting the number of denoising steps applied) are possible but we focus on the basic QSM here and leave exploration design to future work.

Algorithm 2 Q-score matching for diffusion RL.

Initialize critic networks Q_{η_1}, Q_{η_2} and score network ε_θ with random parameters
Initialize target networks $\eta_1' \leftarrow \eta_1$, $\eta_2' \leftarrow \eta_2$
Fix $\lambda > 0$; initialize replay buffer $\mathcal{B}$
while not converged **do**
    Sample action by iterative denoising via (5.1): $a_t \sim \pi_\theta(\cdot \mid s_t)$
    Step environment: $s_{t+1}, r_{t+1} \leftarrow \text{env}(a_t)$
    Store $(s_t, a_t, r_{t+1}, s_{t+1})$ in $\mathcal{B}$
    Sample minibatch of N transitions from $\mathcal{B}$
    Sample $\tilde{a}_{t+1} \sim \pi_\theta(\cdot \mid s_{t+1})$
    Compute critic target: $y_t \leftarrow r_{t+1} + \gamma \min_{i=1,2} Q_{\eta_i'}(s_{t+1}, \tilde{a}_{t+1})$
    Update critics: $\eta_i \leftarrow \arg\min_{\eta_i} N^{-1} \sum (y_t - Q_{\eta_i}(s_t, a_t))^2$
    Sample noised actions $\tilde{a}$ at random denoising indices k
    Update score model: $\theta \leftarrow \arg\min_\theta N^{-1} \sum \|\varepsilon_\theta(\tilde{a}, s_t, k) - \lambda \nabla_a Q(s_t, \tilde{a})\|_2^2$
    Soft-update target networks: $\eta_i' \leftarrow \tau\eta_i + (1-\tau)\eta_i'$
end while

5.8 Experiments

We evaluate QSM on tasks from the DeepMind Control Suite (Tunyasuvunakool et al., 2020) and compare against SAC (Haarnoja et al., 2018), TD3 (Fujimoto et al., 2018b), and Diffusion-QL (Zhendong Wang et al., 2022). The questions:

1. Can QSM learn meaningful policies with limited environment interaction?
2. Does QSM learn complex, multimodal policies?
3. How does QSM compare to standard baselines?

Continuous-control evaluation

In Figure 5.6, we evaluate QSM on six DM Control tasks ranging from easy to medium difficulty. QSM matches or exceeds SAC and TD3 on every task tested, often improving on sample efficiency. Conventional actor-critic methods parameterize the actor as a Gaussian (SAC) or a transformed Gaussian (TD3), which can lead to suboptimal exploration — certain action modes may get dropped, or low-Q actions may be sampled too frequently. QSM, by contrast, learns policies not constrained to a fixed distribution class, and is able to cover multiple action modes.

A toy cartpole-swingup task (Figure 5.5) directly visualizes multimodality. The two optimal trajectories swing the pendulum in opposite directions; both lead to equivalent reward. QSM finds both modes; standard unimodal parameterizations cannot. The quadruped

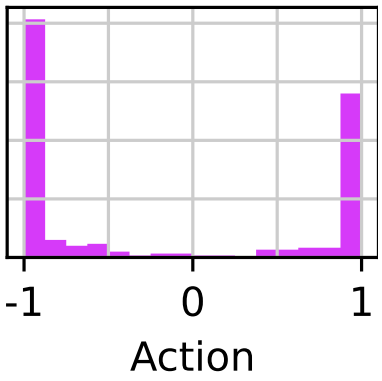


Figure 5.5: QSM learns multimodal policies on a toy cartpole-swingup task. Plotted are sampled actions from the policy at the first state of the task; -1 and $+1$ represent the initial action for the two optimal trajectories. The mass is split between both modes, consistent with two equally-good swingup directions.

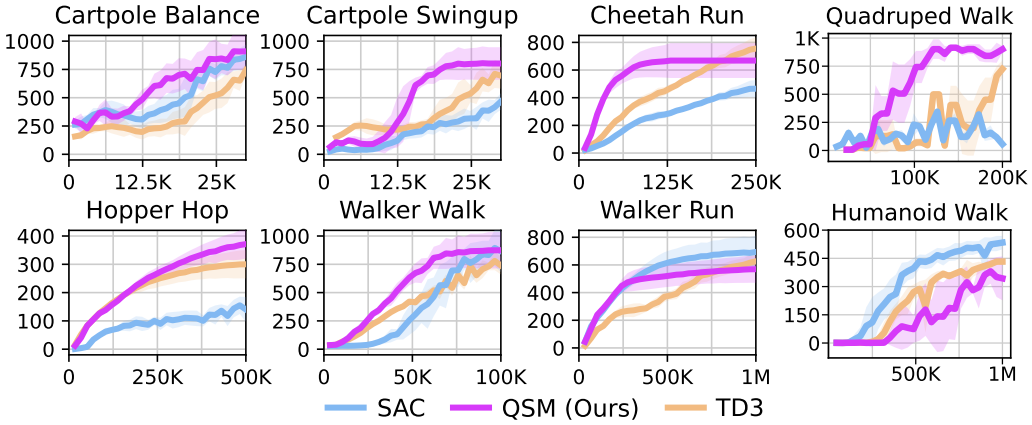


Figure 5.6: Experimental results across a suite of eight continuous-control tasks. QSM matches and sometimes outperforms TD3 and SAC, particularly in samples needed to reach high rewards. Even though QSM trains on expressive diffusion-model policies, it matches the sample efficiency of explicit Gaussian and tanh-Gaussian models.

multimodality figure (Figure 5.4) extends the same observation to a more realistic continuous-control task.

Policy gradient as a baseline

We also compare to a direct application of the policy-gradient formula (5.8) on the same diffusion-model policy architecture. As predicted by the variance argument in Section 5.3, this baseline performs substantially worse than QSM across continuous-control tasks (Figure 5.7). Additional environment-by-environment results are in Appendix B.6.

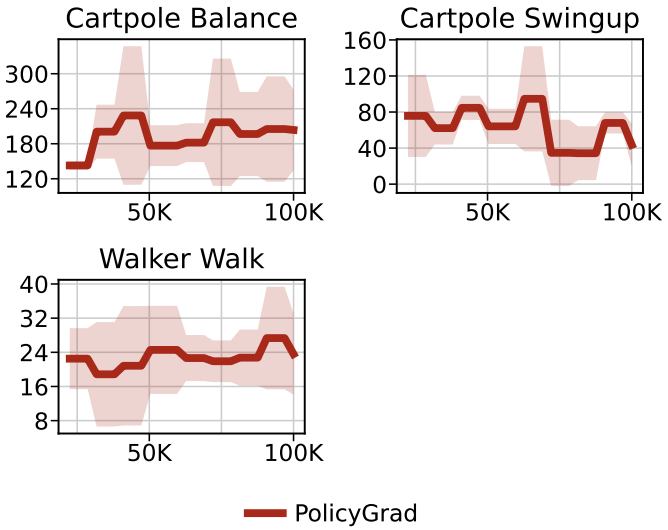


Figure 5.7: Performance when updating a diffusion-model policy via the policy-gradient formula (5.8) on select continuous-control tasks. Notably worse than QSM, matching the high-variance argument of Section 5.3. Further environments in Appendix B.6.

Comparison with Diffusion-QL

Figure 5.8 compares QSM and Diffusion-QL on shared hyperparameters. QSM performs comparably or better on every task, and is computationally cheaper per gradient step because it differentiates only through the denoising network rather than the full reverse-diffusion chain. Diffusion-QL occasionally gets "stuck" at suboptimal solutions, consistent with the unimodal-collapse phenomenon documented in Section 5.6. The QSM-unique parameter λ in Algorithm 2 is increased for harder tasks, which is the only difference between this comparison and the SAC/TD3 comparison of Figure 5.6 where QSM hyperparameters were tuned per environment.

5.9 Related work

Diffusion models in reinforcement learning are a relatively young but rapidly developing area. The closest neighbors of QSM fall into three categories.

Behavior cloning. "Behavior cloning" is a form of imitation learning where the agent mimics expert behavior without explicit reward feedback. Much of the earlier work on diffusion-model policies has been for behavior cloning (Janner et al., 2022; Reuss et al., 2023), since matching a distribution to a given dataset fits more closely to the original design principle of diffusion models, namely score matching (Hyvärinen and Dayan, 2005). Related work uses diffusion models to learn stochastic state dynamics (G. Li et al., 2022).

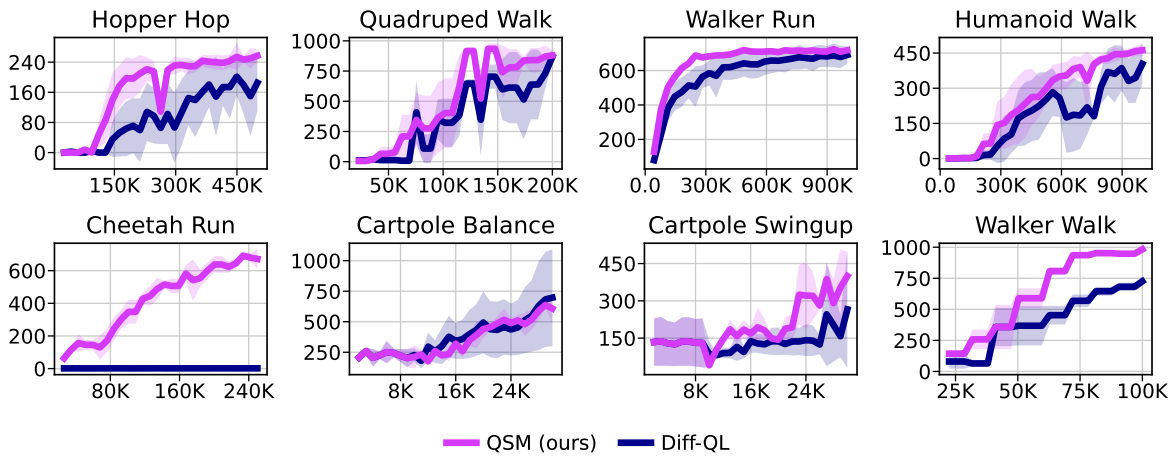


Figure 5.8: Comparison between QSM and Diffusion-QL (Zhendong Wang et al., 2022) on shared hyperparameters. While performance is mostly comparable, QSM takes significantly less computation time for the policy gradient and generally yields more consistent results, while Diffusion-QL can sometimes get stuck at suboptimal solutions. This is further supported by Figure 5.4, which demonstrates QSM's distinct promotion of exploration.

Offline Q-learning. Offline RL leverages existing datasets to learn optimal policies without further environment interaction. Similar to behavior cloning, optimizing the policy using a large fixed dataset fits closely with the common usage of diffusion models for learning distributions. In combination with the diffusion policy gradient formula above, this has enabled work in this area (Zhendong Wang et al., 2022; Suh et al., 2023; Kang et al., 2023; Hansen-Estruch et al., 2023; C. Lu et al., 2023).

Policy gradient. Policy gradient methods directly optimize the policy by computing gradients of the expected reward with respect to the policy parameters (Richard S Sutton et al., 1999). Previous work has derived a policy gradient formula for diffusion models (Black et al., 2023), but the formulas are general and do not exploit the score-based structure. As Section 5.3 showed, the resulting expectation depends on internal action samples, making estimates sample-inefficient.

There are also works that use the action gradient of the Q-function for learning (Silver, Lever, et al., 2014; Berseth et al., 2018; D'Oro and Jaśkowski, 2020; X. Li et al., 2022; Sarafian et al., 2021), where the standard policy gradient is expanded via the chain rule. These methods require an explicit representation of the full policy distribution $\pi(a|s)$, which a diffusion-model policy does not provide.

Diffusion-QL. Although Zhendong Wang et al., 2022 perform experiments in the offline-RL setting with a behavior-cloning term, they also propose pure Q-learning by training on Q itself as the objective and backpropagating through the diffusion model evaluation. This still does

not use the score-based structure of diffusion models and presents computational challenges (e.g., exploding/vanishing gradients from differentiating through composed evaluations of the same model). QSM avoids backpropagating through the entire diffusion evaluation, requiring gradients only through the denoising network, much like standard diffusion-model training.

Stochastic optimal control. At a conceptual level, QSM is rooted in stochastic optimal control (Fleming and Rishel, 2012; Kirk, 2004; Bellman, 1954), which deals with optimizing systems subject to random disturbances over time. Especially relevant is the continuous-time formulation, where control strategies adjust dynamically in response to evolving system states. Most stochastic optimal control literature assumes access to a model of the state dynamics; QSM instead assumes a model for expected discounted rewards (the Q-function), leading to a different theoretical pathway. Exploring this connection further is an interesting direction for future work, in particular given the surface similarities between Theorems 5.2, 5.4 and the Hamilton-Jacobi-Bellman equation (Bellman, 1954).

5.10 Reflection

The QSM loss (5.9) could have been guessed from dimensional analysis alone, without invoking calculus of variations. The variational framing explains its form: it aligns two vector fields induced by learned objects, its fixed points are Boltzmann policies on Q^*, and its structure preserves multimodality.

QSM also gives the simplest instance of smoothing-by-splitting in the dissertation. The reverse-diffusion chain still produces policy samples, but training does not differentiate through that chain. Because $\mathcal{L}_{\text{QSM}}$ uses only local evaluations, its gradient with respect to θ requires only local Jacobians of ε_θ.

Chapter 3 operates on an embedded data manifold, whereas QSM operates on a reverse-diffusion path. Despite this difference, both choose a functional whose stationary points correspond to a desired outcome and then use the variational formulation to make optimization tractable. That common structure is the connection between the two chapters.

Chapter 6

Onsager–Machlup Interpolation for Learned Generative Dynamics

This chapter is based on two papers: Psenka et al., 2025a on molecular conformational transitions, joint work with Sanjeev Raja, Martin Šípka, Tobias Kreiman, Michal Pavelka, and Aditi S. Krishnapriyan; and Psenka et al., 2025b on image-latent interpolation. The underlying SDE construction is the same in both, though the molecular setting uses the full action (Definition 6.1) and the image setting its truncated form (Definition 6.2); the chapter unifies the development, with detailed empirical treatment for both the molecular and image domains. Full proofs, implementation details, hyperparameters, and ablations are collected in Appendix C.

Transition path sampling in molecular systems and image interpolation share a mathematical structure. The former (Bolhuis, Chandler, et al., 2002; Christoph Dellago et al., 2002) seeks probable paths connecting stable conformations across a high-energy barrier on a potential energy landscape. The latter uses a learned generative model to find a smooth, high-likelihood path between two points in image space.

Both problems can be formulated as: given a learned object that supplies a vector field on the data space (an atomistic force field, a diffusion-model score), find a path between two specified endpoints that is most likely under the stochastic dynamics induced by that vector field. The right path-likelihood functional for such problems is the *Onsager–Machlup* (OM) action functional (Onsager and Machlup, 1953), a classical object from statistical physics whose minimizers correspond to most-probable paths under the underlying SDE.

The chapter has two technical contributions. First, the proteins side establishes that one can repurpose pretrained atomistic generative models (denoising diffusion, flow matching) for TPS in a zero-shot manner by interpreting candidate paths as trajectories of the SDE induced by the model's learned score function. Second, the vision side extends this framework to the image domain, where additional subtleties (the roughness of pixel-space transformations, the need for a smoothness-friendly path norm) motivate a generalized OM functional with a Gaussian-blurred kinetic term.

Within the dissertation's argument, this chapter is the second instance of sub-program B:

a variational functional defined on paths through a learned-model landscape, induced this time by a learned score, where the optimization is reshaped to remain well-conditioned against the roughness of the underlying landscape. A defining structural feature of OM, distinct from QSM (Chapter 5), is that the smoothness of the functional is a *continuously tunable parameter* of the same learned model that supplies it: the diffusion time τ parameterizes a continuous family of representations, ranging from sharp-and-faithful (small τ) to smooth-and-blurred (large τ). One picks the operating point on the accuracy–smoothness trade-off, on a per-task basis, without retraining.

The chapter is structured as follows. We first introduce the OM action as a path-likelihood functional under a constant-variance SDE, and describe how to construct a usable SDE from a score-based generative model (diffusion or flow matching). We then present two domains: molecular conformational transitions (the proteins paper) and image-latent interpolation (the vision paper). The chapter closes by formalizing the diffusion-time knob as a recurring structural feature of OM constructions, and reflecting on what makes the OM–QSM pair an instance of the dissertation's variational-unification claim.

6.1 From a generative model to a stochastic dynamical system

Stochastic dynamics with gradient-form drifts

We work throughout with the following constant-variance stochastic differential equation:

$$\mathrm{d}x \;=\; \tfrac{1}{\zeta}\,\mathbf{\Phi}(x)\,\mathrm{d}t + \sqrt{2D}\,\mathrm{d}W_t, \tag{6.1}$$

where $\mathbf{\Phi}(x) : \mathbb{R}^k \to \mathbb{R}^k$ is a drift field, W_t is a standard Wiener process, and $\zeta, D > 0$ are scalar constants governing damping and diffusion noise. We restrict to drifts that are gradient fields, $\mathbf{\Phi}(x) = -\partial\phi(x)/\partial x$, where $\phi : \mathbb{R}^k \to \mathbb{R}$.

In the molecular-dynamics setting, (6.1) is overdamped Langevin dynamics for an atomistic system at finite temperature: ϕ is the potential energy, $\mathbf{\Phi} = -\nabla U$ is the atomistic force field, and ζ, D are physical parameters (friction coefficient and diffusion coefficient, with $D = k_B T/\zeta$).

In the score-based generative-model setting, we will see that a pretrained model induces an SDE of the same form, with $\mathbf{\Phi}$ given by the model's learned score.

Stochastic dynamics under denoising diffusion

Denoising diffusion probabilistic models (DDPM) (Ho et al., 2020) are score-based generative models that learn to de-noise corrupted samples. The DDPM training objective is closely linked to the score-matching objective (Vincent, 2011; Song and Ermon, 2019) for a score model $s_\theta(x, \tau) : \mathbb{R}^k \times \mathbb{R}^+ \to \mathbb{R}^k$:

$$\mathcal{L}_{\mathrm{SM}}(\theta) \;=\; \mathbb{E}_{\tau, x\sim p_\tau}\left[\left\| s_\theta(x,\tau) - \nabla \log p_\tau(x)\right\|_2^2\right], \tag{6.2}$$

where p_τ is the noised marginal distribution. The trained denoising network and the score relate by $\varepsilon_{\theta^*}(x,\tau) \propto -\nabla \log p_\tau(x)$; the precise constant is $s_\theta(x,\tau) = -\varepsilon_\theta(x,\tau)/\sqrt{1-\bar{\alpha}_{\tau-1}}$. A formal derivation of the equivalence between DDPM training and score matching is in Appendix C.1.

To use the OM action with a DDPM, we need an SDE in the form of (6.1). The DDPM's standard sampling process (reverse denoising) is unsuitable: its score field is time-varying, so paths under that dynamics do not have a uniform likelihood interpretation. Instead, we leverage iterative one-step denoising and noising at a fixed latent time τ, which balances the likelihood-maximizing drift of the denoising step against the stochasticity of the noising step. Specifically, with $\beta_\tau, \alpha_\tau = 1-\beta_\tau, \bar{\alpha}_\tau = \prod_{i=1}^{\tau} \alpha_i$ the standard noise-schedule variables:

$$x^{(i,\mathrm{mid})} = \tfrac{1}{\sqrt{1-\beta_\tau}}\Big(x^{(i)} - \tfrac{\beta_\tau}{\sqrt{1-\bar{\alpha}_\tau}}\,\varepsilon_\theta(x^{(i)},\tau)\Big) + \sqrt{\beta_\tau}\,z', \tag{6.3}$$

$$x^{(i+1)} = \sqrt{1-\beta_\tau}\,x^{(i,\mathrm{mid})} + \sqrt{\beta_\tau}\,z, \tag{6.4}$$

with $z, z' \sim \mathcal{N}(0, I)$. Combining the two updates (writing $\mathbf{s}_\theta(x,\tau) = -\varepsilon_\theta(x,\tau)/\sqrt{1-\bar{\alpha}_{\tau-1}}$):

$$x^{(i+1)} \;=\; x^{(i)} + \tfrac{\beta_\tau\sqrt{1-\bar{\alpha}_{\tau-1}}}{\sqrt{1-\bar{\alpha}_\tau}}\,\mathbf{s}_\theta(x^{(i)},\tau) + \sqrt{2\beta_\tau - \beta_\tau^2}\,z. \tag{6.5}$$

Taking the continuum limit $\beta_\tau \to 0$, $2\beta_\tau - \beta_\tau^2 \approx 2\beta_\tau$, and $\bar{\alpha}_\tau \to \bar{\alpha}_{\tau-1}$, equation (6.5) is the Euler-Maruyama discretization of the SDE

$$\mathrm{d}x \;=\; \mathbf{s}_\theta(x,\tau)\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}W_t. \tag{6.6}$$

This is the same form as (6.1) with $\zeta = 1$, $D = 1$, $\mathbf{\Phi}(x) \approx \nabla \log p_\tau(x)$, $\phi \propto -\log p_\tau$. The construction holds for any $\tau \in \{1, \ldots, T_d\}$; a similar derivation appears in Arts et al. (2023) for $\tau = 0$.

Extension to flow matching

The same construction extends to flow matching (Lipman, R. T. Chen, et al., 2022), a more general class of score-based generative models. A flow matching model learns a velocity field $u_\theta(x,\tau)$ via

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{\tau, x\sim p_\tau}\big[\|u_\theta(x,\tau) - u_\tau(x)\|_2^2\big], \tag{6.7}$$

$$u_\tau(x) = \mathbb{E}_{x_0\sim p_0,\, x_1\sim p_1}\big[\dot{\alpha}_\tau x_1 + \dot{\sigma}_\tau x_0 \mid x = \alpha_\tau x_1 + \sigma_\tau x_0\big], \tag{6.8}$$

with affine curves $\alpha_\tau, \sigma_\tau : [0,1] \to [0,1]$. While extracting the score from a DDPM is straightforward, doing so for flow matching requires a conversion. The proof in Appendix C.1 shows that the score is related to the velocity field by

$$\nabla \log p_\tau(x) \;=\; \frac{\alpha_\tau}{\dot{\sigma}_\tau\sigma_\tau\alpha_\tau - \dot{\alpha}_\tau\sigma_\tau^2}\Big(\frac{\dot{\alpha}_\tau}{\alpha_\tau}x - u_\tau(x)\Big). \tag{6.9}$$

Plugging the resulting $\mathbf{s}_\theta^{\mathrm{FM}}(x,\tau)$ into (6.6) gives the same SDE form. The OM framework therefore applies equally to diffusion and flow matching; for concreteness, the rest of the chapter focuses on the diffusion case.

6.2 The Onsager–Machlup functional

Given the SDE (6.1), the probability of a path $x(\cdot) = \{x(t)\}_{t=0}^{1}$ is given by

$$P[x(\cdot)] \;\propto\; e^{-S[x(\cdot)]}, \tag{6.10}$$

where S is the *Onsager-Machlup action functional*, the stochastic analog of the classical principle of least action (Rojo et al., 2018). Maximizing the path probability is equivalent to minimizing S.

In practice we work with discretized paths. The discretized OM action is the following.

Definition 6.1 (Onsager–Machlup action)**.** For a discrete path $\mathbf{X} = \{x^{(0)}, \dots, x^{(L)}\}$ generated by (6.1) with drift $\mathbf{\Phi}$ and timestep Δt, the discretized OM action is

$$\begin{aligned} S[\mathbf{X}] &= \tfrac{1}{2D}\big(A[\mathbf{X}] + B[\mathbf{X}] + C[\mathbf{X}]\big), \\ A[\mathbf{X}] &= \tfrac{1}{2\Delta t}\sum_{i=0}^{L-1} \big\|x^{(i+1)} - x^{(i)}\big\|_2^2, \\ B[\mathbf{X}] &= \tfrac{\Delta t}{2\zeta^2}\sum_{i=1}^{L-1} \big\|\mathbf{\Phi}(x^{(i)})\big\|_2^2, \\ C[\mathbf{X}] &= \tfrac{D\Delta t}{\zeta}\sum_{i=1}^{L-1} \nabla\cdot\mathbf{\Phi}(x^{(i)}). \end{aligned} \tag{6.11}$$

The full action of Definition 6.1 is derived from the Fokker-Planck equation for the path probability, in the Stratonovich convention, in Appendix C.2. A separate δ-tube derivation in the Itô convention is given in Appendix C.2; it yields the *truncated* action of Definition 6.2, as the divergence term C requires the Stratonovich midpoint convention and does not survive the Itô tube limit. The term C also requires the drift $\mathbf{\Phi}$ to be C^1 with well-defined divergence.

Reading the three terms. Each summand of the OM action has an intuitive interpretation. Term A rewards *smoothness*: small displacements between adjacent path points. Term B rewards paths that pass through regions of *low-norm drift*, equilibria or saddle points of the underlying dynamics. Term C rewards paths in regions of *low drift divergence*, which favors convex regions of the underlying potential ϕ where dynamics are more stable. The parameters $\zeta, \Delta t, D$ control the relative weighting of these terms in a physically intuitive manner: e.g., at larger Δt the contribution of term A is diminished, consistent with the intuition that larger jumps are more probable per unit time.

Truncated OM action. In the limit of negligible diffusivity D (low-temperature limit), the divergence term can be dropped, yielding a simpler form we will use frequently in molecular settings.

Definition 6.2 (Truncated OM action)**.** The *truncated* OM action is

$$S[\mathbf{X}] \;=\; \tfrac{1}{2D}\big(A[\mathbf{X}] + B[\mathbf{X}]\big). \tag{6.12}$$

We use the full or truncated form depending on the system: when the divergence term is dominated by quadratic potential terms (whose Laplacian is constant and uninformative), or when D can be taken small, the truncated form is preferred.

Working form for diffusion models. Substituting (6.6) into Definition 6.1 with $\zeta = 1$, $D = 1$, $\Delta t = \beta_\tau$:

$$S(\mathbf{X};\theta^*) = \tfrac{1}{2D}\sum_{i=0}^{L-1}\Big(\tfrac{1}{2\Delta t}\left\|x^{(i+1)} - x^{(i)}\right\|_2^2 + \tfrac{\Delta t}{2\zeta^2}\left\|\mathbf{s}_{\theta^*}(x^{(i)},\tau)\right\|_2^2 + \tfrac{D\Delta t}{\zeta}\nabla\cdot\mathbf{s}_{\theta^*}(x^{(i)},\tau)\Big). \tag{6.13}$$

The diffusion-time parameter β_τ can be tuned by changing the sampling discretization fidelity. For TPS on a Boltzmann-distributed dataset, the parameters $\Delta t, \zeta, D$ can be set to physically interpretable values matching standard MD simulations (see Section 6.2).

Physical interpretation for Boltzmann-distributed data

In the special case where the data are Boltzmann distributed, $p(x) \propto \exp(-U(x)/k_BT)$, the learned score at $\tau = 0$ is interpretable as a physical atomistic force field (Arts et al., 2023):

$$\mathbf{s}_{\theta^*}(x, \tau = 0) \;\approx\; \nabla \log p(x) \;\propto\; -\nabla U(x) \;=\; \mathbf{F}(x). \tag{6.14}$$

The constructed SDE (6.6) at $\tau = 0$ reduces to overdamped Langevin dynamics on a potential energy surface, and we can directly set the OM action constants (Δt, ζ, D, T) to physical values used in MD simulations. Finding high-probability paths between points on a Boltzmann-distributed manifold thus aligns with conventional TPS on an atomistic system at finite temperature.

This is the setup of the proteins side of the chapter; we develop it in Section 6.3 below.

6.3 Domain A: molecular conformational transitions

Sampling configurational ensembles of high-dimensional molecular systems is a grand challenge in statistical mechanics and computational science (Tuckerman, 2023; Frenkel and Smit, 2023). A key sub-problem is the sampling of *rare transition events* between stable configurations, such as in chemical reactions or protein folding (Bolhuis, Chandler, et al., 2002; Christoph Dellago et al., 2002). The challenge is energy barriers: rare events occur on timescales far longer than the fastest dynamical motions of the system.

A rich line of literature on enhanced-sampling techniques (Torrie and Valleau, 1977; Swendsen and J.-S. Wang, 1986; Laio and Parrinello, 2002a; Laio and Gervasio, 2008; Tiwary

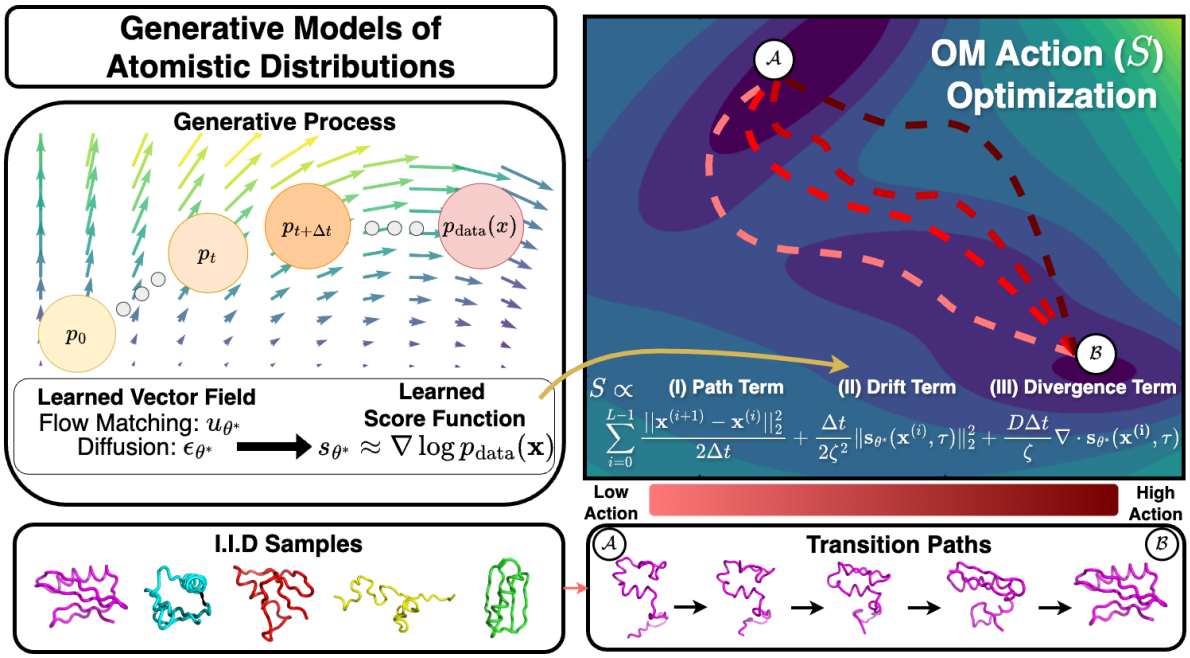


Figure 6.1: Schematic of OM action optimization. **(Left)** An atomistic generative model produces i.i.d. samples by integrating a learned vector field; the same vector field is an estimate of the score $\mathbf{s}_{\theta^*} \approx \nabla \log p_{\text{data}}$, which we interpret as the drift of a stochastic dynamics. **(Right)** The OM action functional, evaluated on a discretized path, has three terms: low displacement between adjacent path points, low-norm drifts, and curvature/stability of the underlying energy landscape. The score $\mathbf{s}_{\theta^*}$ remains frozen throughout OM optimization; only the intermediate path points are updated.

and Parrinello, 2013; Valsson and Parrinello, 2014) addresses this. More recently, machine learning methods have gained popularity for accelerating TPS by learning a control drift term (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) or the increasing availability of high-quality atomistic generative models (Abramson et al., 2024; Lewis et al., 2024; Jing, Berger, et al., 2024; S. Zheng et al., 2024).

Our approach. We propose a conceptually simple post-training method to repurpose generative models for TPS in a zero-shot manner. The core idea exploits that score-based generative models induce stochastic Langevin dynamics on the data manifold, governed by their learned score function (Section 6.1). Drawing on statistical mechanics, the path probabilities of these dynamics are characterized by the Onsager-Machlup action (Section 6.2). We can identify high-probability transition paths between arbitrary points on the data manifold by minimizing the OM action with gradient-based optimizers.

In the special case where the data are Boltzmann distributed, the learned score approximates the underlying atomistic force field, and our approach has the direct physical interpretation of TPS on a potential energy surface at finite temperature.

The approach has three advantages over prior work:

- The method uses pretrained generative models and requires no TPS-specific training procedure, so it can transfer to new systems without retraining.
- Physical parameters such as time horizon and temperature can be incorporated while sampling paths without modifying the underlying generative model.
- The stochasticity of generative models produces diverse candidate paths efficiently.

The TPS setup

Formally, we consider d-dimensional molecular systems with N_p particles interacting under a potential energy function $U : \mathbb{R}^{N_p \times d} \to \mathbb{R}$. A configuration of the system is $x \in \Omega \subset \mathbb{R}^{N_p \times d}$. The goal of TPS is to find most-likely paths $\{x^{(i)}\}_{0 \le i \le L}$ over a time horizon T_p, with Δt the simulation timestep and $L = T_p/\Delta t$ the number of discretization points. The trajectory traverses two fixed endpoints $x^{(0)} \in \mathcal{A}$ and $x^{(L)} \in \mathcal{B}$, where $\mathcal{A}, \mathcal{B} \subset \Omega$ are typically distinct minima of U. The underlying particle dynamics are overdamped Langevin (the SDE (6.1) with $\phi = U$ and $\mathbf{\Phi} = -\nabla U$).

If the forces $\mathbf{F}(x)$ are available (either from a classical force field or from a trained generative model via (6.14)), the OM action (6.13) computes the probability of paths connecting $x^{(0)}, x^{(L)}$ under the governing dynamics.

Method

Given a pretrained generative model with fixed score $\mathbf{s}_{\theta^*}$ and two atomistic configurations $x^{(0)} \in \mathcal{A}, x^{(L)} \in \mathcal{B}$, we aim to sample a transition path $\mathbf{X} = \{x^{(i)}\}_{i=0}^{L}$ of L discrete steps. The core inductive bias is to interpret candidate paths as realizations of the denoise-noise SDE (6.6), enabling tractable computation and optimization of path log-likelihoods via the OM action.

The method has three primary steps: (1) computing an initial guess path, (2) performing OM optimization, and (in some cases) (3) decoding the optimized path back to configurational space.

Initial guess. The choice of initial path connecting $x^{(0)}$ and $x^{(L)}$ matters: naïve linear interpolation in configurational space is unphysical, since plausible atomistic structures lie on a non-convex low-dimensional manifold of Ω. We instead linearly interpolate at a latent level τ_{initial} of the pretrained model, which produces samples closer to the data manifold. After interpolation, we can either (a) decode back to configurational space and optimize the

Algorithm 3 Onsager-Machlup transition path optimization with generative models.

Input: optimization time τ_{opt}; generative model with time-conditional score $\mathbf{s}_{\theta^*}(\cdot, \tau)$; two atomistic configurations $x^{(0)} \in \mathcal{A}, x^{(L)} \in \mathcal{B} \subset \mathbb{R}^{N_p \times d}$
Compute initial guess $\mathbf{X} = \{x^{(0)}, \ldots, x^{(L)}\}$ via Kabsch-aligned linear interpolation at latent level τ_{initial}, then optionally decode
while not converged **do**
    Compute OM action $S[\mathbf{X}; \theta^*]$ with learned vector field $\mathbf{s}_{\theta^*}(\cdot, \tau_{\text{opt}})$ via (6.13)
    $\mathbf{X} \leftarrow \text{OPTIMIZER}(\mathbf{X}, \nabla_{\mathbf{X}} S[\mathbf{X}; \theta^*])$ (endpoints $x^{(0)}, x^{(L)}$ fixed)
end while

OM action there, or (b) optimize directly at the latent level τ_{initial}. Both are valid, since the SDE (6.6) is valid for any τ.

OM optimization. Starting from the initial guess, we minimize the OM action (6.13) via gradient descent on the intermediate path points (keeping the endpoints $x^{(0)}, x^{(L)}$ fixed):

$$\mathbf{X}^* = \underset{\mathbf{X}=\{x^{(i)}\}_{i=0}^{L}}{\arg\min} \; S[\mathbf{X}; \theta^*]. \tag{6.15}$$

Since S is a discretized integral, the entire trajectory is optimized in parallel, which is amenable to multi-device acceleration. In line with Arts et al. (2023), we find that a small nonzero τ_{opt} leads to better alignment with the true forces, and treat it as a hyperparameter. To accelerate computation of the divergence term, we use the Hutchinson trace estimator (Hutchinson, 1989) (Appendix C.10).

Decoding. If OM optimization was performed at a non-zero latent time, we decode the final path back to configurational space via the model's denoising process. If optimization was already done in configurational space, this step is skipped.

Algorithm 3 summarizes the procedure.

Results

We validate the approach on systems of increasing complexity. Detailed implementation, hyperparameters, and additional analyses are in Appendices C.5–C.8.

2D Müller-Brown potential

The 2D Müller-Brown (MB) potential (Müller and L. D. Brown, 1979) is a canonical TPS test system with a global minimum and two local minima, separated by saddle points. We train a denoising diffusion model on samples from the underlying potential, then perform OM optimization between the two deepest minima.

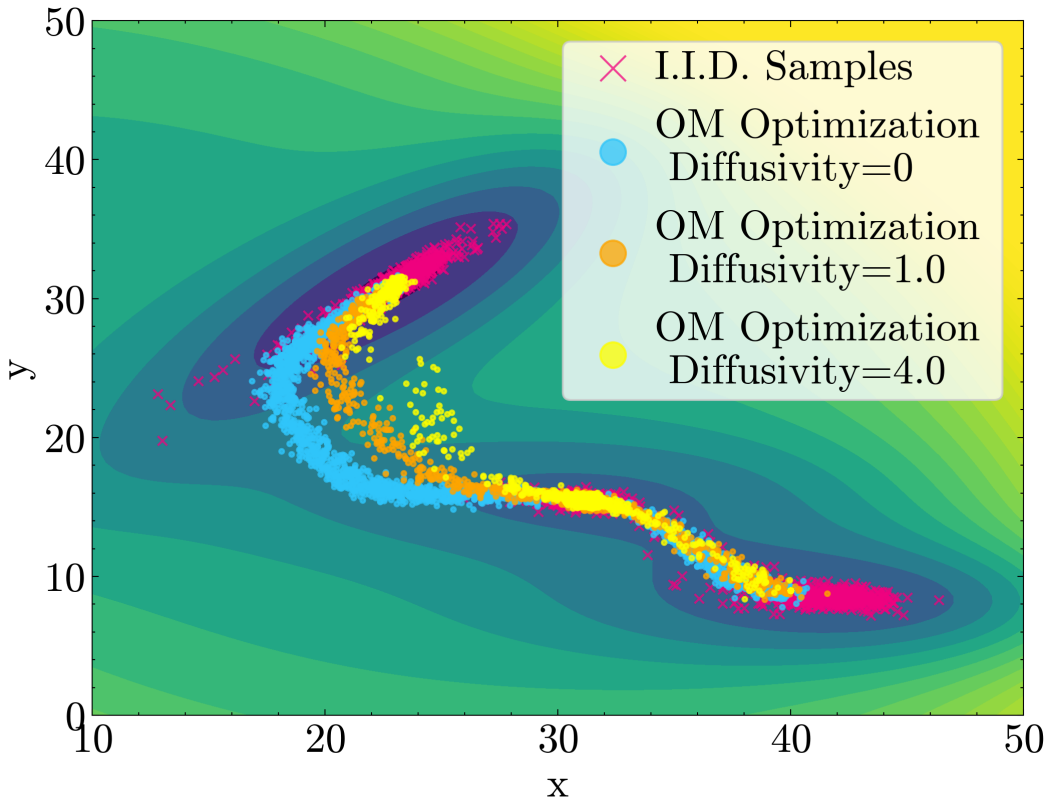


Figure 6.2: OM optimization with a diffusion model on the 2D Müller-Brown potential. Individual points along the OM-optimized paths are shown as dots. Increasing the diffusivity D causes the path to cross at a higher barrier. An equivalent number of i.i.d. samples (red) fails to sample the transition region.

As shown in Figure 6.2, OM optimization with the truncated action yields a transition path (blue) between the energy minima passing through the lowest energy barrier. Due to the stochastic decoding process, samples around the transition path exhibit natural diversity. Increasing the diffusion coefficient $D = k_B T/\gamma M$ results in qualitatively different paths (orange and yellow): samples are more concentrated near the energy minima, and the path crosses a higher barrier, consistent with the larger scale of thermal fluctuations at increased temperatures. Meanwhile, 2500 i.i.d. samples (red) sample only the three energy minima and fail to sample the transition region. This is the proof of concept: OM optimization can repurpose generative models for TPS without specialized training.

Committor and rate estimation. The committor function $q(x)$ (Vanden-Eijnden et al., 2006) is the probability that a trajectory initiated at $x^{(0)} = x$ reaches $\mathcal{B}$ before $\mathcal{A}$. The transition rate k can be estimated from a differentiable committor via

$$k = \frac{k_B T}{\gamma} \langle |\nabla_x q(x)|^2 \rangle_\Omega \approx \frac{k_B T}{\gamma} \hat{\mathbb{E}}_{x \sim \mathcal{D}_{\text{train}}} |\nabla_x q_\theta(x)|^2, \tag{6.16}$$

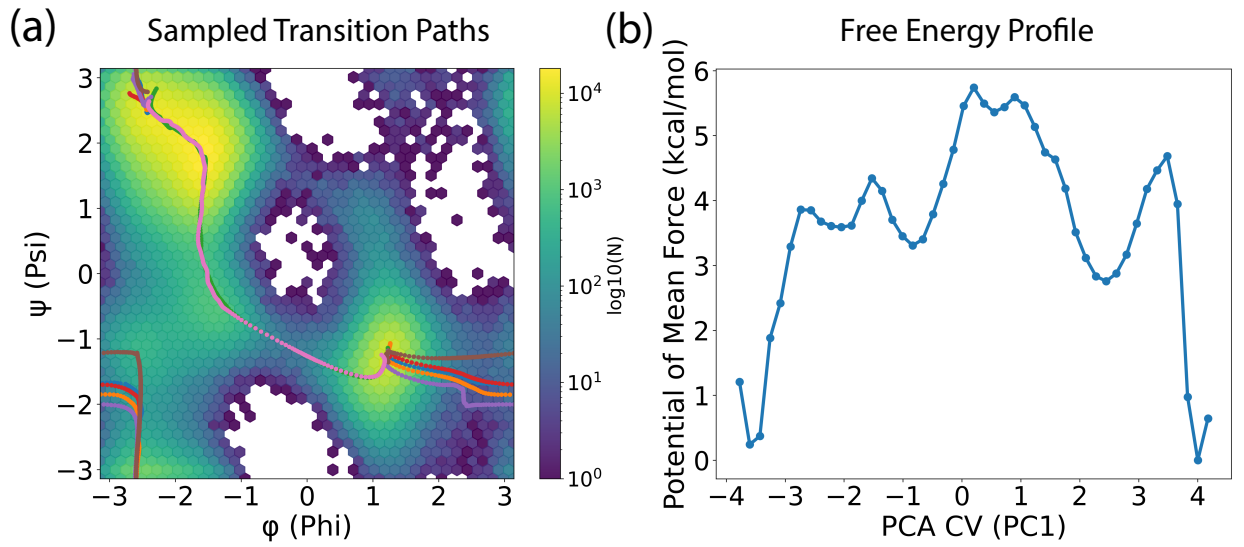


Figure 6.3: OM optimization and free-energy barrier estimation on alanine dipeptide using a pretrained diffusion model. **(a)** Sampled transition paths from OM optimization, overlaid on a Ramachandran plot of sample density at 600 K. **(b)** Free-energy barrier for a selected path, estimated via umbrella sampling, is approximately 6 kcal/mol, in line with metadynamics calculations (Appendix C.6).

where the ensemble average is replaced by an empirical mean over a training set. We can use OM-optimized paths as an enhanced-sampling method to accurately compute q on the Müller-Brown potential: initiate MD simulations from points along the OM paths to collect $\mathcal{D}_{\text{train}}$, train a committor model q_θ by solving a functional optimization problem derived from the Backward Kolmogorov Equation (Hasyim et al., 2022), then estimate the rate. We obtain $k_\theta = 1.3 \times 10^{-5}$, compared with a true rate of 5.4×10^{-5} from numerical solution of the BKE. Estimating a rate within the correct order of magnitude is often challenging (Rotskoff et al., 2022; Hasyim et al., 2022); our estimate matches to within a factor of ≈ 4. Details of the committor training and rate estimation are in Appendix C.5.

Alanine dipeptide

Alanine dipeptide is a classical benchmark for TPS, with 22 atoms and CVs described by the dihedral angles ϕ, ψ. We pretrain a denoising diffusion model on samples from the alanine dipeptide potential energy surface, then use it for OM optimization between the standard CV-space minima.

We successfully sample two likely transition paths between the metastable basins (Figure 6.3a). The headline efficiency comparison is in Table 6.1: OM optimization requires significantly fewer evaluations and lower compute per path than traditional enhanced-sampling methods. We can also use the OM-optimized paths as a natural CV for umbrella sampling

Table 6.1: Benchmarking OM optimization against traditional enhanced-sampling baselines on alanine dipeptide. We report the number of force-field (or score-function) evaluations required per 1,000-step transition path. OM optimization requires significantly fewer evaluations and lower compute per path.

Method	**CVs?**	**# FF evals / path** (↓)	**Runtime / path** (↓)
MCMC (two-way shooting)	No	$\geq 10^9$	$\geq$ 100 hr
Metadynamics	Yes	10^6	10 hr
OM Opt. (diffusion model) (ours)	No	$\mathbf{10^4}$	**50 min**

(Torrie and Valleau, 1977), from which we accurately estimate free-energy profiles (Figure 6.3b).

Fast-folding coarse-grained proteins

We next consider proteins exhibiting fast dynamical transitions, for which millisecond-scale reference MD simulations were performed by Lindorff-Larsen et al. (2011). We adopt a coarse-graining scheme representing each amino acid by the position of its C_α atom, using pretrained diffusion models from Arts et al. (2023) and training our own flow-matching models. Separate models are trained for each protein. We use Markov State Models (MSMs) (Prinz et al., 2011; Noé, H. Wu, et al., 2013) to obtain state transition probabilities and evaluate transition-path quality via three metrics (following Jing, Stärk, et al. (2024)):

1. *Transition negative log-likelihood (TNLL).* The negative log-likelihood of MSM state transitions under the reference MSM, averaged over valid paths.

2. *Fraction of valid paths.* The fraction of paths with nonzero probability under the reference MSM.

3. *Jensen-Shannon divergence (JSD).* Distributional dissimilarity between generated and reference MSM state distributions.

As shown in Figure 6.4a, OM optimization yields diverse transition paths that pass through high-density regions of the free-energy landscape projected onto the two slowest TIC (Noé, H. Wu, et al., 2013) axes. For the BBA protein, the paths robustly sample the transition state ensemble (Figure 6.4b). Sampling transition paths with OM optimization requires considerably less wall-clock time than using the diffusion model's score as a coarse-grained force field for unbiased MD (Figure 6.4c). Across all proteins and both classes of generative model (diffusion, flow matching), OM optimization yields a higher fraction of valid paths and lower transition negative log-likelihood under the reference MSM than unbiased MD of any considered length up to 100 μs (Figure 6.4d). The JSD is also lower than any MD length up to 50 μs.

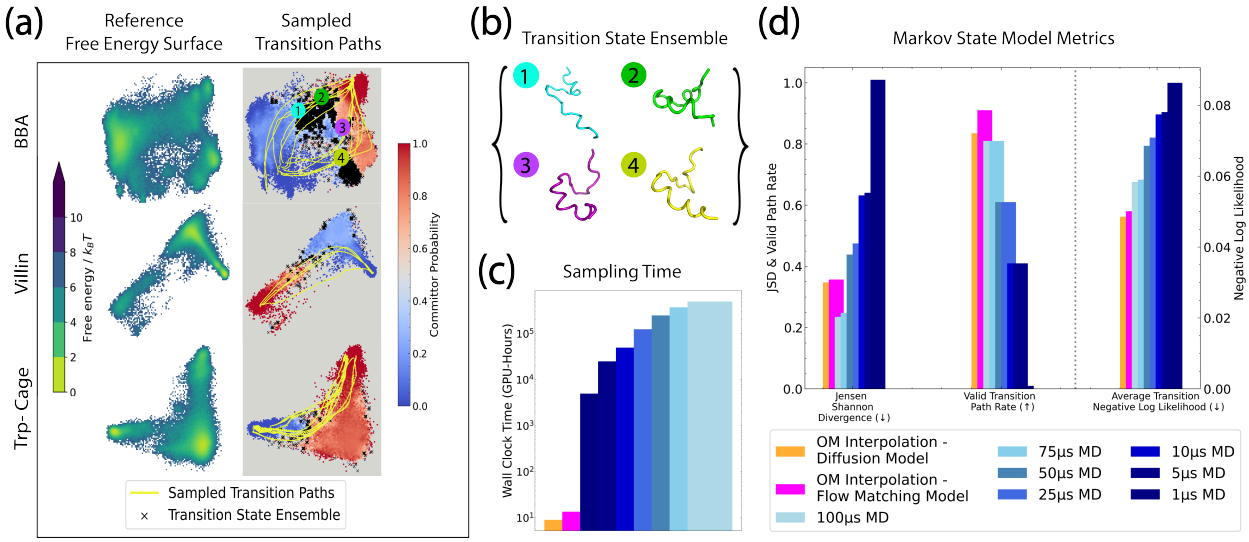


Figure 6.4: OM optimization with diffusion and flow matching models trained on coarse-grained fast-folding proteins. **(a)** Reference free-energy surfaces alongside transition paths from OM optimization (yellow) overlaid on the empirically computed committor function $q(x)$; the transition state ensemble (black) is $\{x : 0.45 \leq q(x) \leq 0.55\}$. **(b)** Samples from the predicted transition state ensemble of BBA. **(c)** Runtime of OM optimization vs. unbiased MD of varying lengths. **(d)** MSM metrics averaged across 5 fast-folding proteins; comparisons are vs. unbiased MD of varying lengths.

Robustness to sparse data. To simulate the scenario where transition states are underrepresented in training data, we retrain diffusion models on datasets with 99% of configurations between committor 0.1 and 0.9 removed. OM optimization still samples plausible transition paths (Appendix C.7), suggesting the approach is useful even when the underlying data distribution is not an exhaustively sampled Boltzmann distribution.

Tetrapeptide generalization

As a final test, we consider all-atom tetrapeptide systems, posing the challenge of generalization to held-out amino acid sequences. We train a flow-matching model on $\approx$ 3,000 tetrapeptides simulated in Jing, Stärk, et al. (2024) and apply OM optimization to generate an ensemble of 16 transition paths for each of 100 tetrapeptides *not seen* during training.

As shown in Figure 6.5, OM optimization achieves MSM metrics competitive with 50-100 ns MD simulations (considerably more computationally expensive). This suggests the approach generalizes across chemical space: a single trained generative model supports TPS on atomistic systems not explicitly seen during training.

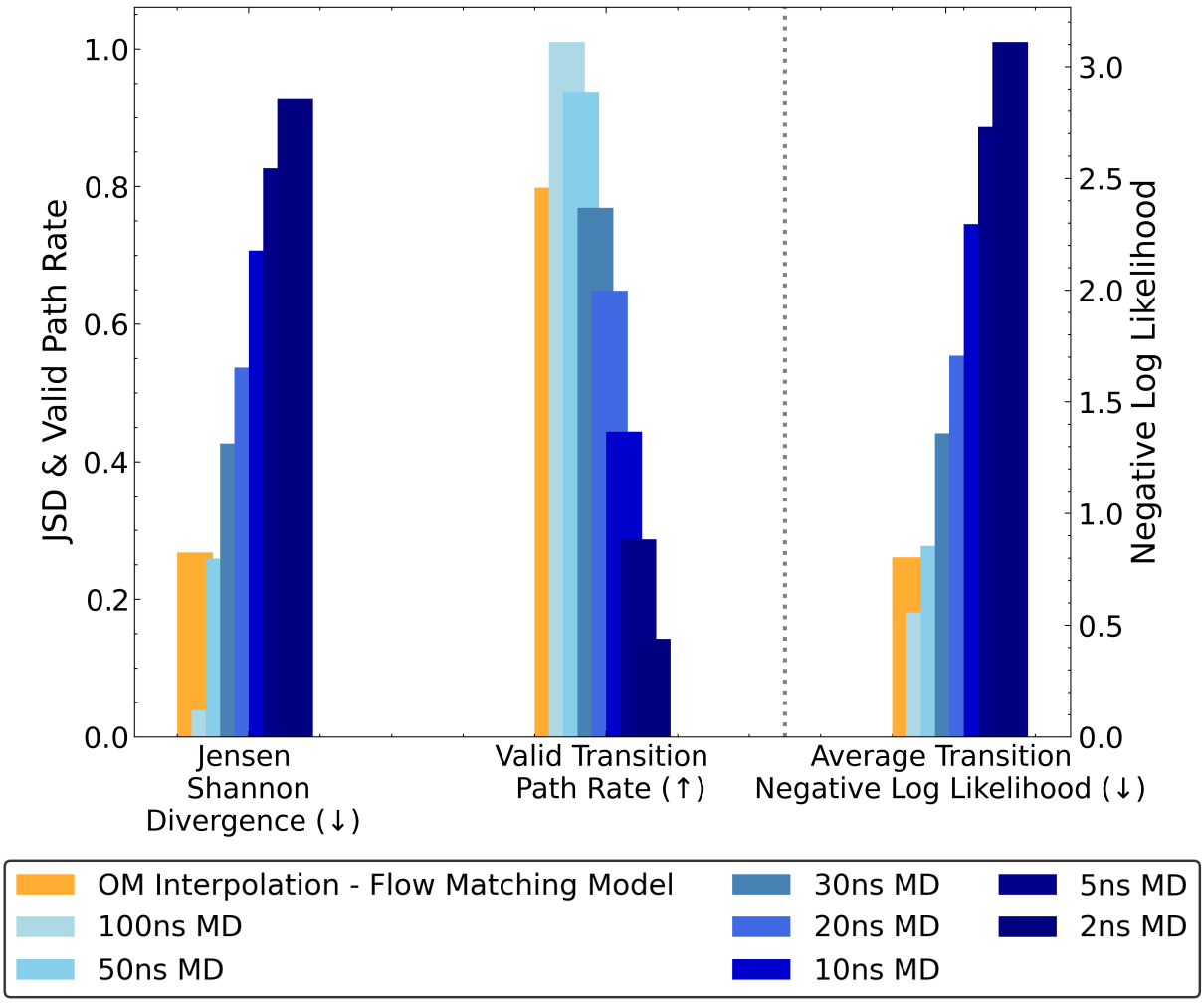


Figure 6.5: OM optimization on unseen tetrapeptide sequences. Transition paths with a flow matching model compare well to variable-length MD simulations, indicating the approach's generalization potential. Plotted are the JSD, fraction of valid paths, and transition negative log-likelihood under the reference MSM.

6.4 Domain B: image-latent interpolation

The OM framework developed above transfers directly from molecular configuration space to image space. In the image setting it also provides a continuous knob that trades representation accuracy against optimization smoothness, with no qualitative analog in the molecular setting.

Image interpolation has applications across computer vision: video frame interpolation, morphing, visualizing latent spaces in generative models. Current approaches struggle with temporal coherence because they lack a physically grounded motion prior. Motion priors can be extracted via optical flow estimation, but the mapping from pixel space to optical

flow is difficult to optimize through. The challenge is generating physically plausible image interpolations from a generative model alone.

Our approach. By transferring the OM machinery to images, we get a method that operates entirely in data space, without latent inversion or training-time editing. Specifically, we use the score network of any off-the-shelf diffusion model as a local force field, analytically differentiate the OM action, and find image geodesics. Generalizing the OM functional with a Gaussian-blur norm makes it well-suited to image trajectories, where pixel-space discontinuities cause naïve L^2 norms to break.

Image manifold geodesics

Let $\mathcal{X} = \mathbb{R}^{c\times w\times h}$ be the image space (with c channels, width w, height h), and let $x_0, x_1 \in \mathcal{X}$ be fixed source and target images. An interpolation is a continuous path $\gamma : [0,1] \to \mathcal{X}$ with $\gamma(0) = x_0$, $\gamma(1) = x_1$. We want γ both perceptually smooth and probable under the data distribution.

A natural formalization of smoothness is via *geodesics*. For an embedded submanifold $\mathcal{M} \subset \mathbb{R}^d$ and two points $x_0, x_1 \in \mathcal{M}$, the geodesic is the curve of minimal length:

$$\gamma_g = \operatorname*{arg\,min}_{\gamma\in\Gamma_{\mathcal{M}}(x_0,x_1)} \int_0^1 \|\dot\gamma(t)\|_\star \,\mathrm{d}t, \tag{6.17}$$

where $\Gamma_{\mathcal{M}}(x_0, x_1)$ is the set of differentiable curves in $\mathbb{R}^d$ lying in $\mathcal{M}$ with endpoints x_0, x_1, and $\|\cdot\|_\star : \mathcal{X} \to \mathbb{R}^+$ is some ambient norm. The constraint set $\mathcal{M}$ is the set of high-likelihood points,

$$\mathcal{M}_\rho = \{x \in \mathbb{R}^d : p(x) \geq \rho\}, \tag{6.18}$$

for some threshold ρ. Minimized path length captures *smoothness*; the manifold constraint captures *realism*.

OM action on image trajectories

We use the SDE construction from Section 6.1 to generate image trajectories. Starting from x_0 and iteratively noising/denoising at fixed latent time τ gives the SDE (6.6). Translating into the path-norm form, minimizing the following quantity finds the maximum-likelihood trajectory under (6.6):

$$\mathcal{A}_{\mathrm{OM}}[\gamma \mid x_0] \;=\; \int_0^1 \tfrac{1}{\varepsilon}\,\|\dot\gamma(t) - F(\gamma(t))\|_2^2 \,\mathrm{d}t, \tag{6.19}$$

where F is the SDE drift (a scaling of $\mathbf{s}_\theta(\cdot,\tau)$). With fixed endpoints x_0, x_1, this expands (using a Cauchy-Schwarz-style argument) to the canonical separated OM action:

$$\mathcal{A}_{\mathrm{OM}}[\gamma \mid x_0, x_1] \;=\; \int_0^1 \left(\|\dot\gamma(t)\|^2 + \|F(\gamma(t))\|^2\right) \mathrm{d}t, \tag{6.20}$$

which reveals a *kinetic term* $\|\dot\gamma(t)\|^2$ and a *force/score term* $\|F(\gamma(t))\|^2$. Plugging in the diffusion drift, the loss to minimize is

$$\mathcal{L}_{\mathrm{OM}}[\gamma] \;=\; \int_0^1 \|\dot\gamma(t)\|^2 \,\mathrm{d}t + \int_0^1 \tfrac{1}{1-\bar\alpha_{\tau-1}} \,\|\varepsilon_{\theta^*}(\gamma(t),\tau)\|^2 \,\mathrm{d}t. \tag{6.21}$$

The simplified OM action defines a path "energy" that combines path smoothness (via the kinetic term) with image realism along the path (via the force term).

Relaxed-constraint geodesic optimization

The L^2 norm in the kinetic term fails for image sequences with hard edges or high-frequency content: it is non-smooth in these cases and can yield infinite path lengths in the continuum limit (Wakin et al., 2005). However, we can generalize to any norm $\|\cdot\|_\star$:

$$\gamma_g = \operatorname*{arg\,min}_{\gamma\in\Gamma_{\mathcal{M}}(x_0,x_T)} \int_0^1 \|\dot\gamma(t)\|_\star \,\mathrm{d}t. \tag{6.22}$$

Ideally $\|\cdot\|_\star$ captures object motion (e.g., centroid velocity of an object in the image), but extracting motion via optical flow is difficult to optimize through. We aim for a tractable $\|\cdot\|_\star$ whose gradient is well-behaved yet whose values are smaller for physically realizable motions.

For explicit control of the smoothness-vs-realism tradeoff, introduce a hyperparameter $\lambda > 0$:

$$\mathcal{L}_I[\gamma] \;=\; \int_0^1 \|\dot\gamma(t)\|_\star^2 \,\mathrm{d}t + \lambda \int_0^1 \|\varepsilon_{\theta^*}(\gamma(t),\tau)\|^2 \,\mathrm{d}t. \tag{6.23}$$

This is no longer a literal OM action (the alternative norm breaks the identification), but can be derived as a penalty-relaxed version of the constrained geodesic problem

$$\begin{aligned} \gamma^* &= \operatorname*{arg\,min}_{\gamma\in\Gamma_{\mathcal{M}}(x_0,x_T)} \int_0^1 \|\dot\gamma(t)\|_\star^2 \,\mathrm{d}t, \\ \mathcal{M}_{\delta,\tau} &= \{x \mid \|\varepsilon_{\theta^*}(x,\tau)\|_2 \le \delta\}. \end{aligned} \tag{6.24}$$

The minimizer is a geodesic over $\mathcal{M}_{\delta,\tau}$ with respect to $\|\cdot\|_\star$. The squared and unsquared formulations have the same minimizers under standard regularity (Appendix C.3), so this is a valid generalization.

Gaussian blur norm

Considering images as functions $x(p) : [0,1]^2 \to [0,1]$, discontinuities make some natural image trajectories have infinite path length: pixels at the edge flip discontinuously as the edge moves. Convolution with a smooth kernel resolves this. We use a Gaussian kernel, often deployed in computer vision as a low-pass filter (Marr and Hildreth, 1980; Zhou Wang et al., 2003; Pham et al., 2008; Kostková et al., 2020; Zhengwu Zhang et al., 2013; Iluz et al., 2023).

Algorithm 4 Relaxed Onsager-Machlup-based image interpolation.

Input: two endpoints $x_1, x_L \in \mathcal{X}$; path length L; force weight λ; blur scale σ; diffusion model $\varepsilon_\theta(x, \tau)$ at fixed latent time τ
Initial path $\gamma = \{x_1, \dots, x_L\}$ via linear or spherical interpolation
while not converged **do**
    Compute blurred trajectory $\gamma_\sigma[i] = \mathcal{G}_\sigma \circledast \gamma[i]$ for all i
    Path derivative via finite differences $\dot{\gamma}_\sigma[i] = L(\gamma_\sigma[i+1] - \gamma_\sigma[i])$
    Compute generalized OM action $\mathcal{E}_I[\gamma] = \|\dot{\gamma}_\sigma\|_2^2 + \lambda \|\varepsilon_\theta(\gamma, \tau)\|_2^2$
    $\gamma \leftarrow \text{OPTIMIZER}(\gamma, \nabla_\gamma \mathcal{E}_I[\gamma])$ (endpoints fixed)
end while
return optimized geodesic γ_g

Define the *Gaussian-blur norm*:

$$\begin{aligned} \|\dot{x}_t\|_\sigma &= \|\mathcal{G}_\sigma \circledast \dot{x}_t\|_2 , \\ \mathcal{G}_\sigma(x) &= e^{-\|x\|_2^2/(2\sigma^2)}. \end{aligned} \tag{6.25}$$

Under this norm, image trajectories with edges and motion have finite length. Trajectories become smoother under $\|\cdot\|_\sigma$ as $\sigma \to \infty$, but very large σ pushes image space to a point and makes the norm trivial. The choice of σ is the fundamental tradeoff: too low and motions are rough; too high and trajectories blur together. In practice we choose the smallest variance that produces visually noticeable blurring.

Theorem C.5 in Appendix C.4 establishes that $\|\cdot\|_\sigma$ is a norm induced by a bounded, injective operator on L^2 images (of operator norm $\leq 2\pi\sigma^2$), and that trajectories of pure-translation motion have finite length under $\|\cdot\|_\sigma$.

We set the image-space norm in the kinetic term of the OM action to the Gaussian-blur norm: $\|\cdot\|_\star = \|\cdot\|_\sigma$.

Latent denoising models

Diffusion models are often implemented in a smaller latent space $\mathcal{Z}$ rather than raw image space $\mathcal{X}$, via a learned encoder/decoder pair $F : \mathcal{X} \to \mathcal{Z}, G : \mathcal{Z} \to \mathcal{X}$. We empirically find that decoders typically preserve locality, in the sense that $G(z \circledast \mathcal{G}_\sigma) \approx \mathcal{G}_\sigma \circledast G(z)$. Hence, applying the Gaussian-blur norm in the latent space is approximately equivalent to applying it in image space, with no need to backprop through the decoder.

Algorithm

The full image interpolation algorithm is given in Algorithm 4.

Controllable image editing

We extend the framework to image editing. Given an original image x_0 and a textual or semantic condition c, an editor produces a new image x_1 satisfying c while changing as little as possible from x_0. We work with a trained conditional noise model $\varepsilon_{\theta^*}(x, \tau, c)$.

A standard sensitivity in editing is the tradeoff between (a) *preservation* of original content and (b) *faithfulness* to the new condition c. Rather than returning a single edited image, we return an entire image trajectory $\gamma(t) : [0, 1] \to \mathcal{X}$ between the original and edit, allowing the user to select where on the preservation-faithfulness curve to operate. We pick the midpoint of the trajectory to lie at the Pareto frontier:

$$\begin{aligned} \gamma(0.5) &= \arg\min_{x \in \mathcal{M}_{\tau,c}} d(x, \gamma(1)), \\ \mathcal{M}_{\delta,\tau,c} &= \{x \mid \|\varepsilon_{\theta^*}(x, \tau, c)\|_2 \le \delta\}, \end{aligned} \tag{6.26}$$

where $\mathcal{M}_{\tau,c}$ is the set of denoising-near-stationary points under fixed condition c.

The trajectory splits into two pieces. For $t \in [0, 0.5]$, $\gamma(t)$ goes from outside $\mathcal{M}_{\delta,\tau,c}$ to inside. We use a higher diffusion time $\mathcal{T}$ to model dynamics near $\gamma(0)$, annealing down to τ:

$$\int_0^{0.5} \|\dot\gamma(t)\|_\star^2 \, \mathrm{d}t + \lambda \int_0^{0.5} \|\varepsilon_{\theta^*}(\gamma(t), \mathcal{T}(1-2t) + 2t\tau, c)\|^2 \, \mathrm{d}t. \tag{6.27}$$

For $t \in [0.5, 1]$, $\gamma(t)$ lies entirely within $\mathcal{M}_{\tau,c}$, and we optimize as a geodesic with fixed τ:

$$\int_{0.5}^1 \|\dot\gamma(t)\|_\star^2 \, \mathrm{d}t + \lambda \int_{0.5}^1 \|\varepsilon_{\theta^*}(\gamma(t), \tau, c)\|^2 \, \mathrm{d}t. \tag{6.28}$$

The combined objective is

$$\begin{aligned} \mathcal{L}_{\text{edit}}[\gamma] &= \int_0^1 \|\dot\gamma(t)\|_\star^2 \, \mathrm{d}t + \lambda \int_0^1 \|\varepsilon_{\theta^*}(\gamma(t), f(t), c)\|^2 \, \mathrm{d}t, \\ f(t) &= \begin{cases} \mathcal{T}(1-2t) + 2t\tau & \text{if } t \le 0.5, \\ \tau & \text{otherwise.} \end{cases} \end{aligned} \tag{6.29}$$

Results

We evaluate on three datasets of increasing complexity: MNIST, CelebA-HQ, and synthetic images from Stable Diffusion (Rombach et al., 2022). For each, we analyze quantitative metrics and qualitative results, comparing against established baselines.

Evaluation metrics. Pixel-space norms penalize sharpness for moving edges; even our Gaussian-blur metric does not penalize small unphysical jitter. To extract physical motion, we run an optical-flow algorithm (RAFT (Teed and J. Deng, 2020)) on the interpolation,

producing pixel-wise velocities. From the velocities, we use the *principle of minimal jerk* (Flash and Hogan, 1985; Yan et al., 2000): physical motion minimizes third-order temporal derivatives. Define the jerk-normalized smoothness metric

$$M_{\text{smooth}}(\gamma) \;=\; \frac{\int_0^1 \left\| \frac{\mathrm{d}^2}{\mathrm{d}t^2} v(t) \right\|_2^2 \mathrm{d}t}{\int_0^1 \|v(t)\|_2^2 \,\mathrm{d}t}, \tag{6.30}$$

where $v(t)$ is the extracted optical flow. Lower M_{smooth} indicates more physically plausible trajectories.

For realism, we measure the perceptual (LPIPS) distance (R. Zhang et al., 2018) of each interpolation frame to the reference dataset and sum:

$$M_{\text{realism}}(\gamma) \;=\; \int_0^1 d(\gamma(t), X)\,\mathrm{d}t, \tag{6.31}$$

where X is the diffusion model's training set and $d(w, X) = \min_{x \in X} \mathrm{LPIPS}(w, x)$.

CelebA-HQ visualizations. Figure 6.6 shows visual results on CelebA-HQ. The optimized trajectories exhibit smooth transitions and maintain high perceptual quality throughout, without relying on fading components in and out (characteristic of L^2-based norms).

Comparison to baselines. Figure 6.7 compares OM-interpolation against linear-latent, spherical-latent (slerp), NoiseDiffusion (P. Zheng et al., 2024), and NAO (Samuel et al., 2023) baselines on CelebA-HQ. OM generates tamer image trajectories under optical-flow-based metrics, capturing physical motion rather than pixel changes while retaining comparable realism. Optimizing the kinetic-vs-force tradeoff slightly degrades realism, but the resulting trajectories appear more physical.

Effect of λ. The parameter λ in (6.23) controls the kinetic-vs-force tradeoff. Higher λ gives sharper individual frames at the cost of larger pixel-space jumps; lower λ gives smoother trajectories at the cost of occasional blurry intermediates. Figure 6.8 shows the effect on MNIST.

6.5 Formalizing the diffusion-time knob

The image-latent setting makes the role of the diffusion time τ especially visible, but the same structural observation applies to any application of OM to a learned generative model. The OM action S_τ depends on τ through the drift $\mathbf{s}_\theta(\cdot, \tau)$, and the diffusion model provides $\mathbf{s}_\theta$ at a continuous range of τ as a byproduct of training.

As noted at the chapter opening, the diffusion time makes the functional's smoothness continuously tunable without retraining the learned model. The operating point on the



Figure 6.6: Visual interpolations on CelebA-HQ using our OM interpolation method. The optimized trajectories transition smoothly while maintaining per-frame quality, without the fading characteristic of L^2-based interpolations.

accuracy–smoothness tradeoff can therefore be chosen separately for each path problem at inference time.

At small τ, the OM minimizer produces interpolations that lock onto high-density regions of the data distribution; the intermediate frames are individually plausible but the path between them is jagged. At large τ, the temporal velocity is much smoother, but the intermediate frames are blurrier, closer to the noised distribution at scale τ than to the underlying clean image distribution. This behavior illustrates the dissertation's broader observation that *the best representation for planning is not the best representation for reconstruction.* A single diffusion model supplies this continuously parametrized family of representations.

In the language of Chapter 4's smoothing-by-splitting principle, the "splitting" here is between diffusion times. We use the learned score at the noise level where the underlying score field is smoothest enough to support well-conditioned optimization, and we pay a price in fidelity to the clean distribution. Unlike the QSM construction of Chapter 5, where the smoothing was structural (we replaced a long backprop by a local matching loss), the smoothing here is parametric and continuously tunable.

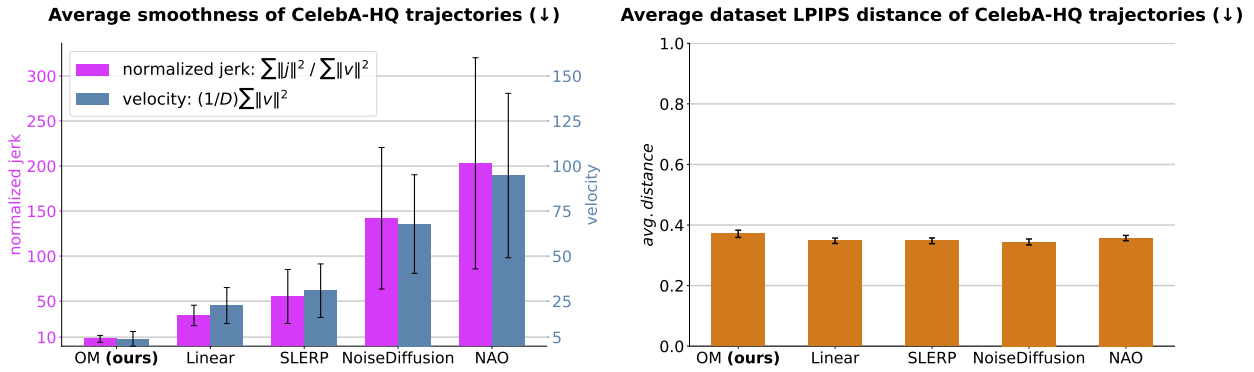


Figure 6.7: OM image-latent interpolations on CelebA-HQ (Karras, Aila, et al., 2017) using a pretrained DDPM (Ho et al., 2020), compared with linear-latent, spherical-linear (slerp), NoiseDiffusion (P. Zheng et al., 2024), and norm-aware (NAO) (Samuel et al., 2023). Optical flow via RAFT (Teed and J. Deng, 2020), derivatives by finite differences. Velocities normalized by embedding dimension. Error bars: 95% confidence intervals.

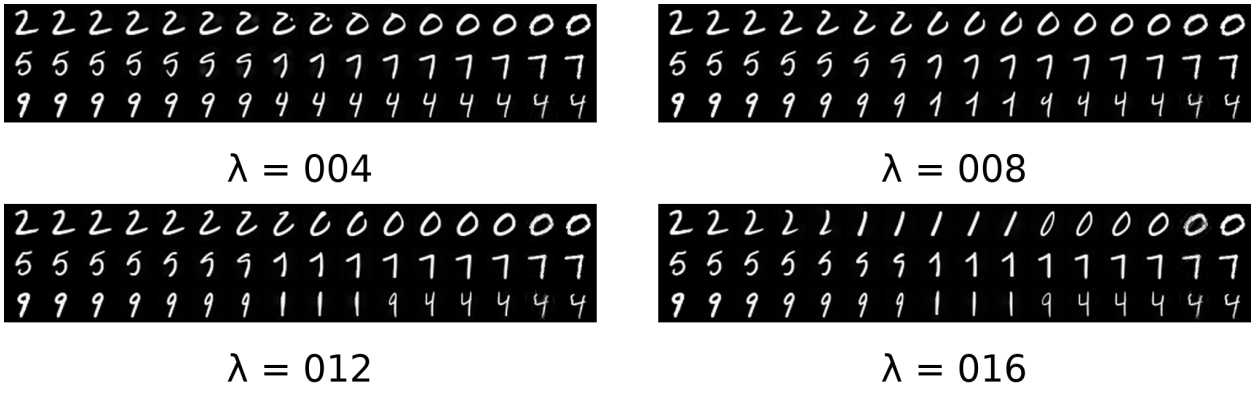


Figure 6.8: Three MNIST interpolations under varying λ. Lower λ values produce smoother trajectories at the cost of occasional blurry intermediate images; higher λ values produce sharper individual frames at the cost of larger pixel-space jumps. The interpolations also explore different topological routes between the endpoints as λ varies.

6.6 Related work

Transition path sampling. Classical TPS (Bolhuis, Chandler, et al., 2002; Christoph Dellago et al., 2002; Vanden-Eijnden et al., 2010) encompasses many methods. Umbrella sampling (Torrie and Valleau, 1977) and metadynamics (Laio and Parrinello, 2002b; Laio and Gervasio, 2008) use biasing potentials along chosen collective variables, which are hard to define near transition states (Sultan and Pande, 2018; Rogal et al., 2019; W. Chen and Ferguson, 2018; Sun et al., 2022; Šípka et al., 2023). Shooting methods (Mullen et al., 2015; Borrero and Dellago, 2016; Jung et al., 2017; Bolhuis and Swenson, 2021) use Metropolis-Hastings sampling but suffer from slow mixing and high rejection rates. ML-based methods

(Das et al., 2021; Rose et al., 2021; Singh and Limmer, 2023; Seong et al., 2024; Liang et al., 2023; Sipka et al., 2023; Singh and Limmer, 2023; Du et al., 2024) require expensive sampling procedures and re-training per system. Interpolation-based methods like Nudged Elastic Band (Henkelman et al., 2000) construct paths via splines between images. Minimizing the OM action has been explored for low-dimensional systems (Faccioli et al., 2006; Vanden-Eijnden and Heymann, 2008; Autieri et al., 2009; Fujisaki et al., 2010; Juyong Lee et al., 2017; Beccara et al., 2012), but has lacked integration with modern auto-differentiation. To our knowledge, our proteins work is the first to combine gradient-based optimization of the complete OM action with the stochastic dynamics induced by pretrained generative models.

Atomistic generative models. Generative models for atomistic configurations include Boltzmann generators (Noé, Olsson, et al., 2019), two-state models (Arts et al., 2023), and protein-specific architectures (S. Zheng et al., 2024; Jing, Berger, et al., 2024; Lewis et al., 2024; Schreiner et al., 2023). Recent work adapts these models for diverse sampling (Corso et al., 2023), rare-event sampling (Falkner et al., 2023), MD simulation via the diffusion-force-field connection (Arts et al., 2023), and trajectory generation (Jing, Stärk, et al., 2024). Our OM framework repurposes such models for dynamical sampling (transition paths) without retraining.

Score-based and flow-based generative models. Diffusion models (Ho et al., 2020; Song and Ermon, 2019) estimate score gradients at varying noise levels. Flow-matching models (Lipman, R. T. Chen, et al., 2022) learn vector fields between noise and data distributions; the two families are interconvertible via standard transforms (Albergo et al., 2023). The OM machinery applies to any generative model from which a time-indexed score can be extracted.

Generative latent-space interpolation. GAN-based methods (I. Goodfellow et al., 2020; Radford et al., 2015; Karras, Laine, et al., 2019) demonstrated semantic transitions via linear latent interpolation, though sensitive to mode collapse. VAEs (Kingma and Welling, 2013; Berthelot et al., 2018) give principled probabilistic encoder/decoder pairs but typically produce blurrier interpolations. Diffusion-based interpolation methods (Samuel et al., 2023; P. Zheng et al., 2024; K. Zhang et al., 2024; Zhaoyuan Yang et al., 2023; Shen et al., 2024; Saito and Matsubara, 2025; Deschenaux et al., 2024) optimize trajectories in latent space and decode back; they tend to produce sharper per-frame results than VAE interpolation but have no direct mechanism for controlling smoothness in image space. The OM approach is distinguished by minimizing a functional whose terms include a smoothness penalty and whose continuous diffusion-time parameter directly controls the smoothness-fidelity tradeoff.

Geometric interpolation in deep learning. Geometric notions of interpolation including geodesics in learned latent spaces (Arvanitidis et al., 2018), manifold-flattening approaches (Psenka, Pai, et al., 2024) (Chapter 3), and image-manifold geodesic methods (Saito and Matsubara, 2025) are conceptual cousins of OM. They differ in what they optimize: classical

geodesic methods minimize a length functional on a learned (smooth) manifold, while OM minimizes a path-likelihood functional under an SDE induced by the learned model. This is exactly the sub-program A vs. sub-program B distinction.

6.7 Reflection: OM as a score-induced functional

The functional (6.13) depends on the score model $\mathbf{s}_\theta$ only through its values and divergences at points on the path. It does not require a particular architecture or training schedule, beyond the regularity needed for the SDE and divergence term to be well-defined. The OM machinery therefore transfers across generative models with relatively little adaptation. Flow-matching (Albergo et al., 2023) models give a drift directly, while rectified-flow models give a more deterministic version; OM-style functionals should apply to both after accounting for their different noise schedules.

The connection to QSM in Chapter 5 is structural. Each method defines a functional on paths through a learned-model landscape, induced by a learned vector field: a score for OM and a critic action gradient for QSM. Their stationary-point characterizations connect minimizers to target distributions, namely the SDE's path likelihood for OM and a Boltzmann policy on Q^* for QSM. Thus molecular dynamics and reinforcement learning use the same variational structure here despite solving different problems.

Chapter 7

Parallel Long-Horizon Planning over Learned World Models

This chapter is based on Psenka, LeCun, et al., 2026, joint work with Michael Rabbat, Aditi Krishnapriyan, Yann LeCun, and Amir Bar. It engages directly with the roughness of learned representations. The functional is induced by a learned dynamics map; the optimization is a parallel penalty-relaxation scheme that exposes, and then accommodates, an adversarial-robustness failure mode of visual world models. Full proofs, additional experiments, and hyperparameter details are in Appendix D.

Intelligent agents carry a small-scale model of external reality which allows them to simulate actions, reason about their consequences, and choose the ones leading to the best outcomes. Attempts to build such models date back to control theory; in recent years researchers have made progress on learning world models from raw sensory inputs (e.g., vision). Recent world models have shown success modeling computer games (Valevski et al., 2024), navigating real-world environments (Bar et al., 2025; Ball et al., 2025), and predicting robot arm motion (Goswami et al., 2025). Applications range from simulating complex medical procedures (Koju et al., 2025) to testing robots in visually realistic environments (Guo et al., 2025).

Current planning algorithms over world models often rely on zeroth-order optimization methods such as the Cross-Entropy Method (CEM) (Rubinstein and Kroese, 2004) or, more generally, *shooting methods* that plan by iteratively rolling out trajectories and choosing actions from the optimal rollout (Bock and Plitt, 1984; Piovesan and Tanner, 2009). These approaches are simple and robust, but their performance degrades with longer planning horizons and higher action dimensionality (Bharadhwaj et al., 2020), motivating the use of gradient information when differentiable world models are available.

Gradient-based planners exploit the differentiability of learned world models to directly optimize action sequences, enabling more sample-efficient planning than zeroth-order methods. Two main challenges:

1. *Local minima.* The action-space objective is non-convex; greedy gradient descent gets

stuck in poor basins (Jyothir et al., 2023).

2. *Instability from rollout differentiation.* Backpropagation through the rollout is akin to BPTT (Werbos, 2002): gradients explode or vanish over long horizons.

While classical optimal control developed multiple-shooting and direct-collocation methods for better conditioning (Von Stryk, 1993), these were typically designed for known dynamical systems and do not scale gracefully to deep neural dynamics (often in latent spaces) with brittle Jacobians.

Our approach. We introduce a gradient-based planner that decouples temporal dynamics into parallel optimized states (rather than serial rollouts), while remaining robust at long horizons and in high-dimensional state spaces. We optimize over *lifted intermediate states* treated as independent variables. Two key additions handle the gradient-sensitivity and local-minima issues:

Suppressing rough state-gradient pathways. In high-dimensional learned state spaces, gradients with respect to state inputs can be brittle or adversarial. To address this, our planner deliberately stops gradients through the state inputs of the world model while retaining gradients with respect to actions. This alone biases trajectories toward the starting state, so we add a dense one-step goal loss whose gradients pull trajectories toward the goal.

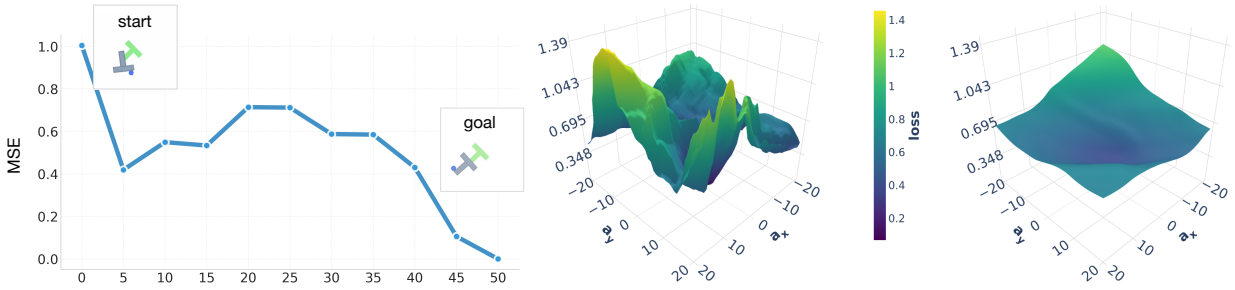
Langevin exploration. To address the remaining non-convexity of the lifted-state approach, the planner incorporates Langevin-style stochastic updates on lifted state variables, encouraging exploration of the state space and facilitating escape from poor basins. We intermittently apply a small step of true serial-rollout GD to fine-tune.

Together, these components yield a practical gradient-based planner for learned visual dynamics that remains stable at long horizons. We call it **GRASP** (Gradient RelAxed Stochastic Planner). In various settings we achieve up to +10% success rate at less than half the compute time. We provide theoretical analysis covering the linear-convergence properties of the lifted approach, the regularity benefits of noise injection, the stationary distribution under state-only Langevin noise, and the nonexistence of exact dynamics-enforcing losses that avoid state gradients. Experiments on visual world models in D4RL (Fu et al., 2020) and the DeepMind Control Suite (Tassa et al., 2018) validate the planner.

Within the dissertation's argument, this chapter gives sub-program B's response to rough learned dynamics. The functional is induced by a learned dynamics map; the smoothing-by-splitting principle of Chapter 4 is realized by replacing a T-deep backprop with a local pairwise dynamics-satisfaction loss; the gradient-cut variant is a targeted intervention against an adversarial-robustness pathology that emerges only in the visual setting.

7.1 Problem formulation

Our object of interest is a learned world model $F_\theta : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ predicting the next state given the current state and action. For visual domains, states are typically represented in



(a) Similarity Loss Across Time (b) Loss Landscape (GD) (c) Loss Landscape (ours)

Figure 7.1: Difficulty of the planning problem. **(a)** Distance to the goal in L^2 norm throughout a successful trajectory: optimal plans frequently first move away from the goal before approaching it, so greedy gradient descent gets stuck. **(b)-(c)** Loss-landscape slices at convergence: the standard serial gradient-descent objective is jagged with many local minima; the GRASP loss is comparatively smooth. Slices along two random orthogonal unit directions in the full action space, Push-T environment at horizon $T = 50$.

a learned latent space to handle high-dimensional observations. We assume $\mathcal{A} = \mathbb{R}^k$ is a continuous Euclidean action space.

We consider the problem of *fixed-goal path planning*: finding an action sequence $\mathbf{a} = (a_0, a_1, \ldots, a_{T-1})$ that, with respect to the dynamics of F_θ and a given initial state $s_0 \in \mathcal{S}$, reaches a set goal state $g \in \mathcal{S}$:

$$\mathcal{F}_\theta^T(\mathbf{a}, s_0) = g, \tag{7.1}$$

where the terminal state s_T is generated recursively through $s_{t+1} = F_\theta(s_t, a_t)$:

$$\mathcal{F}_\theta^T(\mathbf{a}, s_0) \;\coloneqq\; \underbrace{F_\theta(\ldots F_\theta(F_\theta(}_{T \text{ times}} s_0, a_0), a_1), \ldots, a_{T-1}). \tag{7.2}$$

We can sufficiently compute $\mathbf{a}^*$ by solving

$$\mathbf{a}^* \;=\; \arg\min_{\mathbf{a}} \left\| \mathcal{F}_\theta^T(\mathbf{a}, s_0) - g \right\|_2^2. \tag{7.3}$$

Optimizing (7.3) directly is challenging for two reasons. First, it requires T applications of F_θ, which is expensive and difficult to optimize due to poor conditioning from repeated composition (Appendix D.1). Second, it is susceptible to local minima and a jagged loss landscape (Figure 7.1). For these reasons, existing planners rely on zero-order methods like CEM and MPPI (Williams et al., 2016), which are highly stochastic and gradient-free.

In what follows, we propose a gradient-based planner that alleviates these difficulties while using the differentiability of F_θ. In Section 7.2 we lift the optimization by adding states as free variables, giving faster convergence and better conditioning. In Section 7.2 we introduce stochasticity to escape local minima. In Section 7.2 we suppress the brittle state-gradient pathway, which is the chapter's most distinctive intervention.

7.2 Decoupling dynamics for gradient-based planning

We consider planning with F_θ and horizon T. Given $s_0 \in \mathcal{S}$ and goal $g \in \mathcal{S}$, we optimize $\mathbf{a} = (a_0, \ldots, a_{T-1})$ such that the rolled-out state $s_T(\mathbf{a})$ is as close to g as possible. The standard approach defines a trajectory by rolling out the model, but backpropagating through a deep composition of F_θ can be unstable. Following prior work on lifted planning (Tamimi and P. Li, 2009; Rybkin et al., 2021), we introduce auxiliary states $\mathbf{z} = (z_1, \ldots, z_T)$ and enforce dynamics consistency through a penalty function.

Parallelized planning

Rewriting (7.3) in terms of each intermediate dynamics condition:

$$\min_{\mathbf{s},\mathbf{a}} \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a}) \coloneqq \sum_{t=0}^{T-1} \|F_\theta(s_t, a_t) - s_{t+1}\|_2^2, \quad \text{with } s_0 \text{ fixed}, \; s_T = g. \tag{7.4}$$

This is equivalent to (7.3) in that they share global minimizers.

The immediate benefit: *all world-model evaluations are parallel.* There is no need for serial rollouts. However, two issues arise with directly optimizing (7.4):

1. *Local minima.* When optimizing over states, iterates may settle in unphysical regions; for example, states going straight through a barrier (Figure D.2). Addressed via Langevin state updates (Section 7.2).

2. *World-model sensitivity for high-dimensional states.* For high-dimensional (e.g., vision-based) states, the Jacobian $J_s F_\theta(s, a)$ does not necessarily have a nice low-dimensional or convex structure. In practice, the world model can be steered toward any desired output (Figure 7.3). Addressed in Section 7.2 by reshaping descent directions.

Exploration via Langevin state updates

The lifted optimization in (7.4) is still non-convex and can be trapped in poor local minima. Deterministic joint updates in $(\mathbf{a}, \mathbf{s})$ often converge to "bad" stationary points where intermediate variables settle into unfavorable basins (e.g., a linear route ignoring barriers, Figure D.2). To circumvent this, we inject stochasticity into the state iterates, yielding a Langevin-style update that encourages exploration.

Langevin dynamics on state iterates. A standard way to escape spurious basins is to replace deterministic gradient descent on $\mathbf{s}$ with overdamped Langevin dynamics (S. B. Gelfand and Mitter, 1991), whose Euler discretization takes the form

$$\begin{aligned} s_t^{k+1} &\leftarrow s_t^k - \eta_s \nabla_{s_t} \mathcal{L}_{\text{dyn}}(\mathbf{s}^k, \mathbf{a}^k) + \sigma_{\text{state}} \xi_t^k, \\ a_t^{k+1} &\leftarrow a_t^k - \eta_a \nabla_{a_t} \mathcal{L}_{\text{dyn}}(\mathbf{s}^k, \mathbf{a}^k), \end{aligned} \tag{7.5}$$

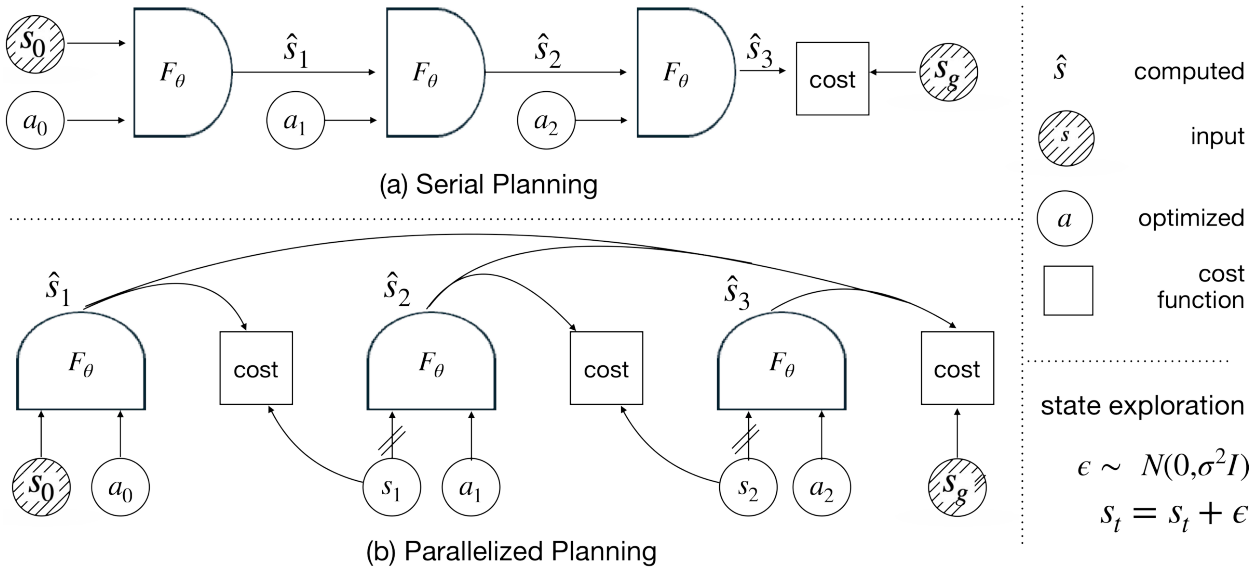


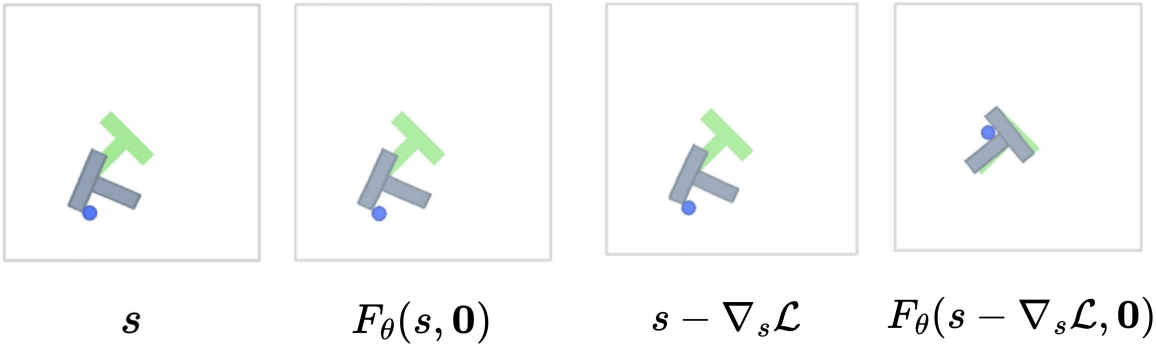
Figure 7.2: Graphical depiction of **(a)** a standard serial-based setup for optimization-based planning, with states rolled out via the actions and the loss evaluated on the goal state; and **(b)** our setup, parallelizing world-model evaluations by optimizing "virtual states" directly and only supervising pairwise dynamics satisfaction. The crossed lines and skipped connections in **(b)** are detailed in Section 7.2: they keep the full planning graph connected while not requiring state gradients of F_θ. In practice we alternate between **(a)** and **(b)**.

with $\xi_t^k \sim \mathcal{N}(0, I)$. Each optimization step performs a gradient update on the intermediate states followed by an isotropic Gaussian perturbation. Intuitively, the noise allows iterates to "hop" between nearby basins.

Noise on states vs. actions. By only noising states, we condition on more dynamically feasible trajectories while still allowing exploration over a wider distribution. Planning problems often have a small number of intermediate states for the solution; noising directly over states finds these faster than noising over actions. Appendix D.3 gives a stationary-distribution characterization.

Sensitivity to state gradients and gradient surgery

In practice, F_θ is learned and can have brittle local geometry. When optimizing (7.4) by gradient descent in both $\mathbf{a}$ and $\mathbf{s}$, we observe empirically that gradients with respect to state inputs $\nabla_s F_\theta(s, a)$ can be exploited. For a local goal-reaching objective $\arg\min_s \|y - F_\theta(s, a)\|_2^2$, rather than learning to find an s on-manifold such that applying a leads to y, the optimizer can find a nearby ambient $s + \delta$ with $\|\delta\|_2 \ll 1$ such that $y \approx F_\theta(s + \delta, a)$, *regardless of the starting state* s.



(a) Expected behavior. (b) Adversarial example.

Figure 7.3: Sensitivity of state gradient structure. From states far from the goal (in-distribution or out-of-distribution), a small step along the gradient $s' = s - \varepsilon \nabla_s \mathcal{L}(s)$, with $\mathcal{L}(s) = \|F_\theta(s, a = \mathbf{0}) - g\|_2^2$, leads to a nearby state s' that solves the planning problem in a single step: $F_\theta(s', \mathbf{0}) = g$. Thus, optimizing states directly through F_θ is challenging.

This is analogous to the adversarial-robustness issue of image classifiers (Szegedy et al., 2013; Shamir et al., 2021): high-dimensional input spaces for trained networks have high Lipschitz constants, hindering optimization performance.

A nonexistence theorem. Unfortunately, any loss function over $\mathbf{s}$ and $\mathbf{a}$ whose minimizers are dynamically feasible must depend on the state gradient $\nabla_s F$ in a meaningful way. The following informal statement (formalized in Appendix D.4) captures the obstruction.

Theorem 7.1 (informal)**.** *A differentiable loss function over state/action trajectories $\mathcal{L} : \mathcal{S}^T \times \mathcal{A}^T \to \mathbb{R}$ given a world model $F_\theta : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ cannot simultaneously satisfy:*

1. *Minimizers of $\mathcal{L}$ correspond to dynamically feasible trajectories: $F_\theta(s_t, a_t) = s_{t+1}$;*
2. *$\mathcal{L}$ is insensitive to the world model state gradient $\nabla_s F_\theta$.*

To address adversarial sensitivity, we detach gradients through the *state inputs* of F_θ while still differentiating with respect to actions. We denote by $\bar{s}_t$ a stop-gradient copy of s_t ($\bar{s}_t = s_t$ in value, treated as constant during differentiation).

Grad-cut dynamics loss. Apply a stop-gradient to state inputs in the dynamics loss:

$$\mathcal{L}^{\text{sg}}_{\text{dyn}}(\mathbf{s}, \mathbf{a}) \;=\; \sum_{t=0}^{T-1} \|F_\theta(\bar{s}_t, a_t) - s_{t+1}\|_2^2 \,. \tag{7.6}$$

Differentiable with respect to $\mathbf{a}$ and the *next* states s_{t+1}, but does not backpropagate through s_t via F_θ.

Dense goal shaping on one-step predictions. While (7.6) improves robustness, it introduces a new degeneracy: paths gravitate toward the current rollout regardless of proximity to the goal. To provide a task-aligned signal at every timestep without state-input gradients, we add a goal loss on one-step predictions:

$$\mathcal{L}^{\text{sg}}_{\text{goal}}(\mathbf{s}, \mathbf{a}) \;=\; \sum_{t=0}^{T-1} \|F_\theta(\bar{s}_t, a_t) - g\|_2^2 \,. \tag{7.7}$$

This encourages each predicted next state to move toward the goal, supplying gradient information to every action a_t while maintaining the grad-cut on s_t (Figure 7.2). Due to the stop-gradient on $\bar{s}_t$, gradients through $F_\theta(\bar{s}_t, a_t)$ flow only with respect to a_t, preventing exploitation of adversarial state-input directions. The full energy is then

$$\mathcal{L}(\mathbf{s}, \mathbf{a}) \;=\; \sum_{t=0}^{T-1} \|F_\theta(\bar{s}_t, a_t) - s_{t+1}\|_2^2 + \gamma \sum_{t=0}^{T-1} \|F_\theta(\bar{s}_t, a_t) - g\|_2^2 \,, \tag{7.8}$$

where $\gamma > 0$ is fixed.

Resulting noisy dynamics. Explicitly writing out $\nabla_a \mathcal{L}$:

$$\begin{aligned} s_t^{k+1} &\leftarrow s_t^k - 2\eta_s \big(s_t^k - F_\theta(s_{t-1}^k, a_{t-1}^k)\big) + \sigma_{\text{state}} \xi_t^k, \\ a_t^{k+1} &\leftarrow a_t^k - \eta_a \nabla_{a_t} \mathcal{L}(\mathbf{s}^k, \mathbf{a}^k). \end{aligned} \tag{7.9}$$

While the action dynamics still follow a gradient flow, the states do not follow a true gradient vector field; *the resulting dynamics are not Langevin*. The dynamics still bias toward valid goal-oriented trajectories, but their efficient convergence requires the extra synchronization step described next.

Full-rollout synchronization

The no-state-gradient updates are designed to be robust to brittle state-input Jacobians. However, the stochastic optimization (7.9) still needs a method of strict descent toward true minima. In practice, we periodically "sync" the plan by briefly running standard full-gradient planning on the original rollout objective.

Every K_{sync} iterations, we perform J_{sync} steps of gradient descent on the original planning loss

$$\min_{\mathbf{a}} \; \|s_T(\mathbf{a}, s_0) - g\|_2^2 \,, \tag{7.10}$$

where $s_T(\mathbf{a}, s_0)$ is computed by sequentially rolling out F_θ. During this synchronization phase we update only the actions:

$$\mathbf{a} \leftarrow \mathbf{a} - \eta_{\text{sync}} \nabla_{\mathbf{a}} \|s_T(\mathbf{a}, s_0) - g\|_2^2 \,, \tag{7.11}$$

using full backpropagation through the T-step rollout. By keeping these GD steps small relative to the stochastic dynamics of (7.9), we benefit from the smoothed loss landscape (Figure 7.1c) for wider exploration and the sharp landscape (Figure 7.1b) for refinement.

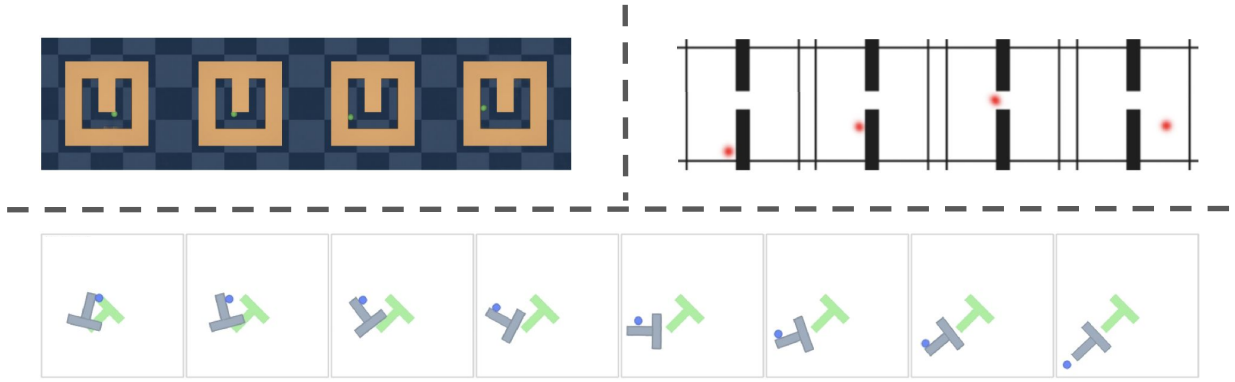


Figure 7.4: Virtual states learned through GRASP planning, at horizon 50 in the PointMaze, Wall-Single, and Push-T environments. Regardless of dynamics-constraint relaxation and state noising, directly optimized states find realistic, non-greedy paths toward the goal.

Algorithm 5 GRASP planner: parallel stochastic gradient-based planning.

Require: Initial observation o_0, goal o_g, world model F_θ, horizon T, steps K, learning rates η_a, η_s.

Ensure: $\mathbf{a}^*$

$s_0 \leftarrow \text{encode}(o_0)$, $s_T \leftarrow \text{encode}(o_g)$

$\mathbf{a} \leftarrow \mathbf{a}^0$; $\mathbf{s} \leftarrow \text{init_states}(s_0, s_T)$

for $k = 0$ to $K - 1$ **do**

  Compute $\mathcal{L}$ via (7.8)

  Joint step: $(\mathbf{a}, \mathbf{s}) \leftarrow (\mathbf{a}, \mathbf{s}) - (\eta_a \nabla_{\mathbf{a}} \mathcal{L}, \eta_s \nabla_{\mathbf{s}} \mathcal{L})$

  Stochastic state: $s_t \leftarrow s_t + \sigma_{\text{state}} \xi_t$ for $t = 1, \ldots, T-1$

  Sync (periodic): if $k \bmod K_{\text{sync}} = 0$, roll out from s_0, $s_{t+1} \leftarrow F_\theta(s_t, a_t)$ for $t = 0, \ldots, T-1$, then take a GD step $\mathbf{a} \leftarrow \mathbf{a} - \eta_{\text{sync}} \nabla_{\mathbf{a}} \|s_T - g\|_2^2$

end for

return $\mathbf{a}^* \leftarrow \mathbf{a}$

Algorithm

Algorithm 5 summarizes the full GRASP procedure.

7.3 Adversarial-robustness reading and the visual world-model wall

The penalty relaxation of Section 7.2 works well on simple problems: low-dimensional state spaces, smooth simulation environments, short-to-medium horizons. The parallel planner finds better trajectories than serial gradient descent and degrades slowly as the horizon grows. It also matches or exceeds classical baselines.

The trouble starts with visual state spaces. On these, the penalty-relaxed planner produces what we call *teleportation failures*: optimized trajectories where state s_t apparently jumps discontinuously between distant configurations, with the dynamics-satisfaction term ostensibly satisfied to high precision (below 10^{-3}, as in the example that follows).

On a Pointmaze visual-control task initialized with linearly interpolated states between s_0 and the goal, run to convergence, we get $\|F_\theta(s_t, a_t) - s_{t+1}\|_2 < 10^{-3}$ at every timestep. By the formal criterion of (7.4), the optimization succeeded. But if we apply F_θ to s_0 with the recovered a_1 and compare to the optimizer's s_1, they differ noticeably; after a few rollout steps the discrepancy is large. The states the optimizer found are dynamically consistent in a pointwise sense (each s_{t+1} lies in the image of F_θ applied to (s_t, a_t)), but the sequence $\{s_t\}$ is not a rollout of F_θ from s_0. The optimizer has discovered configurations s_t in image space that the world model accepts as one-step-consistent but that are not on the orbit of F_θ from s_0.

This is, at structural level, an adversarial-robustness phenomenon (Szegedy et al., 2013; I. J. Goodfellow et al., 2015). The world model $F_\theta(s, a)$ viewed as a function of s with a fixed has the same locally rough behavior described in Chapter 4: for almost any input state and action, there exist small state perturbations producing arbitrarily different outputs. The optimizer in (7.4), given freedom to choose s_t as an independent variable, does exactly the kind of search adversarial-attack algorithms do: finding states that the model maps wherever optimization pressure points, without those states being physically meaningful.

The classical robustness-accuracy tradeoff makes this hard to fix at the world-model level. Training F_θ to be robust to state perturbations costs accuracy on in-distribution rollouts. The chapter's response is to fix the planning loss instead of the world model, which is what GRASP's gradient-cut does.

7.4 Theory

We provide a theoretical analysis of convergence properties of our planning approach compared to traditional shooting methods. We consider a simplified linear-dynamics setting to derive formal guarantees. For the proofs and additional results, see Appendix D.

Convergence in the convex setting

Consider linear dynamics $s_{t+1} = As_t + Ba_t$ with $s_t \in \mathbb{R}^n$, $a_t \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$. We analyze two formulations.

Shooting method. Minimize the terminal-state error with respect to actions:

$$\min_{\mathbf{a}\in\mathbb{R}^{mT}} J_S(\mathbf{a}) = \|\mathcal{F}(s_{\text{init}}, \mathbf{a}) - g\|_2^2 . \tag{7.12}$$

Lifted-states method (multiple shooting). Treat both states and actions as optimization

variables, minimizing dynamics-residual norms:

$$\min_{\mathbf{a},\mathbf{s}} J_L(\mathbf{a},\mathbf{s}) = \sum_{t=0}^{T-1} \|As_t + Ba_t - s_{t+1}\|_2^2, \quad \text{s.t. } s_0 = s_{\text{init}}, s_T = g. \tag{7.13}$$

The Hessian for the shooting method is $H_S = 2\mathcal{C}_T^\top \mathcal{C}_T$ where $\mathcal{C}_T = [A^{T-1}B, A^{T-2}B, \ldots, B]$ is the controllability matrix. The Hessian for the lifted method is $H_L = 2M^\top M$ where M is a sparse, block-banded structure encoding the dynamics residuals at each t.

Theorem 7.2 (Shooting: exploding smoothness)**.** *Suppose $A^\top$ has a real eigenvalue λ with $|\lambda| > 1$ and unit eigenvector v. Assume B aligns with this mode: for some input direction w with $\|w\|_2 = 1$, $|\langle v, Bw\rangle| = \mu > 0$. Then the Lipschitz constant of the shooting gradient grows exponentially with T:*

$$L_S = \lambda_{\max}(H_S) \geq 2\mu^2 |\lambda|^{2(T-1)}. \tag{7.14}$$

Theorem 7.3 (Lifted: stable smoothness)**.** *The Lipschitz constant of the lifted gradient is bounded by a constant independent of T. One valid bound:*

$$L_L = \lambda_{\max}(H_L) \leq 6\big(1 + \|A\|_2^2 + \|B\|_2^2\big). \tag{7.15}$$

The interpretation: many realistic systems are not universally stable within the controllability subspace, so the shooting method forces optimization to traverse a loss landscape with exponentially varying curvature (requiring exponentially small steps to remain stable). The lifted method "preconditions" the problem by creating independent variables for intermediate states, decoupling long-term dependencies into local constraints. This yields a loss landscape with uniform smoothness $O(1)$ with respect to the horizon T.

Proofs are in Appendix D.1.

Gaussian smoothing regularity

The regularity from noisy gradient descent stems primarily from Gaussian-convolution smoothing.

Theorem 7.4 (Gaussian smoothing contracts gradients)**.** *For ϕ_σ the density of $\mathcal{N}(0, \sigma^2 I_d)$ and $L_\sigma(\mathbf{s}) = (\phi_\sigma * L)(\mathbf{s})$:*

1. Regularity never decreases. *If $\nabla_s L \in L^p(\mathbb{R}^d)$ for $1 \leq p \leq \infty$, then $\|\nabla_s L_\sigma\|_{L^p} \leq \|\nabla_s L\|_{L^p}$. In particular, if L is Lipschitz,* $\text{Lip}(L_\sigma) \leq \text{Lip}(L)$.
2. Explicit regularity control by variance. *For $1 \leq p \leq \infty$, if $L \in L^p(\mathbb{R}^d)$, then $\|\nabla_a L_\sigma\|_{L^p} \leq (c_d/\sigma)\|L\|_{L^p}$ where $c_d = \sqrt{2}\,\Gamma((d+1)/2)/\Gamma(d/2)$.*

This is the rigorous version of the intuition that decaying noise schedules anneal from a smoother, less accurate gradient structure to a rigid but more accurate one. See Chaudhari et al. (2019) for further regularity theory of noisy gradient descent. Proof and corollaries are in Appendix D.2.

Stationary distribution under state-only noise

If we also added noise to the actions, the continuous-time Langevin dynamics corresponding to (7.5) would admit a Gibbs (Boltzmann) stationary density $p(\mathbf{s}, \mathbf{a}) \propto \exp(-\beta \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a}))$, with inverse temperature $\beta > 0$ determined by drift-to-diffusion scaling.

However, we noise only the states. The resulting stationary distribution collapses on solutions in action to the local dynamics problems:

$$p(\mathbf{s}, \mathbf{a}) \propto \delta(\mathbf{a} = \mathbf{a}^*(\mathbf{s})) \cdot \exp(-\beta \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a})), \tag{7.16}$$

where $\mathbf{a}^*(\mathbf{s}) = \arg\min_{\mathbf{a}} \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a})$. Detailed analysis is in Appendix D.3.

Behavior under the gradient-cut: tube around predictions

In the stop-gradient setting, with $\mu_t = F_\theta(\bar{s}_t, a_t)$ fixed, the noisy state update is contractive around μ_t.

Theorem 7.5 (Gaussian tube around one-step predictions)**.** *Fix $\{\bar{s}_t^k\}, \{a_t^k\}$ at iteration k, and let $\mu_t^k = F_\theta(\bar{s}_t^k, a_t^k)$. Then the state update with the stop-gradient lifted objective satisfies*

$$\mathbb{E}[s_{t+1}^{k+1} \mid \mu_t^k] = (1 - 2\eta_s)\, \mathbb{E}[s_{t+1}^k \mid \mu_t^k] + 2\eta_s\, \mu_t^k. \tag{7.17}$$

If $\mu_t^k \equiv \mu_t$ is held fixed, s_{t+1}^k converges in distribution to a Gaussian "tube" around μ_t:

$$s_{t+1}^\infty \sim \mathcal{N}\big(\mu_t, \Sigma_{tube}\big), \quad \Sigma_{tube} = \frac{\sigma^2}{1 - (1 - 2\eta_s)^2} I = \frac{\sigma^2}{4\eta_s(1 - \eta_s)} I. \tag{7.18}$$

This tube behavior is essential: it characterizes how the noisy lifted-state dynamics differs structurally from noisy rollout dynamics. The dense one-step goal shaping drifts the tube center toward the goal.

Theorem 7.6 (Goal shaping drifts the tube center)**.** *Let $J_t^k = \nabla_{a_t} F_\theta(\bar{s}_t^k, a_t^k)$ and $P_t^k = J_t^k (J_t^k)^\top \succeq 0$. Assume a first-order linearization $\mu_t^{k+1} \approx \mu_t^k + J_t^k (a_t^{k+1} - a_t^k)$. The action update yields the tube-center evolution*

$$\mu_t^{k+1} \approx \underbrace{(I - \alpha\gamma P_t^k)\, \mu_t^k + \alpha\gamma P_t^k\, g}_{\textit{goal-directed averaging}} + \underbrace{\alpha P_t^k\, \varepsilon_{t+1}^k}_{\textit{exploration forcing}}, \quad \alpha = 2\eta_a, \tag{7.19}$$

where ε_{t+1}^k is the tube residual $s_{t+1}^k = \mu_t^k + \varepsilon_{t+1}^k$.

In contrast, noisy rollouts (with noise applied during rollout rather than during optimization) do not form a tube around the deterministic rollout: the mean evolves by $\mathbb{E}[F_\theta(s_t, a_t)]$, not the deterministic $F_\theta(\mathbb{E}s_t, a_t)$, and the distribution can drift and spread (Appendix D.4).

7.5 Experiments

We evaluate GRASP on two complementary classes of environments testing (i) nonconvex long-horizon planning with obstacles and (ii) data-driven visual control under learned dynamics. The experiments answer three questions:

1. Can the proposed planner overcome the greedy local minima that often trap shooting methods?
2. Does the method remain robust as the planning horizon increases?
3. Does the proposed planner converge faster than rollout-based planners?

Baselines and environments

Baselines. **CEM** optimizes action sequences by iteratively sampling candidates, selecting elites, and refitting a sampling distribution. **GD** directly optimizes the action sequence by backpropagating through the dynamics model. **LatCo** (Rybkin et al., 2021) optimizes in a lifted latent/state space by jointly adjusting intermediate latent variables and actions; we adapt it to our pure-planning setting from its original model-based RL setting.

For all methods we sweep over hyperparameters and report results using the best-performing setting for each environment and horizon. For GRASP, we initialize states $\{s_t\}_{t=0}^{T}$ as noised around the linear interpolation between s_0 and g: $s_t = (t/T)g + (1 - t/T)s_0 + z$, $z \sim \mathcal{N}(0, \varepsilon I)$, and actions at $a_t = 0$.

Environments. Initial evaluation on three visual control environments: PointMaze, Wall-Single, and Push-T. World models trained using the DINO-wm framework (G. Zhou et al., 2024), with each $F_\theta(s, a)$ taking 5 actions and predicting 5 steps ahead. Additional evaluation on Reacher, Cube, and Meta-World Reach (MW Reach) with JEPA-style (Assran et al., 2025) world models, as reported in Table 7.5 below. Success is reaching the goal region within the planning horizon.

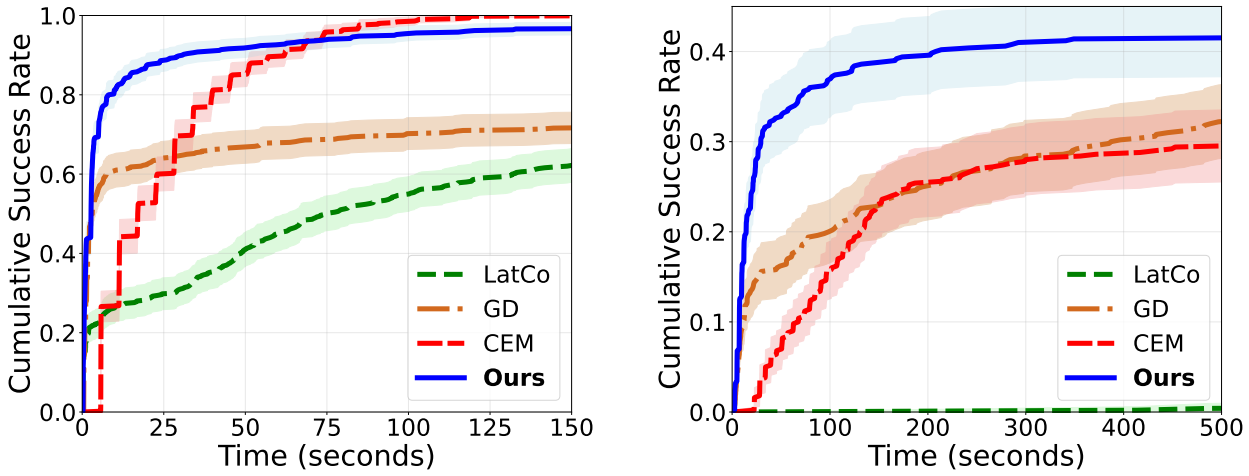
Long-horizon planning and horizon scaling

We evaluate planners in the long-horizon regime where greedy local minima and optimization instability dominate.

GRASP remains reliable as horizon increases: it solves more tasks and finishes its successful trials faster (Table 7.1). At long horizons for Push-T, where it is more important to explore non-greedy optima, GRASP finds more success than baselines (visualized in Figure D.1). Figure 7.5 further illustrates: success rates over time at fixed horizon ($H = 50$) show GRASP converging at an earlier time on most plans.

Table 7.1: Open-loop planning results on long-range Push-T. Success rate (%) and median success time (seconds; successful trials only) across planning horizons. 500 trials per setting. Each cell reports *Success / Time*.

	$H = 40$		$H = 50$		$H = 60$		$H = 70$		$H = 80$	
CEM	**61.4**	/ 35.3s	30.2	/ 96.2s	7.2	/ 83.1s	7.8	/ 156.1s	2.8	/ 132.2s
LatCo	15.0	/ 598.0s	4.2	/ 1114.7s	2.0	/ 231.5s	0.0	/ —	0.0	/ —
GD	51.0	/ 18.0s	37.6	/ 76.3s	16.4	/ 146.5s	12.0	/ 103.1s	6.4	/ 161.3s
GRASP (ours)	59.0	/ **8.5s**	**43.4**	/ **15.2s**	**26.2**	/ **49.1s**	**16.0**	/ **79.9s**	**10.4**	/ **58.9s**



(a) Point Maze ($H = 50$). (b) Push-T ($H = 50$).

Figure 7.5: Success rate over a fixed set of open-loop planning tasks for CEM, GD, LatCo (Rybkin et al., 2021), and GRASP at horizon 50. Curves summarize how quickly each planner makes progress under the learned-world-model setting. Shaded regions: Wald 95% confidence intervals.

Short-term planning

We also evaluate short-horizon planning to demonstrate that GRASP can match strong performance on shorter, easier tasks. Tables 7.2 and 7.3 report success rates and median planning times across environments.

Across all environments and short horizons, GRASP achieves success rates comparable to the baselines while maintaining consistently low planning times, often faster than sampling-based approaches and competitive with gradient-based optimization. GRASP sacrifices a small amount of speed in exchange for a higher success rate.

Table 7.2: Short-term planning. Success rate (%) for Push-T, PointMaze, and WallSingle. 500 trials per setting. GRASP has a comparable success rate with consistently low completion time (Table 7.3).

	Push-T			PointMaze			WallSingle		
Planner	$H = 10$	$H = 20$	$H = 30$	$H = 10$	$H = 20$	$H = 30$	$H = 10$	$H = 20$	$H = 30$
CEM	**100.0**	**96.0**	**76.0**	**100.0**	**100.0**	**100.0**	**98.0**	**100.0**	**100.0**
LatCo	67.4	55.2	35.4	**100.0**	99.8	99.0	91.4	81.6	62.8
GD	99.2	88.4	69.0	94.6	91.8	94.6	91.8	92.8	91.6
GRASP (ours)	**100.0**	89.2	75.2	**100.0**	**100.0**	99.8	95.2	91.8	95.0

Table 7.3: Median completion times (seconds) for short-term experiments. 500 trials per setting. GRASP has consistently low completion times alongside comparable success rates (Table 7.2).

	Push-T			PointMaze			WallSingle		
Planner	$H = 10$	$H = 20$	$H = 30$	$H = 10$	$H = 20$	$H = 30$	$H = 10$	$H = 20$	$H = 30$
CEM	1.3	7.5	23.6	0.7	2.0	6.3	0.9	1.5	2.1
LatCo	39.9	437.2	1800.0	1.6	13.7	36.6	12.1	23.8	111.1
GD	**0.5**	2.4	25.3	**0.3**	**0.6**	**1.2**	**0.4**	**0.5**	**0.7**
GRASP (ours)	1.5	**2.1**	**9.1**	0.7	1.8	2.1	1.6	1.8	2.1

Ablation studies

To isolate the contribution of each component, we conducted ablations on Push-T at $H = 40$. We systematically removed components while keeping other hyperparameters constant (Table 7.4).

The full method achieves the highest success rate, so the components are complementary. The stop-gradient is critical: without it the state optimization is "sticky" and difficult to tune. Noise is similarly important for escaping local minima. Further ablations of alternative stochastic-GD baselines are in Appendix D.5.

Latent-encoder and JEPA-style world models

We further evaluate GRASP on environments with latent-encoder (`le-wm`) and JEPA-style (Assran et al., 2025) (`jepa-wm`) world models, including Reacher, Push-T (le-wm), Cube manipulation, and Meta-World Reach. Two variants of GRASP are evaluated:

- **GRASP-CEM** (single dagger †): uses CEM for the sync step in place of GD.
- **GRASP-dagger** (double dagger ‡): linearly anneals the goal-loss coefficient γ and noise variance σ_{state} between sync steps.

Table 7.5 shows GRASP's advantage on latent-encoder and JEPA-style world models: it outperforms CEM and serial GD across environments, with the larger gains on longer-horizon

Table 7.4: Ablations over GD sync steps, noise level for Langevin dynamics, and the detached-gradient approach. GD sync happens every 100 stochastic steps. Push-T, $H = 40$. Time reported is median over successful trials.

Setting	**Accuracy (%)**	**Time (s)**
Sync Steps		
No Sync	1.6	4.5
10	48.0	11.2
25 (ours)	**59.0**	8.5
50	58.0	9.8
Noise Level		
$\sigma = 0.0$	54.8	10.4
$\sigma = 0.5$ (ours)	**59.0**	8.5
$\sigma = 1.0$	50.6	8.4
Detach $\nabla_s F(s, a)$		
Stop (ours)	**59.0**	8.5
Flow	46.6	10.3

Table 7.5: GRASP performance on latent-encoder and JEPA-style world models. Success rates (%). Horizon shows the executed action horizon (and the corresponding multi-step rollout length in parentheses, e.g. "15 (75)" means 15 executed actions with 75-step total rollout). Asterisk (*) indicates statistical significance over baselines; † uses CEM sync step (GRASP-CEM); ‡ anneals the goal-loss coefficient and noise variance between sync steps (GRASP-dagger).

Environment	**Horizon**	**CEM**	**GD**	**GRASP**
Reacher (le-wm)	15 (75)	38%	42%	70%*
Push-T (le-wm)	15 (75)	22%	26%	46%*†
Cube (le-wm)	10 (50)	58%	56%	66%*‡
Cube (le-wm)	15 (75)	64%	50%	68%*‡
MW Reach (jepa-wm)	6 (30)	12%	6%	19%‡

and harder tasks. The two GRASP variants (CEM sync step, ‡ annealing) each provide additional headroom in specific settings.

Across these results, GRASP matches strong baselines in short-term planning and outperforms them on long horizons, avoiding greedy failures and converging faster in wall-clock time.

7.6 Related work

World modeling. Learned world models (Hafner, Pasukonis, et al., 2025; Ding et al., 2024) have improved sample efficiency for model-based RL by predicting future states from current states and actions, enabling planning without an interactive environment. Recent work emphasizes latent-space representations for high-dimensional observations (Assran et al., 2025) and large-scale generalization across environments (Bar et al., 2025). The development here is downstream of these architectures: we take a trained F_θ as given and ask how to plan with it efficiently.

Sampling-based planning. CEM (Rubinstein and Kroese, 2004) and random shooting are robust and simple but suffer from serial-evaluation bottlenecks and poor scaling with planning-horizon length (Bharadhwaj et al., 2020). Recent work improves CEM with faster variants, action correlation and memory, parallelized sampling, and massively parallel strategies (Pinneri et al., 2021; T. Lai et al., 2022), but fundamental limitations remain for very long horizons.

Gradient-based planning. Gradient-based planners use the differentiability of neural world models to directly optimize action sequences (Thrun et al., 1990; Jyothir et al., 2023). Early approaches applied BPTT (Werbos, 2002) but face vanishing/exploding gradients and poor conditioning at long horizons. Hybrid strategies combining GD with sampling have shown promise; CEM-GD variants interleave backwards passes with population-based search for improved convergence (Bharadhwaj et al., 2020; K. Huang et al., 2021). Recent work improves gradient-based planners by training world models to be adversarially robust (Parthasarathy et al., 2025); we instead improve planning for general world models without pretraining modifications.

State optimization and multiple shooting in optimal control. Treating states as optimization variables separate from dynamics constraints has a rich history in classical optimal control. Non-condensed QP formulations (Jerez et al., 2011) decouple state and input optimization. Multiple-shooting methods (Tamimi and P. Li, 2009; Diedam and Sager, 2018) break long horizons into shorter segments with continuity constraints. Direct collocation (Bordalba et al., 2022; Nie and Kerrigan, 2025) optimizes states and controls simultaneously while enforcing dynamics through collocation constraints. These have primarily been applied to systems with known analytical dynamics. Trajectory optimization methods in robotics have developed parallel shooting and GPU-accelerated planning (Guhathakurta et al., 2022), but most approaches still face limitations when applied to learned world models, particularly visual ones. A parallel work explores a collocation-based approach for grounding action-conditioned video models on real trajectories (Ziakas et al., 2026), focused more on trajectory grounding than planning.

Noise and regularization in optimization. Stochastic optimization and noise injection have long been recognized for improving optimization outcomes (Robbins and Monro, 1951), regularizing and exploring complex loss landscapes (Welling and Teh, 2011; P. Xu et al., 2018; Bras, 2023; Foret et al., 2020). In planning, noise is commonly used in sampling-based methods, but its systematic incorporation into gradient-based planning for learned models remains underexplored.

7.7 Limitations and future work

GRASP shows clear advantages in long-horizon settings, but its benefits are more limited at short horizons where simpler methods already perform well. Several components are designed to mitigate the unreliability of state gradients in learned world models. While effective, these modifications introduce additional structure and hyperparameters that would ideally be unnecessary. If state representations induced by the world model were smoother or more geometrically well-behaved in the state space, many stabilization mechanisms could be removed, potentially leading to further speed improvements. Promising directions include improved representation learning through adversarial training, diffusion-based world models, or techniques that explicitly regularize the geometry of the learned state space.

Hybrid planners combining iterations of a rollout-based planner (like CEM) and a gradient-based planner (like GD) have been implemented for "best of both worlds" (K. Huang et al., 2021). There are many ways to hybridize GRASP, including the GRASP-CEM and GRASP-dagger variants of Section 7.5; further exploration of such methods is future work.

Part IV

Closing

Chapter 8

Conclusion

We now take a final look at the material covered in this thesis, and present the final takeaways as well as the promising next steps from this work.

8.1 The two sub-programs, side by side

The dissertation has presented four variational constructions, distributed across two sub-programs of the larger variational program.

Sub-program A. Chapter 3 treats the path problem in deep learning as a representation-learning problem in disguise. The functional $\mathcal{E}[\mathcal{M}]$, measuring the non-convexity of $\mathcal{M}$ via the gap $\mathrm{conv}(\mathcal{M})\backslash\mathcal{M}$, has the embedded data manifold $\mathcal{M}$ as its argument. A convexification flow is designed to descend it—it promotes convexity, though it is not proven to be the gradient flow of $\mathcal{E}$—and as $\mathcal{M}$ approaches convexity it becomes flat, at which point downstream operations, in particular interpolation between two endpoints, become essentially trivial. The space is the variable; the path is what the geometry of the reshaped space produces for free.

Sub-program B. Chapters 5, 6, and 7 treat the path problem by accepting the underlying space as given and bringing the variational machinery to bear on paths through it. Three different learned objects induce three different functionals: a learned critic gives the Q-score-matching loss (Chapter 5); a learned score gives the Onsager–Machlup action (Chapter 6); a learned dynamics gives the Lagrangian-relaxed planning energy with its gradient-cut variant (Chapter 7). In each case the path is the variable, the space is fixed, and the technical work is to find the right reshaping of the functional so that the optimization remains well-conditioned over a rough learned model.

The unification, made concrete. The dissertation's central claim is that these four constructions belong to one program. The claim is supported by structural commonalities visible only in retrospect. All four constructions write down a functional whose stationary

points correspond to a desired outcome. They exploit structural freedom in choosing the functional (freedom that is hard to articulate without the variational framing) to handle the technical obstacles unique to each domain.

The two sub-programs are not in competition. They differ in what they optimize *over* and in what they take as fixed. Sub-program A applies when the underlying data has recoverable manifold structure; the flattened representation then supports clean downstream analysis with no further variational machinery. For a fixed, rough representation produced by an upstream training procedure, sub-program B acts on paths through that geometry without requiring it to be smooth.

8.2 Smoothing by splitting

A recurring technical theme of sub-program B is *smoothing-by-splitting*. In each of the three chapters of sub-program B, a quantity that would otherwise need to be evaluated by back-propagating through a long, rough computation is replaced by a variational surrogate that depends only on local, well-conditioned ingredients.

In Chapter 5, the long computation is the reverse-diffusion chain, and the smoothing is a local score-matching loss that depends only on the denoising network's output at one action and the critic's action gradient at the same action. In Chapter 6, the long computation is, implicitly, the integration of the SDE to compute a path likelihood; the smoothing is the OM action, which depends only on the path, its derivative, and the drift evaluated locally. The diffusion-time parameter τ also gives an additional, continuously-tunable smoothing knob. In Chapter 7, the long computation is the rolled-out trajectory of the learned dynamics, and the smoothing is the Lagrangian relaxation that turns the rollout into a sum over local dynamics-satisfaction terms, with the gradient-cut variant adding a further structural smoothing of the functional's dependence on intermediate states.

Sub-program A handles roughness differently. It removes roughness by reshaping the space; once the space is flat, no further smoothing is needed for downstream operations. The two sub-programs are complementary responses to the same underlying obstacle, joined by their use of variational principles.

8.3 The physics inspiration, in retrospect

The dissertation borrowed a framing from physics: the calculus of variations as a unifying lens. It is worth asking, after the technical content, how far the analogy actually carries.

The framing helped in two ways. First, it gave a structural vocabulary—functional, action, path, stationary point, fixed endpoints—that fit each of the four technical chapters and made the dissertation's unification claim concrete. Second, it provided a methodological reflex: when one is faced with a path-finding problem, one should look for the variational object

first and worry about the algorithm second. This reflex helped find the QSM loss, the OM action, and the GRASP reshaping.

The framing has limits. Learned models lack the analytic structure that makes physics' Lagrangians yield closed-form solutions; we cannot solve Euler–Lagrange equations to find optimal paths, and the variational principle does not, on its own, give us the algorithmic tools to descend the functionals it constructs. In every chapter, the bulk of the technical work was in figuring out how to handle the roughness of the learned ingredient. The variational framing gave us the right structural target; it did not give us the algorithm for free.

The right reading is that the variational principle is a useful organizing lens for a certain class of problems in deep learning, and that recognizing problems as belonging to that class is a productive methodological move. It is not a unifying theory in the sense that physics' variational unification is.

8.4 Open problems

The framing developed in this dissertation suggests several directions for further work.

Functionals induced by other learned objects. The four technical chapters cover learned representations, critics, scores, and dynamics. Other learned objects could plausibly induce useful variational functionals. A language model's log-probability over token sequences is, mathematically, a sum of local terms that could be read as an action; what variational problems on text trajectories does this induce, and are they useful? A learned reward model in RL-from-human-feedback systems supplies a scalar function over response trajectories. Whether it induces useful variational problems on response generation is unclear. A learned constraint model could be read as a barrier function in a constrained variational problem.

Adaptive smoothing within sub-program B. The diffusion-time parameter in Chapter 6 and the multiplier schedules in Chapter 7 are tuned by hand. A more principled approach would adapt them online based on the local conditioning of the optimization problem. The variational framing suggests an internal diagnostic: when the functional's Hessian is poorly conditioned, increase the smoothing; when the functional looks well-behaved, decrease the smoothing to recover fidelity. Implementing this cleanly across the three constructions of sub-program B is a technically appealing direction.

Variational planning at LLM scale. The path-problem framing applies, in principle, to large language-model agents performing multi-step tasks. The action sequence is a sequence of tool calls or generated tokens; the world model is the language model's distribution over outcomes; the reward is provided by an external evaluator. None of the variational constructions of this dissertation translates without modification, but the structural form of the problem is the same, and the framework's vocabulary may transfer. Whether the

smoothing-by-splitting principle, in particular, finds useful instantiations in this regime is an open question.

Closing the loop. As with any learned dynamical system, the goal is to learn and adapt with feedback in a real-world system. How exactly to adapt to such differences, whether through our variational optimizations or through direct model updates, remains at the time of this thesis an open and interesting problem.

8.5 Closing

The path problems modern deep learning has assembled in different domains, from robotics to molecular dynamics, are largely treated as disjoint engineering challenges. This dissertation has argued that they share a structure, one that becomes visible only when one looks for the variational object underlying each, and that recognizing the structure leads both to better algorithms (the four constructions of Chapters 3 through 7) and to a coherent organizing perspective (the two sub-programs of the larger variational program). It has been argued by exhibiting four instances and showing that they sit naturally inside a common framework. Whether the framework continues to organize the next set of path problems in deep learning is an open question. I hope this dissertation has shown that the framework is worth looking at.

Appendix A

Appendix to Chapter 3 (Manifold Flattening)

This appendix collects technical material in support of Chapter 3: differential-geometric background, proofs of the main theoretical results, a time-complexity bound for Algorithm 1, and additional experimental material.

A.1 Differential-geometric background

For the sake of self-containment we give a brief introduction to the differential-geometric notions used in Chapter 3. The treatment is not meant to be mathematically complete; for full treatments see Boumal, 2020; J.M. Lee, 2012; J. M. Lee, 2018.

Embedded submanifolds

We focus on manifolds relevant to the manifold hypothesis: lower-dimensional submanifolds explicitly embedded in higher-dimensional Euclidean space.

Definition A.1. An *embedded submanifold* $\mathcal{M}$ of $\mathbb{R}^D$ is a subset such that for every $x \in \mathcal{M}$ there is some nonempty neighborhood $N_\varepsilon(x) = \mathcal{M} \cap B_\varepsilon(x)$ diffeomorphic to an open subset of $\mathbb{R}^d$. The integer d is the dimension of the manifold, denoted $\dim(\mathcal{M})$.

Intuitively, an embedded submanifold is a continuous structure that is, at every point, locally invertible to a d-dimensional vector representation. We require manifolds to be *smooth* in order to discuss geodesics and curvature; see J.M. Lee, 2012 for a comprehensive definition.

Tangent space, extrinsic curvature

Smooth manifolds are locally approximated by linear subspaces via their *tangent spaces*.

Definition A.2. Let $\mathcal{M} \subset \mathbb{R}^D$ be a smooth manifold of dimension d and let $x \in \mathcal{M}$. The *tangent space* at x, denoted $T_x\mathcal{M}$, is the set of velocity vectors of smooth curves γ on $\mathcal{M}$ through x:

$$T_x\mathcal{M} \coloneqq \left\{ \frac{\mathrm{d}}{\mathrm{d}t}\gamma(0) \;\middle|\; \gamma : [0,1] \to \mathbb{R}^D,\, \gamma(t) \in \mathcal{M},\, \gamma(0) = x \right\}. \tag{A.1}$$

$T_x\mathcal{M}$ is a linear subspace of $\mathbb{R}^D$ of dimension d. Its orthogonal complement $N_x\mathcal{M} \coloneqq \{v : \langle v, w\rangle = 0 \text{ for all } w \in T_x\mathcal{M}\}$ is the *normal space*. The tangent space encodes the linear approximation to local motion on $\mathcal{M}$.

Extrinsic curvature plays a role analogous to second derivatives. For smooth functions, a constant derivative means f is linear; equivalently, $\frac{\mathrm{d}^2}{\mathrm{d}x^2} f = 0$. The geometric analog uses the *second fundamental form*.

Definition A.3. Let $\mathcal{M} \subset \mathbb{R}^D$ be a smooth manifold of dimension d and let $x \in \mathcal{M}$. The *second fundamental form* at x, denoted $\mathrm{I\!I}_x$, is the bilinear map $\mathrm{I\!I}_x : T_x\mathcal{M} \times T_x\mathcal{M} \to N_x\mathcal{M}$ defined by

$$\mathrm{I\!I}_x(v, w) \coloneqq \mathrm{D}_v\, P_{T_x\mathcal{M}}\{w\}, \tag{A.2}$$

where $P_{T_x\mathcal{M}}$ is orthogonal projection onto $T_x\mathcal{M}$ and $\mathrm{D}_v P_{T_x\mathcal{M}}\{w\}$ is the differential of $P_{T_x\mathcal{M}}\{w\}$ at the base point x along v.

For a more complete and rigorous treatment see Boumal, 2020, eq. (5.37). Intuitively, $\mathrm{I\!I}_x$ measures how much the tangent space changes locally. If $\mathrm{I\!I}_x = 0$ everywhere on $\mathcal{M}$, the tangent space does not change and $\mathcal{M}$ is a globally linear subspace of dimension d. This motivates Definition 3.4 and the following immediate consequence.

Corollary A.4. *If a connected manifold $\mathcal{M} \subset \mathbb{R}^D$ of dimension d is flat, then $\mathcal{M}$ is globally contained in an affine subspace of dimension d: $\mathcal{M} \subset T_x\mathcal{M} + x$ for any $x \in \mathcal{M}$.*

Geodesics

A primary challenge of nonlinear datasets is the inability to interpolate via linear chords; for $x, y \in \mathcal{M}$, the chord $\gamma(t) = (1-t)x + ty$ rarely stays in $\mathcal{M}$. While there are typically many interpolations from x to y that stay in $\mathcal{M}$, we call one of minimal arc length (when a minimizer exists, though it need not be unique) a *geodesic*.

Definition A.5. Let $\mathcal{M} \subset \mathbb{R}^D$ be a manifold of dimension d. For any $x, y \in \mathcal{M}$ such that there exists a smooth curve $\gamma : [0,1] \to \mathcal{M}$ with $\gamma(0) = x$ and $\gamma(1) = y$, the *geodesic* between x and y is the curve of minimal arc length:

$$\gamma^\star \coloneqq \operatorname*{arg\,min}_{\substack{\gamma:[0,1]\to\mathcal{M} \\ \gamma(0)=x,\,\gamma(1)=y}} \int_0^1 \left\| \tfrac{\mathrm{d}}{\mathrm{d}t}\gamma(t) \right\|_2 \mathrm{d}t. \tag{A.3}$$

For Chapter 3 we assume that all pairs of points have geodesics between them, which is essentially the assumption that $\mathcal{M}$ is connected.

A.2 Proofs

We give the proofs of the main theoretical results in the chapter.

Proof of Theorem 3.6

For convenience, write $\mathcal{Z} = \mathbf{f}(\mathcal{M})$.

Part 1: $\mathcal{Z}$ is flat. By Definition 3.5, $\mathcal{Z} = \mathbf{f}(\mathcal{M})$ is a convex subset of a d-dimensional affine subspace $A \subset \mathbb{R}^D$. At every $z \in \mathcal{Z}$ the tangent space $T_z\mathcal{Z}$ therefore equals the direction space of A, a fixed d-dimensional linear subspace independent of z. Since the tangent space does not vary along $\mathcal{Z}$, the second fundamental form $\mathrm{I\!I}$ vanishes identically, and $\mathcal{Z}$ is flat.

Remark A.6. The converse fails: a flat submanifold need not be convex. For example, the planar annulus $\{x \in \mathbb{R}^2 : 1 \leq \|x\|_2 \leq 2\}$ has $\mathrm{I\!I} \equiv 0$ yet is not convex, since the straight chord between two points on opposite sides of the hole leaves the set. Flatness constrains $\mathcal{Z}$ to lie in an affine d-plane but places no constraint on convexity within that plane. Convexity is the strictly stronger property that the convexification flow (Section 3.3) is designed to promote.

Part 2: $\mathcal{Z}$ is contained in a unique d-dimensional affine subspace. We prove (a) $\mathcal{Z} \subset z + T_z\mathcal{Z}$ for any $z \in \mathcal{Z}$, and (b) $z_1 + T_{z_1}\mathcal{Z} = z_2 + T_{z_2}\mathcal{Z}$ for all $z_1, z_2 \in \mathcal{Z}$.

For (a), fix $z \in \mathcal{Z}$. By Part 1, $\mathcal{Z}$ is flat, so by the corollary of Section A.1 it is contained in the affine subspace $z + T_z\mathcal{Z}$. Hence $\mathcal{Z} \subset z + T_z\mathcal{Z}$.

For (b), note that $T_{z_1}\mathcal{Z} = T_{z_2}\mathcal{Z}$ for all $z_1, z_2 \in \mathcal{Z}$ because the second fundamental form is identically zero (i.e., the tangent space cannot change). Calling the shared tangent space $T_z\mathcal{Z}$, it suffices to show $z_1 - z_2 \in T_z\mathcal{Z}$ for all $z_1, z_2 \in \mathcal{Z}$. This holds because the geodesic from z_1 to z_2 is $\gamma(t) = z_1 + t(z_2 - z_1)$ and $\gamma'(0) = z_2 - z_1 \in T_z\mathcal{Z}$.

Part 3: Minimal autoencoding. The autoencoding property $\mathbf{g}'(\mathbf{f}'(x)) = x$ (resp. $\|\mathbf{g}'(\mathbf{f}'(x)) - x\|_2 \leq \varepsilon$) holds by construction. It remains to show that no pair $\tilde{f} : \mathbb{R}^D \to \mathbb{R}^p$ and $\tilde{g} : \mathbb{R}^p \to \mathbb{R}^D$ with $\tilde{g}(\tilde{f}(x)) = x$ on $\mathcal{M}$ exists for $p < d$.

Since $\tilde{g} \circ \tilde{f} = \mathrm{id}$ on $\mathcal{M}$, the map $\tilde{f}$ is injective on $\mathcal{M}$. Fix $x \in \mathcal{M}$ and a chart $\xi : \Omega \to \mathcal{M}$ from an open set $\Omega \subseteq \mathbb{R}^d$ onto a neighborhood of x in $\mathcal{M}$. Then $\tilde{f} \circ \xi : \Omega \to \mathbb{R}^p$ is a continuous injection. Suppose $p < d$; composing with the inclusion $\mathbb{R}^p \hookrightarrow \mathbb{R}^d$ gives a continuous injection of the open set $\Omega \subseteq \mathbb{R}^d$ into $\mathbb{R}^d$ whose image lies in the proper subspace $\mathbb{R}^p \times \{\mathbf{0}\}$ and hence has empty interior. This contradicts invariance of domain, by which the image of an open set under a continuous injection $\mathbb{R}^d \to \mathbb{R}^d$ is open. Therefore $p \geq d$. The argument uses only continuity and injectivity of $\tilde{f}$, so it applies verbatim in the ε-relaxed case, where injectivity of $\tilde{f}$ on $\mathcal{M}$ is part of Definition 3.1.

Proof of Lemma 3.9

Write $P_T = P_{T_{x_0}\mathcal{M}}$, $P_N = I - P_T$, and $z = \mathbf{f}(x)$. From (3.14),

$$\mathbf{f}(x) = \varphi(x)\big(P_T(x - x_c) + x_c\big) + (1 - \varphi(x))\, x. \tag{A.4}$$

Tangent coordinate is fixed. Applying P_T and using $P_T\big(P_T(x - x_c) + x_c\big) = P_T x$ (idempotence of P_T),

$$P_T z = \varphi(x)\, P_T x + (1 - \varphi(x))\, P_T x = P_T x. \tag{A.5}$$

Injectivity on $\mathcal{M} \cap B_r(x_0)$. By hypothesis, within $B_r(x_0)$ the manifold is a graph over $T_{x_0}\mathcal{M}$: each $x \in \mathcal{M} \cap B_r$ is determined by its tangent coordinate $t = P_T(x - x_0)$, with $P_N(x - x_0) = \frac{1}{2} P_N \mathrm{I\!I}_{x_0}(t, t) + O(\varepsilon)$. Since $\mathbf{f}$ fixes the tangent coordinate, $P_T(z - x_0) = t$; hence distinct points of $\mathcal{M} \cap B_r$ (distinct t) have distinct images, and $\mathbf{f}|_{\mathcal{M}\cap B_r}$ is injective.

Recovering φ. Given z, the tangent coordinate $t = P_T(z - x_0)$ is known. Applying P_N to $\mathbf{f}$ (and using $P_N P_T = 0$),

$$P_N(z - x_0) = \varphi(x)\, P_N(x_c - x_0) + (1 - \varphi(x))\, P_N(x - x_0). \tag{A.6}$$

To leading order in the quadratic model both normal displacements lie along the mean-curvature normal $n = P_N(x_c - x_0)/\,\|P_N(x_c - x_0)\|$; projecting (A.6) onto n and writing $s = \langle P_N(x - x_0),\, n\rangle$, $c = \|P_N(x_c - x_0)\|$, and $\varphi(x) = e^{-\lambda(\|t\|^2 + s^2)} =: \varphi(s)$ gives the scalar equation

$$\langle P_N(z - x_0),\, n\rangle \;=\; \varphi(s)\, c + (1 - \varphi(s))\, s \;=:\; g(s), \tag{A.7}$$

with derivative $g'(s) = 1 - \varphi(s)\big[1 + 2\lambda s(c - s)\big]$. On the branch connected to the tangent plane (s in the manifold's normal range $[0, \frac{1}{2}\,\|\mathrm{I\!I}_{x_0}\|\, r^2]$), g is strictly increasing whenever

$$\sup_s\; \varphi(s)\big[1 + 2\lambda s(c - s)\big] < 1, \tag{A.8}$$

a condition on the fitted curvature $\|\mathrm{I\!I}_{x_0}\|$ and kernel scale λ (note $c = O(\|\mathrm{I\!I}_{x_0}\|/\lambda)$). Under (A.8), g is invertible on that branch, so s—and hence $\varphi(x) = \varphi^{-1}(z)$—is recovered uniquely by a scalar root-find (e.g. the secant method initialized at $s = 0$); smoothness of φ^{-1} on $\mathbf{f}(\mathcal{M} \cap B_r)$ follows from the inverse function theorem, since $g' > 0$ there. □

Proof of Proposition 3.10

Let $\alpha_{ij} = \Big\langle \tilde{u}_j,\, \tilde{U}\tilde{U}^\top (x_i - x_0) \Big\rangle = \langle \tilde{u}_j,\, x_i - x_0\rangle$ for convenience. Then

$$\mathcal{L}_\varphi(\tilde{U}, \tilde{V}) = \tfrac{1}{N} \sum_i \varphi(x_i)^2 \left\| \tilde{V}(\tilde{U}\tilde{U}^\top (x_i - x_0), \tilde{U}\tilde{U}^\top (x_i - x_0)) - (I - \tilde{U}\tilde{U}^\top)(x_i - x_0) \right\|_2^2 \tag{A.9}$$

$$= \tfrac{1}{N} \sum_i \varphi(x_i)^2 \left\| \sum_{j,k} \alpha_{ij}\alpha_{ik}\tilde{v}_{jk} - (I - \tilde{U}\tilde{U}^\top)(x_i - x_0) \right\|_2^2. \tag{A.10}$$

Splitting the norm via the orthogonal decomposition $I = \tilde{U}\tilde{U}^\top + (I - \tilde{U}\tilde{U}^\top)$,

$$= \tfrac{1}{N} \sum_i \varphi(x_i)^2 \left(\left\| \tilde{U}\tilde{U}^\top \sum_{j,k} \alpha_{ij}\alpha_{ik}\tilde{v}_{jk} \right\|_2^2 + \left\| (I - \tilde{U}\tilde{U}^\top)\Big(x_i - x_0 - \sum_{j,k} \alpha_{ij}\alpha_{ik}\tilde{v}_{jk}\Big) \right\|_2^2 \right) \quad \text{(A.11)}$$

$$\geq \tfrac{1}{N} \sum_i \varphi(x_i)^2 \left\| (I - \tilde{U}\tilde{U}^\top)\Big(x_i - x_0 - \sum_{j,k} \alpha_{ij}\alpha_{ik}\tilde{v}_{jk}\Big) \right\|_2^2, \quad \text{(A.12)}$$

with equality iff $\tilde{U}\tilde{U}^\top(\sum_{j,k} \alpha_{ij}\alpha_{ik}\tilde{v}_{jk}) = 0$ for all i, equivalent (since $\tilde{U}$ is full rank) to $\tilde{U}^\top(\sum_{j,k} \alpha_{ij}\alpha_{ik}\tilde{v}_{jk}) = 0$. This proves Part 1.

For Part 2, rewrite the loss as a least-squares problem. Define $B_{ijk} = \langle \tilde{u}_j,\, x_i - x_0 \rangle \langle \tilde{u}_k,\, x_i - x_0 \rangle$, flatten the last two indices to a matrix $A \in \mathbb{R}^{N \times d^2}$ with $A_{i,j+dk} = B_{ijk}$, stack the vectors $\tilde{v}_{jk}$ into $M_{\tilde{v}} \in \mathbb{R}^{d^2 \times D}$, and stack the residuals $(I - \tilde{U}\tilde{U}^\top)(x_i - x_0)$ into $C \in \mathbb{R}^{N \times D}$. Then

$$\mathcal{L}_\varphi(\tilde{U}, \tilde{V}) = \left\| D_{\varphi(x_i)} A M_{\tilde{v}} - D_{\varphi(x_i)} C \right\|_F^2, \quad \text{(A.13)}$$

where $D_{\varphi(x_i)}$ is the diagonal matrix with $(D_{\varphi(x_i)})_{ii} = \varphi(x_i)$.

A is never full rank because $B_{ijk} = B_{ikj}$ duplicates rows. Replace A by $A' \in \mathbb{R}^{N \times \frac{1}{2}(d^2+d)}$ taking only the upper-triangular component of B_i, with off-diagonal entries scaled by 2, and similarly modify $M_{\tilde{v}}$. Then A' can in principle be full column rank, and the least-squares solution $M'_{\hat{v}}$ (equivalently $\hat{V}$) is unique iff $D_{\varphi(x_i)}A'$ has full column rank. Since $\varphi(x_i) > 0$, this means $\hat{V}$ is unique iff A' is full rank.

For Part 3, using the notation of Part 2, Part 1 implies $A' M'_{\hat{v}} \tilde{U} = 0$. Since A' is full column rank, $M'_{\hat{v}} \tilde{U} = 0$, which gives $\tilde{U}^\top \hat{v}_{jk} = 0$ for all j, k. Any output of $\hat{V}$ is a linear combination of $\hat{v}_{jk}$, so $\tilde{U}^\top \hat{V}(w_1, w_2) = 0$ for all $w_1, w_2 \in \mathbb{R}^D$. □

A.3 Convergence of the convexification flow: status and proof program

A complete convergence theorem for Algorithm 1 would require structure from the sample size and density, noise level, and curvature of $\mathcal{M}$. We do *not* establish one: the original paper Psenka, Pai, et al., 2024 explicitly leaves convergence to future work, and so do we. This section records the ingredients such a theorem would need, as a proof program rather than as results. The statements below are labeled conjectures; none is proven here or in Psenka, Pai, et al., 2024.

Conjecture A.7 (Hausdorff continuity of the unnormalized flow)**.** *Let $\mathcal{M}$ be a smooth, compact, connected, flattenable embedded submanifold of $\mathbb{R}^D$, and let φ be a smooth integrable kernel whose support diameter is smaller than the reach[1] of $\mathcal{M}$. Then the unnormalized*

[1]The *reach* of $\mathcal{M}$ (Federer, 1959) is the largest r such that every point within distance r of $\mathcal{M}$ has a unique nearest point on $\mathcal{M}$.

convexification flow (3.3) *with* $\mathcal{M}(0) = \mathcal{M}$ *has a smooth solution* $\mathcal{M}(t)$ *on some interval* $[0, T^*)$ *that varies continuously in the Hausdorff metric on compact subintervals.*

A natural line of attack would combine short-time existence (implicit function theorem applied to the velocity field), preservation of flattenability along the flow, and control of the long-time behavior via a convexity potential (Section 3.5). Two obstructions keep this a conjecture. First, the convexification flow is *nonlocal*—its velocity at a point depends on a kernel average over $\mathcal{M}$—so the local parabolic-flow machinery does not apply directly. Second, no monotonicity of $\mathcal{E}$ along the flow has been established; this is consistent with the caution of Section 3.3 that the flow is not known to be a gradient flow of $\mathcal{E}$, and it is why "extract a convergent subsequence from coercivity of $\mathcal{E}$" is not available.

Conjecture A.8 (Discrete convergence of FlatNet)**.** *Let* $\mathcal{M}$ *satisfy the assumptions of Conjecture A.7, and let* $\mathcal{X}$ *be an* $(\varepsilon_s, \delta_s)$*-faithful sample of* $\mathcal{M}$ *(Definition 3.2). Then for any reconstruction tolerance* $\varepsilon > 0$ *there exist a neighborhood radius, a number of layers* L*, and kernel parameters such that the FlatNet maps* $\mathbf{f}_{CC}, \mathbf{g}_{CC}$*, composed with the PCA-style projection of Theorem 3.6, form an* ε*-minimal autoencoding pair for* $\mathcal{M}$*.*

The partial progress toward Conjecture A.8 is the dichotomy of Section 3.5: a single layer at x_i provably decreases the local curvature contribution $H(\mathcal{M}_\ell \cap B_\eta(\mathbf{f}_{1:\ell}(x_i)))$, and the quantity $r_{Q,\ell,\varepsilon}$ separates good from bad fixed points. What remains open—and prevents a theorem—is ruling out that a layer's local improvement is offset by increases at other sample points, the same missing ingredient identified in Psenka, Pai, et al., 2024. We make no claim about how the required number of layers L scales with d or $1/\varepsilon$.

A.4 Time complexity

We give a basic asymptotic time-complexity bound for the main bottleneck in computing a FlatNet layer: the autoencoding loss (3.21).

Theorem A.9. *The loss* $\mathcal{L}_\varphi(\tilde{U}, \tilde{V})$ *in* (3.21) *can be computed in* $O(NDd^2)$ *flops.*

Proof. The cost function is

$$\mathcal{L}_\varphi(\tilde{U}, \tilde{V}) = \tfrac{1}{N} \sum_{i=1}^{N} \varphi(x_i)^2 \left\| \sum_{j,k} \tilde{v}_{jk} \langle \tilde{u}_j,\, x_i - x_0 \rangle \langle \tilde{u}_k,\, x_i - x_0 \rangle - (I - \tilde{U}\tilde{U}^\top)(x_i - x_0) \right\|_2^2 . \tag{A.14}$$

The outer sum has at most N terms; incur an $O(N)$ factor and focus on a single summand.

Since $\tilde{U} \in \mathbb{R}^{D \times d}$, $\tilde{U}^\top(x_i - x_0)$ takes $O(Dd)$ flops, and the subsequent multiplication by $\tilde{U}$ takes another $O(Dd)$. So $(I - \tilde{U}\tilde{U}^\top)(x_i - x_0)$ is computable in $O(Dd)$.

The remaining double sum $\sum_{j,k} \tilde{v}_{jk} \langle \tilde{u}_j,\, x_i - x_0 \rangle \langle \tilde{u}_k,\, x_i - x_0 \rangle$ has $O(d^2)$ terms, and the inner products $\langle \tilde{u}_j,\, x_i - x_0 \rangle$ are already available from $\tilde{U}^\top(x_i - x_0)$. Each term, with $\tilde{v}_{jk} \in \mathbb{R}^D$, costs $O(D)$. So the double sum costs $O(Dd^2)$.



(a) FactorVAE. (b) FlatNet (ours).

Figure A.1: Intrinsic distortion of FactorVAE vs. FlatNet (*brighter is better*), on a random manifold of embedding dimension $D = 100$, intrinsic dimension $d = 10$, and $N = 6000$ training samples. Plots are scaled by the maximum EDM ratio.

The final ℓ_2 norm costs $O(D)$. The total is $O(N(Dd^2 + Dd + D)) = O(NDd^2)$. □

A.5 Additional experiments

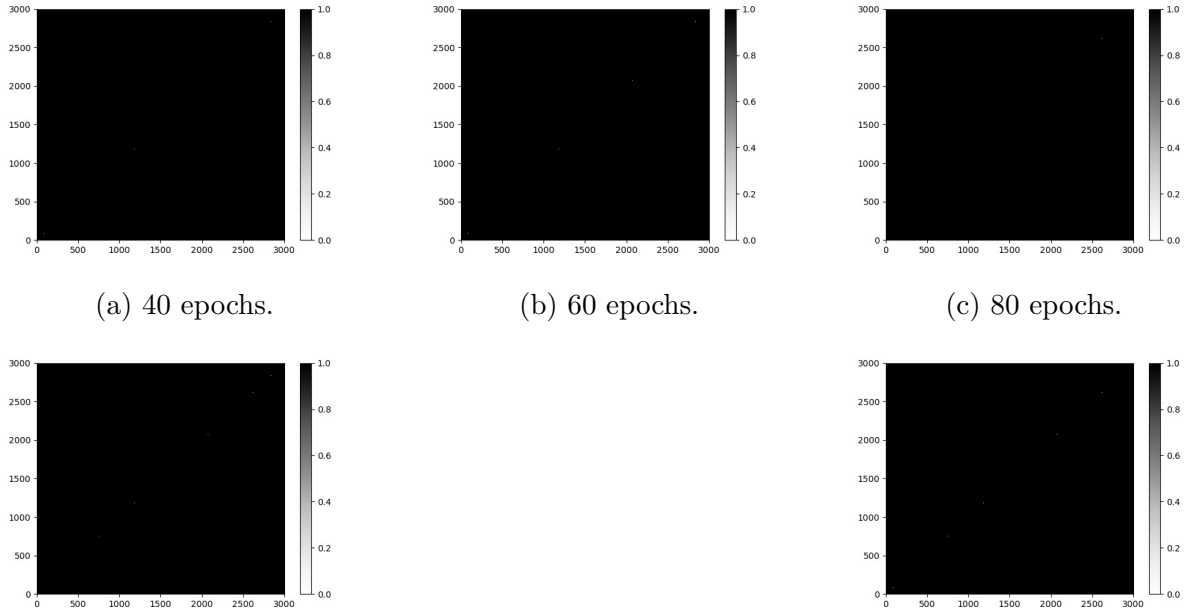
We collect additional experimental figures supporting the main-body results.

Intrinsic distortion vs. FactorVAE

We compare intrinsic distortion of FactorVAE and FlatNet on a random Gaussian-process manifold of $D = 100$, $d = 10$, $N = 6000$. Distortion is measured by the elementwise ratio of Euclidean distance matrices, $\mathrm{EDM}(\mathbf{f}_{\mathrm{CC}}(X))/\mathrm{EDM}(C)$, where $\mathrm{EDM}(\cdot) \in \mathbb{R}^{N\times N}$ is the Euclidean distance matrix of a sample matrix, $X \in \mathbb{R}^{D\times N}$ is the training data, and $C \in \mathbb{R}^{d\times N}$ are the intrinsic coordinates that generated X. Brighter is less distorted. See Figure A.1.

FactorVAE distortion over training epochs

We test FactorVAE's intrinsic distortion across stopping times, on a manifold of $D = 100$, $d = 4$, $N = 3000$. Vanilla VAE and β-VAE showed similar behavior and are omitted (Figure A.2).



(a) 40 epochs. (b) 60 epochs. (c) 80 epochs.

(d) 100 epochs. (e) 120 epochs.

Figure A.2: Intrinsic distortion of FactorVAE on a random manifold of embedding dimension $D = 100$, intrinsic dimension $d = 4$, and $N = 3000$ training samples, across training epochs. Vanilla VAE and β-VAE show similar behavior.

Appendix B

Appendix to Chapter 5 (Q-Score Matching)

This appendix collects technical material for Chapter 5: the proof of the policy-gradient expansion, the proofs of the deterministic and stochastic optimality-condition theorems, a sketch of the Boltzmann fixed-point theorem, the full algorithmic specification, additional experiments, and hyperparameter details.

A note for readability. The proofs in Sections B.2 and B.3 build on each other; please read them in sequence. By far the longest and densest segment is the first part of the proof of Theorem 5.2 (the deterministic case), namely the subsection "Aligning at the origin." This is where the majority of the analysis of the Q-function is built; the remaining results follow from what is established there.

B.1 Proof of the policy-gradient expansion

We justify equation (5.8) with a formal theorem and proof.

Theorem B.1. *Suppose a policy $\pi_\theta(a|s)$ is parameterized through an incremental sequence of steps, with access only to marginal conditional densities $\pi_\theta(a^\tau \mid a^{\tau-1}, s)$ satisfying*

$$\pi(a^K \mid s) \;=\; \int_{a^{K-1}} \cdots \int_{a^1} \pi(a^1) \prod_{\tau=2}^{K} \pi(a^\tau \mid a^{\tau-1}, s)\, \mathrm{d}a^1 \cdots \mathrm{d}a^{K-1}. \tag{B.1}$$

Then the following expectations for the policy gradient are equal:

$$\begin{aligned} &\mathbb{E}_{(s_0,a_0,s_1,\ldots)} \sum_{t=1}^{\infty} Q(s_t, a_t)\, \nabla_\theta \log \pi_\theta(a_t \mid s_t) \\ &\quad = \mathbb{E}_{(s_0,\{a_0^\tau\}_{\tau=1}^K,s_1,\ldots)} \sum_{t=1}^{\infty} Q(s_t, a_t^K) \left(\sum_{\tau=1}^{K} \nabla_\theta \log \pi_\theta(a_t^\tau \mid a_t^{\tau-1}, s_t) \right), \end{aligned} \tag{B.2}$$

where $\pi_\theta(a_t^1 \mid a_t^0, s) \coloneqq \pi_\theta(a_t^1 \mid s)$.

Proof. Substituting the expansion of $\pi(a_t \mid s_t)$ into the LHS,

$$\text{LHS} = \mathbb{E}_{(s_0, a_0^K, s_1, \dots)} \sum_{t=1}^{\infty} Q(s_t, a_t) \nabla_\theta \log\Bigg(\int_{a_t^{K-1}} \cdots \int_{a_t^1} \prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \, \mathrm{d}a^1 \cdots \mathrm{d}a^{K-1} \Bigg).$$

Logs of integrals do not expand directly, but the gradient of a log can be re-expanded:

$$\begin{aligned}
\text{LHS} &= \mathbb{E}_{(s_0, a_0^K, s_1, \dots)} \sum_{t=1}^{\infty} Q(s_t, a_t^K) \frac{1}{\int_{a_t^{K-1}} \cdots \int_{a_t^1} \prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t)} && \text{(B.3)} \\
&\qquad \cdot \nabla_\theta \int_{a_t^{K-1}} \cdots \int_{a_t^1} \prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \, \mathrm{d}a^1 \cdots \mathrm{d}a^{K-1}, \\
&= \mathbb{E}_{(s_0, a_0^K, s_1, \dots)} \sum_{t=1}^{\infty} Q(s_t, a_t^K) \frac{1}{\pi_\theta(a_t^K \mid s_t)} \int_{a_t^{K-1}} \cdots \int_{a_t^1} \nabla_\theta \prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \, \mathrm{d}a^1 \cdots \mathrm{d}a^{K-1}, && \text{(B.4)} \\
&= \mathbb{E}_{(s_0, a_0^K, s_1, \dots)} \sum_{t=1}^{\infty} \int \cdots \int Q(s_t, a_t^K) \frac{1}{\pi_\theta(a_t \mid s_t)} \left(\prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \right) \\
&\qquad \cdot \nabla_\theta \log\left(\prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \right) \mathrm{d}a^1 \cdots \mathrm{d}a^{K-1}, && \text{(B.5)} \\
&= \mathbb{E}_{(s_0, s_1, s_2, \dots)} \sum_{t=1}^{\infty} \int_{a_t^K} \int_{a_t^{K-1}} \cdots \int_{a_t^1} \left(\prod_{\tau=1}^{K} \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \right) && \text{(B.6)} \\
&\qquad \cdot Q(s_t, a_t^K) \nabla_\theta \sum_{\tau=1}^{K} \log \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \, \mathrm{d}a^1 \cdots \mathrm{d}a^K, && \text{(B.7)} \\
&= \mathbb{E}_{(s_0, \{a_0^\tau\}_{\tau=1}^K, s_1, \dots)} \sum_{t=1}^{\infty} Q(s_t, a_t^K) \left(\sum_{\tau=1}^{K} \nabla_\theta \log \pi(a_t^\tau \mid a_t^{\tau-1}, s_t) \right). && \text{(B.8)}
\end{aligned}$$

□

B.2 Proof of Theorem 5.2 (deterministic optimality)

Proof. Write $Q \coloneqq Q^{\Psi}$ for brevity. For any $\tau > 0$, the Q-function decomposes recursively:

$$\begin{aligned} Q(\mathbf{0},\mathbf{0}) &= \int_0^\infty \gamma^t r(s(t))\,\mathrm{d}t \\ &= \int_0^\tau \gamma^t r(s(t))\,\mathrm{d}t + \int_\tau^\infty \gamma^t r(s(t))\,\mathrm{d}t \\ &= \int_0^\tau \gamma^t r(s(t))\,\mathrm{d}t + \gamma^\tau \int_0^\infty \gamma^t r(s(t,s(\tau),a(\tau)))\,\mathrm{d}t \\ &= \int_0^\tau \gamma^t r(s(t))\,\mathrm{d}t + \gamma^\tau Q(s(\tau),a(\tau)). \end{aligned} \tag{B.9}$$

This recursion underlies all local analysis: the Q-function splits into a purely-state integral that can be made arbitrarily small and a boundary term that is another Q-function.

We proceed by contrapositive: suppose for some $t \in [0,\infty)$, $\Psi(s(t),a(t))$ is not aligned with $\nabla_a Q(s(t),a(t))$. We construct a perturbation Ψ' that strictly increases $Q(\mathbf{0},\mathbf{0})$.

Aligning at the origin

Suppose the misalignment occurs at $t=0$. Assume $\nabla_a Q(\mathbf{0},\mathbf{0}) \neq \mathbf{0}$ and $\Psi(\mathbf{0},\mathbf{0})$ is not collinear with $\nabla_a Q(\mathbf{0},\mathbf{0})$. We construct Ψ' such that $Q^{\Psi'}(\mathbf{0},\mathbf{0}) > Q^{\Psi}(\mathbf{0},\mathbf{0})$.

Choose τ and ε small enough that:

1. $s(\tau') = \tau' F(\mathbf{0},\mathbf{0}) + o(\tau')$ for all $0 < \tau' \leq \tau$,
2. $a(\tau') = \tau' \Psi(\mathbf{0},\mathbf{0}) + o(\tau')$ for all $0 < \tau' \leq \tau$,
3. $r(s) = r(\mathbf{0}) + \langle \nabla_s r(\mathbf{0}),\, s\rangle + o(\|s\|)$ for $\|s\| \leq \varepsilon$,
4. $Q(s,a) = Q(\mathbf{0},\mathbf{0}) + \langle \nabla_s Q(\mathbf{0},\mathbf{0}),\, s\rangle + \langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, a\rangle + o(\|(s:a)\|)$ for $\|s\|, \|a\| \leq \varepsilon$, where $(s:a) \in \mathbb{R}^{s+a}$ is the stacked vector,
5. $\|s(\tau')\|, \|a(\tau')\| \leq \varepsilon$ for $0 < \tau' \leq \tau$.

Then

$$\begin{aligned} Q(\mathbf{0},\mathbf{0}) &= \int_0^\tau \gamma^t r(s(t))\,\mathrm{d}t + \gamma^\tau Q(s(\tau),a(\tau)) \\ &= \int_0^\tau \gamma^t \big(r(\mathbf{0}) + \langle \nabla_s r(\mathbf{0}),\, s(t)\rangle + o(\|s(t)\|)\big)\,\mathrm{d}t \qquad \text{(B.10)} \\ &\quad + \gamma^\tau \big(Q(\mathbf{0},\mathbf{0}) + \langle \nabla_s Q(\mathbf{0},\mathbf{0}),\, s(\tau)\rangle + \langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, a(\tau)\rangle + o(\|(s(\tau):a(\tau))\|)\big) \\ &= \int_0^\tau \gamma^t \big(r(\mathbf{0}) + t\,\langle \nabla_s r(\mathbf{0}),\, F(\mathbf{0},\mathbf{0})\rangle\big)\,\mathrm{d}t \qquad \text{(B.11)} \\ &\quad + \gamma^\tau \big(Q(\mathbf{0},\mathbf{0}) + \tau\,\langle \nabla_s Q(\mathbf{0},\mathbf{0}),\, F(\mathbf{0},\mathbf{0})\rangle + \tau\,\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \Psi(\mathbf{0},\mathbf{0})\rangle\big) + o(\tau). \end{aligned}$$

Equation (B.11) splits $Q(\mathbf{0},\mathbf{0})$ into three pieces: an integral depending only on the origin's state dynamics $F(\mathbf{0},\mathbf{0})$; a term with linearly separable dependence on $\Psi(\mathbf{0},\mathbf{0})$; and a remainder decaying strictly faster than τ. We focus on the second piece, which contains separated dependence on Ψ.

We split into cases.

Case 1: no re-entry. Suppose there exists $\varepsilon > 0$ such that, after first escaping the ε-ball around the origin B_ε, $a(t)$ does not re-cross B_ε from any initial condition within B_ε. Then for this reduced neighborhood (and correspondingly reduced τ), $Q(\mathbf{0},\mathbf{0})$ is fully determined by the local integral $\int_0^\tau \gamma^t r(s(t))\,\mathrm{d}t$ while $(s(t),a(t))$ traverses B_ε, plus the boundary $\gamma^\tau Q(s(\tau),a(\tau))$ once it hits the boundary. Changes to Ψ inside B_ε leave $Q(s,a)$ on the boundary of B_ε unchanged.

Let $\phi : \mathbb{R}^a \to \mathbb{R}$ be a partition of unity with respect to B_ε: ϕ is a function of $\|a\|$ only, $\phi(a) = 1$ for $\|a\| \le \varepsilon/2$, $\phi(a) = 0$ for $\|a\| \ge \varepsilon$. Define

$$\Psi'(s,a) \;\coloneqq\; (1-\phi(a))\Psi(s,a) + \phi(a)\,\frac{\|\Psi(\mathbf{0},\mathbf{0})\|}{\|\nabla_a Q(\mathbf{0},\mathbf{0})\|}\,\nabla_a Q(\mathbf{0},\mathbf{0}). \tag{B.12}$$

By construction $\|\Psi'\|_{\mathcal{C}^0} \le C$ (preserving the norm bound on Ψ) and Ψ' is Lipschitz (with a higher but finite constant) as a product of Lipschitz functions. If $\Psi(\mathbf{0},\mathbf{0}) = \mathbf{0}$, replace $\|\Psi(\mathbf{0},\mathbf{0})\|$ with a constant chosen so $\|\Psi'\|_{\mathcal{C}^0} \le C$.

Now consider the improvement in Q:

$$\begin{aligned}
Q^{\Psi'}(\mathbf{0},\mathbf{0}) - Q^{\Psi}(\mathbf{0},\mathbf{0}) &= \gamma^\tau \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, a_{\Psi'}(\tau) - a_\Psi(\tau) \right\rangle + o(\tau) && \text{(B.13)}\\
&= \gamma^\tau \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \int_0^\tau \Psi'(\mathbf{0},a(t))\,\mathrm{d}t - \tau\Psi(\mathbf{0},\mathbf{0}) \right\rangle + o(\tau) && \text{(B.14)}\\
&= \gamma^\tau \int_0^\tau \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \Psi'(\mathbf{0},a(t)) - \Psi(\mathbf{0},\mathbf{0}) \right\rangle \mathrm{d}t + o(\tau) && \text{(B.15)}\\
&= \gamma^\tau \ell(\tau) + o(\tau), && \text{(B.16)}
\end{aligned}$$

where $\ell(\tau) \coloneqq \int_0^\tau \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \Psi'(\mathbf{0},a(t)) - \Psi(\mathbf{0},\mathbf{0}) \right\rangle \mathrm{d}t$. The desired inequality follows if $\ell(\tau) \ge M\tau$ for some $M > 0$, since this implies $\lim_{\tau\to 0} o(\tau)/\gamma^\tau \ell(\tau) = 0$ and hence there exists τ for which $\gamma^\tau \ell(\tau) > |o(\tau)|$, giving $Q^{\Psi'}(\mathbf{0},\mathbf{0}) > Q^{\Psi}(\mathbf{0},\mathbf{0})$.

To find M, split the integral at $\tau_{\varepsilon/2}$, the hitting time of $a(t)$ at $\|a(t)\| = \varepsilon/2$:

$$\begin{aligned}
\int_0^\tau \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \Psi'(\mathbf{0},a(t)) - \Psi(\mathbf{0},\mathbf{0}) \right\rangle \mathrm{d}t &= \tau_{\varepsilon/2} \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \tfrac{\|\Psi(\mathbf{0},\mathbf{0})\|}{\|\nabla_a Q(\mathbf{0},\mathbf{0})\|} \nabla_a Q(\mathbf{0},\mathbf{0}) \right\rangle \\
&\quad + \int_{\tau_{\varepsilon/2}}^\tau \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \Psi'(\mathbf{0},a(t)) - \Psi(\mathbf{0},\mathbf{0}) \right\rangle \mathrm{d}t && \text{(B.17)}\\
&\ge \tau_{\varepsilon/2}\, M', && \text{(B.18)}
\end{aligned}$$

where $M' \coloneqq \left\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \frac{\|\Psi(\mathbf{0},\mathbf{0})\|}{\|\nabla_a Q(\mathbf{0},\mathbf{0})\|} \nabla_a Q(\mathbf{0},\mathbf{0}) \right\rangle > 0$ (the second integral is non-negative by non-collinearity of $\nabla_a Q(\mathbf{0},\mathbf{0})$ and $\Psi(\mathbf{0},\mathbf{0})$ and the construction of Ψ' as a convex combination). Since the flow of $a(t)$ is governed by a constant vector up to $o(\tau)$, the constructed ε scales at most linearly with τ, so we can find $M > 0$ with $\tau_{\varepsilon/2} M' \geq M\tau$, concluding $\ell(\tau) \geq M\tau$ and the no-re-entry case.

Case 2: possible re-entry. Suppose we cannot find a reduced ε as above (e.g., in attractor systems). Leave ε as-is and construct Ψ' from (B.12). Let $\mathcal{T} \coloneqq \{T_1, T_1^e, T_2, T_2^e, \ldots\}$ be the increasing sequence of times at which $a_{\Psi'}(t)$ re-enters B_ε at T_i and exits at T_i^e (with T_1 the first re-entry after the initial escape from $t = 0$). Since $\nabla_a Q(\mathbf{0},\mathbf{0}) \neq \mathbf{0}$, the origin is not an equilibrium and each T_i has a paired T_i^e.

If $\mathcal{T}$ is empty, we repeat the no-re-entry analysis.

If $\mathcal{T}$ is finite, decompose $Q^{\Psi'}(\mathbf{0},\mathbf{0})$ as

$$\begin{aligned} Q^{\Psi'}(\mathbf{0},\mathbf{0}) = &\int_0^{T_1} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \int_{T_1}^{T_1^e} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t \\ &+ \ldots \\ &+ \int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \int_{T_k}^{T_k^e} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \gamma^{T_k^e} Q^{\Psi'}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e)). \end{aligned} \tag{B.19}$$

Focusing on the last two summands, with explanations of each step below:

$$\begin{aligned} &\int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \int_{T_k}^{T_k^e} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \gamma^{T_k^e} Q^{\Psi'}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e)) \\ &\quad = \int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \int_{T_k}^{T_k^e} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \gamma^{T_k^e} Q^{\Psi}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e)) && \text{(B.20)} \\ &\quad > \int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \int_{T_k}^{T_k^e} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \gamma^{T_k^e} Q^{\Psi}(s_{\Psi'_{T_k}}(T_k^e), a_{\Psi'_{T_k}}(T_k^e)) && \text{(B.21)} \\ &\quad = \int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t \\ &\qquad + \gamma^{T_k} \left(\int_0^{T_k^e - T_k} \gamma^t r(s_{\Psi'_{T_k}}(t))\,\mathrm{d}t + o(\tau) + \gamma^{T_k^e - T_k} Q^{\Psi}(s_{\Psi'_{T_k}}(T_k^e), a_{\Psi'_{T_k}}(T_k^e)) \right) && \text{(B.22)} \\ &\quad = \int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \gamma^{T_k} Q^{\Psi}(s_{\Psi'}(T_k), a_{\Psi'}(T_k)) + \gamma^{T_k} o(\tau) && \text{(B.23)} \\ &\quad = \int_{T_{k-1}^e}^{T_k} \gamma^t r(s_{\Psi'_{T_{k-1}^e}}(t))\,\mathrm{d}t + \gamma^{T_k} Q^{\Psi}(s_{\Psi'}(T_k), a_{\Psi'}(T_k)) + \gamma^{T_k} o(\tau) && \text{(B.24)} \\ &\quad = \gamma^{T_{k-1}^e} Q^{\Psi}(s_{\Psi'}(T_{k-1}^e), a_{\Psi'}(T_{k-1}^e)) + \gamma^{T_k} o(\tau), && \text{(B.25)} \end{aligned}$$

where $s_{\Psi'_T}(t)$ for $T < t$ denotes the state from integrating Ψ' until T, then switching to Ψ. The steps are:

1. (B.20): T_k^e is the last time $a(t)$ crosses B_ε, so $\Psi' = \Psi$ thereafter and $Q^{\Psi'}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e)) = Q^{\Psi}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e))$.
2. (B.21): same analysis as in the no-re-entry case (Case 1), focused on the Taylor-approximated Q^Ψ. The neighborhood is now B_ε (not $B_{\varepsilon/2}$), but the linearity argument up to (B.10) still applies.
3. (B.22): reparameterize the integrals; the internal reward integral $\int_{T_k}^{T_k^e} \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t$ depends on action dynamics only at $o(\tau)$, where τ is the maximum difference between T_i and T_i^e and is decreased with ε via Ψ being Lipschitz.
4. (B.23): wrap the recursive equation for Q (B.9), removing dependence on Ψ' from T_k onwards.
5. (B.24): by construction of $\mathcal{T}$, $\Psi' = \Psi$ on $[T_{k-1}^e, T_k]$.
6. (B.25): wrap the recursive equation for Q again.

Iterating this process inductively gives

$$Q^{\Psi'}(\mathbf{0},\mathbf{0}) > \int_0^\tau \gamma^t r(s_{\Psi'}(t))\,\mathrm{d}t + \gamma^\tau Q^\Psi(s_{\Psi'}(\tau), a_{\Psi'}(\tau)) + \left(\sum_{i=1}^k \gamma^{T_i}\right) o(\tau) \tag{B.26}$$

$$= Q^\Psi(\mathbf{0},\mathbf{0}) + \gamma^\tau \ell(\tau) + \left(1 + \sum_{i=1}^k \gamma^{T_i}\right) o(\tau). \tag{B.27}$$

Since $\lim_{\tau\to 0}(1+\sum_i \gamma^{T_i}) o(\tau)/\gamma^\tau \ell(\tau) = 0$, there exists τ for which $Q^{\Psi'}(\mathbf{0},\mathbf{0}) > Q^\Psi(\mathbf{0},\mathbf{0})$.

If $\mathcal{T}$ is infinite, the base case for induction is missing. Using $0 \le Q^\Psi(s,a), Q^{\Psi'}(s,a) \le \int_0^\infty \gamma^t\,\mathrm{d}t = (\log(\gamma^{-1}))^{-1}$, we can write for any fixed k:

$$\gamma^{T_k^e} Q^{\Psi'}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e)) \;=\; \gamma^{T_k^e} Q^{\Psi}(s_{\Psi'}(T_k^e), a_{\Psi'}(T_k^e)) + \gamma^{T_k^e} q_k, \tag{B.28}$$

for some $|q_k| \le (\log(\gamma^{-1}))^{-1}$. Repeating the analysis,

$$Q^{\Psi'}(\mathbf{0},\mathbf{0}) \;>\; Q^{\Psi}(\mathbf{0},\mathbf{0}) + \gamma^\tau \ell(\tau) + \left(1 + \sum_{i=1}^k \gamma^{T_i}\right) o(\tau) + \gamma^{T_k^e} q_k. \tag{B.29}$$

Since $\sum_{i=1}^\infty \gamma^{T_i} < \infty$, there exists τ and a constant $\kappa > 0$ such that $\gamma^\tau \ell(\tau) + (1+\sum_i \gamma^{T_i}) o(\tau) > \kappa$ regardless of k. Choosing k' with $|\gamma^{T_{k'}^e} q_{k'}| < \kappa$ gives $Q^{\Psi'}(\mathbf{0},\mathbf{0}) > Q^\Psi(\mathbf{0},\mathbf{0})$.

Aligning beyond the origin

Suppose Ψ is aligned with $\nabla_a Q$ until some $t' > 0$. Split

$$Q(\mathbf{0},\mathbf{0}) = \int_0^{t'-\tau} r(s(t))\,\mathrm{d}t + \gamma^{t'-\tau}\left(\int_0^{\tau} \gamma^t r(s(t+t'-\tau))\,\mathrm{d}t + \gamma^{\tau} Q(s(t),a(t))\right) \tag{B.30}$$

and repeat the analysis on $Q(s(t),a(t))$. The first summand is unaffected by changing Ψ to Ψ', and τ can be chosen small enough that $\int_0^{\tau} \gamma^t r(s(t+t'-\tau))\,\mathrm{d}t$ is only state-dependent up to error $o(\tau)$. □

Extensions. A trivial extension of Theorem 5.2 is to any starting condition $Q(s_0,a_0)$, and then to Corollary 5.3 where we consider expectations $\mathbb{E}_{\mathbb{P}\times\pi} Q^{\Psi}(s,a)$. Since the analysis made perturbations in strictly positive-measure regions, we can repeat it to nudge Ψ at any non-aligned (s,a) to strictly increase Q at (s,a), and in a measure-positive region around it. Since this change does not decrease $Q(s',a')$ at any other (s',a') by repeating the "aligning beyond the origin" analysis, the constructed Ψ' achieves Corollary 5.3.

B.3 Proof of Theorem 5.4 (stochastic optimality)

We first formalize the support of the full joint distribution.

Definition B.2. A point $(s,a) \in \mathbb{R}^s \times \mathbb{R}^a$ is in the support of the joint distribution between initial-condition distribution $\mathbb{P}\times\pi$ and the stochastic dynamics of $(s(t),a(t))$ if for every $\varepsilon > 0$ the probability that $(a(t),s(t)) \in B_\varepsilon(s,a)$ at some finite t is strictly positive.

Proof. We first repeat the first part of the deterministic proof: focus on $Q(\mathbf{0},\mathbf{0})$ and the case where the misalignment occurs at the origin, then extend.

Assume $\nabla_a Q(\mathbf{0},\mathbf{0})$ is not collinear with $\Psi(\mathbf{0},\mathbf{0})$; we construct Ψ' with $Q^{\Psi'}(\mathbf{0},\mathbf{0}) > Q^{\Psi}(\mathbf{0},\mathbf{0})$.

Start from the stochastic recursive equation for the Q-function for any $\tau > 0$, with expectations $\mathbb{E}$ over the stochastic dynamics:

$$\begin{aligned} Q(\mathbf{0},\mathbf{0}) &= \mathbb{E}\int_0^{\infty} \gamma^t r(s(t))\,\mathrm{d}t \\ &= \mathbb{E}_{s,a}\big[\mathbb{E}[\int_0^{\infty} \gamma^t r(s(t))\,\mathrm{d}t \mid (s(\tau),a(\tau)) = (s,a)]\big] \\ &= \mathbb{E}\big[\int_0^{\tau} \gamma^t r(s(t))\,\mathrm{d}t\big] + \gamma^{\tau}\mathbb{E}_{s,a}[Q(s,a) \mid (s(\tau),a(\tau)) = (s,a)]. \end{aligned} \tag{B.31}$$

The new conditions on ε,τ:

1. $r(s) = r(\mathbf{0}) + \langle \nabla_s r(\mathbf{0}),\, s\rangle + o(\|s\|)$ for $\|s\| \le \varepsilon$,
2. $Q(s,a) = Q(\mathbf{0},\mathbf{0}) + \langle \nabla_s Q(\mathbf{0},\mathbf{0}),\, s\rangle + \langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, a\rangle + o(\|(s:a)\|)$ for $\|s\|, \|a\| \le \varepsilon$,

3. τ chosen so trajectories $\{s(t)\}_{0\le t\le\tau}, \{a(t)\}_{0\le t\le\tau}$ are bounded in an ε-ball with sublinear tail in τ: $\P(\|s(\tau)\|, \|a(\tau)\| > \varepsilon) = o(\tau)$,
4. $\mathbb{E}s(\tau') = \tau' F(\mathbf{0},\mathbf{0}) + o(\tau')$ for $0 < \tau' \le \tau$,
5. $\mathbb{E}a(\tau') = \tau' \Psi(\mathbf{0},\mathbf{0}) + o(\tau')$ for $0 < \tau' \le \tau$.

Construction 3 follows from splitting the joint SDE solution $\mathbf{X}(\tau) = \int_0^\tau \mathbf{F}(\mathbf{X}(t))\,\mathrm{d}t + \int_0^\tau \mathbf{\Sigma}(\mathbf{X}(t))\,\mathrm{d}B_t$ (with $\mathbf{X}(t) = (s(t) : a(t))$ and $\mathbf{F}, \mathbf{\Sigma}$ defined accordingly): both the path integral of a Lipschitz function and the Brownian motion satisfy the bound. Constructions 4 and 5 follow from differentiability of $m(\tau) = \mathbb{E}\mathbf{X}(\tau)$ at $\tau = 0$, using $\mathbb{E}\mathbf{X}(\tau)' = \mathbb{E}\mathbf{F}(\mathbf{X}(\tau))$ (since $\mathbb{E}[B_{\tau'} - B_\tau] = \mathbf{0}$) and $\mathbb{E}\mathbf{F}(\mathbf{X}(0)) = \mathbf{F}(\mathbf{X}(0))$.

Repeat the steps from (B.10) to (B.11), splitting the expectation by whether $(s(\tau), a(\tau)) \in B_\varepsilon$:

$$\begin{aligned} Q(\mathbf{0},\mathbf{0}) = \int_0^\tau \gamma^t \big(r(\mathbf{0}) + t\,\langle \nabla_s r(\mathbf{0}),\, F(\mathbf{0},\mathbf{0})\rangle\big)\,\mathrm{d}t \\ + \gamma^\tau \big(Q(\mathbf{0},\mathbf{0}) + \tau\,\langle \nabla_s Q(\mathbf{0},\mathbf{0}),\, F(\mathbf{0},\mathbf{0})\rangle + \tau\,\langle \nabla_a Q(\mathbf{0},\mathbf{0}),\, \Psi(\mathbf{0},\mathbf{0})\rangle\big) + o(\tau). \end{aligned} \tag{B.32}$$

Since (B.32) matches (B.11), we can proceed as in the deterministic case, conditioning on events (using the tower rule) to exploit specific path structure, e.g., conditioning on the number of times each sampled path $(s(t), a(t))$ re-crosses the ε-ball at the origin. Definition B.2, together with the analysis above, ensures that local perturbations Ψ' produce strict improvements of the Q-function, both at a fixed starting condition and in expectation over starting conditions. □

A note on optimization over vector fields. The proofs consider optimizations over smooth families of vector fields $\Psi : \mathbb{R}^s \times \mathbb{R}^a \to \mathbb{R}^a$ bounded in $\mathcal{C}^0$ norm and Lipschitz with some constant. This choice gives enough control to bound the function space while remaining flexible enough to permit local perturbations toward optima. Tightening the optimization space (e.g., to a fixed Lipschitz constant) and sharpening the ascent rule (cf. (B.12)) are interesting directions for future work; here we focus on the relation of Ψ to its Q-function action gradient.

B.4 The Boltzmann fixed-point reading

We record the argument behind Proposition 5.5 and the continuous-setting idealization that accompanies it, and are explicit about the gap between that idealization and the deployed policy.

Suppose the QSM loss (5.9) is minimized at some $\theta^\star$ with $\varepsilon_{\theta^\star}(a, s, k) = \lambda\,\nabla_a Q_\eta(s, a)$ for all (a, s, k) encountered in training. Then the action update (5.1) is a discretization of the overdamped Langevin SDE

$$\mathrm{d}a \;=\; \lambda\,\nabla_a Q_\eta(s,a)\,\mathrm{d}t \;+\; \sqrt{2}\,\mathrm{d}W. \tag{B.33}$$

Under standard conditions (Bakry et al., 2014), when this SDE admits a stationary distribution it is $\pi^{(\lambda)} \propto \exp(\lambda Q_\eta)$, tending to $\exp(\lambda Q^*)$ as $Q_\eta \to Q^*$ (the continuous analogue of the gridworld observation (5.17), presented informally in Psenka and Abbeel, 2024). Two conditions are required, neither automatic. First, $\exp(\lambda Q_\eta(s, \cdot))$ must be integrable over $\mathcal{A}$: on an unbounded $\mathcal{A} = \mathbb{R}^a$ this fails when Q_η is merely bounded, since the density is then non-normalizable, and it holds when $\mathcal{A}$ is compact or $\lambda Q_\eta(s, a) \to -\infty$ sufficiently fast as $\|a\| \to \infty$. Second, the Boltzmann law is the $t \to \infty$ limit of the Langevin flow, whereas the deployed policy runs a fixed, small number K of steps over a finite horizon, so $a^{(0)}$ realizes $\pi^{(\lambda)}$ only approximately. In the finite-action case both conditions are automatic ($\mathcal{A}$ finite $\Rightarrow \sum_{a'} e^{\lambda Q(s,a')} < \infty$, and (5.18) is a contraction under standard conditions), which gives Proposition 5.5.

B.5 The algorithm in full

The expanded form of Algorithm 2 with all hyperparameter and target-network specifics is given in the main body. The implementation uses TD3-style (Fujimoto et al., 2018a) twin critics with target-network smoothing and the standard min-of-two-critics trick to mitigate overestimation. For each gradient step, actions from the replay buffer are noised via a variance-preserving Markov chain (cf. Hansen-Estruch et al., 2023), and the QSM loss (5.9) is computed at the noised actions against $\nabla_a Q$ from $\min(Q_{\eta_1}, Q_{\eta_2})$. We add a small Gaussian noise to final sampled actions for exploration.

B.6 Additional experiments

Policy gradient baseline across environments

Figure B.1 provides full results for the policy-gradient baseline of Section 5.8 across remaining tested environments. Across the suite, the direct policy-gradient update fails to learn optimal policies for diffusion-model policies in the off-policy setting. Some initializations of `quadruped_walk` admit an unusually broad range of high-reward policies (e.g., when dropped right-side up).

Diffusion depth scaling

Figure B.2 compares QSM and Diffusion-QL on `walker_walk` as the number of diffusion sampling steps K increases. Because QSM uses the score-based structure of the diffusion model and trains the denoising network directly, it scales with depth without instability. Diffusion-QL, which backpropagates through the full reverse-diffusion chain, becomes unstable as K grows.

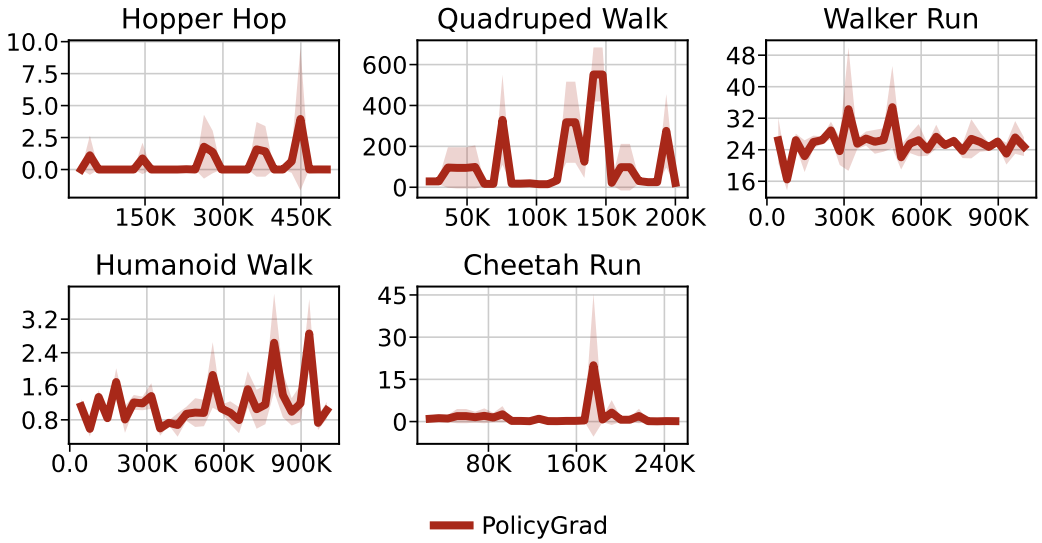


Figure B.1: Results on the policy-gradient update for diffusion-model policies on the remaining environments. The pattern of Section 5.8 persists: direct policy gradients fail to learn optimal policies for diffusion-model policies in the off-policy setting.

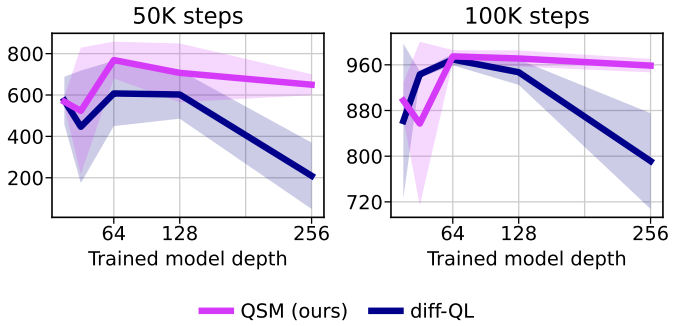


Figure B.2: Comparison of QSM and Diffusion-QL (Zhendong Wang et al., 2022) as the number of diffusion sampling steps K increases, on `walker_walk`. Plotted are validation returns at 50K and 100K training steps. QSM scales with diffusion depth; Diffusion-QL becomes unstable.

B.7 Hyperparameters and benchmark details

Default hyperparameters. Discount factor $\gamma = 0.99$; replay buffer size 10^6; batch size 256; learning rate 3×10^{-4} (Adam (Kingma and Ba, 2017)); twin-critic target soft-update rate $\tau = 5 \times 10^{-3}$; QSM multiplier $\lambda = 1.0$; number of diffusion steps $K = 10$; cosine noise schedule.

The QSM multiplier λ is the most important hyperparameter. On a representative subset of tasks, we swept λ over 0.1, 0.3, 1.0, 3.0, and 10.0. We chose $\lambda = 1.0$ as a default with low

across-task sensitivity. Task-specific sweeps may give modest further gains.

Benchmark tasks. DeepMind Control Suite (Tassa et al., 2018; Tunyasuvunakool et al., 2020) tasks evaluated: humanoid (stand, walk, run), quadruped (walk, run), dog (walk, run), fish, cheetah, walker. Selected MuJoCo Gym tasks were used for cross-benchmark comparison.

Baselines. SAC (Haarnoja et al., 2018), TD3 (Fujimoto et al., 2018a), DDPG (T. P. Lillicrap et al., 2019), Diffusion-QL (Zhendong Wang et al., 2022), with matched architecture and hyperparameters. Reported numbers are mean and standard deviation over 5 seeds per setting.

Compute. Each QSM run takes comparable wall-clock to a matched Diffusion-QL run; the local nature of the QSM loss avoids the expensive backprop chain at the cost of slightly more environment interactions to achieve similar performance.

Full quantitative tables are in Psenka and Abbeel, 2024.

Appendix C

Appendix to Chapter 6 (Onsager–Machlup Interpolation)

This appendix collects technical material for Chapter 6: equivalence of the DDPM and flow-matching training objectives to score matching, derivation of the OM action from a Fokker-Planck equation and via the δ-tubes formalism, the geodesic-energy equivalence theorem, properties of the Gaussian-blur norm, detailed experimental setups and hyperparameters for both the protein and image domains, and additional experiments deferred from the main body.

C.1 Score-based training objectives

For self-containment we replicate the proofs showing that the training objectives for DDPM and flow-matching models are each equivalent to a score-matching objective. See Vincent (2011) for the original DDPM proof and Lipman, Havasi, et al. (2024) for the flow-matching version.

The DDPM reverse process

The sampling process for a DDPM can be written via the terminal $x^{(0)}$ of the following:

$$x^{(T_d)} \sim \mathcal{N}(0, I), \tag{C.1}$$

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

This is an Euler-Maruyama discretization of an SDE, but the vector field $\mathbf{s}_\theta(x^{(i)}, i)$ varies with i; trajectory iterates near the noise scale $i = T_d$ will not follow dynamics determined by the data distribution. This is why the iterative-noise/denoise construction of Section 6.1 is used instead.

DDPM as score matching

Theorem C.1 (DDPM-score matching equivalence)**.** *Let $p_{data}(x_0)$ be the data distribution, and let x_τ be the noised variable via the forward process*

$$x_\tau = \sqrt{\bar\alpha_\tau}\, x_0 + \sqrt{1-\bar\alpha_\tau}\, \varepsilon, \quad \tau \sim \mathrm{Unif}(\{1,\dots,T_d\}), \quad x_0 \sim p_{data}, \quad \varepsilon \sim \mathcal{N}(0,I), \tag{C.3}$$

with $\bar\alpha_\tau \in (0,1)$. Let $p_\tau(x_\tau) = \int p_{data}(x_0)\,\mathcal{N}(x_\tau; \sqrt{\bar\alpha_\tau}x_0, (1-\bar\alpha_\tau)I)\,\mathrm{d}x_0$. Then the DDPM objective

$$\mathcal{L}_{DDPM}(\theta) \;\coloneqq\; \mathbb{E}_{\tau,x_0,\varepsilon}\big[\|\varepsilon - \varepsilon_\theta(x_\tau,\tau)\|_2^2\big] \tag{C.4}$$

satisfies

$$\mathcal{L}_{DDPM}(\theta) \;=\; \mathbb{E}_{\tau,x_\tau}\big[(1-\bar\alpha_\tau)\,\|\nabla_{x_\tau}\log p_\tau(x_\tau) - s_\theta(x_\tau,\tau)\|_2^2\big] + C, \tag{C.5}$$

where $s_\theta(x_\tau,\tau) \coloneqq -\varepsilon_\theta(x_\tau,\tau)/\sqrt{1-\bar\alpha_\tau}$ and C is independent of θ.

Proof. The core of the derivation is Tweedie's formula. For any random variable $z = \mu + \sigma\eta$ where μ is an arbitrary random vector, $\eta \sim \mathcal{N}(0,I)$, $\sigma > 0$:

$$\mathbb{E}[\mu \mid z] = z + \sigma^2 \nabla_z \log p(z), \tag{C.6}$$

where $p(z) = \int \mathcal{N}(z;\mu,\sigma^2 I)p(\mu)\mathrm{d}\mu$ is the marginal of z. Applied to x_τ generated via $x_\tau = \sqrt{\bar\alpha_\tau}\, x_0 + \sqrt{1-\bar\alpha_\tau}\,\varepsilon$ (with $\mu = \sqrt{\bar\alpha_\tau}\, x_0$, $\sigma = \sqrt{1-\bar\alpha_\tau}$):

$$\mathbb{E}[\sqrt{\bar\alpha_\tau}\, x_0 \mid x_\tau] = x_\tau + (1-\bar\alpha_\tau)\,\nabla_{x_\tau}\log p_\tau(x_\tau). \tag{C.7}$$

Dividing by $\sqrt{\bar\alpha_\tau}$ gives the posterior of x_0:

$$\mathbb{E}[x_0 \mid x_\tau] = \tfrac{x_\tau}{\sqrt{\bar\alpha_\tau}} + \tfrac{1-\bar\alpha_\tau}{\sqrt{\bar\alpha_\tau}}\,\nabla_{x_\tau}\log p_\tau(x_\tau). \tag{C.8}$$

Rewriting $\varepsilon = (x_\tau - \sqrt{\bar\alpha_\tau}\, x_0)/\sqrt{1-\bar\alpha_\tau}$ and taking the conditional expectation given x_τ:

$$\begin{aligned}\mathbb{E}[\varepsilon \mid x_\tau] &= \tfrac{x_\tau - \sqrt{\bar\alpha_\tau}\,\mathbb{E}[x_0|x_\tau]}{\sqrt{1-\bar\alpha_\tau}} \\ &= -\sqrt{1-\bar\alpha_\tau}\,\nabla_{x_\tau}\log p_\tau(x_\tau).\end{aligned} \tag{C.9}$$

Using the law of total expectation:

$$\mathcal{L}_{\mathrm{DDPM}}(\theta) = \mathbb{E}_{\tau,x_\tau}\big[\mathbb{E}_\varepsilon[\|\varepsilon - \varepsilon_\theta(x_\tau,\tau)\|_2^2 \mid x_\tau,\tau]\big]. \tag{C.10}$$

The standard bias-variance decomposition gives

$$\mathbb{E}_\varepsilon[\|\varepsilon - \varepsilon_\theta(x_\tau,\tau)\|_2^2 \mid x_\tau,\tau] = \|\mathbb{E}[\varepsilon \mid x_\tau,\tau] - \varepsilon_\theta(x_\tau,\tau)\|_2^2 + \mathbb{E}[\|\varepsilon - \mathbb{E}[\varepsilon \mid x_\tau,\tau]\|_2^2 \mid x_\tau,\tau]. \tag{C.11}$$

The variance term is independent of θ. Substituting the conditional expectation and factoring out $-\sqrt{1-\bar\alpha_\tau}$ yields the score-matching form with $s_\theta = -\varepsilon_\theta/\sqrt{1-\bar\alpha_\tau}$. □

Flow matching to score matching

Theorem C.2 (Flow-matching to score conversion)**.** *Let $p_{data}(x_1)$ be the data distribution, and let $x_\tau = \alpha_\tau x_1 + \sigma_\tau x_0$ for $\tau \sim \mathrm{Unif}([0,1])$, $x_1 \sim p_{data}$, $x_0 \sim \mathcal{N}(0, I)$, where $\alpha_\tau, \sigma_\tau : [0,1] \to [0,1]$ are strictly increasing/decreasing with $\alpha_0 = \sigma_1 = 0$, $\alpha_1 = \sigma_0 = 1$. The flow-matching objective*

$$\mathcal{L}_{FM}(\theta) \coloneqq \mathbb{E}_{\tau,x_0,x_1}\left[\|u_\theta(x_\tau,\tau) - v_\tau(x_0,x_1)\|_2^2\right], \tag{C.12}$$

where $v_\tau(x_0,x_1) = \dot\alpha_\tau x_1 + \dot\sigma_\tau x_0$, satisfies:

1. *Equivalently trains against the unconditional velocities $u_\tau(x)$:*

$$\mathcal{L}_{FM}(\theta) = \mathbb{E}_{\tau,x_0,x_1}\left[\|u_\theta(x_\tau,\tau) - u_\tau(x_\tau)\|_2^2\right] + C, \tag{C.13}$$

   where $u_\tau(x) = \mathbb{E}_{x_0,x_1}[\dot\alpha_\tau x_1 + \dot\sigma_\tau x_0 \mid x_\tau = x]$.

2. *The equality $\nabla_x \log p_\tau(x) = \frac{\alpha_\tau}{\dot\sigma_\tau\sigma_\tau\alpha_\tau - \dot\alpha_\tau\sigma_\tau^2}\left(\frac{\dot\alpha_\tau}{\alpha_\tau}x - u_\tau(x)\right)$ holds, allowing $\mathcal{L}_{FM}$ to be written as a score-matching objective.*

Proof. For Part 1, telescope the integrand around $u_\tau(x_\tau)$:

$$\begin{aligned}\|u_\theta(x_\tau,\tau) - v_\tau(x_0,x_1)\|^2 &= \|u_\theta(x_\tau,\tau) - u_\tau(x_\tau)\|^2 \\ &\quad + 2\,\langle u_\theta(x_\tau,\tau) - u_\tau(x_\tau),\, u_\tau(x_\tau) - v_\tau(x_0,x_1)\rangle \\ &\quad + \|u_\tau(x_\tau) - v_\tau(x_0,x_1)\|^2 .\end{aligned} \tag{C.14}$$

The third term is θ-independent. For the cross term, by the tower rule and the definition of u_τ:

$$\begin{aligned}\mathbb{E}_{x_0,x_1}\left[\langle u_\theta(x_\tau,\tau) - u_\tau(x_\tau),\, u_\tau(x_\tau) - v_\tau(x_0,x_1)\rangle \mid x_\tau,\tau\right] &= \langle u_\theta - u_\tau,\, u_\tau - \mathbb{E}[v_\tau \mid x_\tau]\rangle \\ &= \langle u_\theta - u_\tau,\, u_\tau - u_\tau\rangle = 0.\end{aligned} \tag{C.15}$$

This gives Part 1.

For Part 2, the interpolation $x_\tau = \alpha_\tau x_1 + \sigma_\tau x_0$ satisfies Tweedie's formula:

$$\mathbb{E}[\alpha_\tau x_1 \mid x_\tau] = x_\tau + \sigma_\tau^2\,\nabla_x \log p_\tau(x_\tau). \tag{C.16}$$

Dividing by α_τ:

$$\mathbb{E}[x_1 \mid x_\tau] = \tfrac{x_\tau}{\alpha_\tau} + \tfrac{\sigma_\tau^2}{\alpha_\tau}\,\nabla_x \log p_\tau(x_\tau). \tag{C.17}$$

Noting $x_0 = (x_\tau - \alpha_\tau x_1)/\sigma_\tau$:

$$\begin{aligned}u_\tau(x) &= \dot\alpha_\tau \mathbb{E}[x_1 \mid x_\tau] + \dot\sigma_\tau \mathbb{E}[x_0 \mid x_\tau] \\ &= \tfrac{\dot\sigma_\tau}{\sigma_\tau}x + \left(\dot\alpha_\tau - \tfrac{\alpha_\tau\dot\sigma_\tau}{\sigma_\tau}\right)\mathbb{E}[x_1 \mid x_\tau] \\ &= \tfrac{\dot\alpha_\tau}{\alpha_\tau}x - \tfrac{\dot\sigma_\tau\sigma_\tau\alpha_\tau - \dot\alpha_\tau\sigma_\tau^2}{\alpha_\tau}\,\nabla_x \log p_\tau(x),\end{aligned} \tag{C.18}$$

which is the desired relation. □

C.2 Derivation of the OM action

We give two complementary derivations of the OM action: one via the Fokker-Planck equation for the path probability, and one via δ-tubes around candidate paths.

Fokker-Planck derivation

Considering a single particle (multi-particle extension in Kappler and Adhikari (2020a)), the SDE (6.1) induces a probability density satisfying the Fokker-Planck equation

$$\frac{\partial P}{\partial t} = -\frac{\partial}{\partial x}\big(\tfrac{\mathbf{\Phi}}{\zeta} P\big) + \frac{\partial}{\partial x}\big(D\tfrac{\partial P}{\partial x}\big), \tag{C.19}$$

where $\mathbf{\Phi}(x) = -\partial\phi(x)/\partial x$ is the gradient drift.

Case 1: $\mathbf{\Phi} = 0$. The Fokker-Planck equation reduces to a pure diffusion. The standard heat-kernel solution is

$$P(x, t \mid x_0) = (4\pi D t)^{-d/2}\, e^{-\|x - x_0\|_2^2/(4Dt)}. \tag{C.20}$$

For a sequence (x^j, t^j) at discretized times with $t^j - t^{j-1} = \varepsilon$:

$$\prod_j P(x^j, \varepsilon \mid x^{j-1}) = (4\pi D\varepsilon)^{-dN/2}\, \exp\Big(-\tfrac{1}{4D\varepsilon}\sum_j \big\|x^j - x^{j-1}\big\|_2^2\Big). \tag{C.21}$$

Taking $N \to \infty$ and $\varepsilon = t/N$, the argument of the exponential is a Riemann sum that converges to

$$\tfrac{1}{4D}\int_0^t \|\dot{x}\|^2 \, \mathrm{d}s. \tag{C.22}$$

The prefactor diverges as $N \to \infty$, but it is independent of the path and can be absorbed into a normalization constant (see Gel'fand and Yaglom (1960) for justification). The path probability is then

$$P(x, t) = C\, \exp\Big(-\tfrac{1}{4D}\int_0^t \|\dot{x}(s)\|^2 \, \mathrm{d}s\Big), \tag{C.23}$$

and maximizing the path likelihood is equivalent to minimizing

$$S_0(x(t)) = \tfrac{1}{4D}\int_0^t \|\dot{x}(s)\|^2 \, \mathrm{d}s. \tag{C.24}$$

Case 2: $\mathbf{\Phi} \neq 0$. Use the transformation

$$P(x, t \mid x_0) = G(x, t \mid x_0)\, \exp\Big(\tfrac{1}{2D\zeta}\int_{x(0)}^{x(t)} \mathbf{\Phi}(s)\, \mathrm{d}s\Big). \tag{C.25}$$

Since $\mathbf{\Phi} = -\nabla\phi$, the integral simplifies to $-\bar{\phi}(x) = \frac{1}{2D\zeta}(-\phi(x) + \phi(x_0))$, and $\nabla\bar{\phi} = \frac{1}{2D\zeta}\nabla\phi$. Substituting (C.25) into (C.19) and computing derivatives, the function G satisfies a diffusion-reaction equation

$$\frac{\partial G}{\partial t} = D\,\frac{\partial^2 G}{\partial x^2} - G\left(\frac{1}{4D}\left(\frac{1}{\zeta}\nabla\phi\right)^2 - \frac{1}{2\zeta}\nabla^2\phi\right). \tag{C.26}$$

For $\phi = 0$ this is the heat equation, with solution as in (C.20). For general ϕ, the solution has the form

$$\begin{aligned} G(x,t) &= u_0(x,t)\, e^{-\int_0^t k(s)\,\mathrm{d}s}, \\ k(x) &= \frac{1}{4D}\left(\frac{1}{\zeta}\nabla\phi(x)\right)^2 - \frac{1}{2\zeta}\nabla^2\phi(x). \end{aligned} \tag{C.27}$$

Combining via (C.25) and discretizing in time, the path probability is

$$P(x,t) = C\,\exp\Big[-\frac{1}{4D}\int_0^t \left(\dot{x}^2 + \left(\frac{1}{\zeta}\nabla\phi\right)^2 - \frac{2D}{\zeta}\nabla^2\phi - 2\mathbf{\Phi}\right)\mathrm{d}s\Big]. \tag{C.28}$$

The negative log of the integrand gives the OM action. After fixing endpoints (which makes the $\int \mathbf{\Phi}$ term a boundary contribution $2D^{-1}(\phi(x) - \phi(x_0))$, independent of the path), and recombining to match the form of Definition 6.1, we recover

$$S(x(t)) = \frac{1}{4D}\int_0^t \left(\dot{x}^2 + \left(\frac{1}{\zeta}\nabla\phi\right)^2 - \frac{2D}{\zeta}\nabla^2\phi\right)\mathrm{d}s. \tag{C.29}$$

Discretizing in time (Stratonovich convention, see Adib (2008)) and extending to multi-particle systems gives the discrete form of Definition 6.1.

δ-tubes derivation

An alternative formalization (Dürr and Bach, 1978; Kappler and Adhikari, 2020b; Dekker, 1978; Dutra et al., 2014) considers the relative probability of tubes of width δ around candidate paths and takes $\delta \to 0$.

Theorem C.3 (Path-likelihood ratio for δ-tubes)**.** *Consider an Itô SDE over $\mathbb{R}^d$:*

$$\mathrm{d}x = F(x)\,\mathrm{d}t + \sqrt{\varepsilon}\,\mathrm{d}W_t, \tag{C.30}$$

with globally Lipschitz F. Let $\mathbb{P}_{x_0,T}$ denote the path measure on $C_0([0,T])$. For $\gamma \in C^1([0,T])$, define the δ-tube

$$B_\delta(\gamma) \coloneqq \{\gamma' \in C_0([0,T]) : \sup_{t\in[0,T]} \|\gamma'(t) - \gamma(t)\|_2 \le \delta\}. \tag{C.31}$$

Then for $\gamma_0, \gamma_1 \in C^1([0,T])$ both starting from x_0:

$$\lim_{\delta\to 0} \log \frac{\mathbb{P}_{x_0,T}[B_\delta(\gamma_0)]}{\mathbb{P}_{x_0,T}[B_\delta(\gamma_1)]} = -\frac{1}{2\varepsilon}\Big(\int_0^T \|\dot{\gamma}_0 - F(\gamma_0)\|_2^2\,\mathrm{d}t - \int_0^T \|\dot{\gamma}_1 - F(\gamma_1)\|_2^2\,\mathrm{d}t\Big). \tag{C.32}$$

Proof. Fix $N \in \mathbb{N}$, $h = T/N$, $t_k = kh$. For an arbitrary path X_t let $X_k = X_{t_k}$ and $\Delta X_k = X_{k+1} - X_k$, $\Delta\gamma_k = \gamma(t_{k+1}) - \gamma(t_k)$.

Let $p_N(x_1, \dots, x_N)$ be the density generated by Euler-Maruyama discretization. Explicitly:

$$p_N(x_1, \dots, x_N) = (2\pi\varepsilon h)^{-dN/2} \exp\Big(-\sum_{k=0}^{N-1} \tfrac{\|x_{k+1} - x_k - F(x_k)h\|_2^2}{2\varepsilon h}\Big). \tag{C.33}$$

Define the discrete tube

$$B_\delta^{(N)}(\gamma) \coloneqq \{X : \max_{0 \le k \le N} \|X_{t_k} - \gamma(t_k)\|_2 \le \delta\}. \tag{C.34}$$

For γ continuously differentiable on a compact region, $B_\delta(\gamma) \subset B_\delta^{(N)}(\gamma) \subset B_{2\delta}(\gamma)$ once h is small enough that $\max_k \|\gamma(t) - \gamma(t_k)\| \le \delta$ for $t \in [t_k, t_{k+1}]$.

Taylor-expanding p_N around γ and using the volume $V_d\delta^d$ of a d-ball:

$$\mathbb{P}_{x_0,T}[B_\delta^{(N)}(\gamma)] = C_N(\delta) \exp\Big(-\sum_k \tfrac{\|\Delta\gamma_k - F(\gamma(t_k))h\|_2^2}{2\varepsilon h}\Big)(1 + O_\gamma(\delta)), \tag{C.35}$$

where $C_N(\delta) = [V_d\delta^d(2\pi\varepsilon h)^{-d/2}]^{N-1}$ does not depend on γ. Taking the log ratio cancels $C_N(\delta)$:

$$\log \frac{\mathbb{P}_{x_0,T}[B_\delta^{(N)}(\gamma_0)]}{\mathbb{P}_{x_0,T}[B_\delta^{(N)}(\gamma_1)]} = -\tfrac{1}{2\varepsilon h} \sum_k \big(\|\Delta\gamma_{0,k} - F(\gamma_0(t_k))h\|_2^2 - \|\Delta\gamma_{1,k} - F(\gamma_1(t_k))h\|_2^2\big) + O(\delta). \tag{C.36}$$

As $N \to \infty$ (so $h \to 0$), the Riemann sum converges to

$$-\tfrac{1}{2\varepsilon}\Big(\int_0^T \|\dot\gamma_0 - F(\gamma_0)\|_2^2 \, \mathrm{d}t - \int_0^T \|\dot\gamma_1 - F(\gamma_1)\|_2^2 \, \mathrm{d}t\Big) + O(\delta). \tag{C.37}$$

The squeeze $B_\delta \subset B_\delta^{(N)} \subset B_{2\delta}$ and the limit $\delta \to 0$ yield the desired statement. □

Two corollaries: (i) local/global minima of the OM action correspond to local/global maxima of likelihood (in infinitesimal tubes around the path); (ii) for fixed endpoints and gradient drift $F = -\nabla U$, expanding the squared norm via integration by parts gives the split form

$$\int_0^T \|\dot\gamma - F(\gamma)\|_2^2 \, \mathrm{d}t = \int_0^1 \|\dot\gamma\|^2 \, \mathrm{d}t + \int_0^1 \|F(\gamma)\|^2 \, \mathrm{d}t + 2(U(\gamma(1)) - U(\gamma(0))), \tag{C.38}$$

which after dropping the constant endpoint term gives the form used in the vision experiments.

C.3 Geodesics and energy-minimizing paths

Theorem C.4 (Equivalence of geodesics and energy-minimizers)**.** *Let $\mathcal{M} \subset \mathbb{R}^d$ be an embedded submanifold and let $\|\cdot\|_\star$ be the norm induced by a positive-definite inner product $\langle \cdot, \cdot \rangle_g$ on $\mathbb{R}^d$. Fix $x_0, x_T \in \mathcal{M}$, and for a smooth curve $\gamma : [0,1] \to \mathcal{M}$ define*

$$L(\gamma) \coloneqq \int_0^1 \|\dot\gamma(t)\|_\star \, \mathrm{d}t, \quad E(\gamma) \coloneqq \int_0^1 \|\dot\gamma(t)\|_\star^2 \, \mathrm{d}t. \tag{C.39}$$

Let $\Gamma_{\mathcal{M}}(x_0, x_T) = \{\gamma \in C^\infty([0,1], \mathcal{M}) \mid \gamma(0) = x_0, \gamma(1) = x_T\}$. Then:

(i) $L(\gamma)^2 \le E(\gamma)$ for all γ, with equality iff $\|\dot\gamma(t)\|_\star$ is constant.

(ii) If γ_e minimizes E over $\Gamma_{\mathcal{M}}(x_0, x_T)$, then γ_e has constant speed and also minimizes L.

(iii) Any length-minimizer γ_g with constant speed also minimizes E.

Proof. Part (i): Cauchy-Schwarz on $[0,1]$:

$$L(\gamma) = \int_0^1 1 \cdot \|\dot\gamma(t)\|_\star \, \mathrm{d}t \le \sqrt{\int_0^1 1^2 \, \mathrm{d}t} \sqrt{\int_0^1 \|\dot\gamma(t)\|_\star^2 \, \mathrm{d}t} = \sqrt{E(\gamma)}. \tag{C.40}$$

Equality iff $\|\dot\gamma(t)\|_\star$ is constant.

Parts (ii) and (iii): for paths of constant velocity $V = \|\dot\gamma\|_\star$:

$$L(\gamma) = \int_0^1 V \, \mathrm{d}t = V = \sqrt{V^2} = \sqrt{\int_0^1 V^2 \, \mathrm{d}t} = \sqrt{E(\gamma)}. \tag{C.41}$$

Such paths achieve Cauchy-Schwarz equality, so minimizers of one are minimizers of the other. □

C.4 Gaussian-blur norm

We work in the functional setting: L^2 functions over $\mathbb{R}^2$, with $\|\cdot\|_2, \|\cdot\|_1$ denoting functional L^2, L^1 norms. For clarity we consider codomain dimension 1 (greyscale); the proofs extend straightforwardly to multi-dimensional outputs.

Theorem C.5 (Properties of the Gaussian-blur norm)**.** *Let*

$$\|\dot x_t\|_\sigma \coloneqq \|\mathcal{G}_\sigma \circledast \dot x_t\|_{L^2}, \quad \mathcal{G}_\sigma(x) = e^{-\|x\|_2^2/(2\sigma^2)}, \tag{C.42}$$

defined on $L^2_c(\mathbb{R}^2) = \{f \in L^2(\mathbb{R}^2) \mid \operatorname{supp} f \subset [0,1]^2, \operatorname{range} f \subset [0,1]\}$. Then:

(i) The mapping $T_\sigma : f \mapsto \mathcal{G}_\sigma \circledast f$ is a bounded, injective linear operator on $L^2_c(\mathbb{R}^2)$ (its inverse is unbounded, but injectivity is all that is needed); hence $\|\cdot\|_\sigma$ is a norm.

(ii) T_σ *has operator norm* $\|T_\sigma\|_{L^2\to L^2} \le 2\pi\sigma^2$.

(iii) For any $f \in L^2_c(\mathbb{R}^2)$ *and* $v \in \mathbb{R}^2$, *the curve* $t \mapsto f(\cdot - tv)$ *is rectifiable in the* $\|\cdot\|_\sigma$ *metric.*

Proof. Part (i). The Fourier transform of the kernel is $\widehat{\mathcal{G}}_\sigma(p) = 2\pi\sigma^2 e^{-\sigma^2\|p\|_2^2/2}$. This multiplier is bounded (so T_σ is bounded, with operator norm $2\pi\sigma^2$) and strictly positive (so T_σ is injective, though T_σ^{-1}, with multiplier blowing up as $\|p\| \to \infty$, is unbounded). Linearity from convolution, definiteness from injectivity; other norm axioms inherited from $\|\cdot\|_2$.

Part (ii). Young's convolution inequality with $p = r = 2$, $q = 1$: $\|T_\sigma f\|_2 \le \|\mathcal{G}_\sigma\|_1 \|f\|_2$. Since $\|\mathcal{G}_\sigma\|_1 = 2\pi\sigma^2$ (the L^1 normalization of a 2D Gaussian), $\|T_\sigma\|_{L^2\to L^2} \le 2\pi\sigma^2$.

Part (iii). Let $F(x,t) = f(x - tv)$ and $\gamma(t) = F(\cdot, t)$. The map $t \mapsto (\mathcal{G}_\sigma \circledast f)(\cdot - tv)$ is differentiable due to convolution with the smooth kernel:

$$\tfrac{\mathrm{d}}{\mathrm{d}t}(\mathcal{G}_\sigma \circledast f)(x - tv) = \big(-(v\cdot\nabla\mathcal{G}_\sigma) \circledast f(\cdot - tv)\big)(x). \tag{C.43}$$

Since $f \in L^2_c(\mathbb{R}^2)$, f has compact support and the differentials are L^2-bounded. By Young's inequality:

$$\begin{aligned} L(\gamma) = \int_0^1 \|\dot\gamma\|_\sigma \,\mathrm{d}t &= \int_0^1 \big\|\tfrac{\mathrm{d}}{\mathrm{d}t}(\mathcal{G}_\sigma \circledast f(\cdot - tv))\big\|_2 \,\mathrm{d}t \\ &\le \int_0^1 \|v\cdot\nabla\mathcal{G}_\sigma\|_2 \,\|f(\cdot - tv)\|_1 \,\mathrm{d}t < \infty, \end{aligned} \tag{C.44}$$

since all Gaussian moments are finite and f has bounded support and range. □

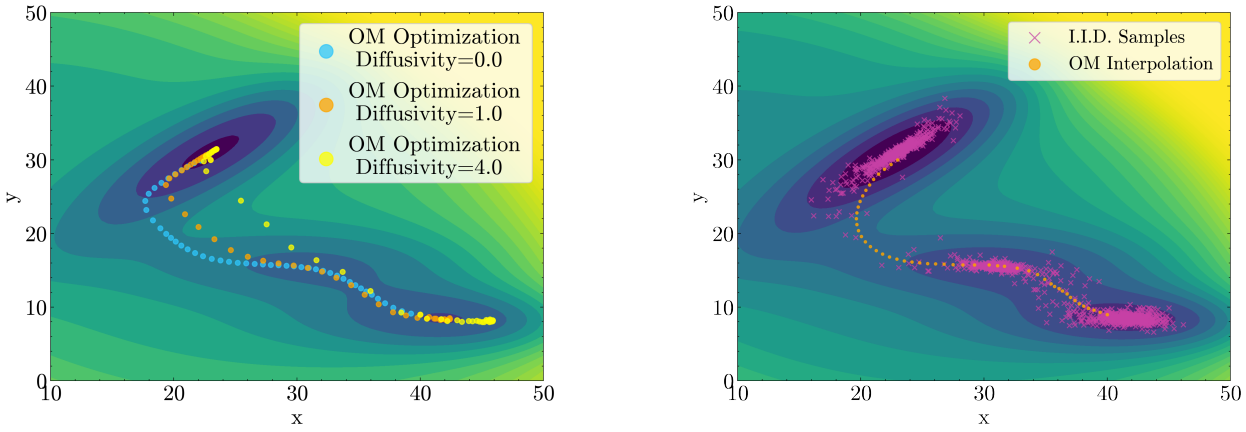
C.5 Implementation details: Müller-Brown experiments

All experiments on a single NVIDIA RTX A6000 GPU.

Potential. The Müller-Brown potential used:

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

Figure C.1a shows OM optimization using the analytical potential as the force field; the diffusion-model results in Section 6.3 align with these.



(a) Analytical potential.

(b) Flow matching model.

Figure C.1: OM optimization on Müller-Brown. **(a)** With the analytical potential as the force field. **(b)** With a flow matching model trained on Müller-Brown data using the extracted score (6.9).

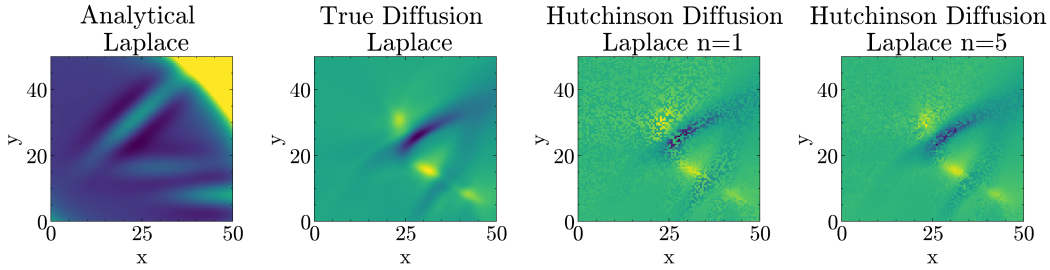


Figure C.2: Hutchinson Trace Estimator accurately estimates the Laplacian of the Müller-Brown potential. The diffusion model learns an estimate that captures the energy wells; the Hutchinson Trace Estimator efficiently approximates the Laplacian, becoming less noisy with more random samples.

Data generation and training. We sample initial conditions from the transition path resulting from OM optimization on the true potential, run unbiased Langevin simulations (1,000 parallel sims, 1,000 steps each, total 1M datapoints, 800K training/200K validation), and train a denoising diffusion model with a 3-layer MLP, GELU activation, hidden dim 256. Trained for 10 epochs, batch size 4096, learning rate 10^{-3} with Adam (Kingma and Ba, 2017).

Laplacian estimation. We estimate the Laplacian of the potential via the Hutchinson trace estimator. As shown in Figure C.2, one random vector ($N = 1$) is enough to capture the local minima and energy barrier; more random vectors give less noisy estimates at increased compute.

Table C.1: Hyperparameters 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

OM hyperparameters. See Table C.1.

Committor function and rate estimation

The committor $q(x)$ is the solution to the steady-state Backward Kolmogorov Equation. For high-dimensional systems, direct solution is infeasible; we train a neural network approximation by minimizing

$$L(\theta) = \tfrac{1}{2}\langle \|\nabla_x q_\theta(x)\|^2 \rangle_{\Omega\setminus\mathcal{A}\cup\mathcal{B}} + \lambda_{\mathcal{A}} \tfrac{1}{2}\langle q_\theta(x)^2 \rangle_{\mathcal{A}} + \lambda_{\mathcal{B}} \tfrac{1}{2}\langle (1 - q_\theta(x))^2 \rangle_{\mathcal{B}}, \tag{C.45}$$

with boundary-penalty weights $\lambda_{\mathcal{A}}, \lambda_{\mathcal{B}}$. The ensemble averages are estimated by importance-sampled Monte Carlo over a training set $\mathcal{D}_{\text{train}}$.

The primary challenge is sampling: the optimization is dominated by transition-region samples, which are rare. Following Hasyim et al. (2022), we use OM-optimized paths as enhanced-sampling seeds: initiate unbiased Langevin simulations from points along OM paths (100 simulations, 50 ps each, $k_B T = 1$), reweight to the Boltzmann distribution via $w_i = e^{-\beta U(x_i)}/p_{\text{OM}}(x_i)/Z$, and train a 5-layer MLP committor (hidden dim 64, sigmoid activation, 2000 steps, batch size 4096, Adam learning rate 10^{-4}, boundary weights $\lambda_{\mathcal{A}} = \lambda_{\mathcal{B}} = 20$).

C.6 Implementation details: alanine dipeptide

Baselines. For MCMC, we report metrics from Du et al. (2024) for the fixed-length two-way exact method. We implement metadynamics ourselves with Ramachandran angles as CVs, hill height 0.2 kcal/mol, width 10Å, temperature 300 K.

Diffusion model training. Training set from MD simulations of alanine dipeptide at 1000 K (high temperature to include transition regions). Graph Transformer architecture as in Arts et al. (2023). 50K training steps, Adam optimizer, learning rate 0.0004 with cosine annealing to 10^{-5}, EMA $\alpha = 0.995$, 1,000 inference steps.

OM hyperparameters. Path length $L = 300$, 300 initial-guess noise/denoise steps out of 1,000, 50,000 OM optimization steps, $\tau_{\text{opt}} = 3$, $\gamma = 0.7$, $\Delta t = 10^{-4}$. After OM optimization: 1,400-step energy minimization with Amber ff19SB, then 21,000 additional steps via BFGS.

Free energy via umbrella sampling. The truncated OM action yields minimum energy paths; to obtain free-energy barriers we use umbrella sampling: short (10 ps) simulations with a harmonic spring potential centered at OM path points (spring constant 5 kcal/mol/rad^2), fit a CV via PCA on the resulting samples, and apply WHAM (Kumar et al., 1992) to obtain the potential of mean force.

C.7 Implementation details: fast-folding proteins

All experiments on a single NVIDIA RTX A6000 GPU.

Markov state model construction. k-means clustering of reference MD into 20 clusters using top-2 TIC dimensions fit on pairwise distances and dihedral angles. MSM lag time 200 ps. To evaluate generated paths, we discretize them under the reference MSM (subsampled to length 20) and compute three metrics: JSD between state distributions, transition negative log-likelihood under reference MSM, fraction of paths with nonzero probability. For length-L paths from the reference MSM conditioned on endpoints s_1, s_L, sample s_t iteratively as

$$s_{t+1} \sim \frac{T^{(L-t-1)}_{:,s_L} T_{s_t,:}}{T^{(L-t)}_{s_t,s_L}}. \tag{C.46}$$

Empirical committor. Divide the TIC space into 100^2 bins; the committor at each bin is the frequency at which reference MD trajectories initiated in that bin reach the target state before the start state. The empirical transition state ensemble is $\{x : 0.45 \leq q(x) \leq 0.55\}$ (Figure C.4).

Model architecture and training. Denoising diffusion and flow matching models parameterized by a Graph Transformer (as in Arts et al. (2023)). Diffusion: pretrained models from Arts et al. (2023). Flow matching: trained from scratch, 3 attention layers, 1M iterations, Adam learning rate 0.0004 with cosine annealing to 10^{-5}, EMA $\alpha = 0.995$. Diffusion inference: 1,000 integration steps; flow matching: 10 steps. Protein-specific architecture in Table C.2.

Table C.2: Architecture and training hyperparameters for diffusion and flow-matching models on fast-folding proteins.

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

Table C.3: Hyperparameters for OM action optimization on fast-folding proteins with diffusion models.

Hyperparameter	**Chignolin**	**Trp-cage**	**BBA**	**Villin**	**Protein G**
Number of paths	8	8	32	4	4
Action type	Trunc	Trunc	Full	Trunc	Trunc
τ_{initial}	250	250	250	250	250
τ_{opt}	20	15	20	10	10
Optimization steps	5000	5000	5000	5000	5000
Optimizer	Adam	Adam	SGD	SGD	SGD
Learning rate	0.2	0.2	10^{-5}	10^{-5}	10^{-5}
Path length L	200	200	200	200	200
Δt	0.001	0.001	0.001	0.005	0.002
γ	1	1	1	1	1
D	0	0	1	0	0

OM optimization hyperparameters. See Tables C.3 (diffusion) and C.4 (flow matching).

Comparison to reference MD. We directly compare our generated paths against unbiased MD transition trajectories from the D.E. Shaw reference simulations (Lindorff-Larsen et al., 2011). Figure C.5 shows our paths sample the correct phase-space regions, with the same transition regions and similar structural transitions, though OM paths are smoother (lacking the fluctuations of unbiased MD).

Visualizations. Figures C.6, C.7, C.8 show additional transition path visualizations for both diffusion and flow matching models.

Varying time horizons. Paths with varying time horizons T_p (with Δt increased to maintain L) explore different parts of the transition state ensemble (Figure C.9): longer horizons spend more time near the folded-state energy well, shorter horizons take more direct paths between endpoints.

Robustness to sparse data. We retrain diffusion models on datasets with 99% of configurations between committor 0.1 and 0.9 removed. For Chignolin, Trp-Cage, and BBA,

Table C.4: Hyperparameters for OM action optimization on fast-folding proteins with flow-matching models.

Hyperparameter	**Chignolin**	**Trp-cage**	**BBA**	**Villin**	**Protein G**
Number of paths	8	8	32	4	4
Action type	Trunc	Trunc	Full	Trunc	Trunc
τ_{initial}	7	7	7	7	7
τ_{opt}	0.5	0.5	0.5	0.5	0.5
Optimization steps	5000	5000	5000	5000	5000
Optimizer	Adam	Adam	SGD	SGD	SGD
Learning rate	0.2	0.2	10^{-4}	10^{-4}	10^{-4}
Path length L	200	200	200	200	200
Δt	0.0008	0.0005	0.001	0.001	0.001
γ	1	1	1	1	1
D	0	0	1	0	0

the subsampling removes 1.4%, 68%, and 85% of the datapoints respectively. As shown in Figure C.10, the produced transition paths are still close to those from the full dataset.

C.8 Implementation details: tetrapeptides

Heavy-atom representation. Following Jing, Stärk, et al. (2024), we use heavy-atom representations of tetrapeptides (no hydrogens), with the terminal C-terminus oxygen also excluded. Tetrapeptides have at most 56 atoms.

Training. Flow matching model with same architecture/training as for fast-folding proteins, with learnable atom-type embeddings and padding tokens (for variable atom counts). Atom embeddings concatenated to residue ordering and flow timestep to form node features. Trained on 3,109 tetrapeptides from Jing, Stärk, et al. (2024), 10,000 evenly spaced configurations per tetrapeptide. 100,000 training steps, batch size 2048.

OM optimization on held-out proteins. 100 held-out tetrapeptide sequences not seen during training (same splits as Jing, Stärk, et al. (2024)). Endpoints chosen as the pair of MSM states with lowest nonzero probability flux. Hyperparameters in Table C.5.

Energy minimization. After OM optimization, we add OXT and hydrogen atoms back using PDBFixer (`https://github.com/openmm/pdbfixer`). Up to 200 steps of energy minimization in OpenMM in vacuum with amber14 force field using L-BFGS, restricted to RMSD ≤ 1 Å between initial and final structures.

Table C.5: Hyperparameters for OM action optimization on tetrapeptides with flow-matching models.

Hyperparameter	**Value**
Number of paths	16
Action type	Truncated
Initial guess time	7
Optimization time	0.5
Optimization steps	250
Optimizer	Adam
Learning rate	0.2
Path length L	100
Δt	0.0001
γ	1
D	0

Evaluation. Same MSM-based pipeline as for fast-folding proteins, with TIC dimensions fit on backbone and sidechain torsion angles. Metrics averaged over 100 held-out test proteins.

C.9 All-atom proteins with a classical force field

We demonstrate that OM optimization is broadly useful for TPS even beyond the generative-modeling setting: finding all-atom paths between unfolded and folded states of 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 with the TIP3P implicit water model.

Physical parameters of the OM action match commonly used MD values (Table C.6). Without a generative model for initial guess, we use a hierarchical-unwrapping warm-up procedure (Algorithm 6): start with small number of replicas in each basin, optimize with a large path term to interpolate, then duplicate replicas and optimize again until reaching desired path length. Since the classical FF is dominated by quadratic terms whose Laplacian is uninformative, we use the truncated OM action. Optimization takes on the order of hours on one NVIDIA RTX A6000 GPU. We obtain physical transition paths of length 2.6 ps (Figure C.12), shorter than previously reported transition path lengths for Chignolin (Sobieraj and Setny, 2022; Lindorff-Larsen et al., 2011) since our trajectories proceed without the fluctuations of unbiased simulations.

Algorithm 6 Initial Guess Path Generation with Iterative Unwrapping.

Function InitialGuess($x^{(0)}, x^{(L)}, L_1, N$)
Initialize trajectory by copying boundary points $L_1/2$ times on both ends
$x \leftarrow \{x^{(0)}, \ldots, x^{(0)}, x^{(L)}, \ldots, x^{(L)}\}$
for $m = 1$ to N **do**
    Duplicate every point: $x \leftarrow \{x^{(0)}, x^{(0)}, x^{(1)}, x^{(1)}, \ldots, x^{(L)}, x^{(L)}\}$
    Optimize truncated OM action: $x \leftarrow \arg\min_x S_\theta^{\text{trunc}}(x)$
end for
return $x = \{x^{(0)}, x^{(1)}, \ldots, x^{(2^N L_1 - 1)}, x^{(L)}\}$

Table C.6: Hyperparameters for OM action optimization on all-atom proteins with a classical force field.

Hyperparameter	**Chig. warmup**	**Chignolin**	**TRP warmup**	**TRP Cage**
Points per path	40-2600	2600	40-2600	2600
Action type	Trunc	Trunc	Trunc	Trunc
Optimizer	Adam	Adam	Adam	Adam
Learning rate	10^{-4}	10^{-5}	10^{-4}	10^{-5}
Δt	1 fs	1 fs	1 fs	1 fs
γ	10 ps^{-1}	10 ps^{-1}	10 ps^{-1}	10 ps^{-1}

C.10 Hutchinson trace estimator and τ_{opt} selection

Hutchinson trace estimator. The third term in (6.13) involves $\nabla \cdot F_\theta(x^{(i)}, \tau_{\text{opt}})$, which naïvely requires computing all diagonal entries of the Jacobian. We approximate the trace via Hutchinson's estimator (Hutchinson, 1989): $\text{tr}(\mathcal{H}) \approx \frac{1}{N}\sum_{i=1}^{N} v_i^\top \mathcal{H} v_i$, where $v_i \sim \mathcal{N}(0, I)$. Vector-Jacobian products allow computing $v^\top \mathcal{H}$ without materializing $\mathcal{H}$.

The estimator has variance $\text{Var}(\hat{\text{Tr}}) = \frac{1}{N}\text{Var}(v_i \cdot \mathcal{H} v_i)$, so error decays as $C/\sqrt{N}$. In practice, $N = 1$ works well, likely because neighboring path points have similar Laplacian values.

Selection of τ_{opt}. In principle $\tau_{\text{opt}} = 0$ maximizes correspondence to the true atomistic force field for Boltzmann-distributed data, but consistent with Arts et al. (2023), a small nonzero value works better. Figure C.13 shows cosine similarity between true forces and the diffusion model's score at various τ_{opt} for Müller-Brown and alanine dipeptide; the optimum is not at $\tau_{\text{opt}} = 0$.

C.11 Implementation details: image interpolation

MNIST. Trained diffusion model with U-Net (Ronneberger et al., 2015) denoising architecture. Path length 16, force scale $\lambda = 0.08$, Gaussian-blur level $\sigma = 2.5$. Adam optimizer, learning rate 5×10^{-2} for 1500 steps. Linear initialization in data space, with endpoint noise such that noise is highest at the middle of the path. Latent-time annealing from $\tau = 1000$ to $\tau = 100$ over first 900 steps, then fixed at $\tau = 100$. Figure C.14 shows random MNIST pairs and their interpolations.

CelebA-HQ. Pretrained data-space diffusion model from (Ho et al., 2020). Path length $L = 256$, Gaussian-blur level $\sigma = 1.0$, $\lambda = 64$ (higher than MNIST because we normalize forces so the mean squared norm matches the kinetic-term mean squared norm). Initial path obtained by noising endpoints to $t = 350$, linearly interpolating, decoding back to $t = 0$. Adam optimizer, learning rate 0.12 for 200 steps. Latent time $\tau = 60$ out of 1000.

Stable Diffusion. Same setup as CelebA-HQ on the pretrained model from (Rombach et al., 2022). $\lambda = 64$, $\sigma = 1.4$, path length $L = 16$. Noise/denoise by 400 out of 1000 steps for initial guess. Adam at learning rate 10^{-2} for 700 steps, latent time $\tau = 50$.

Baselines. All baselines noise/denoise by $T = 400$ steps (out of 1000), the smallest level that removes data-space artifacts (e.g., fading).

- **Linear interpolation.** Endpoints noised, linearly interpolated, denoised. Shared noise for endpoint noising and interpolant denoising.
- **SLERP.** Spherical interpolation in diffusion noised space, motivated by Gaussian concentration around shells of norm $\sqrt{D}$.
- **NoiseDiffusion.** $\alpha(s) = \cos(\frac{\pi}{2}s)$, $\beta(s) = \sin(\frac{\pi}{2}s)$, with $\mu(s), \nu(s)$ small linear deviations at endpoints.
- **NAO.** Hyperparameters mostly from the base implementation, with lower learning rate 5×10^{-4} and norm-penalty weight 0.1.

Latent-vs-image-space Gaussian blur. As described in Section 6.4, applying $\|\cdot\|_\sigma$ in latent space avoids backpropagating through the decoder. Figure C.15 shows that applying Gaussian filters of various noise scales followed by decoding yields visually similar results to applying the Gaussian filter directly in image space.

Integration with existing editors. Figure C.16 shows comparison of OM interpolation between an original image and edited image with a new prompt against interpolation baselines. The OM framework uniquely finds a trajectory that smoothly increases distance from the

origin while approaching the new prompt submanifold (higher CLIP), without sacrificing closeness to the original.

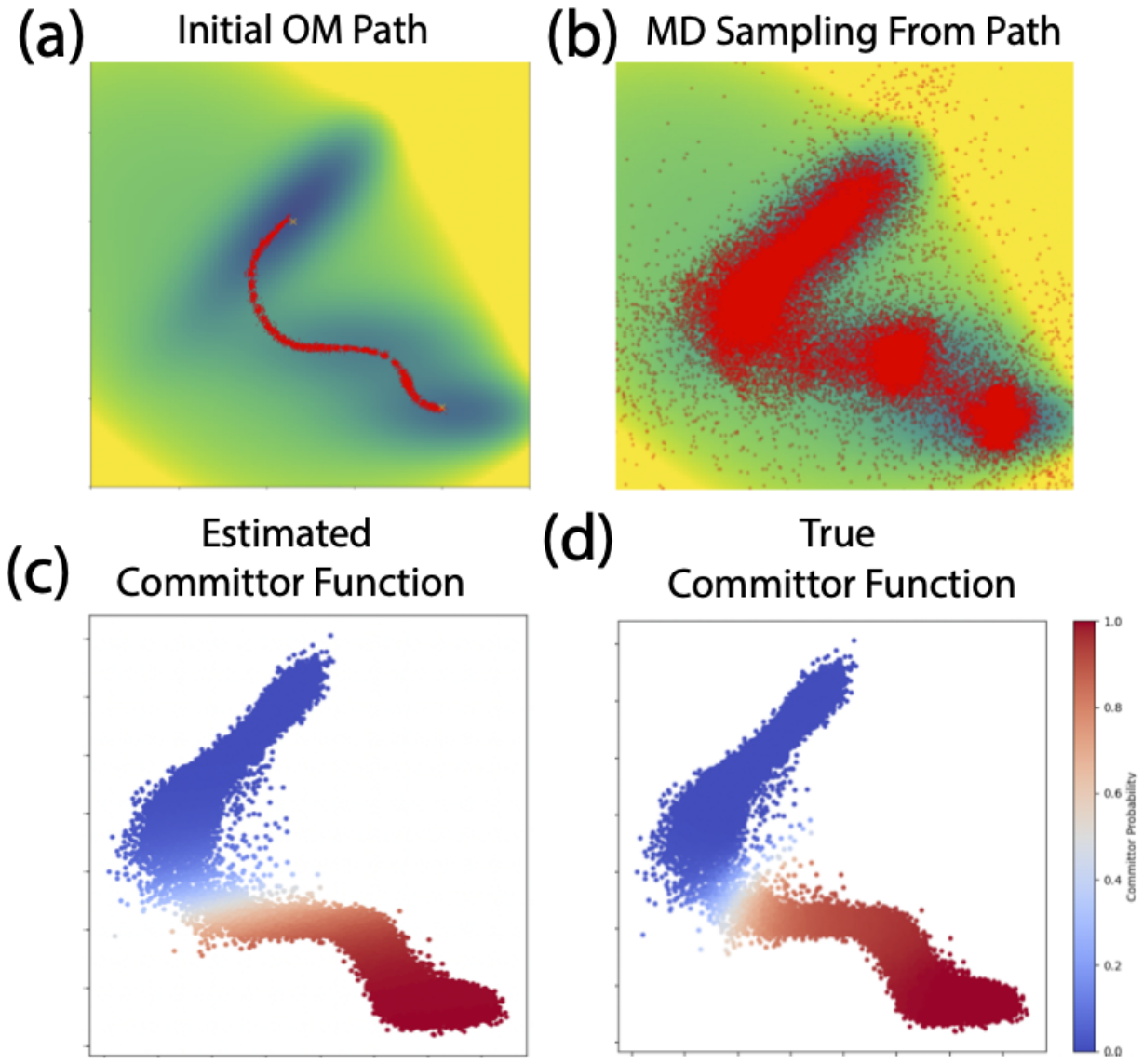


Figure C.3: Efficient committor and transition rate estimation on the 2D Müller-Brown potential using OM optimization. **(a)** Initial OM-optimized paths from the diffusion model. **(b)** Diversified samples via unbiased Langevin simulations initiated along OM paths. **(c)** Predicted committor function $q_\theta(x)$ via minimizing the BKE loss (C.45) over the samples. **(d)** True committor from numerical solution via FEniCS. The predicted reaction rate is 1.3×10^{-5} vs. true 5.38×10^{-5}.

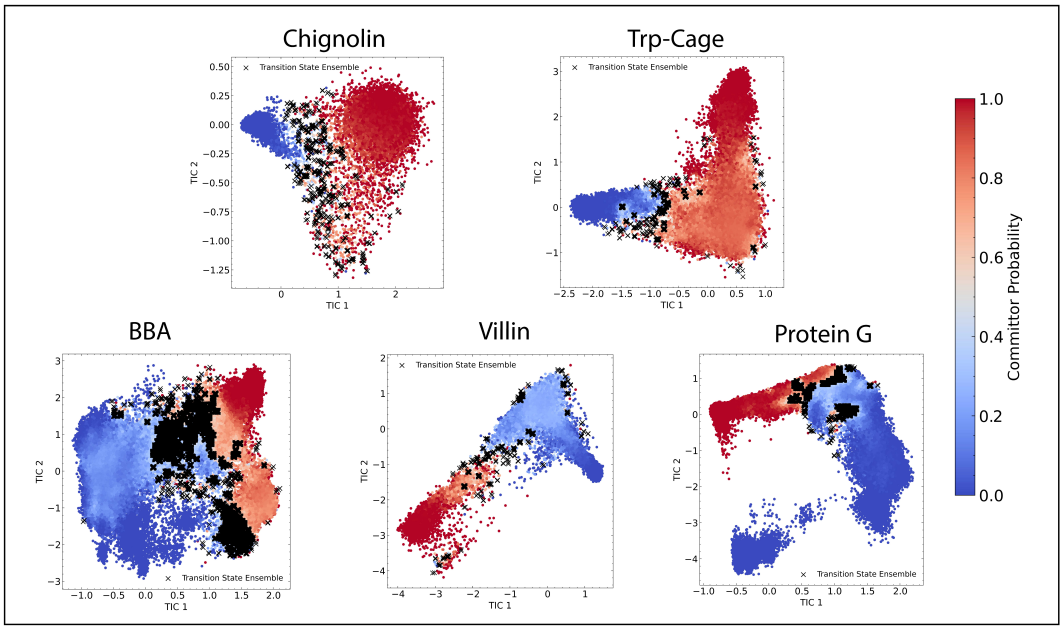


Figure C.4: Empirical committor landscapes for fast-folding proteins, computed by binning the conformational space into 100^2 bins and measuring the frequency at which reference MD trajectories initiated in each bin reach the target before the start. The empirical transition state ensemble (black) is the level set $\{x : 0.45 \leq q(x) \leq 0.55\}$.

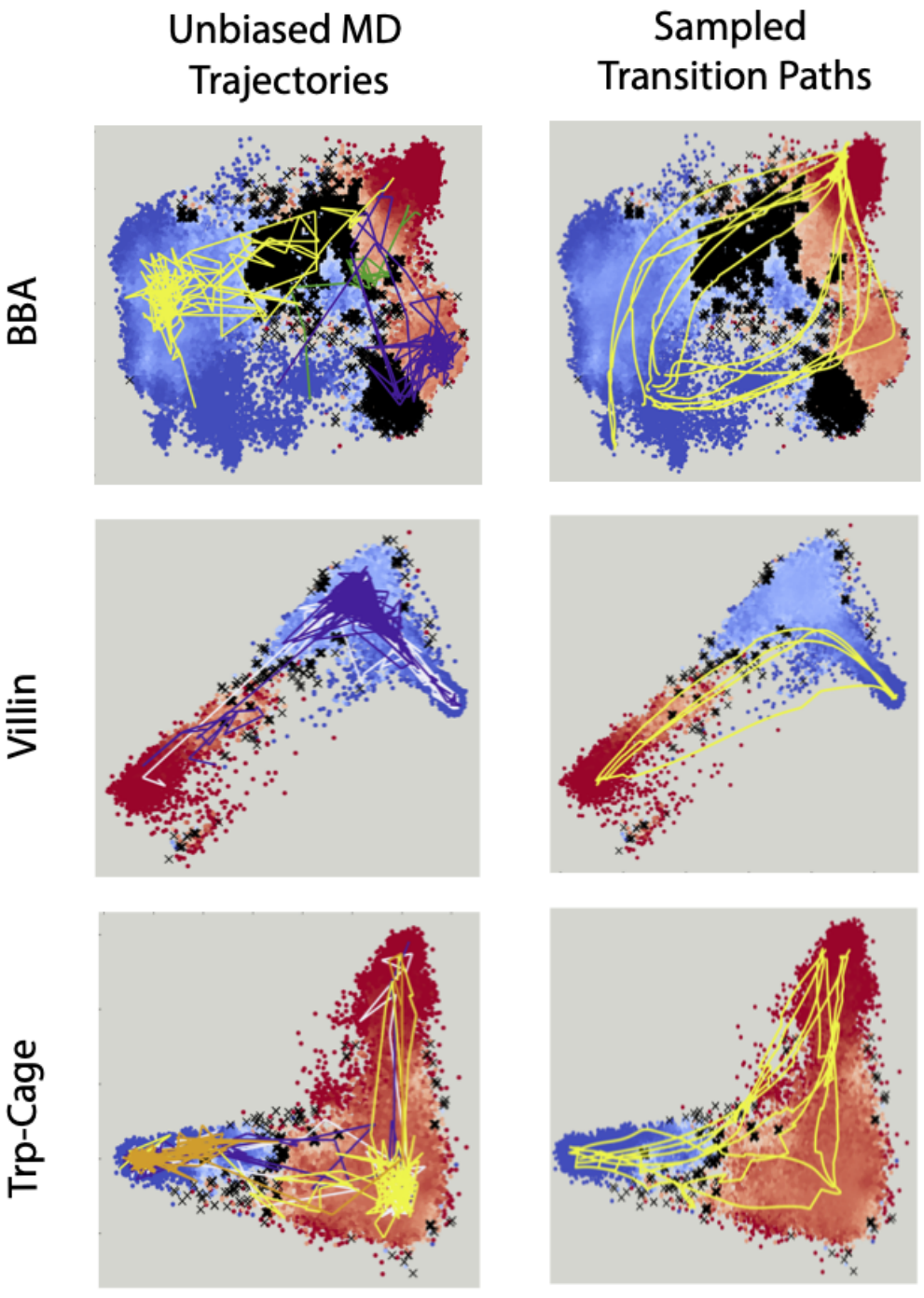


Figure C.5: Sample trajectories from reference unbiased fast-folding protein MD (left) and paths sampled via OM optimization (right), overlaid on TICA plots with committor values shaded from blue to red. OM paths traverse the same transition regions and exhibit similar structural transitions but are smoother (lacking minor fluctuations of unbiased MD).

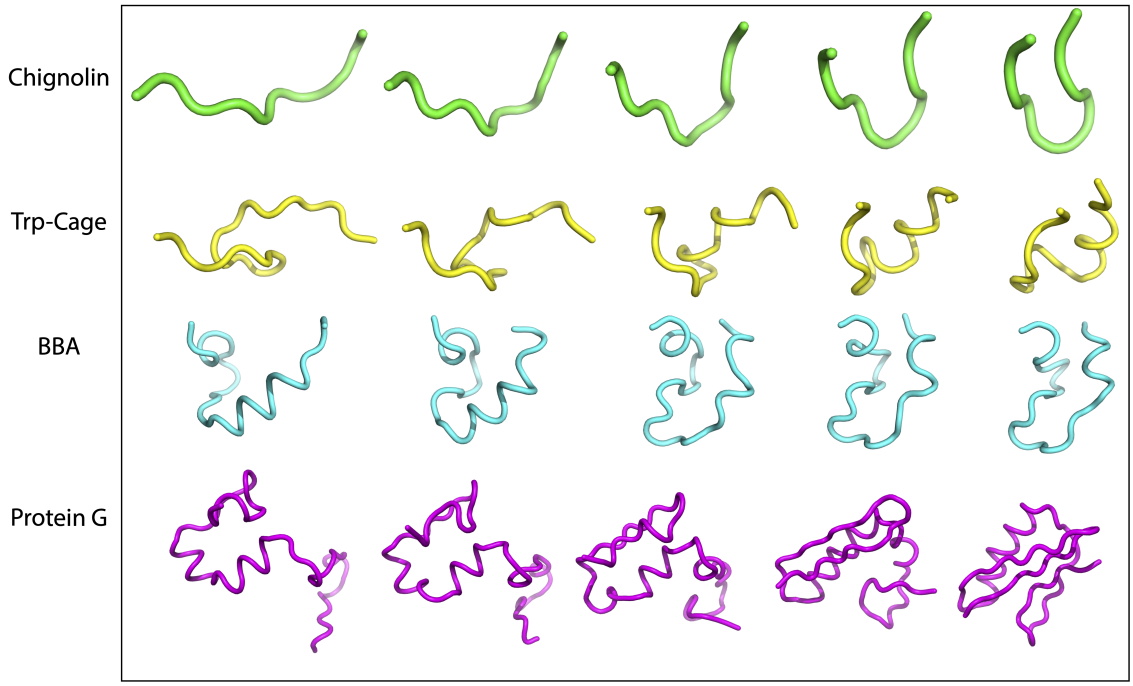


Figure C.6: Sampled transition paths from OM optimization with pretrained diffusion models for fast-folding proteins.

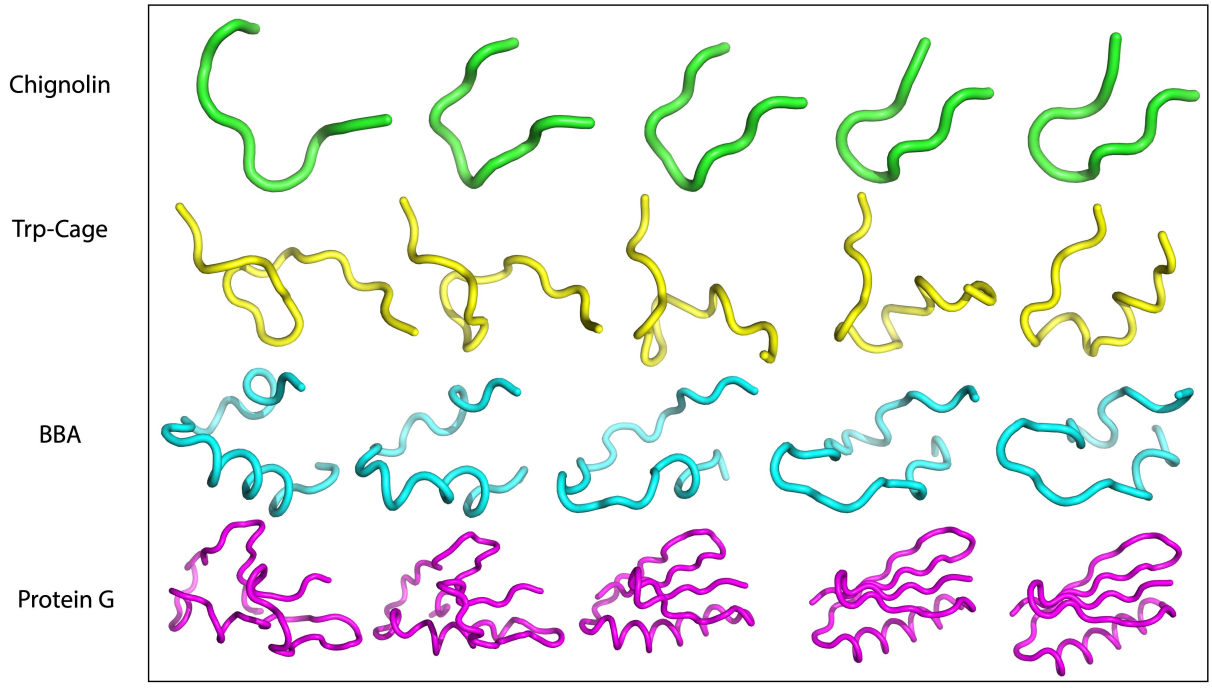


Figure C.7: Sampled transition paths from OM optimization with pretrained flow-matching models for fast-folding proteins.

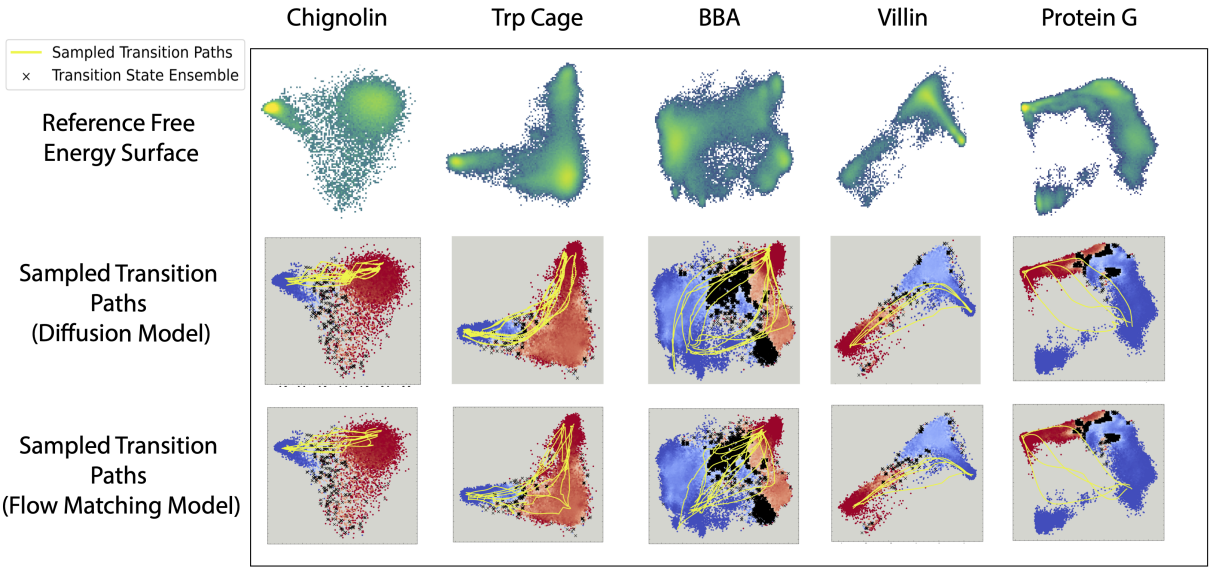


Figure C.8: Visualization of sampled fast-folding protein transition paths in TIC space.

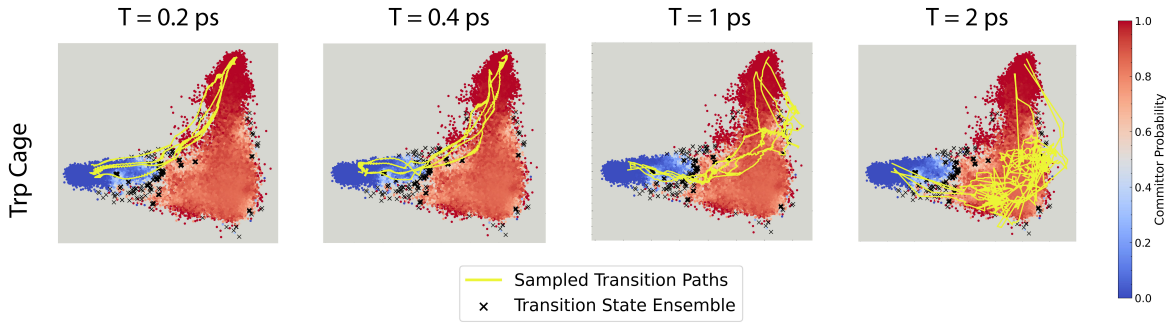


Figure C.9: OM-optimized transition paths for Trp-Cage with $\gamma = 1\,\mathrm{ps}^{-1}$, varying the transition time horizon $T_p \in \{0.2, 0.4, 1, 2\}$ ps and timestep $\Delta t \in \{0.001, 0.002, 0.005, 0.01\}$ ps, fixing $L = 200$ discretization points.

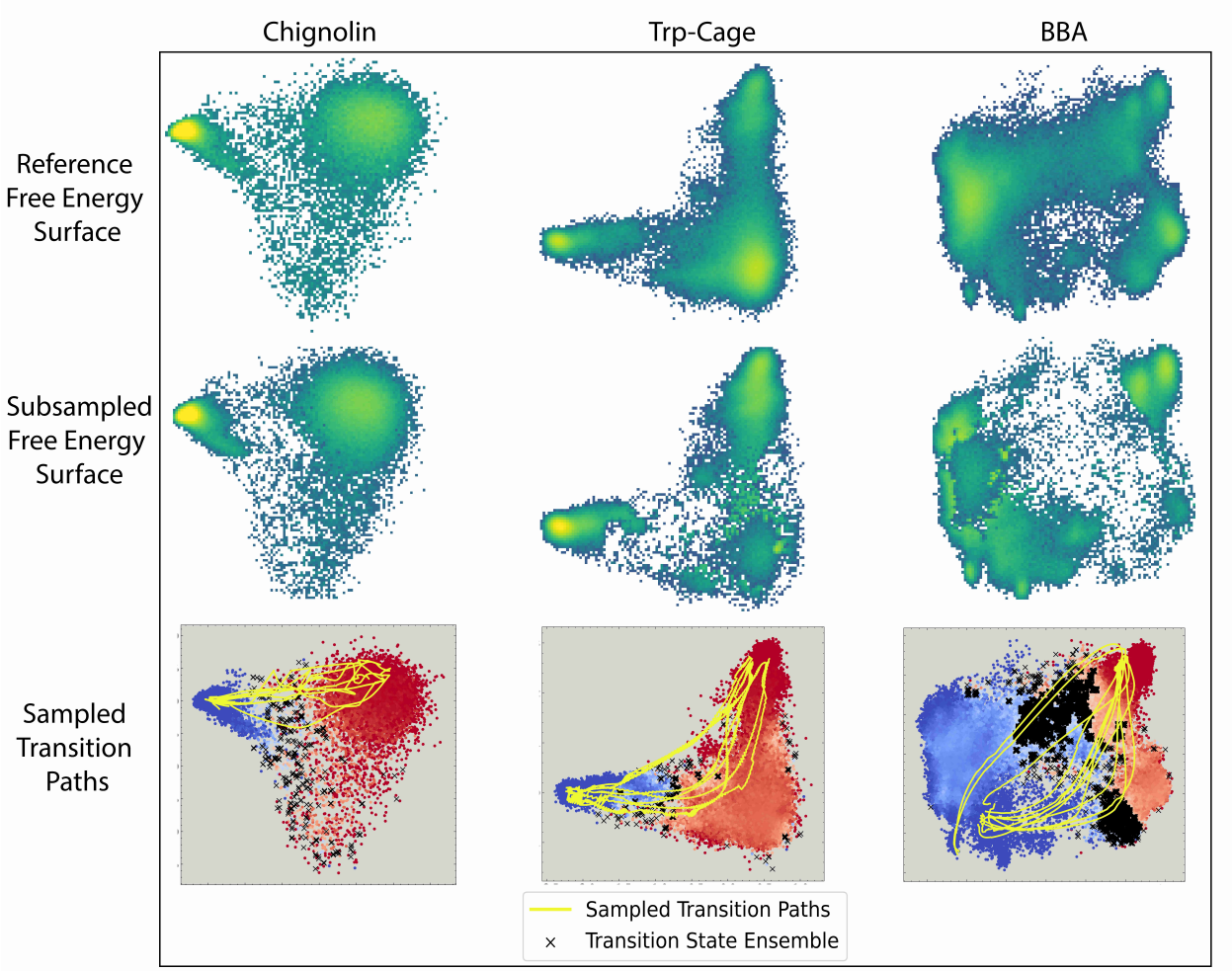


Figure C.10: Training datasets and sampled transition paths from removing intermediate committor values. **(Top row)** Original training datasets. **(Middle row)** Datasets after removing 99% of points with committor values between 0.1 and 0.9. **(Bottom row)** Transition paths from OM optimization with diffusion models trained on the subsampled datasets. The paths still pass through the expected transition state regions despite having seen them at much lower frequency during training.

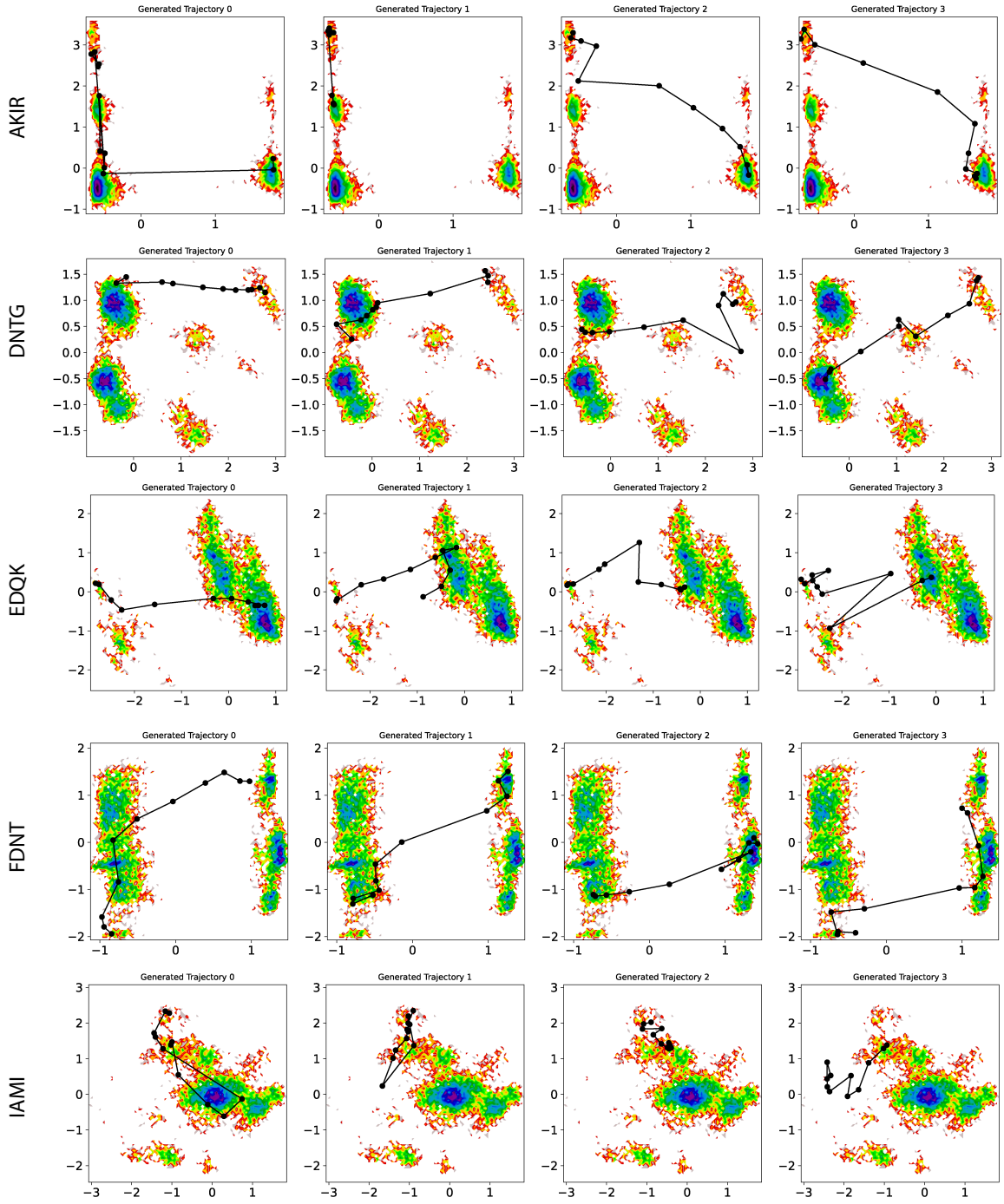


Figure C.11: Sampled transition paths from OM optimization on selected, held-out tetrapeptide sequences. The sampled paths are diverse and intuitively pass through high-density regions in the TIC free-energy landscape.

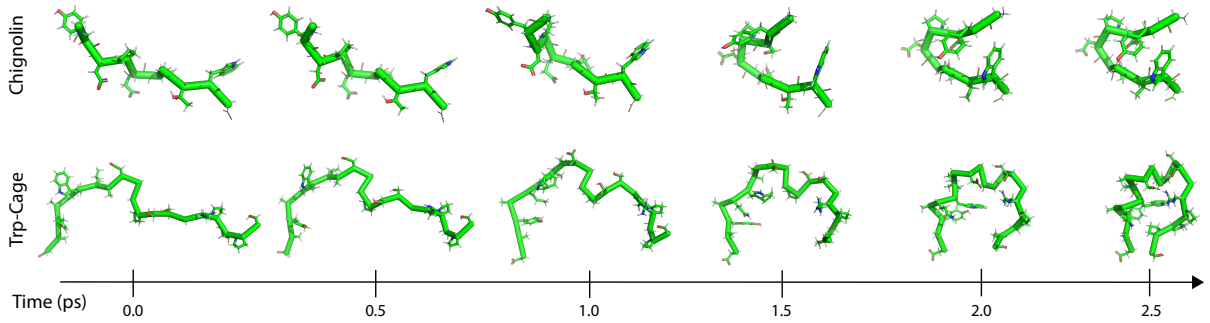
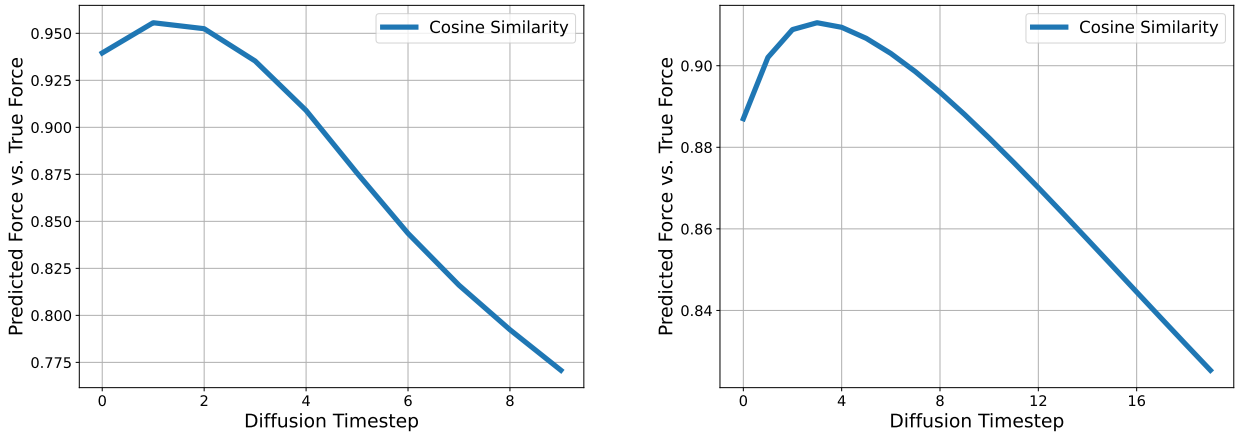


Figure C.12: Transition paths from OM optimization of all-atom Chignolin and Trp-Cage with a classical force field.



(a) Müller-Brown. (b) Alanine Dipeptide.

Figure C.13: Cosine similarity between true forces and the DDPM's learned time-dependent score model $\mathbf{s}_\theta(\cdot, \tau_{\text{opt}})$ over different diffusion latent times τ_{opt}. Notably, $\tau_{\text{opt}} = 0$ is not the optimal time conditioning with respect to true force recovery.

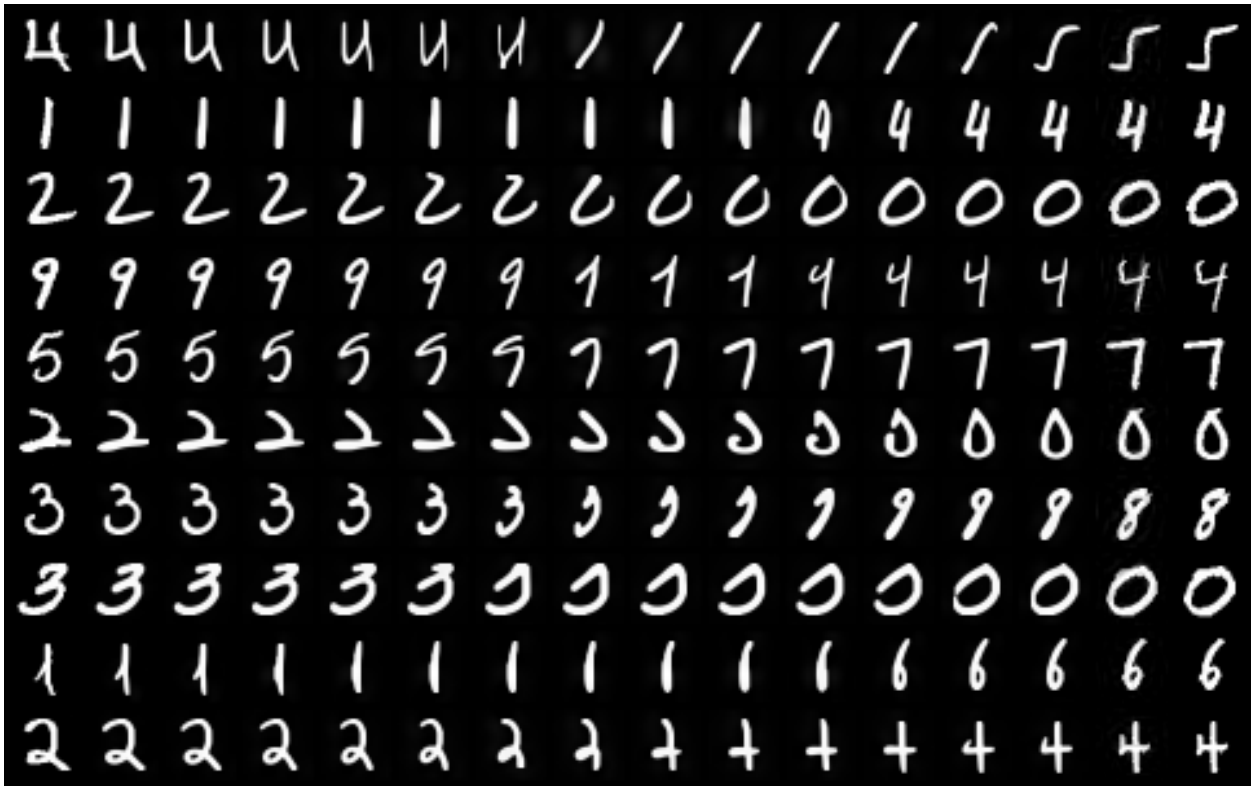


Figure C.14: Randomly selected pairs of MNIST images and their corresponding interpolations.

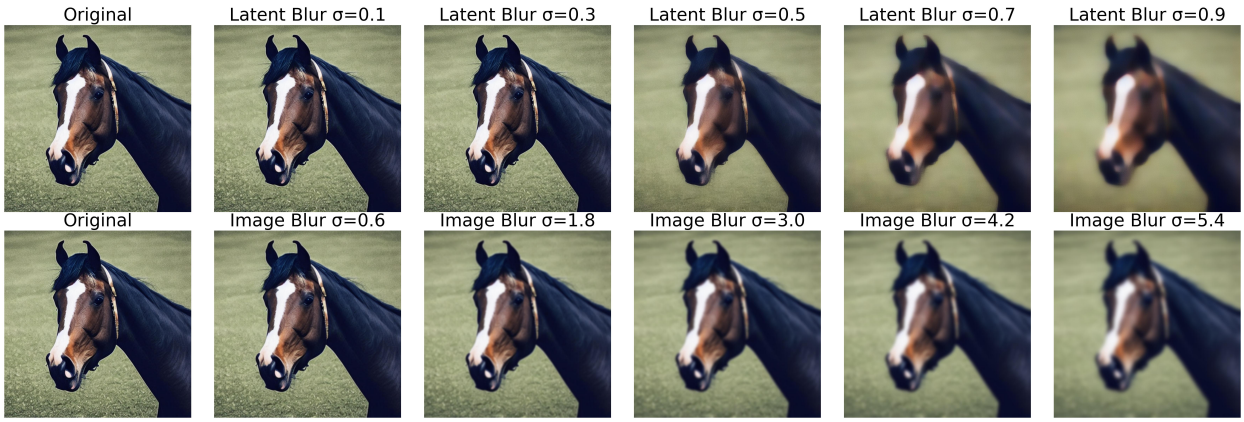


Figure C.15: Applying a Gaussian filter in latent space followed by decoding (top) yields similar results to directly applying Gaussian blurring in image space (bottom). The image-space noise scales are larger than the latent-space scales to account for the difference in dimensionality.

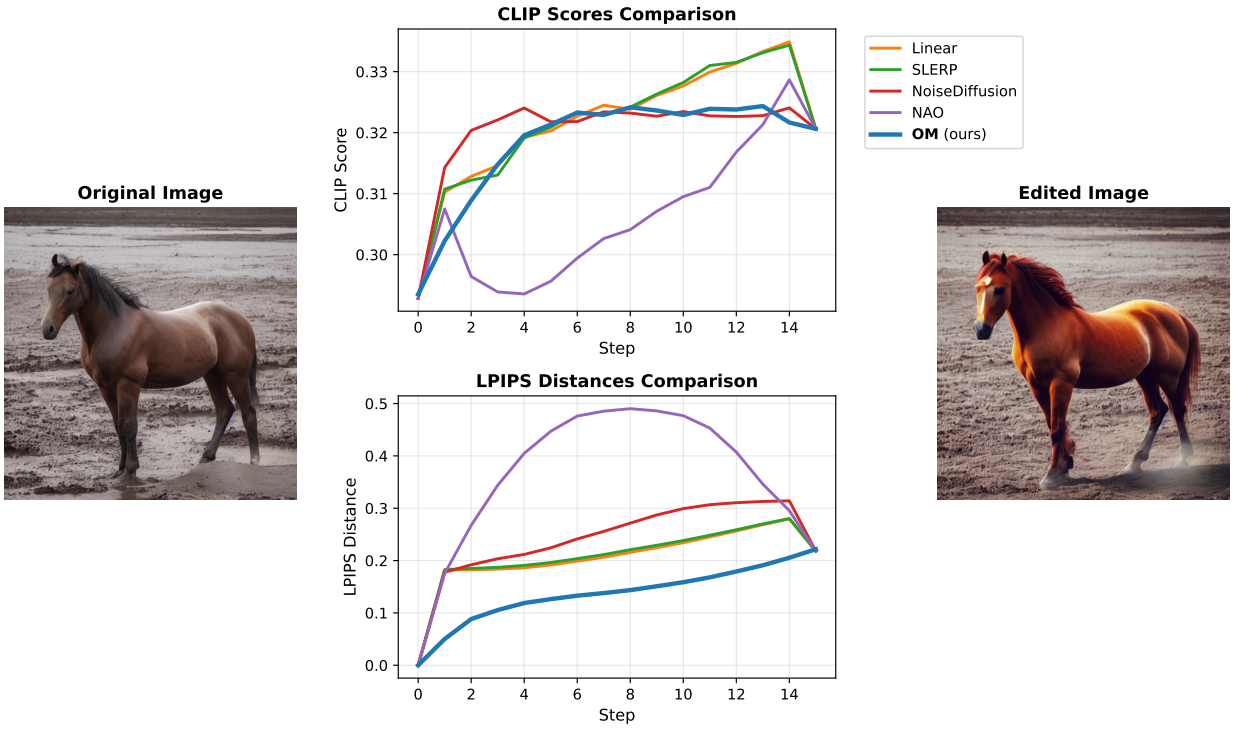


Figure C.16: OM interpolation between an original image and edited image with a new prompt, vs. interpolation baselines. The OM framework finds a trajectory that smoothly increases distance from the origin while going closer to the new prompt submanifold (higher CLIP). Other methods reach higher CLIP score regions sooner but move quickly away from the original image. OM interpolation gives a midpoint with preserved CLIP score and notably closer to the original.

Appendix D

Appendix to Chapter 7 (GRASP)

This appendix collects technical material for Chapter 7: full proofs of the linear-convergence theory in the convex setting, the Gaussian-smoothing regularity theory, a heuristic characterization of the stationary distribution under state-only noise, the nonexistence theorem for exact dynamics-enforcing losses without state gradients, the linear-convergence and Gaussian-tube theory for the stop-gradient regime, additional ablations, and detailed hyperparameter and benchmark setup.

D.1 Linear-convergence theory in the convex setting

We reproduce the convergence analysis for self-containedness; see e.g. Ascher et al. (1995) for other theoretical treatments. Consider linear dynamics

$$s_{t+1} = As_t + Ba_t, \tag{D.1}$$

with $s_t \in \mathbb{R}^n$, $a_t \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$. The rollout has the compact form

$$\mathcal{F}(s_0, \mathbf{a}) = A^T s_0 + \sum_{t=0}^{T-1} A^{T-1-t} B a_t. \tag{D.2}$$

Shooting: matrix representation

Let $\mathbf{a} = (a_0; \ldots; a_{T-1}) \in \mathbb{R}^{mT}$. The final state is linear in $\mathbf{a}$:

$$s_T = A^T s_{\text{init}} + \mathcal{C}_T \mathbf{a}, \tag{D.3}$$

where $\mathcal{C}_T = [A^{T-1}B, A^{T-2}B, \ldots, B] \in \mathbb{R}^{n\times mT}$ is the controllability matrix. The objective is $J_S(\mathbf{a}) = \left\|\mathcal{C}_T\mathbf{a} - (g - A^T s_{\text{init}})\right\|_2^2$. The Hessian is $H_S = 2\mathcal{C}_T^\top \mathcal{C}_T$.

Lifted: matrix representation

Eliminate fixed boundary variables s_0 and s_T and optimize over $\mathbf{z} = (s_1; \dots; s_{T-1}; a_0; \dots; a_{T-1})$. The dynamics residuals form a linear system $M\mathbf{z} - b$. The dynamics equations:

- For $t = 0$: $s_1 - Ba_0 = As_{\text{init}}$ (constant term moved to RHS).
- For $0 < t < T - 1$: $As_t + Ba_t - s_{t+1} = 0$.
- For $t = T - 1$: $-As_{T-1} - Ba_{T-1} = -g$ (s_T fixed to g).

The objective is $J_L(\mathbf{z}) = \|M\mathbf{z} - b\|_2^2$. The Hessian is $H_L = 2M^\top M$. The matrix M has a sparse, block-banded structure.

Smoothness analysis

The Lipschitz constants of the gradients are $L = \lambda_{\max}(H)$.

Proof of Theorem 7.2. The Lipschitz constant is the maximum eigenvalue of $H_S = 2\mathcal{C}_T^\top \mathcal{C}_T$, equal to $2\|\mathcal{C}_T\|_2^2$. By definition,

$$\|\mathcal{C}_T\|_2 = \sup_{\mathbf{a} \neq 0} \frac{\|\mathcal{C}_T \mathbf{a}\|_2}{\|\mathbf{a}\|_2}. \tag{D.4}$$

From the existence of a controllable non-contractive mode, we construct a unit-norm $\mathbf{a}_{\text{test}} = (w; \mathbf{0}; \dots; \mathbf{0})$ giving

$$\mathcal{C}_T \mathbf{a}_{\text{test}} = A^{T-1} B w, \tag{D.5}$$

and projecting onto the unstable eigenvector v:

$$\begin{aligned} \|\mathcal{C}_T \mathbf{a}_{\text{test}}\|_2 &\geq |\langle v,\, A^{T-1} B w \rangle| \\ &= |\langle (A^\top)^{T-1} v,\, B w \rangle| \\ &= |\langle \lambda^{T-1} v,\, B w \rangle| \\ &= |\lambda|^{T-1} |\langle v,\, B w \rangle| \\ &= \mu |\lambda|^{T-1}. \end{aligned} \tag{D.6}$$

Squaring gives the desired bound. □

Proof of Theorem 7.3. $H_L = 2M^\top M$, so $\lambda_{\max}(H_L) = 2\|M\|_2^2$. We upper-bound $\|M\|_2$ by a T-independent constant.

M is block-sparse: each block row (for timestep t) contains at most three non-zero blocks: identity (selecting s_{t+1}), dynamics (A multiplying s_t), and input (B multiplying a_t). The corresponding residual is $r_t = As_t + Ba_t - s_{t+1}$, affine in the free variables.

For optimization vector $\mathbf{z} = (s_1; \dots; s_{T-1}; a_0; \dots; a_{T-1})$, the stacked residuals $M\mathbf{z}$ satisfy, using $\|x+y+z\|^2 \le 3(\|x\|^2 + \|y\|^2 + \|z\|^2)$ and operator-norm bounds:

$$\begin{aligned} \|r_t\|_2^2 &= \|-s_{t+1} + As_t + Ba_t\|_2^2 \\ &\le 3\big(\|s_{t+1}\|_2^2 + \|A\|_2^2 \|s_t\|_2^2 + \|B\|_2^2 \|a_t\|_2^2\big). \end{aligned} \tag{D.7}$$

Summing:

$$\|M\mathbf{z}\|_2^2 = \sum_{t=0}^{T-1} \|r_t\|_2^2 \le 3 \sum_{t=0}^{T-1} \big(\|s_{t+1}\|_2^2 + \|A\|_2^2 \|s_t\|_2^2 + \|B\|_2^2 \|a_t\|_2^2\big). \tag{D.8}$$

Due to the banded structure, each free intermediate state $s_1, \dots, s_{T-1}$ appears in at most two terms (once as s_{t+1}, once as s_t). State contributions bound without accumulation in T:

$$\begin{aligned} \|M\mathbf{z}\|_2^2 &\le 3\Big((1 + \|A\|_2^2) \sum_{t=1}^{T-1} \|s_t\|_2^2 + \|B\|_2^2 \sum_{t=0}^{T-1} \|a_t\|_2^2\Big) \\ &\le 3(1 + \|A\|_2^2 + \|B\|_2^2)\ \|\mathbf{z}\|_2^2 . \end{aligned} \tag{D.9}$$

Thus $\|M\|_2^2 \le 3(1 + \|A\|_2^2 + \|B\|_2^2)$, and $L_L = 2\|M\|_2^2 \le 6(1 + \|A\|_2^2 + \|B\|_2^2)$, independent of T. □

D.2 Gaussian noise regularization

Proof of Theorem 7.4. Convolution commutes with derivatives:

$$\begin{aligned} \nabla_s L_\sigma &= \nabla_s(\phi_\sigma * L) \\ &= (\nabla \phi_\sigma) * L \\ &= \phi_\sigma * (\nabla_s L), \end{aligned} \tag{D.10}$$

where $\nabla \phi_\sigma(\varepsilon) = -(\varepsilon/\sigma^2)\phi_\sigma(\varepsilon)$.

For Part 1, Young's convolution inequality with $\|\phi_\sigma\|_{L^1} = 1$ gives $\|g * \phi_\sigma\|_{L^p} \le \|g\|_{L^p}$ for $g \in L^p$. Setting $g = \nabla_s L$ and using the commutation:

$$\|\nabla_s L_\sigma\|_{L^p} = \|\phi_\sigma * \nabla_s L\|_{L^p} \le \|\nabla_s L\|_{L^p} . \tag{D.11}$$

When $p = \infty$, $\|\nabla_s L\|_{L^\infty}$ is the Lipschitz constant.

For Part 2, again use the commutation and Young's inequality $\|h * k\|_{L^p} \le \|h\|_{L^1} \|k\|_{L^p}$:

$$\|\nabla_a L_\sigma\|_{L^p} = \|(\nabla \phi_\sigma) * L\|_{L^p} \le \|\nabla \phi_\sigma\|_{L^1} \|L\|_{L^p} . \tag{D.12}$$

Compute $\|\nabla \phi_\sigma\|_{L^1}$: since $\|\nabla \phi_\sigma(\varepsilon)\|_2 = \|\varepsilon\|_2 \phi_\sigma(\varepsilon)/\sigma^2$ and $X \sim \mathcal{N}(0, \sigma^2 I_d)$ has density ϕ_σ,

$$\|\nabla \phi_\sigma\|_{L^1} = \int \frac{\|\varepsilon\|_2}{\sigma^2} \phi_\sigma(\varepsilon)\, \mathrm{d}\varepsilon = \frac{1}{\sigma^2} \mathbb{E}\, \|X\|_2 = \frac{1}{\sigma} \mathbb{E}\, \|Z\|_2 = \frac{c_d}{\sigma}, \tag{D.13}$$

where $X = \sigma Z$, $Z \sim \mathcal{N}(0, I_d)$. □

Corollary D.1 (Noise induces smoothed gradients in expectation)**.** *Let* $\boldsymbol{\xi} \sim \mathcal{N}(0, \sigma^2 I_d)$, $L_\sigma(\mathbf{s}) = \mathbb{E}_{\boldsymbol{\xi}}[L(\mathbf{s}+\boldsymbol{\xi})]$. *Assume* $L \in C^1$ *and* $\mathbb{E}_{\boldsymbol{\xi}} \|\nabla L(\mathbf{s}+\boldsymbol{\xi})\| < \infty$ *(so differentiation may be interchanged with expectation). Then*

$$\mathbb{E}_{\boldsymbol{\xi}}[\nabla L(\mathbf{s}+\boldsymbol{\xi})] = \nabla_{\mathbf{s}} L_\sigma(\mathbf{s}). \tag{D.14}$$

If $L \in L^\infty(\mathbb{R}^d)$*, then by Theorem 7.4 (Part 2),*

$$\|\nabla_{\mathbf{s}} L_\sigma\|_{L^\infty} \leq \frac{c_d}{\sigma} \|L\|_{L^\infty}. \tag{D.15}$$

D.3 Connection to Boltzmann sampling

The continuous-time Langevin dynamics corresponding to (7.5), if we noise actions as well, has a stationary distribution that concentrates on low-energy regions of $\mathcal{L}_{\text{dyn}}$:

$$p(\mathbf{s}, \mathbf{a}) \;\propto\; \exp\big(-\beta \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a})\big), \tag{D.16}$$

for inverse temperature $\beta > 0$ determined by drift-to-diffusion scaling. Larger σ_{state} in (7.5) yields broader exploration; smaller concentrates around local minima.

However, we noise only states, so the converged distribution is not Boltzmann. The new distribution, written loosely, collapses on action-space solutions to the local dynamics problems:

$$p(\mathbf{s}, \mathbf{a}) \propto \delta(\mathbf{a} = \mathbf{a}^*(\mathbf{s})) \cdot \exp\big(-\beta \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a})\big), \tag{D.17}$$

where $\mathbf{a}^*(\mathbf{s}) = \arg\min_{\mathbf{a}} \mathcal{L}_{\text{dyn}}(\mathbf{s}, \mathbf{a})$.

D.4 State-gradient-free dynamics

We analyze the gradient-cut variant where gradient flow through state evolution in the dynamics is removed. This is akin to removing the adjoint (backward) pass through time, transforming the global trajectory optimization into a sequence of locally regularized problems.

Nonexistence of exact dynamics-enforcing losses with Jacobian-free state gradients

We first show that no loss function over states and actions can simultaneously (i) have minimizers that correspond to dynamically feasible trajectories, and (ii) be independent of the state gradients through F_θ.

Theorem D.2 (Nonexistence of exact dynamics-enforcing losses with Jacobian-free state gradients)**.** *Let* $\mathcal{S} \subseteq \mathbb{R}^n$ *and* $\mathcal{A} \subseteq \mathbb{R}^m$ *each be open and connected, and fix* $s_0 \in \mathcal{S}$ *and* $g \in \mathcal{S}$.

Consider horizon $T = 2$ with decision variables $(s_1, a_0, a_1) \in \mathcal{S} \times \mathcal{A} \times \mathcal{A}$ and boundary $s_2 = g$ fixed. Let $L_F : \mathcal{S} \times \mathcal{A} \times \mathcal{A} \to \mathbb{R}$ decompose as

$$\begin{aligned} L_F(s_1, a_0, a_1) &= \Phi(s_1, a_0, a_1, y_0, y_1), \\ y_0 &= F(s_0, a_0), \\ y_1 &= F(s_1, a_1), \end{aligned} \tag{D.18}$$

where $\Phi : \mathcal{S} \times \mathcal{A} \times \mathcal{A} \times \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is C^1. Let

$$\mathcal{M}(F) = \{(s_1, a_0, a_1) : s_1 = F(s_0, a_0) \text{ and } g = F(s_1, a_1)\} \tag{D.19}$$

denote feasible trajectories. There does not exist a Φ for which simultaneously, for every C^1 model F:

1. *Minimizers correspond to feasible dynamics:* $\arg\min L_F = \mathcal{M}(F)$*;*
2. *Independence of the loss with respect to the state gradient: for all $G : \mathcal{S} \times \mathcal{A} \to \mathcal{S}, G \in C^1$,*

$$(F(s_0, a_0) = G(s_0, a_0),\ F(s_1, a_1) = G(s_1, a_1)) \Rightarrow \nabla_{s_1} L_F = \nabla_{s_1} L_G. \tag{D.20}$$

Proof. We argue by contradiction. Fix (s_1, a_0, a_1) and write $y_0 = F(s_0, a_0)$, $y_1 = F(s_1, a_1)$. By the chain rule,

$$\nabla_{s_1} L_F = \frac{\partial \Phi}{\partial s_1} + (\nabla_s F(s_1, a_1))^\top \frac{\partial \Phi}{\partial y_1}. \tag{D.21}$$

We claim $\partial\Phi/\partial y_1 \equiv 0$. Construct C^1 models F and G with $F(s_0, a_0) = G(s_0, a_0) = y_0$, $F(s_1, a_1) = G(s_1, a_1) = y_1$, but with prescribed differing state Jacobians at (s_1, a_1):

$$\nabla_s F(s_1, a_1) = J_F, \quad \nabla_s G(s_1, a_1) = J_G. \tag{D.22}$$

Such construction is possible by bump-function surgery in a ball around (s_1, a_1). By Jacobian-invariance, $\nabla_{s_1} L_F = \nabla_{s_1} L_G$ at (s_1, a_0, a_1). Subtracting:

$$(J_F - J_G)^\top \frac{\partial \Phi}{\partial y_1} = 0. \tag{D.23}$$

Since $J_F - J_G$ can be any matrix in $\mathbb{R}^{n \times n}$, $\partial\Phi/\partial y_1 \equiv 0$. The loss is independent of y_1, so there exists $\tilde{\Phi}$ such that $L_F(s_1, a_0, a_1) = \tilde{\Phi}(s_1, a_0, a_1, F(s_0, a_0))$.

Now construct F, G with different feasible sets but $F(s_0, \cdot) = G(s_0, \cdot)$. Pick two actions $u, v \in \mathcal{A}$, two states $s_A, s_B \in \mathcal{S}$, and $a^\star \in \mathcal{A}$. Via bump-function surgery, construct F, G such that $F(s_0, a) = G(s_0, a)$ for all a, with $F(s_0, u) = G(s_0, u) = s_A$, $F(s_0, v) = G(s_0, v) = s_B$, but with swapped second-step goal reachability:

$$\begin{aligned} F(s_A, a^\star) &= g, & F(s_B, a^\star) &\neq g, \\ G(s_A, a^\star) &\neq g, & G(s_B, a^\star) &= g. \end{aligned} \tag{D.24}$$

Then $(s_A, u, a^\star) \in \mathcal{M}(F)$ but $\notin \mathcal{M}(G)$, and $(s_B, v, a^\star) \in \mathcal{M}(G)$ but $\notin \mathcal{M}(F)$, so $\mathcal{M}(F) \neq \mathcal{M}(G)$. Since $F(s_0, \cdot) = G(s_0, \cdot)$, $\arg\min L_F = \arg\min L_G$. By assumption (i), $\mathcal{M}(F) = \mathcal{M}(G)$, a contradiction. □

Linear convergence to a unique fixed point

We introduce the stop-gradient operator $\mathrm{sg}(\cdot)$, $\mathrm{sg}(x) = x$ during forward evaluation but $\nabla \mathrm{sg}(x) = 0$ during backward. The modified objective:

$$\mathcal{L}_{\mathrm{sg}}(\mathbf{s}, \mathbf{a}) = \sum_{t=0}^{T-1} \|s_{t+1} - A\,\mathrm{sg}(s_t) - Ba_t\|_2^2 + \sum_{t=0}^{T-1} \beta_t \|g - A\,\mathrm{sg}(s_t) - Ba_t\|_2^2, \tag{D.25}$$

with $\beta_t \geq 0$ the goal-loss coefficients.

Theorem D.3 (Linear convergence to a unique fixed point)**.** *Consider gradient descent $(\mathbf{s}^{k+1}, \mathbf{a}^{k+1}) = (\mathbf{s}^k, \mathbf{a}^k) - \eta \nabla \mathcal{L}_{sg}(\mathbf{s}^k, \mathbf{a}^k)$, s_0 fixed, gradients with the stop-gradient convention. Assume linear dynamics, $\beta_t \in [\beta_{\min}, \beta_{\max}]$ with $\beta_{\min} > 0$, and B full column rank. Then there exists $\eta \in (0, \bar{\eta})$, where $\bar{\eta}$ depends only on $\beta_{\min}, \beta_{\max}, \|B\|, \sigma_{\min}(B)$ (not on T), such that the update operator $\mathcal{T}$ on $\mathbf{z} = (\mathbf{s}, \mathbf{a})$ has a unique fixed point $\mathbf{z}^\star$ and iterates converge linearly:*

$$\left\|\mathbf{z}^k - \mathbf{z}^\star\right\| \leq C q^k \left\|\mathbf{z}^0 - \mathbf{z}^\star\right\|, \quad q \in (0,1),\, C > 0. \tag{D.26}$$

Proof. Define residuals $r_t = s_{t+1} - As_t - Ba_t$, $e_t = g - As_t - Ba_t$. With stop-gradient, s_t does not receive contributions through $A\,\mathrm{sg}(s_t)$ terms. The state-gradient at time t comes only from s_t's appearance as the *next* state in the previous residual:

$$\nabla_{s_t} \mathcal{L}_{\mathrm{sg}} = 2r_{t-1}, \quad t = 1, \ldots, T, \tag{D.27}$$

with $r_{-1} = 0$ if s_0 is fixed. The action-gradient is

$$\nabla_{a_t} \mathcal{L}_{\mathrm{sg}} = -2B^\top r_t - 2\beta_t B^\top e_t, \quad t = 0, \ldots, T-1. \tag{D.28}$$

Updates with stepsize η:

$$s_t^{k+1} = s_t^k - 2\eta(s_t^k - As_{t-1}^k - Ba_{t-1}^k), \quad t = 1, \ldots, T, \tag{D.29}$$

$$a_t^{k+1} = a_t^k + 2\eta B^\top((s_{t+1}^k - As_t^k - Ba_t^k) + \beta_t(g - As_t^k - Ba_t^k)). \tag{D.30}$$

Stack variables as $\mathbf{z} = (s_1; a_0; s_2; a_1; \ldots; s_T; a_{T-1})$. The updates form an affine map $\mathbf{z}^{k+1} = J\mathbf{z}^k + c$ whose Jacobian J is block lower-triangular: $(s_{t+1}^{k+1}, a_t^{k+1})$ depends only on $(s_t^k, s_{t+1}^k, a_t^k)$ and constants. The eigenvalues of J are the union of eigenvalues of its diagonal blocks.

Each diagonal block at time t is one gradient step on the quadratic

$$\phi_t(s_{t+1}, a_t; s_t) = \|s_{t+1} - As_t - Ba_t\|_2^2 + \beta_t \|g - As_t - Ba_t\|_2^2, \tag{D.31}$$

so the block equals $I - \eta H_t$ where

$$H_t = 2\begin{bmatrix} I & -B \\ -B^\top & (1+\beta_t)B^\top B \end{bmatrix}. \tag{D.32}$$

For $\beta_t \in [\beta_{\min}, \beta_{\max}]$ with $\beta_{\min} > 0$ and B full column rank, the Schur complement of the I block is $\beta_t B^\top B \succ 0$, so $H_t \succ 0$ uniformly:

$$0 < \mu I \preceq H_t \preceq LI < \infty. \tag{D.33}$$

For any η with $0 < \eta < 2/L$, all eigenvalues of $I - \eta H_t$ lie in the unit disk:

$$\rho(I - \eta H_t) \leq \max\{|1 - \eta\mu|, |1 - \eta L|\} = q < 1. \tag{D.34}$$

Since J is block lower-triangular, $\rho(J) = \max_t \rho(I - \eta H_t) \leq q_0 < 1$. By Gelfand's formula, there exists an induced norm $\|\cdot\|_\dagger$ with $\|J\|_\dagger \leq q$ for any $q > q_0$. Hence

$$\left\|\mathbf{z}^k - \mathbf{z}^\star\right\|_\dagger \leq q^k \left\|\mathbf{z}^0 - \mathbf{z}^\star\right\|_\dagger, \tag{D.35}$$

and by norm equivalence in finite dimensions, the ℓ_2 statement follows. □

Remark D.4. The fixed point $\mathbf{z}^\star$ warrants a caveat carried over from the linear analysis. In this linear, convex setting one can show that these stable points are merely the greedy rollouts toward the goal; what makes the optimization nontrivial in the actual setting is what the analysis excludes: the nonconvexity of the learned world model F_θ and the injected state noise. The linear-convergence guarantee accordingly does not extend to the nonconvex learned-dynamics loss, and characterizes only the linearized regime.

Gaussian tube and goal-shaped drift

Proof of Theorem 7.5. Rewrite the state update as an affine Gaussian recursion:

$$s_{t+1}^{k+1} = (1 - 2\eta_s)s_{t+1}^k + 2\eta_s\mu_t^k + \sigma\xi_{t+1}^k. \tag{D.36}$$

Taking conditional expectation yields the mean recursion. If $\mu_t^k \equiv \mu_t$ is fixed, the centered process $u^k = s_{t+1}^k - \mu_t$ satisfies $u^{k+1} = (1 - 2\eta_s)u^k + \sigma\xi^k$, an AR(1) with contraction factor $|1 - 2\eta_s| < 1$ (valid for step size $0 < \eta_s < 1$). The unique stationary covariance solves $\Sigma = (1 - 2\eta_s)^2\Sigma + \sigma^2 I$:

$$\Sigma_{\text{tube}} = \frac{\sigma^2}{1 - (1 - 2\eta_s)^2} I = \frac{\sigma^2}{4\eta_s(1 - \eta_s)} I. \tag{D.37}$$

The continuous-time SDE statement is standard, given μ_t fixed. □

Proof of Theorem 7.6. From the gradient-cut objective, only terms at time t depend on a_t via μ_t. Using $\nabla_{a_t}\mu_t = J_t$:

$$\nabla_{a_t}\mathcal{L} = 2J_t^\top(\mu_t - s_{t+1}) + 2\gamma J_t^\top(\mu_t - g). \tag{D.38}$$

The action update:

$$a_t^{k+1} - a_t^k = -2\eta_a J_t^\top(\mu_t^k - s_{t+1}^k) - 2\eta_a\gamma J_t^\top(\mu_t^k - g). \tag{D.39}$$

Applying the linearization $\mu_t^{k+1} \approx \mu_t^k + J_t(a_t^{k+1} - a_t^k)$:

$$\mu_t^{k+1} \approx \mu_t^k - 2\eta_a J_t J_t^\top(\mu_t^k - s_{t+1}^k) - 2\eta_a \gamma J_t J_t^\top(\mu_t^k - g) \tag{D.40}$$

$$= \mu_t^k - 2\eta_a P_t(\mu_t^k - (\mu_t^k + \varepsilon_{t+1}^k)) - 2\eta_a \gamma P_t(\mu_t^k - g), \tag{D.41}$$

which simplifies to the tube-center update after substituting $\alpha = 2\eta_a$. □

Rollout mean evolution and non-tube behavior

In contrast to the stop-gradient lifted dynamics, the rollout distribution is not contracted toward the current rollout; noise accumulates throughout nonlinear iterations.

Theorem D.5 (Mean evolution and non-tube behavior of noisy rollouts)**.** *Consider a rollout-based stochastic trajectory in model-time:*

$$s_{t+1} = F_\theta(s_t, a_t) + \sigma_{env}\zeta_t, \quad \zeta_t \sim \mathcal{N}(0, I), \quad s_0 \text{ fixed}, \tag{D.42}$$

with dense goal objective along the rollout. Let $m_t = \mathbb{E}[s_t]$, $\Sigma_t = \mathrm{Cov}(s_t)$.

The rollout mean obeys

$$m_{t+1} = \mathbb{E}[F_\theta(s_t, a_t)]. \tag{D.43}$$

For affine F_θ, $m_{t+1} = F_\theta(m_t, a_t)$. For nonlinear F_θ, a second-order expansion yields

$$m_{t+1} \approx F_\theta(m_t, a_t) + \tfrac{1}{2}(\mathrm{Hess}_s F_\theta(m_t, a_t)) : \Sigma_t, \tag{D.44}$$

so the mean generally does not follow the deterministic rollout. A first-order linearization gives covariance propagation

$$\Sigma_{t+1} \approx G_t \Sigma_t G_t^\top + \sigma_{env}^2 I, \quad G_t = \nabla_s F_\theta(m_t, a_t). \tag{D.45}$$

Proof. Taking conditional expectation: $\mathbb{E}[s_{t+1} \mid s_t] = F_\theta(s_t, a_t)$, then total expectation gives $m_{t+1} = \mathbb{E}[F_\theta(s_t, a_t)]$. For affine F_θ, expectation commutes with F_θ. For nonlinear, Taylor-expand F_θ around m_t; first-order vanishes in expectation, second-order produces the Hessian-trace term. Covariance: linearize $F_\theta(s_t, a_t) \approx F_\theta(m_t, a_t) + G_t(s_t - m_t)$, compute $\mathrm{Cov}(\cdot)$, add independent noise variance. □

The non-tube claim: there is no optimization-time contraction repeatedly pulling s_{t+1} back toward a moving center (unlike Theorem 7.5); instead, forward propagation typically increases spread and the mean can deviate from the deterministic path.

D.5 Additional ablations

To complement the main-body ablations, we evaluate alternative simpler stochastic gradient-based planners. Building from a normal rollout-based GD planner, we consider two noise types:

Table D.1: Stochastic GD baselines ablation, showing alternatives from our method for a stochastic gradient-based planner. None beat GRASP's performance (Table 7.4). Reported time is average over experiments; intervals are 95% CI.

Setting	**Accuracy (%)**	**Time (s)**
Default Mode (objective.mode=default)		
$\sigma_s = 0.0$	53.0 ± 4.4	864 ± 77
$\sigma_s = 0.5$	50.6 ± 4.4	913.97 ± 78
$\sigma_s = 1.0$	37.2 ± 4.3	1149 ± 75
$\sigma_a = 0.0$	47.6 ± 4.4	1056 ± 73
$\sigma_a = 0.5$	53.0 ± 4.4	864 ± 77
$\sigma_a = 1.0$	33.4 ± 4.2	1208 ± 73
$\sigma_s = 0.5, \sigma_a = 0.5$	53.8 ± 4.4	854 ± 78
Mode All (objective.mode=all)		
$\sigma_s = 0.0$	54.0 ± 4.4	865 ± 77
$\sigma_s = 0.5$	55.8 ± 4.4	838 ± 77
$\sigma_s = 1.0$	41.0 ± 4.4	1111 ± 75
$\sigma_a = 0.0$	46.6 ± 4.4	1067 ± 73
$\sigma_a = 0.5$	52.8 ± 4.4	873 ± 77
$\sigma_a = 1.0$	52.4 ± 4.4	878 ± 77
$\sigma_s = 0.5, \sigma_a = 0.5$	55.8 ± 4.4	838 ± 77

- *Action noise.* Actions are sampled around Gaussians of fixed variance σ_a before rollout evaluation.
- *State noise.* Noise is added during the rollout directly to states: $s_{t+1} = F(s_t, a_t) + z$, $z \sim \mathcal{N}(0, \sigma_s I)$.

D.6 Additional figures

D.7 Hyperparameters and benchmark details

Default hyperparameters. $\gamma = 0.1$ (goal-loss coefficient); $\sigma_{\text{state}} = 0.5$ (decaying geometrically with factor 0.99); $K = 10^3$ gradient steps; $K_{\text{sync}} = 25$ for standard ablations (Table 7.4); learning rates $\eta_a = \eta_s = 10^{-3}$ (Adam).

Environments. Initial evaluation: PointMaze, Wall-Single, Push-T with DINO-wm world models (G. Zhou et al., 2024). Extended evaluation: Reacher, Cube manipulation, Meta-World Reach (MW Reach) with latent-encoder (`le-wm`) and JEPA-style (Assran et al., 2025) (`jepa-wm`) world models.



Figure D.1: Example of converged plans for GRASP on Push-T at horizon 80, with goals depicted to the right of the dashed line. The parallel optimization converges more consistently at longer horizons, and finds the required non-greedy paths to the goal.

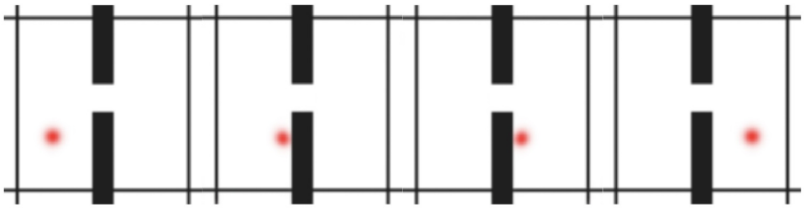


Figure D.2: Example of a bad local minimum when relaxing the dynamics constraint to a penalty function. The dynamics are nonzero for the middle transition, but both states and actions are at a local minimum; there is no direction they can move locally to reduce the dynamics loss further.

Baselines. CEM (Rubinstein and Kroese, 2004), vanilla GD (BPTT through the rolled-out trajectory), LatCo (Rybkin et al., 2021) adapted to our pure-planning setting. For all methods we sweep hyperparameters and report the best-performing setting per environment and horizon.

Success criterion. Reaching the goal region within the planning horizon. 500 trials per setting for short/long-horizon Push-T, PointMaze, WallSingle experiments; sample sizes for the extended evaluation table noted per-environment.

Compute. Each trial runs on a single GPU; full ablation suites complete within hours. Full quantitative tables, ablations, and additional plots are in Psenka, LeCun, et al., 2026.

Bibliography

Abramson, Josh, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. (2024). "Accurate structure prediction of biomolecular interactions with AlphaFold 3". In: *Nature* 630.8016, pp. 493–500.

Abresch, Uwe and Joel Langer (1986). "The normalized curve shortening flow and homothetic solutions". In: *Journal of Differential Geometry* 23.2, pp. 175–196.

Adib, Artur B. (May 2008). "Stochastic Actions for Diffusive Dynamics: Reweighting, Sampling, and Minimization". In: *The Journal of Physical Chemistry B* 112.19, pp. 5910–5916. ISSN: 1520-6106. DOI: 10.1021/jp0751458.

Agostini, Alejandro and Enric Celaya (2010). "Reinforcement learning with a Gaussian mixture model". In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8.

Albergo, Michael S, Nicholas M Boffi, and Eric Vanden-Eijnden (2023). "Stochastic interpolants: A unifying framework for flows and diffusions". In: *arXiv preprint arXiv:2303.08797*.

Arora, Sanjeev, Andrej Risteski, and Yi Zhang (2018). "Do GANs learn the distribution? some theory and empirics". In: *International Conference on Learning Representations*.

Arts, Marloes, Victor Garcia Satorras, Chin-Wei Huang, Daniel Zugner, Marco Federici, Cecilia Clementi, Frank Noé, Robert Pinsler, and Rianne van den Berg (2023). "Two for one: Diffusion models and force fields for coarse-grained molecular dynamics". In: *Journal of Chemical Theory and Computation* 19.18, pp. 6151–6159.

Arvanitidis, Georgios, L. Kai Hansen, and Søren Hauberg (2018). "Latent Space Oddity: On the Curvature of Deep Generative Models". In: *Proceedings of the International Conference on Learning Representations*.

Ascher, Uri M, Robert MM Mattheij, and Robert D Russell (1995). *Numerical solution of boundary value problems for ordinary differential equations*. SIAM.

Assran, Mido, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. (2025). "V-jepa 2: Self-supervised video models enable understanding, prediction and planning". In: *arXiv preprint arXiv:2506.09985*.

Autieri, Emmanuel, Pietro Faccioli, Marcello Sega, Francesco Pederiva, and H Orland (2009). "Dominant reaction pathways in high-dimensional systems". In: *The Journal of chemical physics* 130.6.

Bakry, Dominique, Ivan Gentil, Michel Ledoux, et al. (2014). *Analysis and geometry of Markov diffusion operators*. Vol. 103. Springer.

Ball, Philip J., Jakob Bauer, Frank Belletti, Bethanie Brownfield, Ariel Ephrat, Shlomi Fruchter, Agrim Gupta, Kristian Holsheimer, Aleksander Holynski, Jiri Hron, Christos Kaplanis, Marjorie Limont, Matt McGill, Yanko Oliveira, Jack Parker-Holder, Frank Perbet, Guy Scully, Jeremy Shar, Stephen Spencer, Omer Tov, Ruben Villegas, Emma Wang, Jessica Yung, Cip Baetu, Jordi Berbel, David Bridson, Jake Bruce, Gavin Buttimore, Sarah Chakera, Bilva Chandra, Paul Collins, Alex Cullum, Bogdan Damoc, Vibha Dasagi, Maxime Gazeau, Charles Gbadamosi, Woohyun Han, Ed Hirst, Ashyana Kachra, Lucie Kerley, Kristian Kjems, Eva Knoepfel, Vika Koriakin, Jessica Lo, Cong Lu, Zeb Mehring, Alex Moufarek, Henna Nandwani, Valeria Oliveira, Fabio Pardo, Jane Park, Andrew Pierson, Ben Poole, Helen Ran, Tim Salimans, Manuel Sanchez, Igor Saprykin, Amy Shen, Sailesh Sidhwani, Duncan Smith, Joe Stanton, Hamish Tomlinson, Dimple Vijaykumar, Luyu Wang, Piers Wingfield, Nat Wong, Keyang Xu, Christopher Yew, Nick Young, Vadim Zubov, Douglas Eck, Dumitru Erhan, Koray Kavukcuoglu, Demis Hassabis, Zoubin Gharamani, Raia Hadsell, Aäron van den Oord, Inbar Mosseri, Adrian Bolton, Satinder Singh, and Tim Rocktäschel (2025). "Genie 3: A New Frontier for World Models". In.

Bank, Protein Data (1971). "Protein data bank". In: *Nature New Biol* 233.223, pp. 10–1038.

Bar, Amir, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun (2025). "Navigation world models". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15791–15801.

Beccara, Silvio a, Tatjana Škrbić, Roberto Covino, and Pietro Faccioli (2012). "Dominant folding pathways of a WW domain". In: *Proceedings of the National Academy of Sciences* 109.7, pp. 2330–2335.

Belkin, Mikhail and Partha Niyogi (2003). "Laplacian eigenmaps for dimensionality reduction and data representation". In: *Neural computation* 15.6, pp. 1373–1396.

Bellman, Richard (1954). "Dynamic programming and a new formalism in the calculus of variations". In: *Proceedings of the national academy of sciences* 40.4, pp. 231–235.

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013). "Representation learning: A review and new perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8, pp. 1798–1828.

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (1994). "Learning long-term dependencies with gradient descent is difficult". In: *IEEE transactions on neural networks* 5.2, pp. 157–166.

Berseth, Glen, Alex Kyriazis, Ivan Zinin, William Choi, and Michiel van de Panne (2018). "Model-based action exploration for learning dynamic motion skills". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1540–1546.

Berthelot, David, Colin Raffel, Aurko Roy, and Ian Goodfellow (2018). "Understanding and improving interpolation in autoencoders via an adversarial regularizer". In: *arXiv preprint arXiv:1807.07543*.

Bharadhwaj, Homanga, Kevin Xie, and Florian Shkurti (2020). "Model-predictive control via cross-entropy and gradient-based optimization". In: *Learning for Dynamics and Control*. PMLR, pp. 277–286.

Birsan, T and Dan Tiba (2005). "One Hundred Years Since the Introduction of the Set Distance by Dimitrie Pompeiu." In: *System Modelling and Optimization* 199, pp. 35–39.

Black, Kevin, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine (2023). "Training diffusion models with reinforcement learning". In: *arXiv preprint arXiv:2305.13301*.

Bock, Hans Georg and Karl-Josef Plitt (1984). "A multiple shooting algorithm for direct solution of optimal control problems". In: *IFAC Proceedings Volumes* 17.2, pp. 1603–1608.

Bolhuis, Peter G, David Chandler, Christoph Dellago, and Phillip L Geissler (2002). "Transition path sampling: Throwing ropes over rough mountain passes, in the dark". In: *Annual review of physical chemistry* 53.1, pp. 291–318.

Bolhuis, Peter G and David WH Swenson (2021). "Transition path sampling as Markov chain Monte Carlo of trajectories: Recent algorithms, software, applications, and future outlook". In: *Advanced Theory and Simulations* 4.4, p. 2000237.

Bordalba, Ricard, Tobias Schoels, Lluís Ros, Josep M Porta, and Moritz Diehl (2022). "Direct collocation methods for trajectory optimization in constrained robotic systems". In: *IEEE Transactions on Robotics* 39.1, pp. 183–202.

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

Boumal, Nicolas (2020). "An Introduction to Optimization on Smooth Manifolds". In: *Available online, Aug*.

Bradtke, Steven and Michael Duff (1994). "Reinforcement learning methods for continuous-time Markov decision problems". In: *Advances in neural information processing systems* 7.

Bras, Pierre (2023). "Langevin algorithms for very deep neural networks with application to image classification". In: *Procedia Computer Science* 222, pp. 303–310.

Brown, Bradley CA, Anthony L Caterini, Brendan Leigh Ross, Jesse C Cresswell, and Gabriel Loaiza-Ganem (2022). "The union of manifolds hypothesis and its implications for deep generative modelling". In: *arXiv preprint arXiv:2207.02862*.

Brown, Noam and Tuomas Sandholm (2019). "Superhuman AI for multiplayer poker". In: *Science* 365.6456, pp. 885–890.

Campadelli, Paola, Elena Casiraghi, Claudio Ceruti, and Alessandro Rozza (2015). "Intrinsic dimension estimation: Relevant techniques and a benchmark framework". In: *Mathematical Problems in Engineering* 2015, pp. 1–21.

Caron, Mathilde, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin (2021). "Emerging properties in self-supervised vision transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9650–9660.

Carter, Kevin M, Raviv Raich, and Alfred O Hero III (2009). "On local intrinsic dimension estimation and its applications". In: *IEEE Transactions on Signal Processing* 58.2, pp. 650–663.

Chan, K, Yaodong Yu, Chong You, Haozhi Qi, John Wright, and Yi Ma (2022). "ReduNet: A white-box deep network from the principle of maximizing rate reduction". In: *Journal of machine learning research* 23.114.

Chaudhari, Pratik, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina (2019). "Entropy-sgd: Biasing gradient descent into wide valleys". In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.12, p. 124018.

Chen, CL Philip (1996). "A rapid supervised learning neural network for function interpolation and approximation". In: *IEEE Transactions on Neural Networks* 7.5, pp. 1220–1230.

Chen, Sitan, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R Zhang (2022). "Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions". In: *arXiv preprint arXiv:2209.11215*.

Chen, Wei and Andrew L Ferguson (2018). "Molecular enhanced sampling with autoencoders: On-the-fly collective variable discovery and accelerated free energy landscape exploration". In: *Journal of computational chemistry* 39.25, pp. 2079–2102.

Coll, Vincent E, Jeff Dodd, and David L Johnson (2020). "Ricci flow on surfaces of revolution: an extrinsic view". In: *Geometriae Dedicata* 207.1, pp. 81–94.

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

Costa, Jose A and Alfred O Hero (2006). "Determining intrinsic dimension and entropy of high-dimensional shape spaces". In: *Statistics and analysis of shapes*. Springer, pp. 231–252.

D'Oro, Pierluca and Wojciech Jaśkowski (2020). "How to learn a useful critic? Model-based action-gradient-estimator policy optimization". In: *Advances in Neural Information Processing Systems* 33, pp. 313–324.

Das, Avishek, Dominic C Rose, Juan P Garrahan, and David T Limmer (2021). "Reinforcement learning of rare diffusive dynamics". In: *The Journal of Chemical Physics* 155.13.

Dekker, H (1978). "Derivation of the Onsager-Machlup lagrangian for general diffusion processes by means of Fourier series". In: *Physics Letters A* 68.2, pp. 137–140.

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

Deng, Li (2012). "The mnist database of handwritten digit images for machine learning research [best of the web]". In: *IEEE signal processing magazine* 29.6, pp. 141–142.

Deschenaux, Justin, Igor Krawczuk, Grigorios Chrysos, and Volkan Cevher (2024). "Going beyond compositions, DDPMs can produce zero-shot interpolations". In: *arXiv preprint arXiv:2405.19201*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805*.

Diedam, H and Sebastian Sager (2018). "Global optimal control with the direct multiple shooting method". In: *Optimal Control Applications and Methods* 39.2, pp. 449–470.

Ding, Jingtao, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, et al. (2024). "Understanding world or predicting future? a comprehensive survey of world models". In: *ACM Computing Surveys*.

Doerr, Stefan, Maciej Majewsk, Adrià Pérez, Andreas Krämer, Cecilia Clementi, Frank Noe, Toni Giorgino, and Gianni De Fabritiis (2020). *TorchMD: A deep learning framework for molecular simulations*. arXiv: 2012.12106 [physics.chem-ph].

Doya, Kenji (2000). "Reinforcement learning in continuous time and space". In: *Neural computation* 12.1, pp. 219–245.

Du, Yuanqi, Michael Plainer, Rob Brekelmans, Chenru Duan, Frank Noé, Carla P Gomes, Alán Aspuru-Guzik, and Kirill Neklyudov (2024). "Doob's Lagrangian: A Sample-Efficient Variational Approach to Transition Path Sampling". In: *arXiv preprint arXiv:2410.07974*.

Dürr, Detlef and Alexander Bach (1978). "The Onsager-Machlup function as Lagrangian for the most probable path of a diffusion process". In: *Communications in Mathematical Physics* 60, pp. 153–170.

Dutra, Dimas Abreu, Bruno Otávio Soares Teixeira, and Luis Antonio Aguirre (2014). "Maximum a posteriori state path estimation: Discretization limits and their interpretation". In: *Automatica* 50.5, pp. 1360–1368.

Erickson, Jeffrey Gordon (1996). *Lower bounds for fundamental geometric problems*. University of California, Berkeley.

Faccioli, Pietro, Marcello Sega, Francesco Pederiva, and H Orland (2006). "Dominant pathways in protein folding". In: *Physical review letters* 97.10, p. 108101.

Facco, Elena, Maria d'Errico, Alex Rodriguez, and Alessandro Laio (2017). "Estimating the intrinsic dimension of datasets by a minimal neighborhood information". In: *Scientific reports* 7.1, pp. 1–8.

Falkner, Sebastian, Alessandro Coretti, Salvatore Romano, Phillip L Geissler, and Christoph Dellago (2023). "Conditioning Boltzmann generators for rare event sampling". In: *Machine Learning: Science and Technology* 4.3, p. 035050.

Federer, Herbert (1959). "Curvature measures". In: *Transactions of the American Mathematical Society* 93.3, pp. 418–491.

Fefferman, Charles, Sanjoy Mitter, and Hariharan Narayanan (2016). "Testing the manifold hypothesis". In: *Journal of the American Mathematical Society* 29.4, pp. 983–1049.

Flash, Tamar and Neville Hogan (1985). "The coordination of arm movements: an experimentally confirmed mathematical model". In: *Journal of neuroscience* 5.7, pp. 1688–1703.

Fleming, Wendell H and Raymond W Rishel (2012). *Deterministic and stochastic optimal control*. Vol. 1. Springer Science & Business Media.

Foret, Pierre, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur (2020). "Sharpness-aware minimization for efficiently improving generalization". In: *arXiv preprint arXiv:2010.01412*.

Fraikin, Catherine, K Hüper, and P Van Dooren (2007). "Optimization over the Stiefel manifold". In: *PAMM: Proceedings in Applied Mathematics and Mechanics*. Vol. 7. 1. Wiley Online Library, pp. 1062205–1062206.

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

Fu, Justin, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine (2020). "D4rl: Datasets for deep data-driven reinforcement learning". In: *arXiv preprint arXiv:2004.07219*.

Fujimoto, Scott, Herke van Hoof, and David Meger (2018a). "Addressing Function Approximation Error in Actor-Critic Methods". In: *International Conference on Machine Learning*.

— (2018b). *Addressing Function Approximation Error in Actor-Critic Methods*. arXiv: 1802.09477 [cs.AI].

Fujisaki, Hiroshi, Motoyuki Shiga, and Akinori Kidera (Apr. 2010). "Onsager–Machlup action-based path sampling and its combination with replica exchange for diffusive and multiple pathways". In: *The Journal of Chemical Physics* 132.13, p. 134101. ISSN: 0021-9606. DOI: 10.1063/1.3372802.

Gale, David (1963). "Neighborly and cyclic polytopes". In: *Proc. Sympos. Pure Math.* Vol. 7. 7, pp. 225–232.

Gel'fand, I. M. and A. M. Yaglom (Jan. 1960). "Integration in Functional Spaces and its Applications in Quantum Physics". In: *Journal of Mathematical Physics* 1.1, pp. 48–69. ISSN: 0022-2488. DOI: 10.1063/1.1703636.

Gelfand, I. M. and S. V. Fomin (1963). *Calculus of Variations*. Prentice-Hall.

Gelfand, Saul B and Sanjoy K Mitter (1991). "Recursive stochastic algorithms for global optimization in R^d". In: *SIAM Journal on Control and Optimization* 29.5, pp. 999–1018.

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

— (2020). "Generative adversarial networks". In: *Communications of the ACM* 63.11, pp. 139–144.

Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy (2015). "Explaining and harnessing adversarial examples". In: *International Conference on Learning Representations (ICLR)*.

Goswami, Raktim Gautam, Amir Bar, David Fan, Tsung-Yen Yang, Gaoyue Zhou, Prashanth Krishnamurthy, Michael Rabbat, Farshad Khorrami, and Yann LeCun (2025). "World Models Can Leverage Human Videos for Dexterous Manipulation". In: *arXiv preprint arXiv:2512.13644*.

Guhathakurta, Dipanwita, Fatemeh Rastgar, M Aditya Sharma, K Madhava Krishna, and Arun Kumar Singh (2022). "Fast joint multi-robot trajectory optimization by GPU accelerated batch solution of distributed sub-problems". In: *Frontiers in Robotics and AI* 9, p. 890385.

Guo, Yanjiang, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn (2025). "Ctrl-world: A controllable generative world model for robot manipulation". In: *arXiv preprint arXiv:2510.10125*.

Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, and Sergey Levine (2018). *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. arXiv: 1801.01290 [cs.LG].

Hafner, Danijar, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi (2019). "Dream to control: Learning behaviors by latent imagination". In: *arXiv preprint arXiv:1912.01603*.

Hafner, Danijar, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap (2025). "Mastering diverse control tasks through world models". In: *Nature*, pp. 1–7.

Hamilton, Richard S (1982). "Three-manifolds with positive Ricci curvature". In: *Journal of Differential geometry* 17.2, pp. 255–306.

Hansen-Estruch, Philippe, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine (2023). "Idql: Implicit q-learning as an actor-critic method with diffusion policies". In: *arXiv preprint arXiv:2304.10573*.

Hasyim, Muhammad R, Clay H Batton, and Kranthi K Mandadapu (2022). "Supervised learning and the finite-temperature string method for computing committor functions and reaction rates". In: *The Journal of chemical physics* 157.18.

He, Kaiming, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick (2022). "Masked autoencoders are scalable vision learners". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009.

Hein, Matthias and Jean-Yves Audibert (2005). "Intrinsic dimensionality estimation of submanifolds in Rd". In: *Proceedings of the 22nd international conference on Machine learning*, pp. 289–296.

Henkelman, Graeme, Blas P. Uberuaga, and Hannes Jónsson (Dec. 2000). "A climbing image nudged elastic band method for finding saddle points and minimum energy paths". In: *The Journal of Chemical Physics* 113.22, pp. 9901–9904. ISSN: 0021-9606. DOI: 10.1063/1.1329672.

Higgins, Irina, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner (2017). "beta-vae: Learning basic visual concepts with a constrained variational framework". In: *International conference on learning representations*.

Hinton, Geoffrey (2022). "The forward-forward algorithm: Some preliminary investigations". In: *arXiv preprint arXiv:2212.13345*.

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

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

Hoppe, Hugues, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle (1993). "Mesh optimization". In: *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 19–26.

Huang, Kevin, Sahin Lale, Ugo Rosolia, Yuanyuan Shi, and Anima Anandkumar (2021). "Cem-gd: Cross-entropy method with gradient descent planner for model-based reinforcement learning". In: *arXiv preprint arXiv:2112.07746*.

Huisken, Gerhard (1984). "Flow by mean curvature of convex surfaces into spheres". In: *Journal of Differential Geometry* 20.1, pp. 237–266.

Hutchinson, Michael F (1989). "A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines". In: *Communications in Statistics-Simulation and Computation* 18.3, pp. 1059–1076.

Hyvärinen, Aapo and Peter Dayan (2005). "Estimation of non-normalized statistical models by score matching." In: *Journal of Machine Learning Research* 6.4.

Ibarz, Julian, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine (2021). "How to train your robot with deep reinforcement learning: lessons we have learned". In: *The International Journal of Robotics Research* 40.4-5, pp. 698–721.

Iluz, Shir, Yael Vinker, Amir Hertz, Daniel Berio, Daniel Cohen-Or, and Ariel Shamir (2023). "Word-as-image for semantic typography". In: *ACM Transactions on Graphics (TOG)* 42.4, pp. 1–11.

Izenman, Alan Julian (n.d.). "Introduction to manifold learning". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 4.5 (), pp. 439–446.

Janner, Michael, Yilun Du, Joshua Tenenbaum, and Sergey Levine (2022). "Planning with Diffusion for Flexible Behavior Synthesis". In: *International Conference on Machine Learning*. PMLR, pp. 9902–9915.

Jansen, Aren, Gregory Sell, and Vince Lyzinski (2017). "Scalable out-of-sample extension of graph embeddings using deep neural networks". In: *Pattern Recognition Letters* 94, pp. 1–6.

Jerez, Juan L, Eric C Kerrigan, and George A Constantinides (2011). "A condensed and sparse QP formulation for predictive control". In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE, pp. 5217–5222.

Jia, Yanwei and Xun Yu Zhou (2022). "Policy gradient and actor-critic learning in continuous time and space: Theory and algorithms". In: *Journal of Machine Learning Research* 23.275, pp. 1–50.

— (2023). "q-Learning in continuous time". In: *Journal of Machine Learning Research* 24.161, pp. 1–61.

Jing, Bowen, Bonnie Berger, and Tommi Jaakkola (2024). *AlphaFold Meets Flow Matching for Generating Protein Ensembles*. arXiv: 2402.04845 [q-bio.BM].

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

Jung, Hendrik, Kei-ichi Okazaki, and Gerhard Hummer (2017). "Transition path sampling of rare events by shooting from the top". In: *The Journal of chemical physics* 147.15.

Jyothir, SV, Siddhartha Jalagam, Yann LeCun, and Vlad Sobal (2023). "Gradient-based planning with world models". In: *arXiv preprint arXiv:2312.17227*.

Kang, Bingyi, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan (2023). "Efficient Diffusion Policies for Offline Reinforcement Learning". In: *arXiv preprint arXiv:2305.20081*.

Kappler, Julian and Ronojoy Adhikari (June 2020a). "Stochastic action for tubes: Connecting path probabilities to measurement". In: *Phys. Rev. Res.* 2 (2), p. 023407. DOI: 10.1103/PhysRevResearch.2.023407.

— (2020b). "Stochastic action for tubes: Connecting path probabilities to measurement". In: *Physical Review Research* 2.2, p. 023407.

Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen (2017). "Progressive growing of gans for improved quality, stability, and variation". In: *arXiv preprint arXiv:1710.10196*.

Karras, Tero, Samuli Laine, and Timo Aila (2019). "A style-based generator architecture for generative adversarial networks". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410.

Kim, Hyunjik and Andriy Mnih (2018). "Disentangling by factorising". In: *International Conference on Machine Learning*. PMLR, pp. 2649–2658.

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG].

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

Kirk, Donald E (2004). *Optimal control theory: an introduction*. Courier Corporation.

Kober, Jens, J Andrew Bagnell, and Jan Peters (2013). "Reinforcement learning in robotics: A survey". In: *The International Journal of Robotics Research* 32.11, pp. 1238–1274.

Kobyzev, I, SJD Prince, and MA Brubaker (2019). "Normalizing flows: An introduction and review of current methods, arxiv e-prints". In: *arXiv preprint arXiv:1908.09257*.

Koju, Saurabh, Saurav Bastola, Prashant Shrestha, Sanskar Amgain, Yash Raj Shrestha, Rudra PK Poudel, and Binod Bhattarai (2025). "Surgical vision world model". In: *MICCAI Workshop on Data Engineering in Medical Imaging*. Springer, pp. 1–10.

Kostková, Jitka, Jan Flusser, Matěj Lébl, and Matteo Pedone (2020). "Handling Gaussian blur without deconvolution". In: *Pattern Recognition* 103, p. 107264.

Kramer, Mark A (1991). "Nonlinear principal component analysis using autoassociative neural networks". In: *AIChE journal* 37.2, pp. 233–243.

Kumar, Shankar, John M Rosenberg, Djamal Bouzida, Robert H Swendsen, and Peter A Kollman (1992). "The weighted histogram analysis method for free-energy calculations on biomolecules. I. The method". In: *Journal of computational chemistry* 13.8, pp. 1011–1021.

Lahiri, Subhaneil, Peiran Gao, and Surya Ganguli (2016). "Random projections of random manifolds". In: *arXiv preprint arXiv:1607.04331*.

Lai, Tin, Weiming Zhi, Tucker Hermans, and Fabio Ramos (2022). "Parallelised diffeomorphic sampling-based motion planning". In: *Conference on robot learning*. PMLR, pp. 81–90.

Laio, Alessandro and Francesco L Gervasio (2008). "Metadynamics: a method to simulate rare events and reconstruct the free energy in biophysics, chemistry and material science". In: *Reports on Progress in Physics* 71.12, p. 126601.

Laio, Alessandro and Michele Parrinello (2002a). "Escaping free-energy minima". In: *Proceedings of the national academy of sciences* 99.20, pp. 12562–12566.

— (Oct. 2002b). "Escaping free-energy minima". In: *Proceedings of the National Academy of Sciences of the United States of America* 99.20, pp. 12562–12566.

Langevin, Paul (1908). "Sur la théorie du mouvement brownien". In: *C. R. Acad. Sci.* 146, pp. 530–533.

Lawrence, Neil and Aapo Hyvärinen (2005). "Probabilistic non-linear principal component analysis with Gaussian process latent variable models." In: *Journal of machine learning research* 6.11.

Lee, J.M. (2012). *Introduction to Smooth Manifolds*. 2nd. Vol. 218. Graduate Texts in Mathematics. Springer-Verlag New York. DOI: 10.1007/978-1-4419-9982-5.

Lee, John M (2018). *Introduction to Riemannian manifolds*. Vol. 176. Springer.

Lee, Juyong, In-Ho Lee, InSuk Joung, Jooyoung Lee, and Bernard R. Brooks (2017). "Finding multiple reaction pathways via global optimization of action". In: *Nature Communications* 8.1, p. 15443.

Levina, Elizaveta and Peter Bickel (2004). "Maximum likelihood estimation of intrinsic dimension". In: *Advances in neural information processing systems* 17.

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

Li, Bo, Yan-Rui Li, and Xiao-Long Zhang (2019). "A survey on Laplacian eigenmaps based manifold learning methods". In: *Neurocomputing* 335, pp. 336–351. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2018.06.077. URL: https://www.sciencedirect.com/science/article/pii/S0925231218312645.

Li, Gene, Junbo Li, Anmol Kabra, Nati Srebro, Zhaoran Wang, and Zhuoran Yang (2022). "Exponential family model-based reinforcement learning via score matching". In: *Advances in Neural Information Processing Systems* 35, pp. 28474–28487.

Li, Jun, Li Fuxin, and Sinisa Todorovic (2020). "Efficient riemannian optimization on the stiefel manifold via the cayley transform". In: *arXiv preprint arXiv:2002.01113*.

Li, Jun-Bao, Jeng-Shyang Pan, and Shu-Chuan Chu (2008). "Kernel class-wise locality preserving projection". In: *Information Sciences* 178.7, pp. 1825–1835. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2007.12.001. URL: https://www.sciencedirect.com/science/article/pii/S0020025507005658.

Li, Xihui, Zhongjian Qiao, Aicheng Gong, Jiafei Lyu, Chenghui Yu, Jiangpeng Yan, and Xiu Li (2022). "PRAG: Periodic Regularized Action Gradient for Efficient Continuous Control". In: *Pacific Rim International Conference on Artificial Intelligence*. Springer, pp. 106–119.

Liang, Senwei, Aditya N Singh, Yuanran Zhu, David T Limmer, and Chao Yang (2023). "Probing reaction channels via reinforcement learning". In: *Machine Learning: Science and Technology* 4.4, p. 045003.

Lillicrap, Timothy P., Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra (2019). *Continuous control with deep reinforcement learning*. arXiv: 1509.02971 [cs.LG].

Lin, Tong and Hongbin Zha (n.d.). "Riemannian manifold learning". In: *IEEE transactions on pattern analysis and machine intelligence* 30.5 (), pp. 796–809.

Lindorff-Larsen, Kresten, Stefano Piana, Ron O Dror, and David E Shaw (2011). "How fast-folding proteins fold". In: *Science* 334.6055, pp. 517–520.

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

Lipman, Yaron, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat (2024). "Flow Matching Guide and Code". In: *arXiv preprint arXiv:2412.06264*.

Lu, Cheng, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu (2023). "Contrastive Energy Prediction for Exact Energy-Guided Diffusion Sampling in Offline Reinforcement Learning". In: *arXiv preprint arXiv:2304.12824*.

Lucas, James, George Tucker, Roger Grosse, and Mohammad Norouzi (2019). "Understanding posterior collapse in generative latent variable models". In.

Ma, Yi, Doris Tsao, and Heung-Yeung Shum (2022). "On the principles of parsimony and self-consistency for the emergence of intelligence". In: *Frontiers of Information Technology & Electronic Engineering* 23.9, pp. 1298–1323.

Maier, James A., Carmenza Martinez, Koushik Kasavajhala, Lauren Wickstrom, Kevin E. Hauser, and Carlos Simmerling (2015). "ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB". In: *Journal of Chemical Theory and Computation* 11.8, pp. 3696–3713. DOI: 10.1021/acs.jctc.5b00255.

Marr, David and Ellen Hildreth (1980). "Theory of edge detection". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207.1167, pp. 187–217.

McInnes, Leland, John Healy, and James Melville (2018). "Umap: Uniform manifold approximation and projection for dimension reduction". In: *arXiv preprint arXiv:1802.03426*.

Monera, Maria G, A Montesinos-Amilibia, and Esther Sanabria-Codesal (2014). "The Taylor expansion of the exponential map and geometric applications". In: *Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas* 108.2, pp. 881–906.

Mullen, Ryan Gotchy, Joan-Emma Shea, and Baron Peters (2015). "Easy transition path sampling methods: Flexible-length aimless shooting and permutation shooting". In: *Journal of chemical theory and computation* 11.6, pp. 2421–2428.

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

Nie, Yuanbo and Eric C Kerrigan (2025). "Reliable Solution to Dynamic Optimization Problems using Integrated Residual Regularized Direct Collocation". In: *arXiv preprint arXiv:2503.09123*.

Noé, Frank, Simon Olsson, Jonas Köhler, and Hao Wu (2019). "Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning". In: *Science* 365.6457, eaaw1147.

Noé, Frank, Hao Wu, Jan-Hendrik Prinz, and Nuria Plattner (2013). "Projected and hidden Markov models for calculating kinetics and metastable states of complex molecules". In: *The Journal of chemical physics* 139.18.

Onsager, L. and S. Machlup (Sept. 1953). "Fluctuations and Irreversible Processes". In: *Phys. Rev.* 91 (6), pp. 1505–1512. DOI: 10.1103/PhysRev.91.1505.

Oue, Shingo (1996). "ON ASYMPTOTICS OF LOCAL PRINCIPAL COMPONENT ANALYSIS". In: *Hitotsubashi journal of commerce and management*, pp. 1–11.

Papanicolaou, George (1977). "Martingale approach to some limit theorems". In: *Papers from the Duke Turbulence Conference, Duke Univ., Durham, NC, 1977*.

Parthasarathy, Arjun, Nimit Kalra, Rohun Agrawal, Yann LeCun, Oumayma Bounou, Pavel Izmailov, and Micah Goldblum (2025). "Closing the Train-Test Gap in World Models for Gradient-Based Planning". In: *arXiv preprint arXiv:2512.09929*.

Pham, Tuan Q, LJ v Vliet, and Klamer Schutte (2008). "Robust super-resolution by minimizing a gaussian-weighted l2 error norm". In: *Journal of Physics: Conference Series*. Vol. 124. 1. IOP Publishing, p. 012037.

Pinneri, Cristina, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius (2021). "Sample-efficient cross-entropy method for real-time planning". In: *Conference on Robot Learning*. PMLR, pp. 1049–1065.

Piovesan, Jorge L and Herbert G Tanner (2009). "Randomized model predictive control for robot navigation". In: *2009 IEEE International Conference on Robotics and Automation*. IEEE, pp. 94–99.

Prinz, Jan-Hendrik, Hao Wu, Marco Sarich, Bettina Keller, Martin Senne, Martin Held, John D Chodera, Christof Schütte, and Frank Noé (2011). "Markov models of molecular kinetics: Generation and validation". In: *The Journal of chemical physics* 134.17.

Psenka, Michael et al. (2025a). "Onsager–Machlup Interpolation for Generative Protein Dynamics". In: *International Conference on Machine Learning*.

— (2025b). "Onsager–Machlup Interpolation for Latent Image Dynamics". In: *Advances in Neural Information Processing Systems*.

Psenka, Michael and Pieter Abbeel (2024). "Learning a Diffusion Model Policy from Rewards via Q-Score Matching". In: *International Conference on Machine Learning*.

Psenka, Michael, Yann LeCun, et al. (2026). *Parallel Long-Horizon Planning over Learned World Models via Gradient Surgery*.

Psenka, Michael, Druv Pai, Vishal Raman, Shankar Sastry, and Yi Ma (2024). "Representation learning via manifold flattening and reconstruction". In: *Journal of Machine Learning Research* 25.132, pp. 1–47.

Qiao, Hong, Peng Zhang, Bo Zhang, and Suiwu Zheng (n.d.). "Learning an intrinsic-variable preserving manifold for dynamic visual tracking". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 40.3 (), pp. 868–880.

Qu, Qing, Yuexiang Zhai, Xiao Li, Yuqian Zhang, and Zhihui Zhu (n.d.). "Geometric analysis of nonconvex optimization landscapes for overcomplete learning". In: *International Conference on Learning Representations*.

Radford, Alec, Luke Metz, and Soumith Chintala (2015). "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434*.

Reuss, Moritz, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov (2023). "Goal-conditioned imitation learning using score-based diffusion policies". In: *arXiv preprint arXiv:2304.02532*.

Rezende, Danilo Jimenez and Fabio Viola (2018). "Taming vaes". In: *arXiv preprint arXiv:1810.00597*.

Robbins, Herbert and Sutton Monro (1951). "A stochastic approximation method". In: *The annals of mathematical statistics*, pp. 400–407.

Rogal, Jutta, Elia Schneider, and Mark E Tuckerman (2019). "Neural-network-based path collective variables for enhanced sampling of phase transformations". In: *Physical Review Letters* 123.24, p. 245701.

Rojo, Alberto G, Alberto Rojo, and Anthony Bloch (2018). *The principle of least action: History and physics*. Cambridge University Press.

Rombach, Robin, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer (2022). "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, pp. 234–241.

Rose, Dominic C, Jamie F Mair, and Juan P Garrahan (2021). "A reinforcement learning approach to rare trajectory sampling". In: *New Journal of Physics* 23.1, p. 013013.

Rotskoff, Grant M, Andrew R Mitchell, and Eric Vanden-Eijnden (2022). "Active importance sampling for variational objectives dominated by rare events: Consequences for optimization and generalization". In: *Mathematical and Scientific Machine Learning*. PMLR, pp. 757–780.

Roweis, Sam T and Lawrence K Saul (2000). "Nonlinear dimensionality reduction by locally linear embedding". In: *science* 290.5500, pp. 2323–2326.

Rubinstein, Reuven Y and Dirk P Kroese (2004). *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media.

Rybkin, Oleh, Chuning Zhu, Anusha Nagabandi, Kostas Daniilidis, Igor Mordatch, and Sergey Levine (2021). "Model-based reinforcement learning via latent-space collocation". In: *International Conference on Machine Learning*. PMLR, pp. 9190–9201.

Saito, Shinnosuke and Takashi Matsubara (2025). "Image Interpolation with Score-based Riemannian Metrics of Diffusion Models". In: *arXiv preprint arXiv:2504.20288*.

Samuel, Dvir, Rami Ben-Ari, Nir Darshan, Haggai Maron, and Gal Chechik (2023). "Norm-guided latent space exploration for text-to-image generation". In: *Advances in Neural Information Processing Systems* 36, pp. 57863–57875.

Sarafian, Elad, Shai Keynan, and Sarit Kraus (2021). "Recomposing the reinforcement learning building blocks with hypernetworks". In: *International Conference on Machine Learning*. PMLR, pp. 9301–9312.

Schmidt, Wouter F, Martin A Kraaijveld, Robert PW Duin, et al. (1992). "Feed forward neural networks with random weights". In: *International conference on pattern recognition*. IEEE Computer Society Press, pp. 1–1.

Schreiner, Mathias, Ole Winther, and Simon Olsson (2023). "Implicit transfer operator learning: Multiple time-resolution models for molecular dynamics". In: *Advances in Neural Information Processing Systems* 36, pp. 36449–36462.

Schwartz, Ariel and Ronen Talmon (n.d.). "Intrinsic isometric manifold learning with application to localization". In: *SIAM Journal on Imaging Sciences* 12.3 (), pp. 1347–1391.

Seong, Kiyoung, Seonghyun Park, Seonghwan Kim, Woo Youn Kim, and Sungsoo Ahn (2024). *Transition Path Sampling with Improved Off-Policy Training of Diffusion Path Samplers*. arXiv: 2405.19961 [cs.LG].

Shamir, Adi, Odelia Melamed, and Oriel BenShmuel (2021). "The dimpled manifold model of adversarial examples in machine learning". In: *arXiv preprint arXiv:2106.10151*.

Shen, Liao, Tianqi Liu, Huiqiang Sun, Xinyi Ye, Baopu Li, Jianming Zhang, and Zhiguo Cao (2024). "Dreammover: Leveraging the prior of diffusion models for image interpolation with large motion". In: *European Conference on Computer Vision*. Springer, pp. 336–353.

Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, pp. 484–489.

Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. (2017). "Mastering chess and shogi by self-play with a general reinforcement learning algorithm". In: *arXiv preprint arXiv:1712.01815*.

Silver, David, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller (2014). "Deterministic policy gradient algorithms". In: *International conference on machine learning*. Pmlr, pp. 387–395.

Singh, Aditya N and David T Limmer (2023). *Variational deep learning of equilibrium transition path ensembles*.

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

Šípka, Martin, Andreas Erlebach, and Lukáš Grajciar (2023). "Constructing Collective Variables Using Invariant Learned Representations". In: *Journal of Chemical Theory and Computation* 19.3, pp. 887–901. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.2c00729.

Sobieraj, Marcin and Piotr Setny (2022). "Granger Causality Analysis of Chignolin Folding". In: *Journal of Chemical Theory and Computation* 18.3, pp. 1936–1944. DOI: 10.1021/acs.jctc.1c00945.

Sohl-Dickstein, Jascha, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli (2015). "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International Conference on Machine Learning*. PMLR, pp. 2256–2265.

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

Suh, HJ, Glen Chou, Hongkai Dai, Lujie Yang, Abhishek Gupta, and Russ Tedrake (2023). "Fighting Uncertainty with Gradients: Offline Reinforcement Learning via Diffusion Score Matching". In: *arXiv preprint arXiv:2306.14079*.

Sultan, Mohammad M and Vijay S Pande (2018). "Automated design of collective variables using supervised machine learning". In: *The Journal of Chemical Physics* 149.9, p. 94106. DOI: 10.1063/1.5029972.

Sun, Lixin, Jonathan Vandermause, Simon Batzner, Yu Xie, David Clark, Wei Chen, and Boris Kozinsky (2022). *Multitask machine learning of collective variables for enhanced sampling of rare events*.

Sünderhauf, Niko, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, et al. (2018). "The limits and potentials of deep learning for robotics". In: *The International journal of robotics research* 37.4-5, pp. 405–420.

Sutton, Richard S, David McAllester, Satinder Singh, and Yishay Mansour (1999). "Policy gradient methods for reinforcement learning with function approximation". In: *Advances in neural information processing systems* 12.

Sutton, Richard S. and Andrew G. Barto (2018). *Reinforcement Learning: An Introduction*. Second. The MIT Press. URL: http://incompleteideas.net/book/the-book-2nd.html.

Swendsen, Robert H and Jian-Sheng Wang (1986). "Replica Monte Carlo simulation of spin-glasses". In: *Physical review letters* 57.21, p. 2607.

Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus (2013). "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199*.

Takács, Gábor and Domonkos Tikk (2012). "Alternating least squares for personalized ranking". In: *Proceedings of the sixth ACM conference on Recommender systems*, pp. 83–90.

Tamimi, Jasem and Pu Li (2009). "Nonlinear model predictive control using multiple shooting combined with collocation on finite elements". In: *IFAC Proceedings Volumes* 42.11, pp. 703–708.

Tassa, Yuval, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. (2018). "Deepmind control suite". In: *arXiv preprint arXiv:1801.00690*.

Teed, Zachary and Jia Deng (2020). "Raft: Recurrent all-pairs field transforms for optical flow". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, pp. 402–419.

Tenenbaum, Joshua B, Vin de Silva, and John C Langford (2000). "A global geometric framework for nonlinear dimensionality reduction". In: *science* 290.5500, pp. 2319–2323.

Thrun, Sebastian, Knut Möller, and Alexander Linden (1990). "Planning with an adaptive world model". In: *Advances in neural information processing systems* 3.

Tiwary, Pratyush and Michele Parrinello (2013). "From metadynamics to dynamics". In: *Physical review letters* 111.23, p. 230602.

Torrie, Glenn M and John P Valleau (1977). "Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling". In: *Journal of computational physics* 23.2, pp. 187–199.

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

Tunyasuvunakool, Saran, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa (2020). "dm control: Software and tasks for continuous control". In: *Software Impacts* 6, p. 100022. ISSN: 2665-9638. DOI: https://doi.org/10.1016/j.simpa.2020.100022. URL: https://www.sciencedirect.com/science/article/pii/S2665963820300099.

Vahdat, Arash, Karsten Kreis, and Jan Kautz (2021). "Score-based generative modeling in latent space". In: *Advances in Neural Information Processing Systems* 34, pp. 11287–11302.

Valevski, Dani, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter (2024). "Diffusion models are real-time game engines". In: *arXiv preprint arXiv:2408.14837*.

Valsson, Omar and Michele Parrinello (2014). "Variational approach to enhanced sampling and free energy calculations". In: *Physical review letters* 113.9, p. 090601.

Van Den Oord, Aaron, Oriol Vinyals, et al. (2017). "Neural discrete representation learning". In: *Advances in neural information processing systems* 30.

Van der Maaten, Laurens and Geoffrey Hinton (2008). "Visualizing data using t-SNE." In: *Journal of machine learning research* 9.11.

Vanden-Eijnden, Eric et al. (2006). "Towards a theory of transition paths". In: *Journal of statistical physics* 123.3, pp. 503–523.

— (2010). "Transition-path theory and path-finding algorithms for the study of rare events." In: *Annual review of physical chemistry* 61, pp. 391–420.

Vanden-Eijnden, Eric and Matthias Heymann (2008). "The geometric minimum action method for computing minimum energy paths". In: *The Journal of chemical physics* 128.6.

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

Vandereycken, Bart (2013). "Low-rank matrix completion by Riemannian optimization". In: *SIAM Journal on Optimization* 23.2, pp. 1214–1236.

Vidal, Rene, Yi Ma, and Shankar Sastry (2005). "Generalized principal component analysis (GPCA)". In: *IEEE transactions on pattern analysis and machine intelligence* 27.12, pp. 1945–1959.

Vincent, Pascal (2011). "A connection between score matching and denoising autoencoders". In: *Neural computation* 23.7, pp. 1661–1674.

Von Stryk, Oskar (1993). "Numerical solution of optimal control problems by direct collocation". In: *Optimal Control: Calculus of Variations, Optimal Control Theory and Numerical Methods*. Springer, pp. 129–143.

Wakin, Michael B, David L Donoho, Hyeokho Choi, and Richard G Baraniuk (2005). "The multiscale structure of non-differentiable image manifolds". In: *Wavelets XI*. Vol. 5914. SPIE, pp. 413–429.

Wang, Zhendong, Jonathan J Hunt, and Mingyuan Zhou (2022). "Diffusion policies as an expressive policy class for offline reinforcement learning". In: *arXiv preprint arXiv:2208.06193*.

Wang, Zhou, Eero P Simoncelli, and Alan C Bovik (2003). "Multiscale structural similarity for image quality assessment". In: *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. Vol. 2. Ieee, pp. 1398–1402.

Watson, Daniel, William Chan, Ricardo Martin-Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi (2022). "Novel view synthesis with diffusion models". In: *arXiv preprint arXiv:2210.04628*.

Welling, Max and Yee W Teh (2011). "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688.

Werbos, Paul J (2002). "Backpropagation through time: what it does and how to do it". In: *Proceedings of the IEEE* 78.10, pp. 1550–1560.

Williams, Grady, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou (2016). "Aggressive driving with model predictive path integral control". In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 1433–1440.

Wu, Philipp, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel (2022). "Daydreamer: World Models for Physical Robot Learning". In: *Proceedings of Machine Learning Research*.

Xu, Jian-Wu, António RC Paiva, Il Park, and Jose C Principe (2008). "A reproducing kernel Hilbert space framework for information-theoretic learning". In: *IEEE Transactions on Signal Processing* 56.12, pp. 5891–5902.

Xu, Pan, Jinghui Chen, Difan Zou, and Quanquan Gu (2018). "Global convergence of Langevin dynamics based algorithms for nonconvex optimization". In: *Advances in Neural Information Processing Systems* 31.

Yan, Jin H, Richard N Hinrichs, V Gregory Payne, and Jerry R Thomas (2000). "Normalized jerk: a measure to capture developmental characteristics of young girls' overarm throwing". In: *Journal of Applied Biomechanics* 16.2, pp. 196–203.

Yang, Zhaoyuan, Zhengyang Yu, Zhiwei Xu, Jaskirat Singh, Jing Zhang, Dylan Campbell, Peter Tu, and Richard Hartley (2023). "Impus: Image morphing with perceptually-uniform sampling using diffusion models". In: *arXiv preprint arXiv:2311.06792*.

Yu, Yaodong, Sam Buchanan, Druv Pai, Tianzhe Chu, Ziyang Wu, Shengbang Tong, Benjamin Haeffele, and Yi Ma (2024). "White-box transformers via sparse rate reduction". In: *Advances in Neural Information Processing Systems* 36.

Yu, Yaodong, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma (2020). "Learning diverse and discriminative representations via the principle of maximal coding rate reduction". In: *Advances in Neural Information Processing Systems* 33, pp. 9422–9434.

Zhai, Yuexiang, Hermish Mehta, Zhengyuan Zhou, and Yi Ma (n.d.). "Understanding l4-based dictionary learning: Interpretation, stability, and robustness". In: *International conference on learning representations*.

Zhai, Yuexiang, Zitong Yang, Zhenyu Liao, John Wright, and Yi Ma (2020). "Complete dictionary learning via l 4-norm maximization over the orthogonal group". In: *The Journal of Machine Learning Research* 21.1, pp. 6622–6689.

Zhang, Junping, Stan Z Li, and Jue Wang (n.d.). "Manifold learning and applications in recognition". In: *Intelligent multimedia processing with soft computing* (), pp. 281–300.

Zhang, Kaiwen, Yifan Zhou, Xudong Xu, Bo Dai, and Xingang Pan (2024). "Diffmorpher: Unleashing the capability of diffusion models for image morphing". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7912–7921.

Zhang, Peng, Hong Qiao, and Bo Zhang (2011). "An improved local tangent space alignment method for manifold learning". In: *Pattern Recognition Letters* 32.2, pp. 181–189.

Zhang, Richard, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang (2018). "The unreasonable effectiveness of deep features as a perceptual metric". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595.

Zhang, Zhengwu, Eric Klassen, and Anuj Srivastava (2013). "Gaussian blurring-invariant comparison of signals and images". In: *IEEE Transactions on Image Processing* 22.8, pp. 3145–3157.

Zhang, Zhenyue and Hongyuan Zha (2003). "Nonlinear dimension reduction via local tangent space alignment". In: *Intelligent Data Engineering and Automated Learning: 4th International Conference, IDEAL 2003, Hong Kong, China, March 21-23, 2003. Revised Papers 4*. Springer, pp. 477–481.

Zhao, Hanyang, Wenpin Tang, and David Yao (2024). "Policy optimization for continuous reinforcement learning". In: *Advances in Neural Information Processing Systems* 36.

Zheng, Nanning, Jianru Xue, Nanning Zheng, and Jianru Xue (n.d.). "Manifold learning". In: *Statistical learning and pattern analysis for image and video processing* (), pp. 87–119.

Zheng, PengFei, Yonggang Zhang, Zhen Fang, Tongliang Liu, Defu Lian, and Bo Han (2024). "Noisediffusion: Correcting noise for image interpolation with diffusion models beyond spherical linear interpolation". In: *The Twelfth International Conference on Learning Representations*.

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

Zhou, Gaoyue, Hengkai Pan, Yann LeCun, and Lerrel Pinto (2024). "Dino-wm: World models on pre-trained visual features enable zero-shot planning". In: *arXiv preprint arXiv:2411.04983*.

Ziakas, Christos, Amir Bar, and Alessandra Russo (2026). "Grounding Generated Videos in Feasible Plans via World Models". In: *arXiv*. arXiv: 2602.01960 [cs.CV].