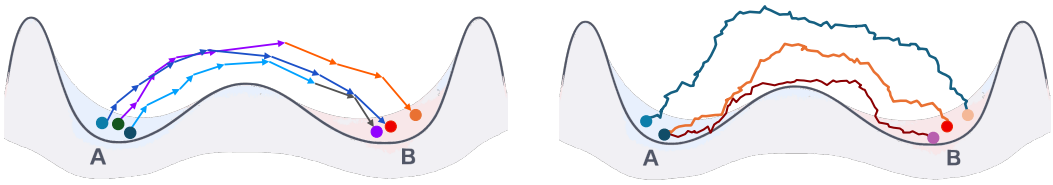
ACTION LEARNING FOR SCIENTIFIC SIMULATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding and controlling how systems evolve is central to science and engineering, where many problems are defined on trajectories rather than individual states. Existing learned methods often predict local dynamics or generate trajectories, while representative path inference, mechanism steering and constrained trajectory optimization require an objective on complete paths. We introduce **LEAP** (**L**earned **E**nergy on **A**ction **P**aths), which treats the stochastic action itself as the learning target and recovers it directly from trajectory samples. LEAP decomposes the unknown action to an evaluable reference process and learns the remaining path space correction from complete trajectories, yielding a differentiable objective on paths. Once learned, the same action can be reused for path ranking, representative path inference, mechanism tilting, constraints and sampling. Across stochastic transition, orbital and molecular systems, LEAP recovers physical drift from terminally selected trajectories, represents a non-additive action correction induced by a persistent latent regime and reuses a fixed molecular action for completion, basin preferences, angular constraints and temperature transfer. Code is available here.

1 FROM TRAJECTORY SAMPLES TO PATH INFERENCE



(a) **Local dynamics learning.** One-step transition laws induce a likelihood over complete trajectories.

(b) **Trajectory action learning.** LEAP learns relative path costs from complete trajectory comparisons.

Figure 1: **Local dynamics learning versus trajectory action learning.** Local models learn short-range transition laws that induce a path likelihood, whereas LEAP learns an action from complete trajectories while retaining structured local dynamics when appropriate.

Understanding how systems move between states requires reasoning about the trajectories that connect them. In molecular dynamics, two conformations may be connected by several transition routes that visit different intermediate configurations (Bolhuis et al., 2002; Shaw et al., 2010); in robotics, agents act through sequences of decision (Kalakrishnan et al., 2011; Janner et al., 2022). In these settings, the object of interest is often the trajectory itself: which paths realize a transition, which mechanisms are plausible and which paths remain likely under perturbations or constraints.

Trajectory-level questions require objectives on complete paths. To infer representative transitions or impose new preferences, we need an objective that assigns probabilities or costs to trajectories. When the dynamics are known, stochastic action functionals provide such an objective. In statistical physics and stochastic control, actions such as the Onsager-Machlup functional, Freidlin-Wentzell rate functional and path-integral control costs define probabilities or costs on path space (Onsager & Machlup, 1953; Freidlin & Wentzell, 2012; Kappen, 2005; Adib, 2007). Once the action is available, we can use it for downstream tasks like constrained inference and sampling.

In many applications, however, we do not know the action. We instead observe trajectories from simulation or experiment, like molecular simulation traces, state transitions, policy rollouts or logs of a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

stochastic process (Lindorff-Larsen et al., 2011; Shaw et al., 2010; Glielmo et al., 2021). Classical trajectory inference methods typically assume a known action, while learned methods often predict local dynamics, learn state or trajectory representations or generate trajectories directly (Chen et al., 2019; Li et al., 2020; Song et al., 2021; Jing et al., 2024b). These objects are useful, but they do not directly give the reusable path objective needed for downstream trajectory inference. We therefore call the task of recovering an evaluable action from trajectory samples *action learning*.

Existing learned methods often model dynamics locally by fitting one-step transition laws and using the resulting local model to evaluate complete trajectories (Chua et al., 2018). This requirement can become restrictive when trajectories are selected by global events, e.g., successful transitions or task level constraints. A local predictor trained on such data estimates the one-step conditionals of the selected process, which can differ from the underlying dynamics. Errors in local transition log likelihoods can also accumulate along a path and then distort the induced optimization landscape. Since path-level quantities do not have to decompose into independent local decisions, optimizing locally plausible segments can still produce a poor local minimum; Fig. 1 shows this distinction.

We propose **LEAP** (**L**earned **E**nergy on **A**ction **P**aths), a framework for learning a stochastic action directly from trajectory samples. LEAP treats the action itself as the learning target, rather than a local predictor or trajectory generator. It anchors the unknown action to a tractable reference process and learns the remaining correction from complete trajectories, yielding an evaluable objective on path space. In structured settings, this correction recovers changes in the underlying dynamics; more generally, it can capture nonadditive interactions across transitions. Once learned, the action can be reused with preferences that were not specified during training. For example, a reward for visiting an intermediate basin can favor a different transition route, while a geometric penalty can discourage paths outside admissible regions.

Our approach applies broadly to trajectories from dynamical systems, including molecular simulations, stochastic physical systems and logged sequential processes (Shaw et al., 2010; Kalakrishnan et al., 2011; Janner et al., 2022). We evaluate LEAP across controlled physical systems, a 31-body orbital benchmark with 186-dimensional states and alanine dipeptide molecular benchmark, allowing us to test physical recovery, nonadditive dependence across time and reuse across inference objectives.

Our contributions are as follows: (i) we formulate *action learning* as learning a reusable objective directly on complete trajectories; (ii) we introduce LEAP, which learns this action from trajectory samples by correcting a tractable reference process using complete path comparisons; (iii) we prove population action recovery up to context constants and show that persistent latent regimes can induce nonadditive corrections that transitionwise objectives cannot represent; and (iv) we demonstrate physical drift recovery under terminal selection, recovery of a nonadditive orbital action and reuse across molecular completion, temperature transfer, preferences and constraints.

1.1 Related work

Sampling trajectories from known dynamics. Classical trajectory-sampling methods usually assume access to the dynamics, energy landscape or action. When trajectories of interest occur often, direct simulation can be sufficient. When relevant paths are rare, e.g., because an energy barrier separates metastable states, one can search for or bias toward rare paths using the known model. The string method and related path optimization methods evolve discretized curves on a known energy landscape to find transition pathways (E et al., 2002). Minimum-action methods optimize an action functional derived from the governing stochastic dynamics (Freidlin & Wentzell, 2012). Other methods optimize transition structures with respect to a force field or potential energy surface, including classical, learned or diffusion-assisted energies (Holdijk et al., 2023; Yuan et al., 2024; Du et al., 2024; Raja et al., 2025; Amin et al., 2025). Path integral control and sampling-based control methods define trajectory objectives from controlled dynamics or costs (Kappen, 2005; Todorov, 2009; Williams et al., 2017), providing methods to infer trajectory inference when the action, force field or cost is available. LEAP targets the complementary setting, where trajectories are observed but the trajectory objective must be learned.

Learning from trajectories. Learned trajectory models usually target evolution or generation. Neural differential equation and latent dynamics models learn evolution from observations (Chen et al., 2019; Kidger et al., 2021; Hafner et al., 2019), while score-based, flow matching, diffusion

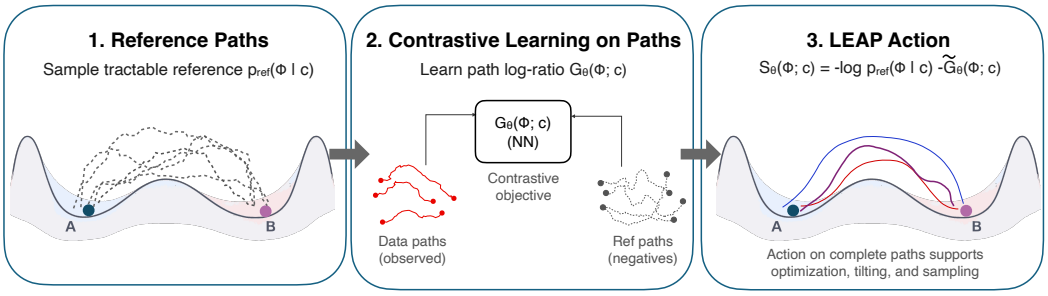


Figure 2: **LEAP.** Given data trajectories and a tractable reference process, LEAP learns a centered path space correction and combines it with the reference action to obtain $S_\theta(\Phi; c)$.

and action matching methods learn vector fields or scores that transport or generate samples from state, trajectory or marginal distributions (Song et al., 2021; Jing et al., 2024b;a; Neklyudov et al., 2023). These methods model evolution or generation, but they do not generally return the stochastic action needed for path optimization, mechanism tilting or Metropolis-Hastings correction. Energy-based models have also been used for planning by learning energies over individual state transitions and composing them along a trajectory (Du et al., 2020). Relatedly, reinforcement learning and imitation learning learn rewards or policies from demonstrations, e.g., maximum-entropy inverse reinforcement learning and adversarial imitation learning (Ziebart et al., 2008; Ho & Ermon, 2016; Fu et al., 2018; Nuti et al., 2023). LEAP also learns from trajectories, but targets a different object: a stochastic action on complete paths, i.e., the trajectory level objective used directly for optimization, control and sampling.

2 Learning Action Functionals from Trajectory Samples

LEAP learns a differentiable stochastic action $S_\theta(\Phi; c)$ directly from trajectory samples. The action assigns a scalar objective to each complete trajectory, with lower action corresponding to higher probability under the observed path distribution. LEAP requires trajectory samples for each context, but not the underlying drift, potential, or simulator likelihood. We construct the action by correcting an evaluable reference path measure, writing the unknown data action as a known reference action plus a learned correction estimated from observed and reference trajectories. The resulting objective can be evaluated and differentiated on new trajectories. See Alg. 1 for the algorithm.

2.1 Target action and training objective

Let $\Phi = (x_0, x_1, \ldots, x_N)$ denote a discretized trajectory with $t_k = k\Delta t$ and horizon $T = N\Delta t$. Let c denote the conditioning variables, e.g., endpoints, terminal regions, horizon or diffusion level. We observe trajectory samples $\Phi \sim p_{\text{data}}(\cdot \mid c)$ for each context.

Target action. Following stochastic action formalisms such as Onsager-Machlup and Freidlin-Wentzell theory, we use the negative log path density as the trajectory objective (Onsager & Machlup, 1953; Freidlin & Wentzell, 2012):

$$S^*(\Phi; c) = -\log p_{\text{data}}(\Phi \mid c) + \alpha(c), \tag{1}$$

where $\alpha(c)$ is constant across paths with the same context and therefore does not affect their relative action, including for path ranking or representative path optimization (§2.4).

Reference-based decomposition. Directly evaluating $p_{\text{data}}(\Phi \mid c)$ is generally not possible from trajectory samples. LEAP therefore introduces a tractable reference process with density $p_{\text{ref}}(\Phi \mid c)$. The reference gives an evaluable baseline action and trajectories from a known path distribution. LEAP learns the remaining correction needed to recover the data action from this reference.

Specifically, the correction is the path log density ratio

$$G^*(\Phi; c) = \log \frac{p_{\text{data}}(\Phi \mid c)}{p_{\text{ref}}(\Phi \mid c)}. \tag{2}$$

With $S_{\text{ref}}(\Phi; c) = -\log p_{\text{ref}}(\Phi \mid c)$, the target action is

$$S^*(\Phi; c) = \underbrace{S_{\text{ref}}(\Phi; c)}_{\text{reference action}} - \underbrace{G^*(\Phi; c)}_{\text{learned correction}} + \alpha(c). \tag{3}$$

We estimate G^* from comparisons between observed and reference trajectories and combine it with the reference action. The resulting S_θ is the learned stochastic action used for trajectory inference.

Learning the action correction. To estimate G^*, we compare observed trajectories with those drawn from the reference process (Gutmann & Hyvärinen, 2010). For each context c, we draw one observed trajectory $\Phi^+ \sim p_{\text{data}}(\cdot \mid c)$ and K reference trajectories $\Phi_k^- \sim p_{\text{ref}}(\cdot \mid c)$. We parameterize the correction by $G_\theta(\Phi; c)$ and optimize the setwise comparison objective

$$\mathcal{L}_K(\theta) = -\mathbb{E}\left[\log \frac{\exp G_\theta(\Phi^+; c)}{\exp G_\theta(\Phi^+; c) + \sum_{k=1}^{K} \exp G_\theta(\Phi_k^-; c)}\right]. \tag{4}$$

This objective has the desired correction as its population solution. Under the conditions in § A.1, every population minimizer satisfies

$$G_{\theta^*}(\Phi; c) = G^*(\Phi; c) + a(c), \tag{5}$$

for every $K \geq 1$. The context dependent term cancels from each comparison, so the objective identifies the trajectory dependent part of the correction.

Because the comparison objective leaves this context dependent constant undetermined, we fix the additive convention after training by centering the correction under the reference distribution,

$$\bar{G}_\theta(c) = \mathbb{E}_{\Phi \sim p_{\text{ref}}(\cdot|c)}\left[G_\theta(\Phi; c)\right], \qquad \widetilde{G}_\theta(\Phi; c) = G_\theta(\Phi; c) - \bar{G}_\theta(c). \tag{6}$$

The learned action is then

$$S_\theta(\Phi; c) = -\log p_{\text{ref}}(\Phi \mid c) - \widetilde{G}_\theta(\Phi; c). \tag{7}$$

When the correction is recovered, S_θ equals the data action up to a context dependent additive constant. Within a fixed context, $S_{\text{ref}} - G_\theta$ gives the same path rankings and optimization solutions.

2.2 Reference process

The reference defines the baseline path measure and the inductive bias of the learned action. This means we require a tractable path density, efficient sampling and sufficient probability mass over the trajectories represented in the data. While its drift can include an available nominal drift or force model, LEAP does not necessarily require one. For rare event trajectories, we condition the reference on the same event as the data so that both distributions contain trajectories selected by the same condition.

We use a discrete time Gaussian Markov reference because its transition law is analytic and parameterized directly by drift and diffusion. Under shared diffusion, differences in drift induce an exact path log likelihood ratio, giving the learned correction a direct dynamical interpretation. We define

$$x_{k+1} = x_k + b_{\text{ref}}(x_k, t_k; c)\Delta t + \sqrt{2D\Delta t}\,\xi_k, \qquad \xi_k \sim \mathcal{N}(0, I), \tag{8}$$

where b_{ref} is the reference drift and D is the diffusion coefficient. This induces the path density $p_{\text{ref}}(\Phi \mid c)$ and reference action $S_{\text{ref}}(\Phi; c)$. For endpoint or terminal region conditioning, p_{ref} denotes the corresponding conditional density over the interior knots; its normalizer contributes only a context dependent constant (see § A.3).

2.3 Local and global action corrections

When the data and reference processes are Gaussian Markov processes with the same diffusion coefficient and differ only in drift, their exact path log likelihood ratio decomposes over transitions. We use this structure when the drift difference is the quantity of interest. Let $v_k = (x_{k+1} - x_k)/\Delta t$ and let u_θ parameterize $b_{\text{data}} - b_{\text{ref}}$. The correction is

$$G_\theta(\Phi; c) = \sum_{k=0}^{N-1}\left[\underbrace{\frac{1}{2D}\, u_\theta(x_k, t_k; c)^\top \big(v_k - b_{\text{ref}}(x_k, t_k; c)\big)}_{\text{velocity alignment}} - \underbrace{\frac{1}{4D}\,\|u_\theta(x_k, t_k; c)\|^2}_{\text{normalization cost}}\right]\Delta t. \tag{9}$$

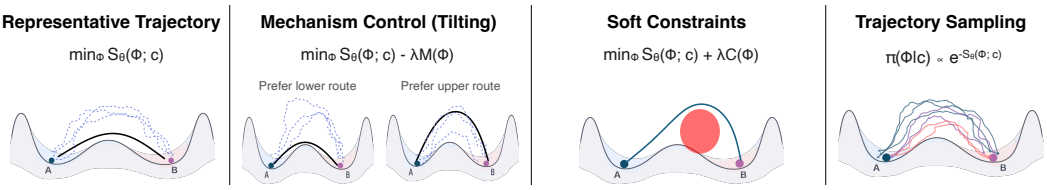


Figure 3: **Downstream trajectory inference with the learned action.** Once $S_\theta(\Phi; c)$ is learned, the same action supports representative path inference, mechanism control, constrained inference and sampling.

When the data and reference processes also share the initial law and time grid, and differ only through their drift, Eq. (9) is the exact path log likelihood ratio for $b_{\text{data}} = b_{\text{ref}} + u^*$; see §A.2. For stopped or padded trajectories, we sum only over active transitions as described in the appendix.

More generally, LEAP does not require the Gaussian drift parameterization. If the data to reference log density ratio factorizes over transitions, LEAP-Local learns a scalar correction g_θ to each transition and sums the resulting corrections over the path,

$$G_\theta^{\text{local}}(\Phi; c) = \sum_{k=0}^{N-1} g_\theta(x_k, x_{k+1}, t_k; c). \tag{10}$$

This factorization can fail when information is shared across the trajectory. For example, marginalizing a latent variable that remains fixed throughout a trajectory can couple multiple transitions. LEAP-Global therefore models the correction directly from the complete path,

$$G_\theta^{\text{global}}(\Phi; c) = r_\theta(\Phi; c). \tag{11}$$

LEAP-Hybrid combines both the two parameterizations,

$$G_\theta^{\text{hybrid}}(\Phi; c) = G_\theta^{\text{local}}(\Phi; c) + G_\theta^{\text{global}}(\Phi; c). \tag{12}$$

The global term captures interactions across transitions that cannot be written as an additive transition correction. When both terms are learned jointly, only their sum is identified, so interpreting the local term as a physical drift correction requires additional restrictions.

For conservative systems, we optionally parameterize the total drift as the gradient of a potential,

$$b_{\text{ref}}(x, t; c) + u_\theta(x, t; c) = -\nabla U_\theta(x; c). \tag{13}$$

2.4 Downstream trajectory inference

Because S_θ is an action on complete trajectories, we can reuse it after training. The same learned object can rank candidate paths, infer representative trajectories, incorporate mechanism preferences, impose soft constraints and define trajectory distributions. See §B.2 for details.

Trajectory ranking and representative inference. Given feasible trajectories $\Gamma(c)$, e.g., paths satisfying boundary conditions, we can rank candidate trajectories by their action or infer the lowest-action representative trajectory by solving

$$\Phi^*(c) = \arg\min_{\Phi \in \Gamma(c)} S_\theta(\Phi; c). \tag{14}$$

Mechanism control and constraints. We steer the representative path by adding trajectory-level functionals. Given a deired mechanism preference $M(\Phi)$ and a constraint penalty $C(\Phi)$, we solve

$$\arg\min_{\Phi \in \Gamma(c)} \Big[S_\theta(\Phi; c) - \lambda_M M(\Phi) + \lambda_C C(\Phi) \Big]. \tag{15}$$

Changing λ_M or λ_C changes the inference problem without retraining the model.

Compositional trajectory distributions. The learned action defines an unnormalized trajectory distribution. Given mechanism preferences M_j with weights α_j and penalties C_ℓ with weights λ_ℓ,

$$\pi(\Phi \mid c) \propto \exp\left(-S_\theta(\Phi; c) + \sum_j \alpha_j M_j(\Phi) - \sum_\ell \lambda_\ell C_\ell(\Phi) \right). \tag{16}$$

Algorithm 1 LEAP training

Require: dataset $\mathcal{D} = \{(\Phi_i^+, c_i)\}$, reference process $p_{\mathrm{ref}}(\Phi \mid c)$, reference count K

while not converged **do**
  sample minibatch $\{(\Phi_i^+, c_i)\}_{i=1}^B \subset \mathcal{D}$
  for $i = 1$ to B **do**
    sample $\{\Phi_{i,k}^-\}_{k=1}^K \sim p_{\mathrm{ref}}(\cdot \mid c_i)$
    evaluate $G_i^+ \leftarrow G_\theta(\Phi_i^+; c_i)$ and $G_{i,k}^- \leftarrow G_\theta(\Phi_{i,k}^-; c_i)$
  end for
  update θ by a gradient step on the minibatch estimate of Eq. (4)
end while
return learned action S_θ in Eq. (7)

Sampling from π gives a family of plausible paths weighted by the specified preferences/penalties.

Metropolis-Hastings (MH) sampling. At fixed context, S_θ defines an unnormalized distribution over paths. For a proposal kernel $q(\Phi' \mid \Phi)$ on $\Gamma(c)$, the MH acceptance probability is

$$\alpha(\Phi, \Phi') = \min\left\{1,\ \exp\big(-S_\theta(\Phi'; c) + S_\theta(\Phi; c)\big)\frac{q(\Phi \mid \Phi')}{q(\Phi' \mid \Phi)}\right\}. \tag{17}$$

For symmetric proposals, this reduces to $\alpha(\Phi, \Phi') = \min\left\{1,\ \exp\big(-S_\theta(\Phi'; c) + S_\theta(\Phi; c)\big)\right\}$. Any context-dependent additive constant in S_θ cancels in the acceptance ratio.

3 Why LEAP recovers an action

We give three guarantees for action learning. The first recovers the data action from the comparison objective, the second identifies the drift correction under the structured model, and the third bounds downstream inference error from action error. Full assumptions and proofs are in § A.

The comparison objective recovers the action. Fix a context c. Assume that the data and reference distributions have positive densities on the same support, the candidates are conditionally independent and the score class contains G^*. Every population minimizer of Eq. (4) satisfies

$$G_{\theta^*}(\Phi; c) = \log \frac{p_{\mathrm{data}}(\Phi \mid c)}{p_{\mathrm{ref}}(\Phi \mid c)} + a(c). \tag{18}$$

If $\mathbb{E}_{p_{\mathrm{ref}}}|G^*| < \infty$, reference centering then gives

$$S_{\theta^*}(\Phi; c) = S^*(\Phi; c) + \beta(c). \tag{19}$$

The context dependent constant keeps optimization solutions and MH ratios the same.

The structured correction recovers drift differences. For Euler-Maruyama dynamics with the same diffusion, time step, and initial distribution, changing the drift changes the path log likelihood by the structured correction in Eq. (9). Matching the unconditioned path distribution identifies the drift at states and times visited by the process. If data and reference trajectories are selected using the terminal event, this changes the path log likelihood ratio by condition-dependent constants. Recovering the unconditioned drift from selected trajectories needs variation across terminal conditions and the structured drift parameterization. See § 4.1 for the recovery; § A.2 and A.3 for derivations.

Action error controls later inference. Fix a context c and an inference domain $\Gamma(c)$. Suppose

$$\sup_{\Phi \in \Gamma(c)} |S_\theta(\Phi; c) - S^*(\Phi; c) - \beta(c)| \le \varepsilon.$$

For the same added functional H, let $F_\theta = S_\theta + H$ and $F^* = S^* + H$. If $\widehat{\Phi}$ is δ suboptimal for F_θ,

$$F^*(\widehat{\Phi}) - \inf_{\Phi \in \Gamma(c)} F^*(\Phi) \le 2\varepsilon + \delta. \tag{20}$$

If $\pi_\theta \propto e^{-F_\theta}$ and $\pi^* \propto e^{-F^*}$ have finite, positive normalizing constants, then

$$\left|\log \frac{\pi_\theta(\Phi \mid c)}{\pi^*(\Phi \mid c)}\right| \le 2\varepsilon. \tag{21}$$

Thus the same action error bounds both the objective value of an inferred trajectory and the change in its normalized trajectory density.

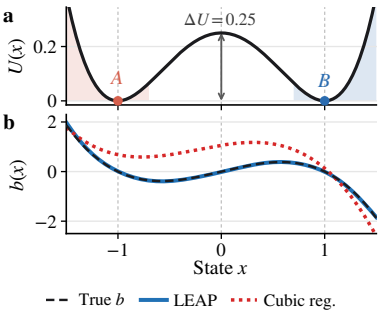


Figure 5: **Double well recovery.**

Method	Drift MSE ↓	True path cost ↓	
		$T = 4$	$T = 16$
Kramers-Moyal	0.813	—	—
Cubic velocity reg.	0.775	0.319	0.613
Zero field	0.0759	0.264	0.445
Geometric path	—	0.264	0.264
Analytic drift	0	0.253	0.250
LEAP	$\mathbf{3.30 \times 10^{-4}}$	**0.253**	**0.251**

Table 1: **Physical recovery and path inference.** Drift error and path cost under new endpoints and horizons.

4 Experiments

Our experiments evaluate four properties of action learning. We show that LEAP can recover physical dynamics from selected trajectories, capture dependencies across multiple transitions, stay accurate as the reference changes, and reuse the learned action under new physical conditions and inference requests. §B gives the full experimental details.

4.1 Recovering dynamics under terminal selection

We first show that LEAP recovers the underlying drift when training trajectories are selected by their final state. We use the overdamped diffusion $dX_t = b(X_t)\,dt + \sqrt{2D}\,dW_t$ with $b(x) = x - x^3$ and $U(x) = (x^2 - 1)^2/4$. In this double well setting, trajectories start at $x_0 = -1$ and are kept when $x_T \in [0.9, 1.1]$ at $T = 3$. We use 5,000 selected trajectories per seed with $D = 0.20$ and $\Delta t = 0.01$, and a Brownian reference conditioned on the same terminal window. Conditioning on the final state changes the distribution of one step increments, so local regression does not directly estimate the original drift. LEAP instead compares complete selected paths with the matched reference and learns the drift through the structured action correction.

Drift recovery. We compare LEAP with Kramers-Moyal, which estimates drift from binned one-step increments, and cubic velocity regression using $\{1, x, x^2, x^3\}$, which contains the exact drift $x - x^3$. LEAP achieves drift MSE $(3.3 \pm 2.7) \times 10^{-4}$, compared to 0.8132 ± 0.0030 for Kramers-Moyal, 0.7752 ± 0.0043 for cubic regression, and 0.0759 for the zero field (Fig. 5 and Table 1). Even though cubic regression can represent the exact drift, it fits the one-step dynamics of the terminally selected trajectories, which differ from the original drift.

Path inference under new endpoints. Next, we keep the learned potential fixed and change the endpoint from the training region to the saddle $x_T = 0$, and vary the horizon. At $T = 4$, LEAP gives true path cost 0.2533 ± 0.0001 versus the analytic value 0.2532. At $T = 16$, LEAP gives 0.2506 ± 0.0006 versus 0.2500, while cubic velocity regression gives 0.6126 ± 0.0079 and the geometric path gives 0.2637. The same learned action can still infer near optimal paths after changing both the endpoint and the horizon.

Reference choice. We retrain LEAP on the same double well data using different references. Changing the reference drift from 0 to $0.5b$ reduces correction NRMSE from 0.6641 to 0.2229, showing that a closer reference leaves a smaller correction to learn. See §D for the full sweep.

4.2 Recovering a nonadditive trajectory action

Next, we test whether LEAP can learn a correction that depends on several transitions. These settings may appear when an unobserved physical property is fixed during a trajectory, like an unknown mass in orbital dynamics, friction coefficient in Langevin dynamics or material parameter in a mechanical system, and its effect accumulates over the full path. We model this by sampling $z \in \{-1, +1\}$ once and change the solar mass to $m_\odot(1 + z\eta)$. The same unobserved z is fixed for the entire trajectory, so recovering the observed path likelihood needs combining information across time. We use a 31 body orbital system initialized from JPL HORIZONS, giving a 186 dimensional state space (Giorgini et al., 1997).

Conditional on z, the path density factorizes over transitions. After marginalizing z, the exact data to reference correction becomes

$$G^*(\Phi; c) = \log\left[\tfrac{1}{2}e^{\ell_{+1}(\Phi;c)} + \tfrac{1}{2}e^{\ell_{-1}(\Phi;c)}\right] - \ell_0(\Phi; c), \tag{22}$$

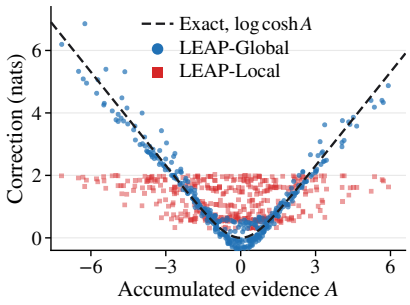


Figure 6: **Recovery of the nonadditive orbital correction.** LEAP-Global follows the exact $\log\cosh A$ dependence across held out trajectories.

Model	Input	$\mathrm{NRMSE}_{\Delta G}\downarrow$
LEAP-Local	One transition	$1.0013_{\pm 0.0015}$
Transition NCE	One transition	$1.0008_{\pm 0.0013}$
LEAP-Global	**Complete path + features**	$\mathbf{0.228}_{\pm \mathbf{0.011}}$

Table 2: **Ablation on trajectory information.** Models using one transition at a time remain near the constant prediction level, while the complete-path model reduces NRMSE.

Objective	Input	$\mathrm{NRMSE}_{\Delta G}\downarrow$
Categorical NRE	Transformer + features	$0.228_{\pm 0.015}$
Binary NRE	Path encoder	$1.00000_{\pm 3\times 10^{-5}}$
Binary NRE	**Transformer + features**	$\mathbf{0.175}_{\pm \mathbf{0.011}}$

Table 3: **Ablation on accumulated path summaries.** Adding the same path summaries reduces error for both binary and categorical ratio objectives.

where $\ell_z = \sum_k \ell_{z,k}$ and ℓ_0 uses the nominal solar mass. Defining $d_k = (\ell_{+1,k} - \ell_{-1,k})/2$ and $m_k = (\ell_{+1,k} + \ell_{-1,k})/2 - \ell_{0,k}$ gives

$$G^*(\Phi; c) = \sum_k m_k + \log\cosh\left(\sum_k d_k\right). \tag{23}$$

Its mixed derivatives satisfy $\partial^2 G^*/(\partial d_i \partial d_j) > 0$ for $i \neq j$, whereas a sum of separate transition corrections has zero mixed derivatives. Therefore no additive transition model can represent the exact correction; see §A.4.

Local and complete path models. We evaluate whether the correction can be recovered from individual transitions or need information from the whole trajectory. LEAP-Local uses the structured local correction from Eq. (10). As a baseline with different supervision, Transition NCE trains directly on data and reference transitions using the same transition inputs. LEAP-Global uses the full trajectory and features accumulated from the nominal gravitational dynamics (Table 2). These inputs do not have the hidden variable z or the exact correction G^*.

LEAP-Local and Transition NCE stay at the constant prediction level, with NRMSE 1.0013 and 1.0008, while LEAP-Global reaches 0.2280. This shows that the correction cannot be recovered from separate transitions and requires information across the trajectory. Consistent with this result, Fig. 6 shows that LEAP-Global follows the exact $\log\cosh A$ dependence and recovers the hidden mass perturbation as 1.02×10^{-4}, compared to the true 1.04×10^{-4}. As a trajectory diffusion comparison, GeoTDM (Han et al., 2024) is trained separately on the same data and reference paths, and the difference in its objectives gives AUROC 0.4999; see §E.5.

Accumulated transition features. We ablate these features while keeping a complete path model. Without them, binary NRE gives NRMSE $1.0000 \pm 3.0 \times 10^{-5}$. Adding them reduces error to 0.1746 ± 0.0113 with binary NRE and 0.2278 ± 0.0149 with categorical NRE (Table 3). Both objectives improve, showing that the accumulated information from Eq. (23) drives the decrease.

Scaling. At 42 bodies and 256 transitions, each path contains 64,764 coordinates. LEAP-Global reaches NRMSE 0.1107, while LEAP-Local stays at 1.0015. Thus, even as the system grows to more bodies and longer trajectories, LEAP-Global can accumulate the small effect of the hidden mass perturbation across transitions, while models that treat each transition separately still stay near the constant prediction baseline. See §E.3 for full sweep.

4.3 Reusing learned actions across conditions and objectives

In molecular simulation, trajectories may come from a high dimensional simulator but be analyzed in a small set of collective coordinates, i.e., the effective dynamics are not available in closed form. These coordinates are used to identify metastable states and and describe transition pathways. They are also used in enhanced sampling and restrained simulations, where one may require a path to pass through a particular basin or compare conformational pathways across temperatures. We therefore evaluate whether LEAP can learn one action in these reduced coordinates and reuse it across these different settings. We use alanine dipeptide represented by the backbone angles (ϕ, ψ) on the Ramachandran torus (Vargas et al., 2002). OpenMM Langevin simulations provide 169 reactive C7eq→C7ax trajectories at 500 K and 613 at 600 K (Eastman et al., 2023).

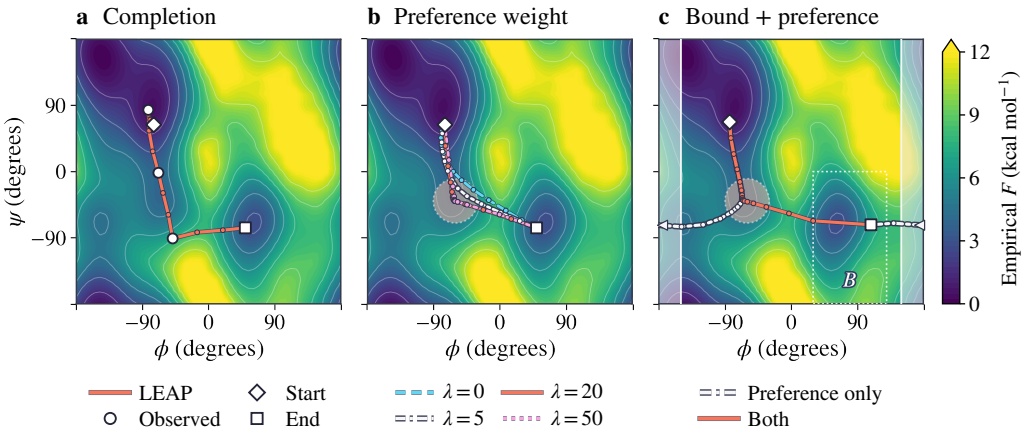


Figure 7: **Reuse of a fixed molecular action.** All panels use the same 600 K LEAP action. (a) Completion from endpoints and three interior observations. (b) Increasing the basin preference raises M from 0.23 to 0.81 for the shown path. (c) Adding $|\phi| \leq 150^\circ$ changes the route while preserving the terminal condition. Diamonds and squares are endpoints, dots are saved frames, and the background is empirical free energy.

Completion from partial observations. We keep the learned action fixed and optimize the missing states. When the endpoints and three interior states are observed, LEAP gives circular MSE 0.0931 ± 0.0016 at 600 K, compared to 0.1191 ± 0.0124 for conditional diffusion and 0.1247 ± 0.0105 for conditional flow matching. LEAP is trained only on complete trajectories, so the observed states are specified only at inference. The conditional generators, by contrast, are trained with completion masks. With endpoints only, the conditional generators get a lower reconstruction error. LEAP uses the same learned action in both cases, so the observation pattern can change without training a new model. Fig. 7a shows one example, with full results in §F.

Transfer across temperatures. We keep the learned correction fixed and use the diffusion, temperature and reference process of the target system. Training at 600 K gives mean within-context AUROC 0.9579 ± 0.0057 at 500 K, compared to 0.9403 ± 0.0099 for direct 500 K training. Training at 500 K gives 0.9719 ± 0.0068 at 600 K, compared to 0.9754 ± 0.0012 for direct 600 K training. The learned correction therefore remains useful after changing temperature.

Adding preferences and constraints. Next, using the same learned 600 K action, we change the inference objective. Increasing the basin preference weight from 0 to 20 increases the mean reward from 0.0658 to 0.2517, while the projected transition cost increases from 0.9389 to 0.9809 nats per active step. We also impose $|\phi| \leq 150^\circ$ while keeping the same basin preference. In Fig. 7c, the preference-only path enters the excluded angular region, while the constrained path stays within the bound and reaches the same endpoint. Thus, we can add preferences and geometric constraints without retraining the learned action.

5 Conclusion

In this work, LEAP treats the stochastic action as a reusable learning target for trajectory data. In the selected double well, the learned action recovers the underlying physical field and supports path inference under new endpoints and horizons. In the orbital benchmark, it captures complete path interactions induced by a persistent latent regime that additive transition models cannot do. In molecular coordinates, the same learned action enables completion, temperature transfer, basin preferences and angular constraints with fixed learned parameters. These settings show that learning an objective on path space enables a common interface between trajectory data and diverse downstream inference problems, including cases where the reduced dynamics are unavailable in closed form. Looking forward, we hope action learning encourages a broader view for trajectory modeling, in particular, shifting the focus to learning reusable objectives over complete paths.

Limitations. LEAP requires overlap between the reference and trajectory distribution; poor overlap increases finite sample error in the learned correction. While the structured correction in Eq. (9) is exact for Gaussian Markov processes with shared diffusion, state dependent diffusion requires more likelihood ratio terms. Finally, projecting onto reduced coordinates can also introduce memory and state dependent friction or diffusion (Ma et al., 2016; Vroylandt & Monmarché, 2022), so our alanine experiments learn an effective action in (ϕ, ψ) instead of the full molecular action.

Impact Statement

This paper advances machine learning methods for trajectory samples by learning reusable objectives on complete paths. Such methods may help analyze and steer dynamical systems in science, robotics, planning and other sequential domains. As with other learned objectives, they be used to steer systems toward harmful outcomes, so deployment requires domain validation and careful attention to the data.

Use of generative AI

In this work, we used generative AI tools for organizing and editing the paper, checking/iterating the proofs and improving their exposition and assisting with writing code. We did not use generative AI tools to propose or refine hypotheses, and synthetic data generation is not applicable to this work. We have reviewed all AI-assisted work. For example, we have checked the proof steps sequentially and wrote unit tests for any LLM-generated code.

References

Artur B. Adib. Stochastic actions for diffusive dynamics: Reweighting, sampling, and minimization. *The Journal of Physical Chemistry B*, 112(19):5910–5916, November 2007. ISSN 1520-5207. doi: 10.1021/jp0751458. URL http://dx.doi.org/10.1021/jp0751458.

Ishan Amin, Sanjeev Raja, and Aditi Krishnapriyan. Towards fast, specialized machine learning force fields: Distilling foundation models via energy hessians, 2025. URL https://arxiv.org/abs/2501.09009.

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

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations, 2019. URL https://arxiv.org/abs/1806.07366.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 4759–4770, Red Hook, NY, USA, 2018. Curran Associates Inc.

Yilun Du, Toru Lin, and Igor Mordatch. Model-based planning with energy-based models. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 374–383. PMLR, 30 Oct–01 Nov 2020. URL https://proceedings.mlr.press/v100/du20a.html.

Yuanqi Du, Michael Plainer, Rob Brekelmans, Chenru Duan, Frank Noe, Carla P Gomes, Alan Aspuru-Guzik, and Kirill Neklyudov. Doob's lagrangian: A sample-efficient variational approach to transition path sampling. *Advances in Neural Information Processing Systems*, 37:65791–65822, 2024.

Weinan E, Weiqing Ren, and Eric Vanden-Eijnden. String method for the study of rare events. *Physical Review B*, 66(5), August 2002. ISSN 1095-3795. doi: 10.1103/physrevb.66.052301. URL http://dx.doi.org/10.1103/PhysRevB.66.052301.

Peter Eastman, Raimondas Galvelis, Raúl P. Peláez, Charlles R. A. Abreu, Stephen E. Farr, Emilio Gallicchio, Anton Gorenko, Michael M. Henry, Frank Hu, Jing Huang, Andreas Krämer, Julien Michel, Joshua A. Mitchell, Vijay S. Pande, João PGLM Rodrigues, Jaime Rodriguez-Guerra, Andrew C. Simmonett, Sukrit Singh, Jason Swails, Philip Turner, Yuanqing Wang, Ivy Zhang, John D. Chodera, Gianni De Fabritiis, and Thomas E. Markland. Openmm 8: Molecular dynamics simulation with machine learning potentials, 2023. URL https://arxiv.org/abs/2310.03121.

Mark I. Freidlin and Alexander D. Wentzell. *Random Perturbations of Dynamical Systems*, volume 260 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, Heidelberg, 3 edition, 2012. ISBN 978-3-642-25847-3. doi: 10.1007/978-3-642-25847-3.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning, 2018. URL https://arxiv.org/abs/1710.11248.

J. D. Giorgini, D. K. Yeomans, A. B. Chamberlin, P. W. Chodas, R. A. Jacobson, M. S. Keesey, J. H. Lieske, S. J. Ostro, E. M. Standish, and R. N. Wimberly. JPL's On-Line Solar System Ephemeris and Data Service. In *Bulletin of the American Astronomical Society*, volume 28, pp. 1099, September 1997.

I. V. Girsanov. On transforming a certain class of stochastic processes by absolutely continuous substitution of measures. *Theory of Probability & Its Applications*, 5(3):285–301, 1960. doi: 10.1137/1105027. URL https://doi.org/10.1137/1105027.

Aldo Glielmo, Brooke E. Husic, Alex Rodriguez, Cecilia Clementi, Frank Noé, and Alessandro Laio. Unsupervised learning methods for molecular simulation data. *Chemical Reviews*, 121:9722 – 9758, 2021. URL https://api.semanticscholar.org/CorpusID:233743060.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/gutmann10a.html.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Jiaqi Han, Minkai Xu, Aaron Lou, Haotian Ye, and Stefano Ermon. Geometric trajectory diffusion models. *Advances in Neural Information Processing Systems*, 37:25628–25662, 2024.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning, 2016. URL https://arxiv.org/abs/1606.03476.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Lars Holdijk, Yuanqi Du, Ferry Hooft, Priyank Jaini, Berend Ensing, and Max Welling. Stochastic optimal control for collective variable free sampling of molecular transition paths. *Advances in Neural Information Processing Systems*, 36:79540–79556, 2023.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022. URL https://arxiv.org/abs/2205.09991.

Bowen Jing, Bonnie Berger, and Tommi Jaakkola. Alphafold meets flow matching for generating protein ensembles, 2024a. URL https://arxiv.org/abs/2402.04845.

Bowen Jing, Hannes Stärk, Tommi Jaakkola, and Bonnie Berger. Generative modeling of molecular dynamics trajectories. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 40534–40564. Curran Associates, Inc., 2024b. doi: 10.52202/079017-1282. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/478b06f60662d3cdc1d4f15d4587173a-Paper-Conference.pdf.

Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE international conference on robotics and automation*, pp. 4569–4574. IEEE, 2011.

H J Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(11):P11011–P11011, November 2005. ISSN 1742-5468. doi: 10.1088/1742-5468/2005/11/p11011. URL http://dx.doi.org/10.1088/1742-5468/2005/11/P11011.

Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. Neural sdes as infinite-dimensional gans. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5453–5463. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/kidger21b.html.

Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3870–3882. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/li20i.html.

Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, and David E. Shaw. How fast-folding proteins fold. *Science*, 334(6055):517–520, 2011. doi: 10.1126/science.1208351. URL https://www.science.org/doi/abs/10.1126/science.1208351.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Lina Ma, Xiantao Li, and Chun Liu. The derivation and approximation of coarse-grained dynamics from langevin dynamics. *The Journal of Chemical Physics*, 145(20), November 2016. ISSN 1089-7690. doi: 10.1063/1.4967936. URL http://dx.doi.org/10.1063/1.4967936.

Kirill Neklyudov, Rob Brekelmans, Daniel Severo, and Alireza Makhzani. Action matching: Learning stochastic dynamics from samples, 2023. URL https://arxiv.org/abs/2210.06662.

Felipe Nuti, Tim Franzmeyer, and João F. Henriques. Extracting reward functions from diffusion models, 2023. URL https://arxiv.org/abs/2306.01804.

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

Sanjeev Raja, Martin Šípka, Michael Psenka, Tobias Kreiman, Michal Pavelka, and Aditi S. Krishnapriyan. Action-minimization meets generative modeling: Efficient transition path sampling with the onsager-machlup functional, 2025. URL https://arxiv.org/abs/2504.18506.

David E. Shaw, Paul Maragakis, Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, Michael P. Eastwood, Joseph A. Bank, John M. Jumper, John K. Salmon, Yibing Shan, and Willy Wriggers. Atomic-level characterization of the structural dynamics of proteins. *Science*, 330(6002):341–346, 2010. doi: 10.1126/science.1187409. URL https://www.science.org/doi/abs/10.1126/science.1187409.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Emanuel Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences of the United States of America*, 106(28):11478–11483, 2009. doi: 10.1073/pnas.0710743106. URL https://doi.org/10.1073/pnas.0710743106.

Rubicelia Vargas, Jorge Garza, Benjamin P. Hay, and David A. Dixon. Conformational study of the alanine dipeptide at the mp2 and dft levels. *The Journal of Physical Chemistry A*, 106(13):3213–3218, 2002. doi: 10.1021/jp013952f. URL https://doi.org/10.1021/jp013952f.

Hadrien Vroylandt and Pierre Monmarché. Position-dependent memory kernel in generalized langevin equations: Theory and numerical estimation. *The Journal of Chemical Physics*, 156 (24), June 2022. ISSN 1089-7690. doi: 10.1063/5.0094566. URL http://dx.doi.org/10.1063/5.0094566.

Grady Williams, Andrew Aldrich, and Evangelos A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017. doi: 10.2514/1.G001921. URL https://doi.org/10.2514/1.G001921.

Eric C.-Y. Yuan, Anup Kumar, Xingyi Guan, Eric D. Hermes, Andrew S. Rosen, Judit Zádor, Teresa Head-Gordon, and Samuel M. Blau. Analytical ab initio hessian from a deep learning potential for transition state optimization. *Nature Communications*, 15(1), 10 2024. doi: 10.1038/s41467-024-52481-5.

Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pp. 1433–1438. AAAI Press, 2008. ISBN 9781577353683.

APPENDIX

The appendix is organized as follows:

- § A gives the action recovery, structured Gaussian, matched conditioning, nonadditive action and reuse theoretical results.
- § B defines the shared metrics, training and path inference protocol.
- § C gives the double well construction, baselines, and path inference details.
- § D studies reference choice, temporal resolution and training budget.
- § E gives the exact orbital action and the estimator, scaling and trajectory diffusion controls.
- § F gives the molecular models, completion and transfer controls, preference and constraint sweeps and molecular checks.

A THEORY AND DERIVATIONS

This section gives the assumptions and derivations in § 3, which includes population action recovery, the structured Gaussian correction and drift interpretation, matched trajectory selection, nonadditive actions and downstream stability.

Notation. Fix a context c and let Ω_c be the feasible trajectory space. The densities $p_{\mathrm{data}}(\cdot \mid c)$ and $p_{\mathrm{ref}}(\cdot \mid c)$ are defined with respect to same base measure on Ω_c. The equalities here hold almost everywhere on their common support. We use the target correction G^* and target action S^* defined in Eqs. (1) and (2).

A.1 ACTION RECOVERY FROM TRAJECTORY COMPARISONS

Recall that LEAP learns the correction by identifying one observed trajectory among K reference trajectories drawn under the same context. This comparison objective has the exact trajectory density ratio as its population target for every $K \geq 1$.

Fix a context c and write $p = p_{\mathrm{data}}(\cdot \mid c)$ and $q = p_{\mathrm{ref}}(\cdot \mid c)$, so that

$$G^*(\Phi; c) = \log \frac{p(\Phi)}{q(\Phi)}.$$

Draw an index I uniformly from $\{0, \ldots, K\}$. Conditional on $I = i$, independently draw

$$\Phi_i \sim p, \qquad \Phi_j \sim q, \quad j \neq i. \tag{A.1}$$

The corresponding population objective is

$$\mathcal{L}_K(G; c) = -\mathbb{E}\left[\log \frac{\exp G(\Phi_I; c)}{\sum_{j=0}^{K} \exp G(\Phi_j; c)} \,\middle|\, c\right]. \tag{A.2}$$

By symmetry, this is the population form of Eq. (4).

Theorem A.1 (Population action recovery)**.** *Suppose p and q are positive on the same support, the candidates in Eq. (A.1) are conditionally independent and the model class contains G^*. Then, for every $K \geq 1$, every population minimizer satisfies*

$$G_{\theta^*}(\Phi; c) = G^*(\Phi; c) + a(c) \tag{A.3}$$

almost everywhere, where $a(c)$ is independent of Φ. If $\mathbb{E}_q|G^*| < \infty$, then the reference-centered action satisfies

$$S_{\theta^*}(\Phi; c) = S^*(\Phi; c) + \beta(c) \tag{A.4}$$

for a constant $\beta(c)$ independent of Φ.

Proof. By Bayes' rule,

$$\mathbb{P}(I = i \mid \Phi_{0:K}, c) = \frac{p(\Phi_i)\prod_{j\neq i} q(\Phi_j)}{\sum_{\ell=0}^{K} p(\Phi_\ell)\prod_{j\neq \ell} q(\Phi_j)} = \frac{p(\Phi_i)/q(\Phi_i)}{\sum_{\ell=0}^{K} p(\Phi_\ell)/q(\Phi_\ell)} = \frac{\exp G^*(\Phi_i; c)}{\sum_{\ell=0}^{K} \exp G^*(\Phi_\ell; c)}. \tag{A.5}$$

The comparison loss is minimized when the model matches this posterior. Therefore, for any two candidates Φ_i and Φ_j, $G_{\theta^*}(\Phi_i; c) - G_{\theta^*}(\Phi_j; c) = G^*(\Phi_i; c) - G^*(\Phi_j; c)$, which implies $G_{\theta^*}(\Phi; c) = G^*(\Phi; c) + a(c)$ almost everywhere. The context-dependent constant $a(c)$ cancels from the softmax.

Reference centering then gives

$$\widetilde{G}_{\theta^*}(\Phi; c) = G_{\theta^*}(\Phi; c) - \mathbb{E}_q G_{\theta^*}(\Phi'; c) = G^*(\Phi; c) - \mathbb{E}_q G^*(\Phi'; c). \tag{A.6}$$

Substituting into Eq. (7),

$$S_{\theta^*}(\Phi; c) = -\log q(\Phi) - \widetilde{G}_{\theta^*}(\Phi; c) = -\log p(\Phi) + \mathbb{E}_q G^*(\Phi'; c) = S^*(\Phi; c) + \beta(c), \tag{A.7}$$

where $\beta(c) = \mathbb{E}_q G^*(\Phi'; c) - \alpha(c)$. □

Reference centering fixes the undetermined context-dependent constant. With finite data, correction accuracy depends on reference coverage, the number of reference trajectories, model capacity and optimization.

A.2 Structured correction and drift identification

Consider the Euler-Maruyama process

$$x_{k+1} = x_k + b(x_k, t_k; c)\Delta t + \sqrt{2D\Delta t}\,\xi_k, \qquad \xi_k \overset{\text{iid}}{\sim} \mathcal{N}(0, I). \tag{A.8}$$

Let $v_k = (x_{k+1} - x_k)/\Delta t$. Its transition density is

$$q_b(x_{k+1} \mid x_k, c) = (4\pi D\Delta t)^{-d/2} \exp\left[-\frac{\Delta t}{4D}\|v_k - b(x_k, t_k; c)\|^2\right], \tag{A.9}$$

so for a fixed initial state,

$$-\log p_b(\Phi \mid c) = C_{\mathrm{EM}}(c) + \sum_{k=0}^{N-1} \frac{\Delta t}{4D}\|v_k - b(x_k, t_k; c)\|^2, \tag{A.10}$$

where $C_{\mathrm{EM}}(c) = \frac{Nd}{2}\log(4\pi D\Delta t)$.

Suppose the data and reference processes share the diffusion, time grid and initial law and differ via $b_{\mathrm{data}} = b_{\mathrm{ref}} + u^*$. Writing $u_k = u^*(x_k, t_k; c)$, the Gaussian likelihood ratio gives

$$G^*(\Phi; c) = \sum_{k=0}^{N-1}\left[\frac{u_k^\top\left(v_k - b_{\mathrm{ref}}(x_k, t_k; c)\right)}{2D} - \frac{\|u_k\|^2}{4D}\right]\Delta t. \tag{A.11}$$

This is the structured correction in Eq. (9). Minus it from the reference action and we get

$$S_{\mathrm{ref}}(\Phi; c) - G^*(\Phi; c) = C_{\mathrm{EM}}(c) + \sum_{k=0}^{N-1}\frac{\Delta t}{4D}\left\|v_k - b_{\mathrm{data}}(x_k, t_k; c)\right\|^2. \tag{A.12}$$

Thus the structured correction parameterizes the drift difference exactly under the shared-diffusion assumptions. This discrete likelihood ratio is the Euler-Maruyama analogue of the shared-diffusion Girsanov likelihood ratio (Girsanov, 1960).

Proposition A.2 (Drift identification under shared diffusion)**.** *Let P_b and $P_{\widetilde{b}}$ be unconditioned Euler-Maruyama trajectory distributions with the same initial law, step size $\Delta t > 0$ and diffusion coefficient $D > 0$. Then*

$$\mathrm{KL}(P_b \| P_{\widetilde{b}}) = \frac{\Delta t}{4D}\sum_{k=0}^{N-1}\mathbb{E}_{P_b}\left[\left\|b(X_k, t_k; c) - \widetilde{b}(X_k, t_k; c)\right\|^2\right]. \tag{A.13}$$

Therefore, equality of the trajectory distributions implies equality of their drifts on the occupied state-time support.

Proof. The KL chain rule decomposes the path divergence into expected transition divergences. Because the corresponding Gaussian transitions have covariance $2D\Delta t\, I$,

$$\mathrm{KL}\left(q_b(\cdot \mid X_k, c) \| q_{\widetilde{b}}(\cdot \mid X_k, c)\right) = \frac{\Delta t}{4D} \left\| b(X_k, t_k; c) - \widetilde{b}(X_k, t_k; c) \right\|^2 .$$

Take sum over k to get Eq. (A.13). If the path distributions are equal, the KL divergence is zero, so the drift difference vanishes almost surely on every occupied state-time pair. □

For structured LEAP, $b = b_{\mathrm{ref}} + u^*$ and $\widetilde{b} = b_{\mathrm{ref}} + u_{\theta^*}$. Hence recovery of the unconditioned trajectory distribution identifies $u_{\theta^*} = u^*$ on the occupied state-time support.

A.3 Matched conditioning and selected trajectories

Usually trajectory datasets are often restricted to a fixed endpoint or terminal event. Then when the data and reference use the same condition, conditioning changes their path log likelihood ratio only by a constant within that condition.

Proposition A.3 (Matched conditioning preserves action differences). *Let $Y = T(\Phi)$ be a trajectory statistic. Suppose the data and reference admit conditional densities given $Y = y$ with respect to the same base measure and both assign positive density to $Y = y$. Then, for almost every Φ with $T(\Phi) = y$,*

$$\log \frac{p_{\mathrm{data}}(\Phi \mid Y = y, c)}{p_{\mathrm{ref}}(\Phi \mid Y = y, c)} = \log \frac{p_{\mathrm{data}}(\Phi \mid c)}{p_{\mathrm{ref}}(\Phi \mid c)} + \log \frac{p^Y_{\mathrm{ref}}(y \mid c)}{p^Y_{\mathrm{data}}(y \mid c)}. \tag{A.14}$$

Proof. For $T(\Phi) = y$,

$$p_{\mathrm{data}}(\Phi \mid Y = y, c) = \frac{p_{\mathrm{data}}(\Phi \mid c)}{p^Y_{\mathrm{data}}(y \mid c)}, \qquad p_{\mathrm{ref}}(\Phi \mid Y = y, c) = \frac{p_{\mathrm{ref}}(\Phi \mid c)}{p^Y_{\mathrm{ref}}(y \mid c)}.$$

Taking their log ratio gives Eq. (A.14). The last term depends on y and c. Therefore conditioning preserves action differences between trajectories with the same condition. □

Setting $Y = X_N$ gives exact endpoint conditioning and setting $Y = \mathbb{1}\{X_N \in B\}$ and conditioning on $Y = 1$ gives terminal region selection.

Conditioning preserves complete path likelihood ratios up to a constant, but it changes the one step transition distribution. For the terminal event $\mathcal{E} = \{X_N \in B\}$, define $h_k(x; c) = P(\mathcal{E} \mid X_k = x, c)$. The conditioned transition kernel is

$$q^{\mathcal{E}}_k(x' \mid x, c) = q_k(x' \mid x, c) \frac{h_{k+1}(x'; c)}{h_k(x; c)}. \tag{A.15}$$

For smooth positive h in continuous time, the corresponding drift is

$$b^{\mathcal{E}}(x, t) = b(x, t) + 2D\nabla \log h(x, t). \tag{A.16}$$

Thus local regression on selected trajectories estimates $b^{\mathcal{E}}$.

Remark A.4 (Drift identification under fixed endpoints). We note that only having matched conditioning does not mean we can identify the unconditioned drift. Consider constant drift $b(x, t) = v$ relative to zero drift. The path log likelihood ratio is

$$\log \frac{p_v(\Phi \mid x_0)}{p_0(\Phi \mid x_0)} = \sum_{k=0}^{N-1} \left[\frac{v^\top (x_{k+1} - x_k)}{2D} - \frac{\Delta t \|v\|^2}{4D} \right] = \frac{v^\top (x_N - x_0)}{2D} - \frac{T\|v\|^2}{4D}. \tag{A.17}$$

The intermediate states cancel, so the ratio depends only on (x_0, x_N). Fixing both endpoints therefore makes this ratio constant across all admissible trajectories and removes the information needed to identify v from within that conditioned set. Hence Thm. A.2 applies to unconditioned trajectory distributions. We test drift recovery from terminally selected trajectories directly in §4.1.

A.4 NONADDITIVE ACTION FROM A PERSISTENT LATENT REGIME

A latent physical parameter may remain fixed for an entire trajectory without appearing in the observed state. After marginalizing this shared variable, the path action can contain interactions across transitions.

Let $z \in \{-1, +1\}$ be sampled once with equal probability and held fixed for the complete trajectory. Conditional on z, assume the path density factorizes as $p_z(\Phi \mid c) = \prod_{k=0}^{N-1} q_{z,k}(x_{k+1} \mid x_k, c)$, with reference $p_0(\Phi \mid c) = \prod_{k=0}^{N-1} q_{0,k}(x_{k+1} \mid x_k, c)$. Any shared initial density cancels from the likelihood ratio.

Define $\ell_{z,k} = \log q_{z,k}(x_{k+1} \mid x_k, c)$ and $\ell_{0,k} = \log q_{0,k}(x_{k+1} \mid x_k, c)$ and let

$$d_k = \frac{\ell_{+1,k} - \ell_{-1,k}}{2}, \qquad m_k = \frac{\ell_{+1,k} + \ell_{-1,k}}{2} - \ell_{0,k}. \tag{A.18}$$

Proposition A.5 (Persistent latent regimes induce nonadditive actions)**.** *Let* $p_{\mathrm{data}}(\Phi \mid c) = \frac{1}{2}p_{+1}(\Phi \mid c) + \frac{1}{2}p_{-1}(\Phi \mid c)$. *Then*

$$G^*(\Phi; c) = \sum_{k=0}^{N-1} m_k + \log\cosh\left(\sum_{k=0}^{N-1} d_k\right). \tag{A.19}$$

For any $i \neq j$,

$$\frac{\partial^2 G^*}{\partial d_i\, \partial d_j} = \operatorname{sech}^2\left(\sum_{k=0}^{N-1} d_k\right) > 0. \tag{A.20}$$

Therefore, on any region where d_i *and* d_j *can vary independently,* G^* *cannot be represented by a twice differentiable additive model* $G(\Phi; c) = \sum_k g_k(d_k, m_k)$.

Proof. Using the conditional factorization,

$$G^*(\Phi; c) = \log\left[\frac{1}{2}\exp\left(\sum_k \ell_{+1,k} - \ell_{0,k}\right) + \frac{1}{2}\exp\left(\sum_k \ell_{-1,k} - \ell_{0,k}\right)\right].$$

By Eq. (A.18), $\ell_{+1,k} - \ell_{0,k} = m_k + d_k$ and $\ell_{-1,k} - \ell_{0,k} = m_k - d_k$. Therefore,

$$G^*(\Phi; c) = \sum_k m_k + \log\left[\frac{1}{2}e^{\sum_k d_k} + \frac{1}{2}e^{-\sum_k d_k}\right] = \sum_k m_k + \log\cosh\left(\sum_k d_k\right).$$

Let $D = \sum_k d_k$. Then $\partial G^*/\partial d_i = \tanh D$, so for $i \neq j$, $\frac{\partial^2 G^*}{\partial d_i\, \partial d_j} = \operatorname{sech}^2 D > 0$.

In contrast, any additive model satisfies $\partial^2(\sum_k g_k(d_k, m_k))/(\partial d_i\, \partial d_j) = 0$ for $i \neq j$. □

A.5 DOWNSTREAM STABILITY UNDER ACTION REUSE

Fix a context c and feasible set $\Gamma(c)$ and suppose

$$\sup_{\Phi \in \Gamma(c)} |S_\theta(\Phi; c) - S^*(\Phi; c) - \beta(c)| \leq \varepsilon. \tag{A.21}$$

For any added path functional H, define $F_\theta = S_\theta + H$ and $F^* = S^* + H$.

Theorem A.6 (Stability under action reuse)**.** *If* $\widehat{\Phi}$ *is* δ *suboptimal for* F_θ*, then*

$$F^*(\widehat{\Phi}; c) - \inf_{\Phi \in \Gamma(c)} F^*(\Phi; c) \leq 2\varepsilon + \delta. \tag{A.22}$$

If $\pi_\theta \propto e^{-F_\theta}$ *and* $\pi^* \propto e^{-F^*}$ *have finite positive normalizing constants, then*

$$\left|\log\frac{\pi_\theta(\Phi \mid c)}{\pi^*(\Phi \mid c)}\right| \leq 2\varepsilon \qquad \forall \Phi \in \Gamma(c). \tag{A.23}$$

Proof. Write $e(\Phi) = S_\theta(\Phi; c) - S^*(\Phi; c) - \beta(c)$, so $|e(\Phi)| \le \varepsilon$. If $\widehat{\Phi}$ is δ suboptimal for F_θ, then for every $\Phi \in \Gamma(c)$, $F^*(\widehat{\Phi}; c) \le F^*(\Phi; c) + e(\Phi) - e(\widehat{\Phi}) + \delta \le F^*(\Phi; c) + 2\varepsilon + \delta$, which gives Eq. (A.22) after taking the inf.

For the normalized distributions,

$$\log \frac{\pi_\theta(\Phi \mid c)}{\pi^*(\Phi \mid c)} = -e(\Phi) - \log \mathbb{E}_{\pi^*}[e^{-e(\Phi)}].$$

Since $|e| \le \varepsilon$, the second term has magnitude at most ε, giving Eq. (A.23). □

Because the same H appears in both objectives, these bounds continue hold after adding new preferences or constraints.

B Experimental details

We use the metrics, reporting procedure and inference methods below for all experiments in § 4. Each experiment gives its system specific simulation, reference and model settings.

Compute resources. We run each training job on a single GPU. The experiments use NVIDIA A100-SXM GPUs with 8 CPU cores and 64 GB of memory per job.

B.1 Metrics and uncertainty

Drift recovery. When the true drift is known, we report

$$\mathrm{MSE}_b = \frac{1}{M} \sum_{i=1}^{M} \left\| \widehat{b}(x_i) - b^*(x_i) \right\|^2. \tag{B.1}$$

We evaluate all methods on the same states. For the double well, we use 301 states uniformly spanning $[-1, 1]$.

Action correction error. Because the action is identified up to a context dependent constant, we compare correction differences within each context. For $i, j \in I_c$, let $\Delta G_{ij} = G(\Phi_i; c) - G(\Phi_j; c)$. We define

$$\mathrm{NRMSE}_{\Delta G} = \left[\frac{\sum_c n_c^{-2} \sum_{i,j \in I_c} \left(\Delta G_{\theta,ij} - \Delta G^*_{ij} \right)^2}{\sum_c n_c^{-2} \sum_{i,j \in I_c} \left(\Delta G^*_{ij} \right)^2} \right]^{1/2}. \tag{B.2}$$

Exact correction differences give 0 and a constant prediction gives 1.

Trajectory discrimination and ordering. We report within context AUROC for separating data and reference trajectories. The metric does not measure correction scale, so we also use $\mathrm{NRMSE}_{\Delta G}$ when we need numerical action differences.

Path quality. When the true drift is known, we evaluate inferred paths with

$$Q_{\mathrm{true}}(\Phi) = \frac{\Delta t}{4} \sum_{k=0}^{N-1} \left\| \frac{x_{k+1} - x_k}{\Delta t} - b^*(x_k) \right\|^2. \tag{B.3}$$

For the double well, $b^*(x) = x - x^3$. We use this cost only for evaluation; each method performs inference with its own objective.

Uncertainty. We use 3 independent random seeds for every experiment that involves training. For all the tables, we report the sample mean±sample standard deviation,

B.2 Path inference

Algorithm 2 Path optimization with a learned LEAP action

Require: learned action $S_\theta(\Phi; c)$, initialization $\Phi^{(0)} \in \Gamma(c)$, optional rewards $\{M_j\}$, optional penalties $\{C_\ell\}$

1: define

$$F(\Phi; c) = S_\theta(\Phi; c) - \sum_j \alpha_j M_j(\Phi) + \sum_\ell \lambda_\ell C_\ell(\Phi)$$

2: initialize $\Phi \leftarrow \Phi^{(0)}$
3: **for** $r = 1$ to R **do**
4: update the free trajectory coordinates by one optimizer step on $F(\Phi; c)$
5: project onto hard constraints when required
6: **if** using knot redistribution **then**
7: redistribute knots and reevaluate F before the next optimizer step
8: **end if**
9: **end for**
10: **return** optimized path Φ^*

	Double well	Orbital	Alanine
Training paths	5,000	1,016	130 / 463
Reference paths K	8	4	8
Training budget	500 epochs	100 epochs	1,200 updates
Batch size	64	8	32
Learning rate	10^{-3}	3×10^{-4}	3×10^{-4}
Training seeds	3	3	3

Table 4: **Training settings for the principal experiments.** Molecular counts correspond to the 500 K and 600 K training sets. Orbital architecture details are given in § E.2.

B.3 Training and model selection

Table 4 summarizes the main settings. Additionally, for the double well, we use a potential network with three hidden layers of width 512, GELU activations and LayerNorm. We train with AdamW, weight decay 10^{-5}, cosine learning rate decay to 10^{-5} and gradient clipping at 1. For the orbital experiments, we use validation stopping with patience 20. § F gives the molecular architecture, reference and regularization settings.

C Double well experiments

We give the full double well protocol for § 4.1, including trajectory construction, baselines, path inference and the analytic long horizon cost.

C.1 Trajectory construction and baselines

Data and reference. We simulate from $x_0 = -1$ to $T = 3$ with $D = 0.20$ and $\Delta t = 0.01$ and retain trajectories with $x_T \in [0.9, 1.1]$. Each seed uses 5,000 selected training trajectories. We use three independent training seeds. The reference is Brownian motion with the same diffusion and terminal condition. We sample x_T from its terminal density restricted to $[0.9, 1.1]$, then sample the corresponding Gaussian bridge. The data and reference use the same terminal selection event as in Thm. A.3.

LEAP model. We parameterize the drift by a scalar potential, $b_\theta(x) = -dU_\theta(x)/dx$. Training settings are given in Table 4.

Local drift baselines. Kramers-Moyal averages $(x_{k+1} - x_k)/\Delta t$ in state bins of width $2/300$. Cubic velocity regression fits the same local target by least squares using $\{1, x, x^2, x^3\}$. This basis contains the exact drift $x - x^3$, so its error does not come from function class misspecification. Both

Method	Definition
Kramers-Moyal	Binned mean local increment
Cubic velocity regression	Cubic least squares fit to local increments
Zero field	$b(x) = 0$
Geometric path	Fixed waiting and linear crossing path
Analytic	Exact drift and action
LEAP	Complete path comparison objective

Table 5: **Double well methods and controls.**

	Drift MSE ↓	True path cost ↓	
Method		$T = 4$	$T = 16$
Kramers-Moyal	$0.8132_{\pm 0.0030}$	—	—
Cubic velocity reg.	$0.7752_{\pm 0.0043}$	$0.31890_{\pm 0.00080}$	$0.6126_{\pm 0.0079}$
Zero field	0.0759	0.2637	0.4454
Geometric path	—	0.2637	0.2637
Analytic drift	0	0.2532	0.2500
LEAP	$\mathbf{(3.3_{\pm 2.7}) \times 10^{-4}}$	$\mathbf{0.25330_{\pm 0.00010}}$	$\mathbf{0.25060_{\pm 0.00060}}$

Table 6: **Physical recovery and path inference.** Drift error and path cost under new endpoints and horizons.

methods use the same selected trajectories as LEAP. By §A.3, terminal selection changes their local regression target.

C.2 Path inference under new boundary conditions

Inference setting. We keep the learned potential fixed and change the endpoint and horizon. Training paths end in $[0.9, 1.1]$ at $T = 3$, whereas inference fixes $x_0 = -1$ and $x_T = 0$. We evaluate $T \in \{1, 4, 8, 16\}$ and optimize only the free path coordinates.

Optimization. We use the same four fixed linear and waiting initializations for every differentiable objective and run 500 L-BFGS iterations. Each method selects the candidate with lowest value under its own objective. We then evaluate the returned path using the ground truth cost Q_{true} from Eq. (B.3). For LEAP, we differentiate through the learned drift during path optimization.

C.3 Long horizon barrier scale

For

$$\mathcal{S}_D^T(\phi) = \frac{1}{4D} \int_0^T \left| \dot{\phi}(t) + U'(\phi(t)) \right|^2 dt,$$

with $\phi(0) = -1$ and $\phi(T) = 0$, the barrier $U(0) - U(-1) = 1/4$ sets the long horizon cost. Using $|\dot{\phi} + U'|^2 = |\dot{\phi} - U'|^2 + 4U'\dot{\phi}$,

$$D\mathcal{S}_D^T(\phi) = U(\phi(T)) - U(\phi(0)) + \frac{1}{4} \int_0^T |\dot{\phi} - U'(\phi)|^2 dt \quad \geq U(0) - U(-1) = \frac{1}{4}.$$

The reverse gradient path $\dot{\phi} = U'(\phi)$ approaches this bound as $T \to \infty$, so

$$\lim_{T \to \infty} \inf_{\substack{\phi(0) = -1 \\ \phi(T) = 0}} D\mathcal{S}_D^T(\phi) = \frac{1}{4}.$$

We use this analytic long horizon limit to evaluate inferred paths in Table 6.

D Reference choice and robustness

We test sensitivity to the reference process on an unconditioned double well with an exact discrete path density. We use $b(x) = x - x^3$, $D = 0.12$, and $X_0 \sim \mathcal{N}(-0.8, 0.15^2)$. Each of the three seeds uses 256 training, 96 validation and 128 test trajectories. Unless varied below, we use 24 transitions with $\Delta t = 0.04$, $T = 0.96$ and a zero drift reference with $D_{\text{ref}} = 0.20$.

Setting	LEAP	Transition NCE	Gaussian MLE
Reference diffusion			
$D_{\mathrm{ref}} = 0.06$	$0.31_{\pm 0.13}$	$\mathbf{0.24}_{\pm \mathbf{0.13}}$	$0.347_{\pm 0.014}$
$D_{\mathrm{ref}} = 0.12$	$\mathbf{0.58}_{\pm \mathbf{0.16}}$	$0.64_{\pm 0.12}$	$0.644_{\pm 0.084}$
$D_{\mathrm{ref}} = 0.20$	$\mathbf{0.66}_{\pm \mathbf{0.14}}$	$0.71_{\pm 0.11}$	$0.729_{\pm 0.084}$
$D_{\mathrm{ref}} = 0.32$	$0.795_{\pm 0.089}$	$\mathbf{0.77}_{\pm \mathbf{0.10}}$	$0.855_{\pm 0.062}$
$D_{\mathrm{ref}} = 0.48$	$0.939_{\pm 0.040}$	$\mathbf{0.837}_{\pm \mathbf{0.080}}$	$0.931_{\pm 0.034}$
Reference drift $b_{\mathrm{ref}} = a_{\mathrm{ref}} b$			
$a_{\mathrm{ref}} = 0.25$	$\mathbf{0.42}_{\pm \mathbf{0.15}}$	$0.45_{\pm 0.19}$	$0.486_{\pm 0.066}$
$a_{\mathrm{ref}} = 0.50$	$\mathbf{0.223}_{\pm \mathbf{0.063}}$	$0.26_{\pm 0.11}$	$0.356_{\pm 0.028}$
$a_{\mathrm{ref}} = 0.75$	$0.1255_{\pm 0.0054}$	$\mathbf{0.121}_{\pm \mathbf{0.029}}$	$0.325_{\pm 0.013}$
$a_{\mathrm{ref}} = 1.00$	$0.079_{\pm 0.044}$	$\mathbf{0.0384}_{\pm \mathbf{0.0087}}$	$0.3170_{\pm 0.0062}$

Table 7: **Reference choice.** Correction NRMSE ↓. The diffusion sweep uses $b_{\mathrm{ref}} = 0$; the drift sweep uses $D_{\mathrm{ref}} = 0.20$. Contrastive models use $K = 4$.

Setting	LEAP	Transition NCE	Gaussian MLE
$\Delta t = 0.02$, $N = 48$	$0.828_{\pm 0.096}$	$0.769_{\pm 0.081}$	$0.730_{\pm 0.033}$
$\Delta t = 0.04$, $N = 24$	$0.66_{\pm 0.14}$	$0.71_{\pm 0.11}$	$0.729_{\pm 0.084}$
$\Delta t = 0.08$, $N = 12$	$0.67_{\pm 0.15}$	$0.671_{\pm 0.070}$	$0.66_{\pm 0.13}$

Table 8: **Temporal resolution.** Correction NRMSE ↓, mean ± sample SD over three seeds at fixed $T = 0.96$.

Transition NCE compares each observed next state with reference next states sampled from the same current state. Gaussian MLE maximizes the Gaussian one-step transition likelihood and learns a scalar diffusion coefficient. Both use the same drift network as LEAP.

Reference choice. Adding physical drift to the reference decreases LEAP correction NRMSE from 0.6641 at $a_{\mathrm{ref}} = 0$ to 0.2229 at $a_{\mathrm{ref}} = 0.5$ and 0.0791 at $a_{\mathrm{ref}} = 1$ (Table 7).

Temporal and observation robustness. Across all three temporal grids, LEAP correction NRMSE remains below the constant prediction level (Table 8). Increasing the hidden integration resolution from 2 to 8 substeps changes LEAP AUROC from 0.9171 to 0.9216. Observation noise decreases AUROC from 0.9156 on clean trajectories to 0.5948 at $\sigma_{\mathrm{obs}} = 0.10$.

Data and reference budget. Increasing the training set from 32 to 256 trajectories decreases LEAP NRMSE from 0.8693 to 0.6641. Increasing K from 1 to 4 improves the most, with small changes after $K = 4$. At the default setting, lowest action selection gives oracle-action regret 0.0075 nats for LEAP and 0.0230 nats for Gaussian MLE.

E Orbital experiments

We give the exact orbital correction and controls for path structure, likelihood parameterization and comparison objective. This is the same setting as §4.2. The canonical benchmark contains 31 bodies, a 186 dimensional state and 32 transitions. We split initial contexts chronologically and evaluate on held-out contexts.

E.1 Exact trajectory action

For each trajectory, we draw $z \in \{-1, +1\}$ uniformly and set the solar mass to $m_\odot(1 + z\eta)$. The same unobserved z remains fixed for every transition; the reference uses the nominal mass $z = 0$.

Model	Input	$\mathrm{NRMSE}_{\Delta G} \downarrow$
LEAP-Local	Transition	$1.0013_{\pm 0.0015}$
Transition NCE	Transition	$1.0008_{\pm 0.0013}$
LEAP-Global	Complete path + accumulated features	$\mathbf{0.228}_{\pm \mathbf{0.011}}$
LEAP-Hybrid	Local + complete path	$0.232_{\pm 0.014}$
Gaussian MLE	State + time	$11.92_{\pm 0.66}$
History Gaussian MLE	Causal history	$10.162_{\pm 0.093}$
Objective control with complete path model		
Binary NRE	Path encoder	$1.00000_{\pm 3\times 10^{-5}}$
Binary NRE	Transformer + accumulated features	$\mathbf{0.175}_{\pm \mathbf{0.011}}$
Categorical NRE	Transformer + accumulated features	$0.228_{\pm 0.015}$

Table 9: **Orbital controls.** Correction NRMSE ↓ on the 31 body benchmark; a constant prediction gives 1. Bold marks the lowest mean within each control block.

Setting	Training	$\mathrm{NRMSE}_{\Delta G} \downarrow$
$K = 4$	validation stop	$0.2208_{\pm 0.0056}$
$K = 4$	275 epochs	$0.2207_{\pm 0.019}$
$K = 16$	validation stop	$0.147_{\pm 0.017}$
$K = 32$	validation stop	$0.138_{\pm 0.032}$
Training data		
Training paths	LEAP-Global	Binary NRE
4,064	$0.149_{\pm 0.069}$	$\mathbf{0.129}_{\pm \mathbf{0.017}}$
8,128	$\mathbf{0.110}_{\pm \mathbf{0.033}}$	$0.119_{\pm 0.015}$

Table 10: **Comparison and data budgets.** Correction NRMSE ↓. Increasing K changes the number of reference paths per observed trajectory; the lower panel increases the number of training paths.

Let $\mu_z(x)$ be the velocity-Verlet update under regime z and let Σ be the shared transition covariance. Conditional on the initial context,

$$\begin{aligned} \ell_z(\Phi) &= \sum_{k=0}^{N-1} \log \mathcal{N}\left(x_{k+1}; \mu_z(x_k), \Sigma\right), \\ G^*(\Phi) &= \log\left[\tfrac{1}{2} e^{\ell_{+1}(\Phi)} + \tfrac{1}{2} e^{\ell_{-1}(\Phi)}\right] - \ell_0(\Phi). \end{aligned} \tag{E.1}$$

Marginalizing the shared z therefore mixes likelihoods only after information has accumulated across the complete path, producing the nonadditive correction derived in § A.4.

E.2 Path structure and estimator controls

LEAP-Local sums separate transition corrections. Transition NCE uses the same transition level information without the structured LEAP form. LEAP-Global models a correction from the complete path and accumulated features computed from the nominal gravitational dynamics. LEAP-Hybrid adds the local and global corrections. Gaussian MLE predicts a Gaussian next state distribution from the current state and time, while History Gaussian MLE also conditions on the preceding trajectory.

Path structure. Transition-level models remain at NRMSE 1, while LEAP-Global reaches 0.228 (Table 9). The Gaussian likelihood models also fail to represent the exact correction in Eq. (E.1), which marginalizes a regime shared across the complete trajectory.

Objective control. With the same Transformer and accumulated inputs, binary and categorical ratio objectives reach NRMSE 0.175 and 0.228, respectively. Therefore the reduction comes from using the complete path information by Eq. (E.1).

Transitions	Path coords.	LEAP-Local	LEAP-Global
31 bodies			
32	6,138	$0.9994_{\pm 1.2\times 10^{-3}}$	$0.2584_{\pm 0.0097}$
64	12,090	$0.9985_{\pm 7\times 10^{-4}}$	$0.127_{\pm 0.015}$
128	23,994	$1.0052_{\pm 0.0066}$	$0.113_{\pm 0.027}$
256	47,802	$0.9965_{\pm 7\times 10^{-4}}$	$0.117_{\pm 0.055}$
42 bodies			
32	8,316	$1.0001_{\pm 7\times 10^{-4}}$	$0.251_{\pm 0.016}$
64	16,380	$1.0011_{\pm 1.6\times 10^{-3}}$	$0.144_{\pm 0.024}$
128	32,508	$0.9997_{\pm 1.4\times 10^{-3}}$	$0.163_{\pm 0.053}$
256	64,764	$1.0015_{\pm 1.7\times 10^{-3}}$	$\mathbf{0.111}_{\pm \mathbf{0.010}}$

Table 11: **Scaling with system size and path length.** Correction NRMSE ↓. Each configuration uses its exact mixture correction and $K = 4$.

Objective difference	Error ↓	AUROC ↑	Spearman ↑
Denoising	$1.0000_{\pm 2.3\times 10^{-6}}$	$0.4999_{\pm 2\times 10^{-4}}$	—
Variational	$0.9883_{\pm 1\times 10^{-4}}$	$0.4999_{\pm 2\times 10^{-4}}$	$0.0066_{\pm 0.0055}$

Table 12: **Trajectory diffusion baseline.** GeoTDM objective differences on the 31 body orbital benchmark. The denoising row reports correction NRMSE; the variational row reports affine relative error.

E.3 Comparison and data budgets

Increasing K from 4 to 16 decreases LEAP-Global NRMSE from 0.2208 to 0.1474, while extending the $K = 4$ model to 275 epochs leaves it at 0.2207 (Table 10). Increasing the training set from 4,064 to 8,128 paths further decreases LEAP-Global NRMSE from 0.1493 to 0.1099.

E.4 Longer paths and larger systems

We vary the number of bodies and transitions and evaluate each setting against its exact mixture correction. At 42 bodies and 256 transitions, each path contains 64,764 coordinates. LEAP-Global reaches NRMSE 0.1107, while LEAP-Local remains at 1.0015 (Table 11). The global model therefore has low correction error with increasing state dimension and path length.

E.5 Trajectory diffusion baseline

We adapt GeoTDM (Han et al., 2024) to the orbital benchmark and train separate trajectory diffusion models on the data and reference paths. We use the difference between their denoising or variational objectives as an estimate of G^*. In Table 12, both give chance AUROC (0.4999) and the variational estimate has Spearman correlation 0.0066 with G^*. Thus trajectories with larger exact corrections do not systematically receive larger GeoTDM estimates.

F Molecular inference and action reuse

F.1 Molecular data and models

Data and reference. We use 130/17/22 train/validation/test trajectories at 500 K and 463/73/77 at 600 K, with simulation replicas assigned to disjoint splits. Vacuum Langevin dynamics use a 1 fs integration step and we save the backbone angles (ϕ, ψ) every 0.1 ps. Paths have at most 201 frames.

We estimate a binned angular drift $\widehat{b}(x)$ and effective diffusion D_{eff} from the training replicas and define

$$b_{\text{ref}}(x, t) = \widehat{b}(x) + 15\frac{\text{wrap}(x_B - x)}{T - t + 2\Delta t}.$$

We condition reference paths on the same reactive event as the data and evaluate wrapped Gaussian transition densities by summing periodic images.

Method	Sparse completion	Endpoints only
Reference action	$0.1031_{\pm 0.0016}$	$0.8183_{\pm 0.0042}$
Untrained action	$0.0947_{\pm 0.0000}$	$0.7953_{\pm 0.0027}$
Velocity regression	$0.0900_{\pm 0.0003}$	$0.6609_{\pm 0.0032}$
Binary NRE	$0.0897_{\pm 0.0002}$	$0.6682_{\pm 0.0118}$
LEAP	$0.0931_{\pm 0.0016}$	$0.6225_{\pm 0.0080}$
Conditional DDPM	$0.119_{\pm 0.012}$	$0.409_{\pm 0.034}$
Conditional flow matching	$0.125_{\pm 0.011}$	$0.350_{\pm 0.012}$

Table 13: **Molecular completion at 600 K.** Circular MSE ↓ on unobserved frames for the same observation patterns and durations.

Train	Test	LEAP	Binary NRE	Velocity	Untrained
500	500	$0.9403_{\pm 0.0099}$	$0.9337_{\pm 0.0030}$	$0.9427_{\pm 0.0033}$	$0.8385_{\pm 0.0050}$
600	500	$\mathbf{0.9579_{\pm 0.0057}}$	$0.9489_{\pm 0.0051}$	$0.9437_{\pm 0.0022}$	$0.8385_{\pm 0.0050}$
500	600	$\mathbf{0.9719_{\pm 0.0068}}$	$0.9386_{\pm 0.0026}$	$0.9533_{\pm 0.0025}$	$0.8847_{\pm 0.0028}$
600	600	$0.9754_{\pm 0.0012}$	$0.9692_{\pm 0.0007}$	$0.9559_{\pm 0.0029}$	$0.8847_{\pm 0.0028}$

Table 14: **Temperature transfer.** Within context AUROC ↑. Bold marks LEAP evaluated at a temperature different from its training temperature.

Models and controls. We embed angles periodically and parameterize LEAP with a drift network of three width 128 hidden layers and a scalar residual with two width 64 hidden layers. LEAP uses $K = 8$ reference paths drawn from a bank of 32 successful references per initial state. Binary NRE uses the same inputs and network capacity, while velocity regression fits wrapped one step increments with the same drift network.

For conditional generation, we compare with DDPM (Ho et al., 2020) and flow matching (Lipman et al., 2023) using the same molecular coordinates, observation masks and active durations. Both use a temporal convolutional network with width 96 and seven residual layers. DDPM uses a 100 step cosine schedule and flow matching uses 100 Euler steps at inference.

F.2 Completion controls

We evaluate two conditioning settings. In the first, we observe the endpoints together with three interior frames; in the second, we observe only the endpoints and trajectory duration. For action-based methods, we optimize eight shared initializations for 400 projected Adam steps and retain the path with the lowest action. Conditional generators produce eight candidate trajectories, from which we select using the reference action. Candidate selection never uses the held-out target frames.

With endpoints and three interior observations, LEAP gets circular MSE 0.0931, compared with 0.1191 for conditional DDPM and 0.1247 for conditional flow matching (Table 13). With endpoints only, the conditional generators achieve lower reconstruction error. The same LEAP checkpoint is used for both observation patterns.

F.3 Temperature transfer

We evaluate each learned correction at both temperatures without retraining. At the target temperature, we use the corresponding reference process and effective diffusion.

Training at 600 K gives AUROC 0.9579 at 500 K, while training at 500 K gives 0.9719 at 600 K (Table 14). Thus the learned correction transfers across temperatures when combined with the target reference process.

	$\lambda = 0$	$\lambda = 5$	$\lambda = 20$	$\lambda = 50$
Preference $M \uparrow$	$0.0658_{\pm 0.0059}$	$0.1095_{\pm 0.0088}$	$0.2517_{\pm 0.0045}$	$0.3641_{\pm 0.0116}$
Transition cost $\downarrow$	$0.9389_{\pm 0.0545}$	$0.9159_{\pm 0.0228}$	$0.9809_{\pm 0.0231}$	$1.3774_{\pm 0.0252}$

Table 15: **Preference reuse with a fixed LEAP action.** Increasing λ raises the basin preference while changing the unguided projected transition cost in nats per active step.

F.4 Preference and constraint reuse

We keep the learned 600 K action fixed and add a periodic preference centered at $c_M = (-60^\circ, -40^\circ)$,

$$M(\Phi) = \frac{1}{N-1} \sum_{k=1}^{N-1} \exp\left[\frac{1}{s_M^2} \sum_{j=1}^{2} \big(\cos(x_{k,j} - c_{M,j}) - 1\big) \right], \qquad s_M = 30^\circ,$$

with inference objective $J_\lambda(\Phi) = S_\theta(\Phi) - \lambda M(\Phi)$. We evaluate $\lambda \in \{0, 5, 20, 50\}$ on the same 77 endpoint contexts.

Increasing λ from 0 to 20 raises the mean preference from 0.0658 to 0.2517 while the transition cost changes from 0.9389 to 0.9809 (Table 15). At $\lambda = 50$, the preference increases further to 0.3641 with a larger transition cost of 1.3774.

Constraints and composed requests. We also keep the learned action fixed while imposing $|\phi_k| \le 150^\circ$ and $|\phi_{k+1} - \phi_k| \le \pi - 10^{-3}$ through projection during inference. We combine these bounds with basin preferences, route constraints and additional observed states by changing only the inference objective or feasible set.

Molecular geometry verification. We reconstruct optimized (ϕ, ψ) paths in the same molecular system used for simulation. At both 500 K and 600 K, the maximum torsion error is less than 0.24° and the relative bond length change is also less than 1.5%. All reconstructed configurations preserve C_α chirality and trans peptide geometry.