STITCH-OPE: Trajectory Stitching with Guided Diffusion for Off-Policy Evaluation

Hossein Goli[1,2,4] Michael Gimelfarb[1,2,4] Nathan De Lara[1,2,4] Haruki Nishimura[3]
Masha Itkina[3] Florian Shkurti[1,2,4]
[1]Department of Computer Science, University of Toronto
[2]University of Toronto Robotics Institute, Toronto, Canada
[3]Toyota Research Institute, Los Altos, California
[4]Vector Institute, Toronto, Canada
{hossein.goli,mike.gimelfarb,nathan.delara}@mail.utoronto.ca

***Abstract*—Off-policy evaluation (OPE) estimates the performance of a target policy using offline data collected from a behavior policy, and is crucial in domains such as robotics or healthcare where direct interaction with the environment is costly or unsafe. Existing OPE methods are ineffective for high-dimensional, long-horizon problems, due to exponential blow-ups in variance from importance weighting or compounding errors from learned dynamics models. To address these challenges, we propose STITCH-OPE, a model-based generative framework that leverages denoising diffusion for long-horizon OPE in high-dimensional state and action spaces. Starting with a diffusion model pre-trained on the behavior data, STITCH-OPE generates synthetic trajectories from the target policy by guiding the denoising process using the score function of the target policy. STITCH-OPE proposes two technical innovations that make it advantageous for OPE: (1) prevents over-regularization by subtracting the score of the behavior policy during guidance, and (2) generates long-horizon trajectories by stitching partial trajectories together end-to-end. We provide a theoretical guarantee that, under mild assumptions, these modifications result in an exponential reduction in variance versus long-horizon trajectory diffusion. Experiments on the D4RL and OpenAI Gym benchmarks show substantial improvement in mean squared error, correlation, and regret metrics compared to state-of-the-art OPE methods.**

I. Introduction

Given the slow and risky nature of online data collection, real-world applications of reinforcement learning and robot evaluation often require offline data for policy learning and evaluation [27, 51]. An important problem of working with offline data is *off-policy evaluation* (OPE), which aims to evaluate the performance of a target policy π using offline data collected from another behavior policy β. One practical advantage of OPE is that it saves the cost of evaluation on hardware in embodied applications in the real world [32]. However, a central challenge of OPE is the presence of *distribution shift* induced by differences in behavior and target policies [27, 4]. This can lead to inaccurate estimates of policy values, making it difficult to trust or select between multiple target policies before they are deployed [55, 31].

Numerous approaches have attempted to address the distribution shift in offline policy evaluation by reducing either the variance of the policy value or its bias, but they are typically ineffective in high-dimensional long-horizon problems. For example, Importance Sampling (IS) [43] estimates the value of the target policy by weighing the behavior policy rollouts according to the ratio of their likelihoods. However, it suffers from the so-called *curse of horizon* where the variance of the estimate increases exponentially in the evaluation horizon [30]. More recent model-free OPE estimators reduce or eliminate the explosion in variance by estimating the long-run state-action density ratio $d^\pi(s,a)/d^\beta(s,a)$ between the target and behavior policy [30, 39, 54], yet they have demonstrated poor empirical performance on high-dimensional tasks where the behavior and target policies are different (i.e. the behavior policy is not a noisy version of the target policy) [16].

As an alternative approach, model-based OPE estimators typically learn an empirical autoregressive model of the environment and reward function from the behavior data, which is used to generate synthetic rollouts from the target policy for offline evaluation [21, 50, 57]. Some advantages of the model-based paradigm include sample efficiency [28], exploitation of prior knowledge about the dynamics [14], and better generalization to unseen states [56]. Although model-based OPE methods often scale well to high-dimensional short-horizon problems – owing to the scalability of the deep model-based RL paradigm – their robustness diminishes in long-horizon tasks due to the compounding of errors in the approximated dynamics model [16, 19, 20].

Driven by the recent successes of generative diffusion in RL [36, 60, 1, 18, 37, 45], we propose ***S**ub-**T**rajectory **I**mportance-Weighted **T**rajectory **C**omposition for Long-**H**orizon **OPE*** for model-based off-policy evaluation in long-horizon high-dimensional problems. STITCH-OPE first trains a diffusion model on behavior data, allowing it to generate dynamically feasible behavior trajectories [20]. STITCH-OPE differs from prior work [1, 20, 18, 45] by training the diffusion model on short sub-trajectories instead of full rollouts, where sub-trajectory generation is conditioned on the final state of the previous generated sub-trajectory. This enables accurate trajectory "stitching" using short-horizon rollouts, while minimizing compounding error of full-trajectory rollouts, thus bridging the gap between model-based OPE and full-trajectory offline diffusion.

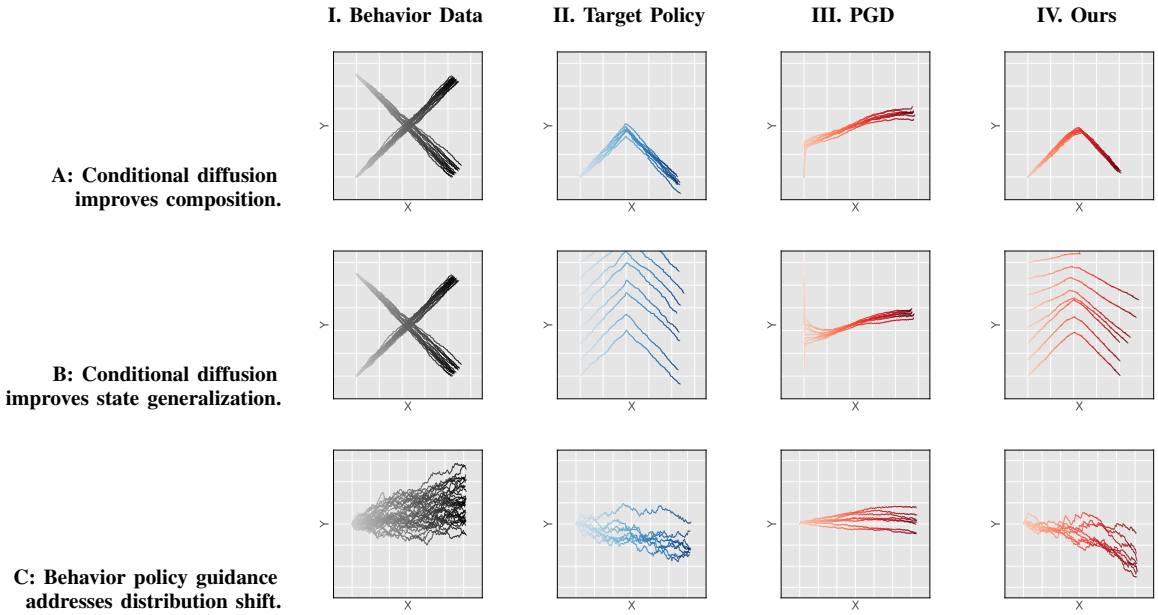

TABLE I

A 2D TOY PROBLEM WITH GAUSSIAN DYNAMICS ILLUSTRATES THE ADVANTAGES OF STITCH-OPE. **ROW A:** BEHAVIOR DATA FROM TWO POLICIES; TARGET IS PIECEWISE. STITCH-OPE CORRECTLY STITCHES TRAJECTORIES, WHILE PGD [18] STRUGGLES WITH COMPOSITIONALITY. **ROW B:** STITCH-OPE IS TRAINED ON SUB-TRAJECTORIES FROM ARBITRARY STATES AND THUS GENERALIZES BETTER. **ROW C:** NEGATIVE BEHAVIOR GUIDANCE PREVENTS OVER-REGULARIZATION IN THE PRESENCE OF SEVERE DISTRIBUTION SHIFT, REDUCING BIAS.

STITCH-OPE explicitly accounts for distribution shift in OPE by guiding the diffusion denoising process [10, 20] during inference. This can be achieved by selecting the guidance function to be the difference between the score functions of the target and behavior policies. A significant advantage of guided diffusion is that it eliminates the need to retrain the diffusion model for each new target policy. By pretraining the model on a variety of behavior datasets, generalization can be achieved during guided sampling to produce feasible trajectories under the target policy, leading to robust off-policy estimates for target policies that lack offline data.

II. PROPOSED METHODOLOGY

The *direct method* for off-policy evaluation [13] estimates the single-step autoregressive model $\hat{P}(s_t|s_{t-1}, a_{t-1})$ and the reward function $\hat{R}(s_t, a_t)$ from the behavior data. Then, it draws target policy trajectories $\tau \sim p_\pi(\tau)$ by forward sampling. That is, drawing initial state s_0 from the initial state distribution d_0, i.e. $s_0 \sim d_0$, we simulate $a_0 \sim \pi(\cdot|s_0), s_1 \sim \hat{P}(\cdot|s_0, a_0), \dots s_T \sim \hat{P}(\cdot|s_{T-1}, a_{T-1})$. However, even small errors in $\hat{P}$ can lead to significant bias in the value estimate, $J(\pi)$, due to the compounding of errors over long horizon T [22, 19]. STITCH-OPE avoids the compounding problem by generating the trajectory in short chunks, leading to more accurate OPE estimates over a long horizon.

A. Guided Diffusion for Off-Policy Evaluation

It is possible to approximate p_π using guided diffusion by interpreting each input/output to the diffusion model as a full trajectory τ. Given a behavior policy β and corresponding length-T trajectory distribution $p_\beta(\tau)$, the corresponding length-T trajectory distribution of target policy π can be written as:

$$p_\pi(\tau) = p_\beta(\tau)\, \rho_{0:T}(\tau), \tag{1}$$

where $\rho_{u:v}(\tau) = \prod_{t=u}^{v-1} \frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}$ is the standard importance sampling correction [43]. We address the question of tractably learning $p_\beta(\tau)$ by training a diffusion model $\hat{p}_\beta(\tau)$ on the offline behavior data set $\mathcal{D}_\beta$ [20], thus approximating $\hat{p}_\beta(\tau) \approx p_\beta(\tau)$. Specifically, the diffusion model learns to map a trajectory consisting of Gaussian noise, $\tau^k = (s_0^k, a_0^k, \dots s_T^k) \sim \mathcal{N}(0, I)$, to a noiseless behavior trajectory $\tau^0 = (s_0^0, a_0^0, \dots s_T^0)$.

A key observation is that we can bypass importance sampling in (1) by guiding the generation process $\hat{p}_\beta(\tau)$ towards $p_\pi(\tau)$ using diffusion guidance [20]. Specifically, let τ^k denote a noisy behavior trajectory at step k of the forward diffusion process, and let $y \in \{0, 1\}$ be a binary outcome with $p(y = 1|\tau) \propto \rho_{0:T}(\tau)$. Intuitively, y indicates whether the trajectory τ is generated by the target policy π ($y = 1$) or the behavior policy β ($y = 0$), and the likelihood ratio determines the odds that $y = 1$ given τ. By (1), $p_\pi(\tau) \propto p_\beta(\tau)\, p(y = 1|\tau)$, and thus the backward diffusion process for generating target policy trajectories for OPE can be approximated with guidance:

$$\begin{aligned}
&\log p_\pi(\tau^k|\tau^{k+1}) \\
&\propto \log(p_\beta(\tau^k|\tau^{k+1})p(y = 1|\tau^{k+1})) \\
&\approx \log \mathcal{N}(\tau^{k+1}; \mu_k + \Sigma_k \nabla_\tau \log p(y = 1|\tau)|_{\tau^{k+1}}, \Sigma_k),
\end{aligned} \tag{2}$$

where $p_\beta(\tau^k|\tau^{k+1}) = \mathcal{N}(\mu_k, \Sigma_k)$ is the backward diffusion process. Therefore, we can obtain feasible target policy

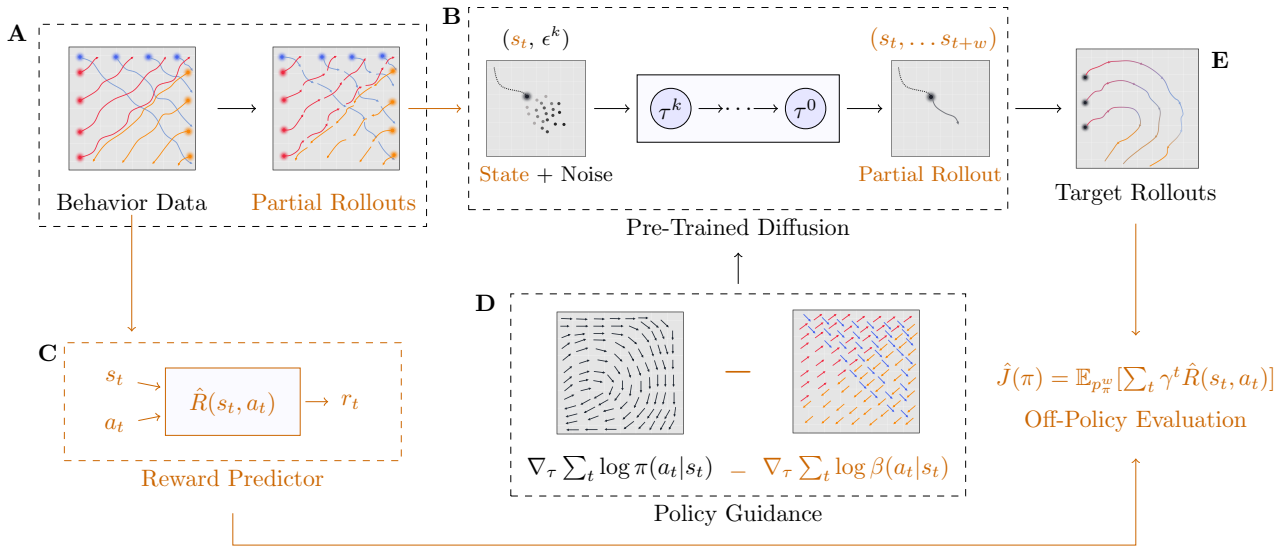

Fig. 1. A conceptual illustration of STITCH-OPE, with novel contributions highlighted in orange. **A:** Behavior data is sliced into partial trajectories of length w. **B:** Denoising diffusion process to generate partial behavior trajectories starting from the initial state s_t. **C:** A reward function $\hat{R}(s, a)$ is estimated from behavior transitions. **D:** Policy guidance with the negative behavior score function guides the diffusion process towards partial target trajectories. **E:** Estimation of the return by stitching generated partial trajectories end-to-end and evaluating their empirical cumulative returns.

trajectories using the guidance function:

$$g(\tau) = \nabla_\tau \log p(y = 1|\tau) = \nabla_\tau \log \rho_{0:T}(\tau) \\ = \nabla_\tau \sum_{t=0}^{T-1} \log \pi(a_t|s_t) - \nabla_\tau \sum_{t=0}^{T-1} \log \beta(a_t|s_t). \quad (3)$$

In our empirical evaluation, we employ the following generalization of (3) to allow fine-grained control over the relative importance of the target and behavior policy guidance

$$g(\tau) = \alpha \nabla_\tau \sum_{t=0}^{T-1} \log \pi(a_t|s_t) - \lambda \nabla_\tau \sum_{t=0}^{T-1} \log \beta(a_t|s_t), \quad (4)$$

where α and λ are hyper-parameters. Ignoring the normalizing constant which does not dependent on τ, (4) is equivalent to sampling from a *tempered posterior* [2, 6] distribution over τ,

$$q_\pi(\tau) \propto p_\beta(\tau) \prod_{t=0}^{T-1} \frac{\pi(a_t|s_t)^\alpha}{\beta(a_t|s_t)^\lambda}. \quad (5)$$

Given a set of denoised trajectories τ^0 from the guided diffusion process, and an empirical reward function $\hat{R}(s, a)$, it is straightforward to estimate the expected return (or a statistic such as variance or quantile) given *any* target policy, i.e. $\hat{J}(\pi) = \mathbb{E}_{\tau=\tau^0 \sim \hat{p}_\pi}\left[\sum_t \gamma^t \hat{R}(s_t, a_t)\right]$.

B. Sub-Trajectory Stitching with Conditional Diffusion

Recent work has shown that full-length diffusion models do not provide sufficient compositionality for accurate long-horizon sequence generation [7]. In addition, full-length prediction requires the generation of sequences of length $T \times (\dim(\mathcal{A}) + \dim(\mathcal{S}))$; this may be infeasible or inefficient on resource-constrained systems, when T is large or when $\mathcal{A}$ or $\mathcal{S}$ is high-dimensional.

To tackle these limitations, STITCH-OPE trains a conditional diffusion model to generate behavior sub-trajectories of length $w \ll T$. To allow for a more flexible composition of behavior trajectories during guidance, generation in STITCH-OPE is performed in a semi-autoregressive manner from the diffusion model, which is conditioned on the last state of the previously generated sub-trajectory.

Writing $p_\beta(\tau_{t:t+w}|s^0_t)$ to denote the sampling distribution over fully denoised sub-trajectories $\tau^0_{t:t+w}$ conditioned on s^0_t, the sampling process of STITCH-OPE can be written as:

$$p^w_\pi(\tau) = \prod_{t=0}^{T/w-1} \left(p_\beta(\tau_{wt:w(t+1)}|s^0_{wt})\, \rho_{wt:w(t+1)}(\tau)\right), \quad (6)$$

and thus each sub-trajectory can be generated by guiding the conditional diffusion with

$$g(\tau_{wt:w(t+1)}) = \nabla_{\tau_{wt:w(t+1)}} \log \rho_{wt:w(t+1)}(\tau).$$

A complete algorithm description of STITCH-OPE is provided in Appendix D.

To understand the intuition that the conditional diffusion model offers better compositionality than the full-horizon prediction, we decompose the behavior trajectory distribution as a mixture over the trajectories τ_j in $\mathcal{D}_\beta$:

$$p_\beta(s_t, a_t, \dots s_{T-1}|s_0, a_0, \dots s_t) \\ \approx \sum_{\tau_j \in \mathcal{D}_\beta} p_\beta(s_t, a_t, \dots s_{T-1}|s_t, \tau_j) p(\tau_j|s_0, a_0, \dots s_t).$$

Meanwhile, the conditional diffusion model ignores the full history of past states, i.e.:

$$p_\beta(s_t, a_t, \dots s_{T-1}|s_0, a_0, \dots s_t) \\ \approx \sum_{\tau_j \in \mathcal{D}_\beta} p_\beta(s_t, a_t, \dots s_{T-1}|s_t, \tau_j) p(\tau_j|s_t).$$

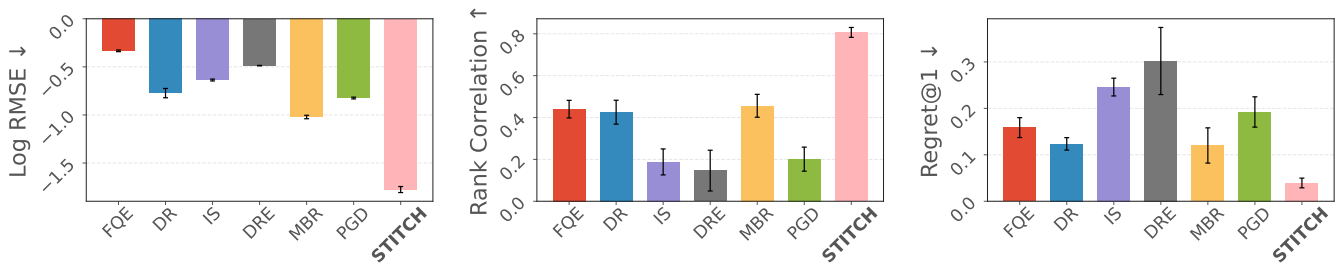

Fig. 2. Mean overall performance of all baselines, averaged across environments. Error bars represent +/- one standard error.

$p(\tau_j|s_t)$ has higher entropy than $p(\tau_j|s_0, a_0, \dots s_t)$ since it is conditioned on less information (see Appendix B for a proof), and thus provides a broader coverage of the diverse modes in the behavior dataset. This improves the compositionality of guided long-horizon trajectory generation. Row A of Table I illustrates this claim empirically using the GaussianWorld problem. A further claim is that the STITCH-OPE model can generalize better across initial states with low, or even zero, probability under d_0 (see row B of Table I). This occurs because p_β^w is trained on sub-trajectories starting in arbitrary states in $\mathcal{D}_\beta$, as opposed to states sampled only from d_0. We also provide a formal bias-variance tradeoff analysis in Appendix C, showing that STITCH-OPE achieves an exponential reduction in variance while maintaining bounded bias under mild assumptions (see Theorems C.8,C.11).

III. Empirical Evaluation

A. Experiment Details

a) Experimental Setup: We assess STITCH-OPE on high-dimensional, long-horizon tasks from the D4RL benchmark [15] (halfcheetah-medium, hopper-medium, walker2d-medium) with 10 target policies of varying proficiency [16], as well as Pendulum and Acrobot from OpenAI Gym [5]. Domain specifics (horizons, γ, policy training) and full training hyper-parameters are detailed in Appendices E and I.

b) Comparisons and Metrics: We compare against four model-free OPE estimators—**FQE** [26], **DR** [49], **IS** [43], **DRE** [39]—and two model-based methods—**MB** [21, 52] and **PGD** [18] (implementation details in Appendix G). Each method is run over 5 seeds per dataset–policy pair; true returns are obtained by executing each policy in the simulator. We report **LogRMSE**, **Spearman Correlation**, and **Regret@1** (difference between the estimated-best and true-best policy; further details in Appendix H). Results are plotted in Figure 2; per-environment breakdowns and additional experiments are provided in Appendix K. We see that STITCH-OPE provides state-of-the-art results on OPE across all metrics.

B. Off-Policy Evaluation with Diffusion Policies

To demonstrate the ability of STITCH-OPE to evaluate more complex policy classes, we replace target policies with diffusion policies, which have led to significant advances in robotics [8, 53] (see Appendix J for details). Since STITCH-OPE only requires the score of the target policy, it is computationally straightforward to perform OPE with diffusion policies, which is not the case for other estimators that require an explicit probability distribution $\pi_i(a|s)$ over actions (i.e. IS, DR). D4RL results are provided in Appendix J Table X. We see that STITCH-OPE outperforms all other baselines in 6 out of 9 instances, demonstrating robust OPE performance across multiple target policy classes.

IV. Real-World Deployability and Generalization Discussion

By training a conditional diffusion model on sub-trajectories (Section II-B), our approach improves compositionality and generalization to out-of-distribution (OOD) states. As shown in Table I, this enables generalization to low-probability or entirely unseen initial states, a prerequisite for robustness in real-world settings.

Furthermore, our use of guided diffusion with negative behavior guidance (Section II-B) mitigates over-regularization, allowing effective correction for distribution shift without retraining the generative model for each target policy. This is particularly important for real-world deployability, where behavior data could be collected from a variety of past policies, and new target policies could differ substantially.

Finally, we demonstrate in Appendix J that STITCH-OPE generalizes not only across tasks but also across policy classes, achieving strong performance when evaluating diffusion-based policies—a class of policies that are becoming increasingly prevalent in real-world robotics. Compatibility with diverse policy formats further supports the practical deployability of STITCH-OPE.

V. Conclusion

We presented STITCH-OPE for off-policy evaluation in high-dimensional, long-horizon environments. We showed that STITCH-OPE outperforms state-of-the-art OPE methods across MSE, correlation and regret metrics. Future work could investigate online data collection to address severe distribution shift, or explore ways to adapt the guidance coefficients or incorporate prior knowledge (e.g. about dynamics or policies) into guidance. It also remains an open question whether the advantages of STITCH-OPE apply to offline policy optimization.

References

[1] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

[2] Pierre Alquier and James Ridgway. Concentration of tempered posteriors and of their variational approximations. *The Annals of Statistics*, 48(3):1475–1497, 2020.

[3] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[4] David Brandfonbrener, Will Whitney, Rajesh Ranganath, and Joan Bruna. Offline rl without off-policy evaluation. *Advances in neural information processing systems*, 34: 4933–4946, 2021.

[5] G Brockman. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[6] Frédéric Cérou, Patrick Héas, and Mathias Rousset. Adaptive reduced tempering for bayesian inverse problems and rare event simulation. *arXiv preprint arXiv:2410.18833*, 2024.

[7] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[8] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[9] Stephen Dankwa and Wenfeng Zheng. Twin-delayed ddpg: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent. In *Proceedings of the 3rd international conference on vision, image and signal processing*, pages 1–5, 2019.

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[11] Zibin Dong, Yifu Yuan, Jianye Hao, Fei Ni, Yi Ma, Pengyi Li, and Yan Zheng. Cleandiffuser: An easy-to-use modularized library for diffusion models in decision making. *arXiv preprint arXiv:2406.09509*, 2024. URL https://arxiv.org/abs/2406.09509.

[12] Miroslav Dudık, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511, 2014.

[13] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456. PMLR, 2018.

[14] M Fard and Joelle Pineau. Pac-bayesian model selection for reinforcement learning. *Advances in Neural Information Processing Systems*, 23, 2010.

[15] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[16] Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, ziyu wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, Cosmin Paduraru, Sergey Levine, and Thomas Paine. Benchmarks for deep off-policy evaluation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=kWSeGEeHvF8.

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

[18] Matthew Thomas Jackson, Michael Matthews, Cong Lu, Benjamin Ellis, Shimon Whiteson, and Jakob Nicolaus Foerster. Policy-guided diffusion. In *Reinforcement Learning Conference*, 2024.

[19] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 1273–1286. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/099fe6b0b444c23836c4a5d07346082b-Paper.pdf.

[20] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

[21] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International conference on machine learning*, pages 652–661. PMLR, 2016.

[22] Nan Jiang, Alex Kulesza, Satinder Singh, and Richard Lewis. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 international conference on autonomous agents and multiagent systems*, pages 1181–1189, 2015.

[23] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on robot learning*, pages 651–673. PMLR, 2018.

[24] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020.

[25] Ilya Kostrikov and Ofir Nachum. Statistical bootstrapping for uncertainty estimation in off-policy evaluation. *arXiv preprint arXiv:2007.13609*, 2020.

[26] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR,

2019.

[27] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[28] Gen Li, Laixi Shi, Yuxin Chen, Yuejie Chi, and Yuting Wei. Settling the sample complexity of model-based offline reinforcement learning. *The Annals of Statistics*, 52(1):233–260, 2024.

[29] Guanghe Li, Yixiang Shan, Zhengbang Zhu, Ting Long, and Weinan Zhang. Diffstitch: Boosting offline reinforcement learning with diffusion-based trajectory stitching. In *International Conference on Machine Learning*, pages 28597–28609. PMLR, 2024.

[30] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. *Advances in neural information processing systems*, 31, 2018.

[31] Vincent Liu, Prabhat Nagarajan, Andrew Patterson, and Martha White. When is offline policy selection sample efficient for reinforcement learning? *arXiv preprint arXiv:2312.02355*, 2023.

[32] Yang Liu, Weixing Chen, Yongjie Bai, Guanbin Li, Wen Gao, and Liang Lin. Aligning cyber space with physical world: A comprehensive survey on embodied ai. *CoRR*, 2024.

[33] Yao Liu, Pierre Luc Bacon, and Emma Brunskill. Understanding the curse of horizon in off-policy evaluation via conditional importance sampling. In *International Conference on Machine Learning*, pages 6184–6193. PMLR, 2020.

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[35] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2022.

[36] Cong Lu, Philip J. Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=6jNQ1AY1Uf.

[37] Liyuan Mao, Haoran Xu, Xianyuan Zhan, Weinan Zhang, and Amy Zhang. Diffusion-DICE: In-sample diffusion guidance for offline reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=EIl9qmMmvy.

[38] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[39] Ali Mousavi, Lihong Li, Qiang Liu, and Denny Zhou. Black-box off-policy estimation for infinite-horizon reinforcement learning. In *International Conference on Learning Representations*, 2020.

[40] S. A. Murphy, M. J. van der Laan, J. M. Robins, and Conduct Problems Prevention Research Group. Marginal mean models for dynamic regimes. *Journal of the American Statistical Association*, 96(456):1410–1423, 2001. ISSN 01621459. URL http://www.jstor.org/stable/3085909.

[41] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *Advances in neural information processing systems*, 32, 2019.

[42] Xinkun Nie, Emma Brunskill, and Stefan Wager. Learning when-to-treat policies. *Journal of the American Statistical Association*, 116(533):392–409, 2021.

[43] Doina Precup, Richard S Sutton, and Satinder P Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 759–766, 2000.

[44] Aniruddh Raghu, Omer Gottesman, Yao Liu, Matthieu Komorowski, Aldo Faisal, Finale Doshi-Velez, and Emma Brunskill. Behaviour policy estimation in off-policy policy evaluation: Calibration matters. *International Conference on Machine Learning: workshop on Causal Machine Learning*, 2018.

[45] Marc Rigter, Jun Yamada, and Ingmar Posner. World models via policy-guided trajectory diffusion. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=9CcgO0LhKG.

[46] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 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*, pages 234–241. Springer, 2015.

[47] C Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

[48] Georgios Theocharous, Philip S. Thomas, and Mohammad Ghavamzadeh. Personalized ad recommendation systems for life-time value optimization with guarantees. In *International Joint Conference on Artificial Intelligence*, 2015. URL https://api.semanticscholar.org/CorpusID:8081523.

[49] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

[50] Masatoshi Uehara and Wen Sun. Pessimistic model-based offline rl: Pac bounds and posterior sampling under partial coverage. *CoRR*, 2021.

[51] Masatoshi Uehara, Chengchun Shi, and Nathan Kallus. A review of off-policy evaluation in reinforcement learning. *arXiv preprint arXiv:2212.06355*, 2022.

[52] Cameron Voloshin, Hoang Minh Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. In *Thirty-fifth Conference*

on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1), 2021.

[53] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

[54] Mengjiao Yang, Ofir Nachum, Bo Dai, Lihong Li, and Dale Schuurmans. Off-policy evaluation via the regularized lagrangian. *Advances in Neural Information Processing Systems*, 33:6551–6561, 2020.

[55] Mengjiao Yang, Bo Dai, Ofir Nachum, George Tucker, and Dale Schuurmans. Offline policy selection under uncertainty. In *International Conference on Artificial Intelligence and Statistics*, pages 4376–4396. PMLR, 2022.

[56] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33: 14129–14142, 2020.

[57] Michael R Zhang, Thomas Paine, Ofir Nachum, Cosmin Paduraru, George Tucker, ziyu wang, and Mohammad Norouzi. Autoregressive dynamics models for offline policy evaluation and optimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=kmqjgSNXby.

[58] Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. Gendice: Generalized offline estimation of stationary values. In *International Conference on Learning Representations*, 2020.

[59] Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=j5JvZCaDM0.

[60] Zhengbang Zhu, Hanye Zhao, Haoran He, Yichao Zhong, Shenyu Zhang, Haoquan Guo, Tingting Chen, and Weinan Zhang. Diffusion models for reinforcement learning: A survey. *arXiv preprint arXiv:2311.01223*, 2023.

APPENDIX A
GAUSSIANWORLD DOMAIN

The GaussianWorld domain is a toy 2-dimensional Markov decision process defined designed to illustrate and compare generalization and compositionality of diffusion models (Table I). It is defined as follows:

a) State Space: $\mathcal{S} = \mathbb{R}^2$ describes all positions (x_t, y_t) of a particle in space at every decision epoch t. It is assumed that x_t is the x-coordinate and y_t is the y-coordinate.

b) Action Space: $\mathcal{A} = \mathbb{R}$ describes the (counterclockwise) angle of the movement vector of the particle at every decision epoch, relative to the horizontal.

c) Transitions: Letting a_t be the angle of movement of the particle at time t, the transitions of x_t and y_t are defined as follows:

$$x_{t+1} = x_t + 0.02\cos(a_t + \varepsilon_t), \qquad y_{t+1} = y_t + 0.02\sin(a_t + \varepsilon_t), \qquad \varepsilon_t \sim \mathcal{N}(0, 0.2^2).$$

Here, ε_t is an i.i.d. Gaussian noise added to the actions before they are applied by the controller.

d) Reward Function: The problem is not solved so we leave the reward unspecified. We also leave the discount factor unspecified.

APPENDIX B
PROOF THAT CONDITIONAL DIFFUSION INCREASES ENTROPY

We begin with the following definitions.

Definition B.1 (Entropy)**.** Let $p(x)$ be a density function of a random variable X with support $\mathcal{X}$. The *entropy* of X is defined as

$$H(X) = \int_{\mathcal{X}} p(x) \log\left(\frac{1}{p(x)}\right) \mathrm{d}x.$$

Definition B.2 (Conditional Entropy)**.** The *conditional entropy* of X given Y on support $\mathcal{Y}$ is defined as

$$H(X|Y) = \mathbb{E}_{y\in\mathcal{Y}}[H(X|Y = y)].$$

Our goal is to prove

Theorem B.3. *Let S_t be the random state at time t sampled according to the conditional distribution $p(S_{t+1} = s|S_t = x, A_t = u)$, and let A_t be a random action following some conditional distribution $p(A_t = a|S_t = x)$. Then $H(\tau|S_t) \geq H(\tau|S_0, A_0 \dots S_t)$, where τ is a (random) sub-trajectory beginning in state S_t.*

Proof: First, letting $U = (S_0, A_0, \dots S_{t-1}, A_{t-1})$, observe that:

$$\begin{aligned}
H(U, \tau|S_t = s) &= \iint p(U = u, \tau|S_t = s) \log\left(\frac{1}{p(U = u, \tau|S_t = s)}\right) \mathrm{d}u\, \mathrm{d}\tau \\
&= \iint p(U = u, \tau|S_t = s) \log\left(\frac{1}{p(U = u|S_t = s)p(\tau|U = u, S_t = s)}\right) \mathrm{d}u\, \mathrm{d}\tau \\
&= \int p(U = u|S_t = s) \log\left(\frac{1}{p(U = u|S_t = s)}\right) \mathrm{d}u \\
&\quad + \int p(U = u|S_t = s) \int p(\tau|U = u, S_t = s) \log\left(\frac{1}{p(\tau|U = u, S_t = s)}\right) \mathrm{d}u\, \mathrm{d}\tau \\
&= H(U|S_t = s) + \mathbb{E}_{u\in\mathcal{U}|S_t=s}[H(\tau|U = u, S_t = s)].
\end{aligned}$$

Next, using the additivity property of expectation and law of total expectation:

$$H(U, \tau|S_t) = \mathbb{E}_{s\in\mathcal{S}_t}[H(U|S_t = s)] + \mathbb{E}_{s\in\mathcal{S}_t, u\in\mathcal{U}}[H(\tau|U = u, S_t = s)] = H(U|S_t) + H(\tau|U, S_t).$$

Next, we prove sub-additivity of conditional entropy:

$$\begin{aligned}
&H(U,\tau|S_t=s) - H(U|S_t=s) - H(\tau|S_t=s) \\
&= \iint p(U=u,\tau|S_t=s)\log\left(\frac{1}{p(U=u,\tau|S_t=s)}\right)\mathrm{d}u\,\mathrm{d}\tau \\
&\quad - \int p(U=u|S_t=s)\log\left(\frac{1}{p(U=u|S_t=s)}\right)\mathrm{d}u - \int p(\tau|S_t=s)\log\left(\frac{1}{p(\tau|S_t=s)}\right)\mathrm{d}\tau \\
&= \iint p(U=u,\tau|S_t=s)\log\left(\frac{p(\tau|S_t=s)p(U=u|S_t=s)}{p(U=u,\tau|S_t=s)}\right)\mathrm{d}u\,\mathrm{d}\tau \\
&\leq \log\iint p(U=u,\tau|S_t=s)\left(\frac{p(\tau|S_t=s)p(U=u|s_t=s)}{p(U=u,\tau|S_t=s)}\right)\mathrm{d}u\,\mathrm{d}\tau \\
&= \log 1 = 0,
\end{aligned}$$

where the inequality in the derivation follows by Jensen's inequality. This implies that

$$H(U,\tau|S_t=s) \leq H(U|S_t=s) + H(\tau|S_t=s).$$

Taking expectation of both sides with respect to S_t, and using the monotonicity and additivity properties of expectation:

$$\begin{aligned}
H(U,\tau|S_t) &= \mathbb{E}_{s\in\mathcal{S}_t}[H(U,\tau|S_t=s)] \\
&\leq \mathbb{E}_{s\in\mathcal{S}_t}[H(U|S_t=s) + H(\tau|S_t=s)] = H(U|S_t) + H(\tau|S_t).
\end{aligned}$$

Finally, putting it all together:

$$H(\tau|U,S_t) = H(U,\tau|S_t) - H(U|S_t) \leq H(U|S_t) + H(\tau|S_t) - H(U|S_t) = H(\tau|S_t),$$

which completes the proof. ∎

APPENDIX C
THEORETICAL ANALYSIS

A. Assumptions and Definitions

We decompose a full trajectory of length T into $N = T/w$ non-overlapping sub-trajectories (or chunks), each of length w. Each *chunk* $S_i \in \mathcal{T}^{(w)}$ is defined as

$$S_i := (s_{iw}, a_{iw}, s_{iw+1}, a_{iw+1}, \ldots, s_{(i+1)w}).$$

Let the *full trajectory* be defined as

$$S = (S_0, S_1, \ldots, S_{N-1}).$$

We define the *boundary state* X_i as the initial state of chunk S_i:

$$X_i := s_{iw}, \quad i = 0, 1 \ldots N,$$

which form the backbone of the generative process.

We assume the following factored generative process for trajectories

$$p(S_0, S_1, \ldots, S_{N-1}) = p(X_0)\prod_{i=0}^{N-1} p(S_i \mid X_i)\, p(X_{i+1} \mid S_i).$$

This implies that the boundary state sequence $X = (X_0, X_1, \ldots, X_N)$ forms a first-order Markov chain

$$p(X_{i+1} \mid S_i) = p(X_{i+1} \mid X_i).$$

Each chunk S_i produces a scalar discounted return Y_i, defined as

$$Y_i := f(S_i) = \sum_{j=0}^{w-1} \gamma^j \hat{R}(s_{iw+j}, a_{iw+j}),$$

where $\hat{R}$ is a learned reward model, and $\gamma \in [0,1]$ is the discount factor.

Given a bound $R_{\max} < \infty$ on the absolute reward, we define the *maximum per-chunk return bound* as:

$$B_w := \sum_{j=0}^{w-1} \gamma^j R_{\max} = \frac{R_{\max}(1-\gamma^w)}{1-\gamma} \quad \Rightarrow \quad |Y_i| \leq B_w.$$

The cumulative return over the full trajectory is approximated by

$$\hat{J} = \sum_{i=0}^{N-1} \gamma^{iw} Y_i,$$

and the expected return under the target policy π is:

$$J(\pi) := \mathbb{E}_{p_\pi}[\hat{J}] = \mathbb{E}_{p_\pi}\left[\sum_{i=0}^{N-1} \gamma^{iw} Y_i\right].$$

Definition C.1 (Chunked Behavior Distributions)**.** Let $p_\beta^{(w)}$ denote the true distribution over behavior chunks S_i, and let $\hat{p}_\beta^{(w)}$ be the learned conditional distribution modeled by the diffusion process. These distributions describe how chunks are generated given boundary states:

$$p_\beta^{(w)}(S_i \mid X_i), \qquad \hat{p}_\beta^{(w)}(S_i \mid X_i).$$

Definition C.2 (Total Variation Distance)**.** The *total variation distance* between two probability distributions P and Q over the same measurable space $\mathcal{X}$ is defined as

$$\mathrm{TV}(P, Q) := \sup_{A \subseteq \mathcal{X}} |P(A) - Q(A)|.$$

We require the following assumptions.

Assumption C.3 (Bounded Likelihood Ratio)**.** There is a constant κ such that $\frac{\pi(a|s)}{\beta(a|s)} \leq \kappa$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$.

Note that this assumption can be easily verified in our experimental setting. Since the action spaces are closed intervals and the behavior and target policy distributions are both represented as truncated Gaussian distributions, the ratio of the two policies is bounded over the action space.

Assumption C.4 (Chunk-wise Model Fit)**.** The total variation distance between the true chunk distribution $p_\beta^{(w)}$ and the learned conditional distribution $\hat{p}_\beta^{(w)}$ is bounded by some constant $\delta_\beta > 0$,

$$\mathrm{TV}\left(p_\beta^{(w)}, \hat{p}_\beta^{(w)}\right) \leq \delta_\beta.$$

This assumption is stated in terms of p_β rather than p_π, and is thus easier to validate in practice.

B. Analysis of the Bias

We begin by bounding the total variation distance between the true target distribution $p_\pi^{(w)}$ and the guided model $\hat{p}_\pi^{(w)}$.

Lemma C.5. *The total variation distance between the guided model $\hat{p}_\pi^{(w)}$ and the true target distribution $p_\pi^{(w)}$ satisfies*

$$\mathrm{TV}\left(p_\pi^{(w)}, \hat{p}_\pi^{(w)}\right) \leq \kappa^2 \cdot \delta_\beta$$

Proof: By the definition of total variation distance

$$\mathrm{TV}(p_\pi^{(w)}, \hat{p}_\pi^{(w)}) = \frac{1}{2} \int \left| p_\pi^{(w)}(\tau) - \hat{p}_\pi^{(w)}(\tau) \right| \mathrm{d}\tau.$$

Using the reweighted form of each distribution

$$\mathrm{TV}(p_\pi^{(w)}, \hat{p}_\pi^{(w)}) = \frac{1}{2} \int \left| \left(p_\beta^{(w)}(\tau) - \hat{p}_\beta^{(w)}(\tau)\right) \cdot \prod_{j=0}^{w-1} \frac{\pi(a_j \mid s_j)}{\beta(a_j \mid s_j)} \right| \mathrm{d}\tau,$$

and applying the bound on the likelihood ratio (Assumption C.3):

$$\mathrm{TV}(p_\pi^{(w)}, \hat{p}_\pi^{(w)}) \leq \frac{\kappa^w}{2} \int \left| p_\beta^{(w)}(\tau) - \hat{p}_\beta^{(w)}(\tau) \right| \mathrm{d}\tau = \kappa^w \cdot \mathrm{TV}(p_\beta^{(w)}, \hat{p}_\beta^{(w)}) \leq \kappa^w \cdot \delta_\beta.$$

This completes the proof. ∎

Let the total variation distance between the true target distribution and the guided diffusion model be denoted by

$$\delta_\pi := \mathrm{TV}\left(p_\pi^{(w)}, \hat{p}_\pi^{(w)}\right).$$

By Lemma C.5, we have the bound

$$\delta_\pi \leq \kappa^w \cdot \delta_\beta.$$

We now derive a bound on the absolute bias of the estimated return when sampling chunks from the guided model $\hat{p}_\pi^{(w)}$ instead of the true target distribution $p_\pi^{(w)}$.

Lemma C.6 (Expectation Difference Bound via Total Variation)**.** *Let p and q be two probability densities on a probability space $\mathcal{X}$. Let*

$$\|f\|_\infty \;=\; \sup_{x\in\mathcal{X}} \big|f(x)\big|$$

be the supremum norm of a bounded function $f : \mathcal{X} \to \mathbb{R}$, and let:

$$\|p-q\|_1 = \int_{\mathcal{X}} \big|p(x)-q(x)\big|\,\mathrm{d}x, \qquad \mathrm{TV}(p,q) = \tfrac{1}{2}\,\|p-q\|_1.$$

Then

$$\big|\mathbb{E}_{x\sim p}[f(x)] \;-\; \mathbb{E}_{x\sim q}[f(x)]\big| \le 2\,\|f\|_\infty\,\mathrm{TV}(p,q).$$

Proof:

$$\begin{aligned}
\big|\mathbb{E}_p[f]-\mathbb{E}_q[f]\big| &= \Big|\int_{\mathcal{X}} f(x)\,p(x)\,\mathrm{d}x \;-\; \int_{\mathcal{X}} f(x)\,q(x)\,\mathrm{d}x\Big| \\
&= \Big|\int_{\mathcal{X}} f(x)\,\big(p(x)-q(x)\big)\,\mathrm{d}x\Big| \;\le\; \int_{\mathcal{X}} \big|f(x)\big|\,\big|p(x)-q(x)\big|\,\mathrm{d}x \\
&\le\; \|f\|_\infty \int_{\mathcal{X}} \big|p(x)-q(x)\big|\,\mathrm{d}x \;=\; 2\,\|f\|_\infty\,\mathrm{TV}(p,q).
\end{aligned}$$

This completes the proof. ∎

Lemma C.7 (Marginal TV Bound via Conditional TV)**.** *Let $p(x \mid s)$ and $\hat{p}(x \mid s)$ be conditional densities over chunk $x \in \mathcal{T}^{(w)}$, given state $s \in \mathcal{S}$, and let $\mu(s)$ denote the marginal distribution over s. Then*

$$\mathrm{TV}\left(\int p(x \mid s)\mu(s)\mathrm{d}s, \int \hat{p}(x \mid s)\mu(s)\mathrm{d}s\right) \le \int \mathrm{TV}\left(p(\cdot \mid s), \hat{p}(\cdot \mid s)\right)\mu(s)\mathrm{d}s.$$

In particular, if $\mathrm{TV}(p(\cdot \mid s), \hat{p}(\cdot \mid s)) \le \epsilon$ for all s, then $\mathrm{TV}(p, \hat{p}) \le \epsilon$.

Proof: Let $p(x) = \int p(x \mid s)\mu(s)\mathrm{d}s$, $\hat{p}(x) = \int \hat{p}(x \mid s)\mu(s)\mathrm{d}s$. Then:

$$\begin{aligned}
\mathrm{TV}(p,\hat{p}) &= \frac{1}{2}\int |p(x)-\hat{p}(x)|\,\mathrm{d}x = \frac{1}{2}\int \left|\int \mu(s)\,[p(x \mid s)-\hat{p}(x \mid s)]\,\mathrm{d}s\right|\mathrm{d}x \\
&\le \frac{1}{2}\iint \mu(s)\,|p(x \mid s)-\hat{p}(x \mid s)|\,\mathrm{d}s\,\mathrm{d}x \quad \text{(by Jensen's inequality)} \\
&= \int \mu(s)\left[\frac{1}{2}\int |p(x \mid s)-\hat{p}(x \mid s)|\,\mathrm{d}x\right]\mathrm{d}s = \int \mu(s)\cdot \mathrm{TV}(p(\cdot \mid s),\hat{p}(\cdot \mid s))\mathrm{d}s.
\end{aligned}$$

If $\mathrm{TV}(p(\cdot \mid s), \hat{p}(\cdot \mid s)) \le \epsilon$ uniformly, the integral is bounded by ϵ. ∎

Theorem C.8 (Bias Bound for STITCH-OPE)**.** *The bias of the return estimate under the guided diffusion model satisfies*

$$\left|\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - J(\pi)\right| \le \frac{2B_w}{1-\gamma^w}\cdot\delta_\pi.$$

Proof: The return estimator is:

$$\hat{J} = \sum_{i=0}^{N-1}\gamma^{iw}Y_i, \qquad \text{where} \quad Y_i = f(S_i) = \sum_{j=0}^{w-1}\gamma^j \hat{R}(s_{iw+j}, a_{iw+j}).$$

Thus, the bias is:

$$\left|\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - \mathbb{E}_{p_\pi}[\hat{J}]\right| = \left|\sum_{i=0}^{N-1}\gamma^{iw}\left(\mathbb{E}_{\hat{p}_\pi}[Y_i] - \mathbb{E}_{p_\pi}[Y_i]\right)\right| \le \sum_{i=0}^{N-1}\gamma^{iw}\left|\mathbb{E}_{\hat{p}_\pi}[Y_i] - \mathbb{E}_{p_\pi}[Y_i]\right|.$$

For each chunk i, Y_i depends only on S_i, with marginal distributions $\hat{p}_\pi^{(w,i)}$ and $p_\pi^{(w,i)}$ under $\hat{p}_\pi$ and p_π, respectively. By Lemma C.6 and Lemma C.7

$$|\mathbb{E}_{\hat{p}_\pi}[Y_i] - \mathbb{E}_{p_\pi}[Y_i]| \le 2\cdot\sup|Y_i|\cdot\mathrm{TV}(p_\pi^{(w,i)}, \hat{p}_\pi^{(w,i)}).$$

Since $|\hat{R}(s,a)| \le R_{\max}$, the per-chunk return is bounded:

$$|Y_i| \le \sum_{j=0}^{w-1} \gamma^j R_{\max} = R_{\max} \cdot \frac{1-\gamma^w}{1-\gamma}.$$

Using Lemma C.5, we know that $\mathrm{TV}(p_\pi^{(w,i)}, \hat{p}_\pi^{(w,i)}) \le \delta_\pi$, Thus we have

$$|\mathbb{E}_{\hat{p}_\pi}[Y_i] - \mathbb{E}_{p_\pi}[Y_i]| \le 2 \cdot \frac{R_{\max}(1-\gamma^w)}{1-\gamma} \cdot \delta_\pi.$$

Summing over chunks:

$$\left|\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - \mathbb{E}_{p_\pi}[\hat{J}]\right| \le \sum_{i=0}^{N-1} \gamma^{iw} \cdot 2 \cdot \frac{R_{\max}(1-\gamma^w)}{1-\gamma} \cdot \delta_\pi = 2 \cdot \frac{R_{\max}(1-\gamma^w)}{1-\gamma} \cdot \delta_\pi \cdot \sum_{i=0}^{N-1} \gamma^{iw}.$$

The geometric sum is:

$$\sum_{i=0}^{N-1} \gamma^{iw} \le \sum_{i=0}^{\infty} \gamma^{iw} = \frac{1}{1-\gamma^w}.$$

Thus:

$$\left|\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - \mathbb{E}_{p_\pi}[\hat{J}]\right| \le 2 \cdot \frac{R_{\max}(1-\gamma^w)}{1-\gamma} \cdot \delta_\pi \cdot \frac{1}{1-\gamma^w} = \frac{2B_w}{1-\gamma^w} \cdot \delta_\pi.$$

This completes the proof. ■

Corollary C.9 (Bias Bound in Terms of Model Fit δ_β)**.** *Under the assumptions* $\sup_i \mathrm{TV}(p_\pi^{(w,i)}, \hat{p}_\pi^{(w,i)}) \le \delta_\pi \le \kappa^w \cdot \delta_\beta$ *and* $\sup_\tau |\hat{J}(\tau)| \le \frac{R_{\max}}{1-\gamma}$*, the bias satisfies*

$$\left|\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - J(\pi)\right| \le \frac{2B_w}{1-\gamma^w} \cdot \kappa^w \cdot \delta_\beta.$$

C. Analysis of the Variance

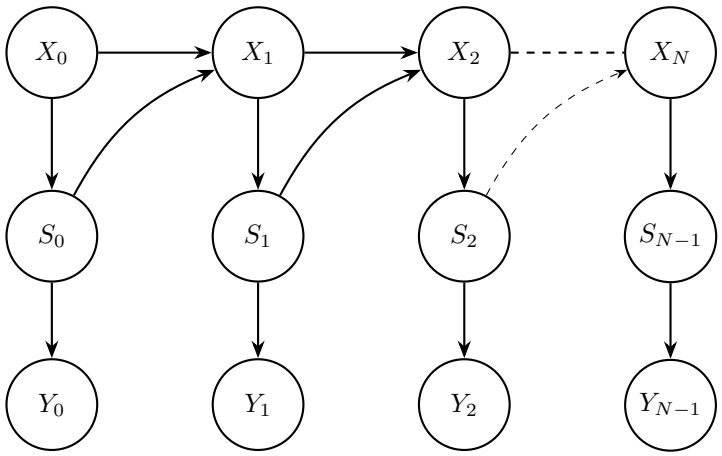

Fig. 3. Illustration of the sub-trajectory decomposition. Each chunk S_i generates a reward sequence Y_i and leads to a boundary state X_{i+1}.

Lemma C.10 (Conditional Independence of Chunk Rewards)**.** *Let* $X_i := s_{iw}$ *be the boundary state at the start of chunk* S_i*, and define:*

$$Y_i := f(S_i) = \sum_{j=0}^{w-1} \gamma^j \hat{R}(s_{iw+j}, a_{iw+j}).$$

Assume the generative process satisfies the following properties:

- *Each chunk* S_i *is generated independently given* X_i
- *The return* Y_i *is a deterministic function of* S_i*.*

Then for all $i \neq j$*, the returns* Y_i *and* Y_j *are conditionally independent given the full boundary state chain* $X_0, X_1, \ldots, X_N$*,*

$$Y_i \perp\!\!\!\perp Y_j \mid X_0, \ldots, X_N.$$

Proof: Refer to the graphical model in Figure 3. The nodes $X_0, X_1, \ldots, X_N$ form a Markov chain. Each chunk S_i is a child of X_i, and each return Y_i is a child of S_i.

Now consider any path from Y_i to Y_j. Such a path must go through:

$$Y_i \leftarrow S_i \leftarrow X_i \rightsquigarrow X_{i+1} \rightsquigarrow \cdots \rightsquigarrow X_j \rightarrow S_j \rightarrow Y_j.$$

All such paths must traverse through at least one boundary node X_k. Since we are conditioning on all $X_0, \ldots, X_N$, and these nodes are non-colliders on every path from Y_i to Y_j, all such paths are blocked. By the criterion of d-separation (see, e.g. Chapter 8 in [3]), this implies $Y_i \perp\!\!\!\perp Y_j \mid X_0, \ldots, X_N$. ■

Theorem C.11 (Variance Bound)**.** *Let $\hat{p}_\pi$ denote the trajectory distribution induced by the guided diffusion model, and p_π the true trajectory distribution under the target policy. Let $\hat{J}$ be the return estimator using a learned reward model. Then*

$$\mathrm{Var}_{\hat{p}_\pi}(\hat{J}) \;\le\; \mathrm{Var}_{p_\pi}(J) + 10\left(\frac{T}{w}\right)^2 B_w^2 \kappa^w \delta_\beta + \frac{2B_w^2}{1-\gamma^{2w}}\kappa^w \delta_\beta,$$

where B_w denotes the maximum per-chunk discounted return.

Proof: We begin by applying the law of total variance under the guided model distribution $\hat{p}_\pi$

$$\mathrm{Var}_{\hat{p}_\pi}(\hat{J}) = \mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(\hat{J} \mid X)\right] + \mathrm{Var}_{\hat{p}_\pi}\left(\mathbb{E}_{\hat{p}_\pi}[\hat{J} \mid X]\right).$$

Using Lemma C.10 we have that the chunk-level rewards Y_i and Y_j are conditionally independent given the boundary states $X_0, X_1, \ldots, X_N$:

$$Y_i \;\perp\!\!\!\perp\; Y_j \;\mid\; X_0, X_1, \ldots, X_N \qquad \text{for all } i \neq j.$$

Using this conditional independence, the variance of the total return under $\hat{p}_\pi$ factorizes:

$$\mathrm{Var}_{\hat{p}_\pi}[\hat{J} \mid X] = \mathrm{Var}_{\hat{p}_\pi}\left[\sum_{i=0}^{N-1} \gamma^{iw} Y_i \;\middle|\; X\right] = \sum_{i=0}^{N-1} \gamma^{2iw} \cdot \mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i).$$

To bound the difference in conditional variances, we apply the law of variance

$$\mathrm{Var}(Y_i \mid X_i) = \mathbb{E}[Y_i^2 \mid X_i] - \left(\mathbb{E}[Y_i \mid X_i]\right)^2.$$

Let us define a bound on the per-chunk return magnitude:

$$B_w := \frac{R_{\max}(1-\gamma^w)}{1-\gamma} \quad \Rightarrow \quad |Y_i| \le B_w, \quad Y_i^2 \le B_w^2.$$

Using Lemma C.6 (Expectation Difference Bound via Total Variation), we have

$$|\mathbb{E}_{p_\pi}[f] - \mathbb{E}_{\hat{p}_\pi}[f]| \le 2\delta_\pi \cdot \|f\|_\infty.$$

Applying this with $f = Y_i$ and $f = Y_i^2$, and using the bound $|Y_i| \le B_w$, we obtain:

$$|\mathbb{E}_{p_\pi}[Y_i] - \mathbb{E}_{\hat{p}_\pi}[Y_i]| \le 2\delta_\pi B_w, \qquad \left|\mathbb{E}_{p_\pi}[Y_i^2] - \mathbb{E}_{\hat{p}_\pi}[Y_i^2]\right| \le 2\delta_\pi B_w^2.$$

We analyze the difference in conditional variances:

$$\begin{aligned}
&|\mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i) - \mathrm{Var}_{p_\pi}(Y_i \mid X_i)| \\
&= \left|\mathbb{E}_{\hat{p}_\pi}[Y_i^2] - \mathbb{E}_{p_\pi}[Y_i^2] - \left(\mathbb{E}_{\hat{p}_\pi}[Y_i]^2 - \mathbb{E}_{p_\pi}[Y_i]^2\right)\right| \\
&\le \left|\mathbb{E}_{\hat{p}_\pi}[Y_i^2] - \mathbb{E}_{p_\pi}[Y_i^2]\right| + \left|\mathbb{E}_{\hat{p}_\pi}[Y_i]^2 - \mathbb{E}_{p_\pi}[Y_i]^2\right| \\
&= \left|\mathbb{E}_{\hat{p}_\pi}[Y_i^2] - \mathbb{E}_{p_\pi}[Y_i^2]\right| + |\mathbb{E}_{\hat{p}_\pi}[Y_i] - \mathbb{E}_{p_\pi}[Y_i]| \cdot |\mathbb{E}_{\hat{p}_\pi}[Y_i] + \mathbb{E}_{p_\pi}[Y_i]| \\
&\le 2\delta_\pi B_w^2 + (2\delta_\pi B_w)(2B_w) = 6\delta_\pi B_w^2.
\end{aligned}$$

This uses the triangle inequality and the identity $|a^2 - b^2| = |a-b||a+b|$, along with the bounds $|Y_i| \le B_w$, $\|Y_i\|_\infty^2 \le B_w^2$, and total variation guarantees from Lemma C.6. Then

$$|\mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i) - \mathrm{Var}_{p_\pi}(Y_i \mid X_i)| \le 6\delta_\pi B_w^2.$$

We now return to bounding the first term in the law of total variance

$$\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(\hat{J} \mid X)\right] = \mathbb{E}_{\hat{p}_\pi}\left[\sum_{i=0}^{N-1} \gamma^{2iw} \cdot \mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i)\right].$$

Using the bound from the previous step

$$\mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i) \le \mathrm{Var}_{p_\pi}(Y_i \mid X_i) + 6\delta_\pi B_w^2.$$

Taking expectation over $\hat{p}_\pi$ on both sides

$$\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i)\right] \le \mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right] + 6\delta_\pi B_w^2.$$

Now, using the expectation difference bound from Lemma C.6 again:

$$|\mathbb{E}_{\hat{p}_\pi}[f] - \mathbb{E}_{p_\pi}[f]| \le 2\delta_\pi \|f\|_\infty, \qquad \text{where} \quad f(X_i) := \mathrm{Var}_{p_\pi}(Y_i \mid X_i) \le B_w^2.$$

So

$$\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right] \le \mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right] + 2\delta_\pi B_w^2.$$

Combining both components

$$\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i)\right] \le \mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right] + 8\delta_\pi B_w^2.$$

Summing across all chunks:

$$\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(\hat{J} \mid X)\right] = \sum_{i=0}^{N-1} \gamma^{2iw} \cdot \mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(Y_i \mid X_i)\right] \le \sum_{i=0}^{N-1} \gamma^{2iw} \left(\mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right] + 8\delta_\pi B_w^2\right).$$

We can split the sum and factor out constants:

$$\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(\hat{J} \mid X)\right] = \sum_{i=0}^{N-1} \gamma^{2iw} \cdot \mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right] + 8\delta_\pi B_w^2 \sum_{i=0}^{N-1} \gamma^{2iw}.$$

Let us define the chunk-level return variance

$$\mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(\hat{J} \mid X)\right] := \sum_{i=0}^{N-1} \gamma^{2iw} \cdot \mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(Y_i \mid X_i)\right].$$

Therefore

$$\boxed{\mathbb{E}_{\hat{p}_\pi}\left[\mathrm{Var}_{\hat{p}_\pi}(\hat{J} \mid X)\right] \le \mathbb{E}_{p_\pi}\left[\mathrm{Var}_{p_\pi}(\hat{J} \mid X)\right] + \frac{8\delta_\pi B_w^2}{1-\gamma^{2w}}.}$$

To complete the law of total variance, we now analyze the second term:

$$\mathrm{Var}_{\hat{p}_\pi}\left(\mathbb{E}_{\hat{p}_\pi}[\hat{J} \mid X]\right) = \mathrm{Var}_{\hat{p}_\pi}(Z_{\hat{p}}), \quad \text{where } Z_{\hat{p}} := \sum_{k=0}^{N-1} g_k(X_k), \quad g_k(x) := \mathbb{E}_{\hat{p}_\pi}[Y_k \mid X_k = x].$$

We define the corresponding ideal (true model) version:

$$Z_p := \sum_{k=0}^{N-1} \tilde{g}_k(X_k), \qquad \tilde{g}_k(x) := \mathbb{E}_{p_\pi}[Y_k \mid X_k = x].$$

Our goal is to bound the variance difference:

$$\Delta_{\text{mean}} := \mathrm{Var}_{\hat{p}_\pi}(Z_{\hat{p}}) - \mathrm{Var}_{p_\pi}(Z_p) = (M_{\hat{p}} - M_p) - (m_{\hat{p}} - m_p)(m_{\hat{p}} + m_p),$$

where $M_{\hat{p}} := \mathbb{E}_{\hat{p}_\pi}[Z_{\hat{p}}^2]$, $m_{\hat{p}} := \mathbb{E}_{\hat{p}_\pi}[Z_{\hat{p}}]$, and similarly for M_p, m_p.

Insert and subtract a common term:

$$m_{\hat{p}} - m_p = \sum_{k=0}^{N-1} \left(\mathbb{E}_{\hat{p}_\pi}[g_k(X_k)] - \mathbb{E}_{\hat{p}_\pi}[\tilde{g}_k(X_k)]\right) + \sum_{k=0}^{N-1} \left(\mathbb{E}_{\hat{p}_\pi}[\tilde{g}_k(X_k)] - \mathbb{E}_{p_\pi}[\tilde{g}_k(X_k)]\right).$$

Each term is bounded by $2\delta_\pi B_w$, so $|m_{\hat{p}} - m_p| \le 4N\delta_\pi B_w$.

Expand both squares:

$$Z_{\hat{p}}^2 = \sum_{k=0}^{N-1} g_k^2(X_k) + 2 \sum_{0 \le k < \ell \le N-1} g_k(X_k) g_\ell(X_\ell),$$

$$Z_p^2 = \sum_{k=0}^{N-1} \tilde{g}_k^2(X_k) + 2 \sum_{0 \le k < \ell \le N-1} \tilde{g}_k(X_k) \tilde{g}_\ell(X_\ell).$$

Each term (both diagonal and cross terms) is bounded in total variation with sup-norm B_w^2, yielding

$$|M_{\hat{p}} - M_p| \leq 2N^2 \delta_\pi B_w^2.$$

From the bound on the means:

$$|m_{\hat{p}}|, |m_p| \leq NB_w \quad \Rightarrow \quad |m_{\hat{p}} + m_p| \leq 2NB_w.$$

So, the product term:

$$|(m_{\hat{p}} - m_p)(m_{\hat{p}} + m_p)| \leq (4N\delta_\pi B_w)(2NB_w) = 8N^2 \delta_\pi B_w^2.$$

Combining both:

$$|\Delta_{\text{mean}}| = |\text{Var}_{\hat{p}_\pi}(Z_{\hat{p}}) - \text{Var}_{p_\pi}(Z_p)| \leq 2N^2 \delta_\pi B_w^2 + 8N^2 \delta_\pi B_w^2 = 10N^2 \delta_\pi B_w^2,$$

which yields

$$\boxed{\left|\text{Var}_{\hat{p}_\pi}\left(\mathbb{E}_{\hat{p}_\pi}[\hat{J} \mid X]\right) - \text{Var}_{p_\pi}\left(\mathbb{E}_{p_\pi}[J \mid X]\right)\right| \leq 10 \cdot \frac{T^2}{w^2} \cdot \delta_\pi B_w^2.}$$

Combining the two components from the law of total variance, we conclude:

$$\begin{aligned}
\text{Var}_{\hat{p}_\pi}(\hat{J}) &= \mathbb{E}_{\hat{p}_\pi}\left[\text{Var}_{\hat{p}_\pi}(\hat{J} \mid X)\right] + \text{Var}_{\hat{p}_\pi}\left(\mathbb{E}_{\hat{p}_\pi}[\hat{J} \mid X]\right) \\
&\leq \mathbb{E}_{p_\pi}\left[\text{Var}_{p_\pi}(\hat{J} \mid X)\right] + \frac{8\delta_\pi B_w^2}{1-\gamma^{2w}} + \text{Var}_{p_\pi}\left(\mathbb{E}_{p_\pi}[J \mid X]\right) + 10\left(\frac{T}{w}\right)^2 \delta_\pi B_w^2 \\
&= \text{Var}_{p_\pi}(J) + 10\left(\frac{T}{w}\right)^2 \delta_\pi B_w^2 + \frac{8\delta_\pi B_w^2}{1-\gamma^{2w}}.
\end{aligned}$$

By Lemma C.5,

$$\boxed{\text{Var}_{\hat{p}_\pi}(\hat{J}) \leq \text{Var}_{p_\pi}(J) + 10\left(\frac{T}{w}\right)^2 B_w^2 \kappa^w \delta_\beta + \frac{8B_w^2}{1-\gamma^{2w}} \kappa^w \delta_\beta,}$$

and the proof is complete. ∎

D. Proof of the Bias-Variance Decomposition

Finally, we can bound the mean squared error of STITCH-OPE.

Theorem C.12. *Under Assumption C.3 and C.4, and using the notation of Theorem C.8 and Theorem C.11, the mean squared error of STITCH-OPE is bounded by*

$$\mathbb{E}_{\hat{p}_\pi}\left[(\hat{J} - J(\pi))^2\right] \leq \left(\frac{2B_w}{1-\gamma^w}\kappa^w \delta_\beta\right)^2 + 10\left(\frac{T}{w}\right)^2 B_w^2 \kappa^w \delta_\beta + \frac{8B_w^2}{1-\gamma^{2w}}\kappa^w \delta_\beta + \text{Var}_{p_\pi}(J).$$

Proof: We start by adapting the standard bias-variance decomposition to our setting:

$$\begin{aligned}
\mathbb{E}_{\hat{p}_\pi}\left[(\hat{J} - J(\pi))^2\right] &= \mathbb{E}_{\hat{p}_\pi}\left[(\hat{J} - \mathbb{E}_{\hat{p}_\pi}[\hat{J}] + \mathbb{E}_{\hat{p}_\pi}[\hat{J}] - J(\pi))^2\right] \\
&= \mathbb{E}_{\hat{p}_\pi}\left[(\hat{J} - \mathbb{E}_{\hat{p}_\pi}[\hat{J}])^2\right] + \mathbb{E}_{\hat{p}_\pi}\left[(\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - J(\pi))^2\right] \\
&\quad + \mathbb{E}_{\hat{p}_\pi}\left[(\hat{J} - \mathbb{E}_{\hat{p}_\pi}[\hat{J}])(\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - J(\pi))\right] \\
&= \text{Var}_{\hat{p}_\pi}(\hat{J}) + \text{Bias}_{\hat{p}_\pi}(\hat{J})^2 + (\mathbb{E}_{\hat{p}_\pi}[\hat{J}] - J(\pi))(\mathbb{E}_{\hat{p}_\pi}[\hat{J} - \mathbb{E}_{\hat{p}_\pi}[\hat{J}]]) \\
&= \text{Var}_{\hat{p}_\pi}(\hat{J}) + \text{Bias}_{\hat{p}_\pi}(\hat{J})^2,
\end{aligned}$$

since the last term is zero. Plugging in the bounds of Theorems C.8 and C.11 completes the proof. ∎

APPENDIX D
PSEUDOCODE

A high-level pseudocode of conditional diffusion model training in STITCH-OPE is provided as Algorithm 1. A pseudocode of the off-policy evaluation subroutine for a single rollout is provided as Algorithm 2. Empirically, we have found that per-term normalization of the guidance function (line 9) resulted in more consistent performance, and allowed the guidance coefficients α and λ to be more easily tuned.

Algorithm 1 Conditional Diffusion Model Training in STITCH-OPE

Require: diffusion model $\epsilon_\theta(\tau, k|s)$, behavior data $\mathcal{D}_\beta$, $w \geq 0$, learning rate $\eta > 0$, $\{\sigma_k\}_{k=1}^K$ and $\{\alpha_k\}_{k=1}^K$ positive

$\bar{\alpha}_k \leftarrow \prod_{t=1}^k \alpha_t$ **for** $k = 1 \dots K$
initialize θ randomly
repeat
  sample length-w sub-trajectory $\tau^0 = (s_0, a_0, s_1, \dots s_w)$ from $\mathcal{D}_\beta$
  sample $k \sim \text{Uniform}(\{1, \dots K\})$ ▷ Sample denoising time step k
  sample $\epsilon \sim \mathcal{N}(0, I)$ ▷ Sample pure noise sub-trajectory
  $\nabla_\theta \mathcal{L}(\theta) \leftarrow \nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_k}\tau^0 + \sigma_k \epsilon, k|s_0)\|^2$ ▷ Gradient descent step on θ
  $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$
until converged
return ϵ_θ

Algorithm 2 Off-Policy Evaluation in STITCH-OPE

Require: diffusion model $\epsilon_\theta(\tau, k|s)$ (Algorithm 1), empirical reward function $\hat{R}(s, a)$, behavior policy $\beta(a|s)$, target policy $\pi(a|s)$, $\alpha \geq 0$, $\lambda \geq 0$, $w \geq 0$ (divides T), $\{\sigma_k\}_{k=1}^K$ and $\{\alpha_k\}_{k=1}^K$ positive

$\hat{J} \leftarrow 0$
sample $s_0^0 \sim d_0$ ▷ Sample initial state
for $t = 0$ **to** $T/w - 1$ **do** ▷ Generation for decision epochs wt to $w(t+1)$
  sample $\tau^K_{wt:w(t+1)} \sim \mathcal{N}(0, I)$ ▷ Sample pure noise sub-trajectory
  for $k = K$ **to** 1 **do** ▷ Denoising step k
    $\mu_{k-1} \leftarrow \frac{1}{\sqrt{\alpha_k}} \left(\tau^k_{wt:w(t+1)} - \frac{1-\alpha_k}{\sigma_k} \epsilon_\theta(\tau^k_{wt:w(t+1)}, k \mid s^0_{wt}) \right)$ ▷ Mean of diffusion
    $g_k^\pi \leftarrow \sum_{u=wt}^{w(t+1)-1} \nabla_\tau \log \pi(a_u^k | s_u^k)$ ▷ Compute π guidance term
    $g_k^\beta \leftarrow \sum_{u=wt}^{w(t+1)-1} \nabla_\tau \log \beta(a_u^k | s_u^k)$ ▷ Compute β guidance term
    $g_k \leftarrow \alpha(g_k^\pi / \|g_k^\pi\|_2) - \lambda(g_k^\beta / \|g_k^\beta\|_2)$ ▷ Compute normalized guidance
    sample $\tau^{k-1}_{wt:w(t+1)} \sim \mathcal{N}\left(\mu_k + \sigma_k^2 g_k, \sigma_k^2 I\right)$ ▷ Apply guided diffusion step
  end for
  $\hat{J} \leftarrow \hat{J} + \sum_{u=wt}^{w(t+1)-1} \gamma^u \hat{R}(s_u^0, a_u^0)$ ▷ Update π return using denoised $\tau^0_{wt:w(t+1)}$
end for
return $\hat{J}$

APPENDIX E
DOMAINS

We include experiments on the medium datasets from the D4RL offline suite [15], and Pendulum and Acrobot domains from the OpenAI Gym suite [5]. We set the evaluation horizon to $T = 768$ for D4RL, $T = 256$ for Acrobot and $T = 196$ for Pendulum, and we use $\gamma = 0.99$ in all experiments. Furthermore, Acrobot uses a discrete action space and is incompatible with our method, so we modified the domain to take continuous actions. Table II summarizes the key properties of each domain.

Description	Hopper	Walker	HalfCheetah	Pendulum	Acrobot
state dimension	11	17	17	3	6
action dimension	3	6	6	1	3
range of action	$[-1, 1]$	$[-1, 1]$	$[-1, 1]$	$[-2, 2]$	$[-1, 1]$
rollout length T	768	768	768	196	256
discount factor γ	0.99	0.99	0.99	0.99	0.99

TABLE II
PROPERTIES OF D4RL [15] AND OPENAI GYM [5] BENCHMARK PROBLEMS.

APPENDIX F
POLICIES

a) D4RL Offline Suite: Behavior and target policies and their trained procedures are described in [16], and the policy parameters are borrowed from the official repository at https://github.com/google-research/deep_ope (Apache 2.0 licensed). The 10 target policies of varying ability, $\pi_{\theta_1}, \pi_{\theta_2}, \dots \pi_{\theta_{10}}$, are obtained by checkpointing the policy parameters $\theta_1, \theta_2 \dots \theta_{10}$

at various points during training. Each target policy network models the action probability distribution $\pi_i(a|s)$ using a set of independent Gaussian distributions, predicting the mean and variance (μ_i, σ_i^2) of each action component a_i independently. This allows the score function of the target policy to be easily computed. As discussed in the main text, all policies are derived from the medium datasets in all experiments.

b) OpenAI Gym: We model target policies $\pi_1, \pi_2 \dots \pi_5$ as MLPs and train them in each environment following the Twin-Delayed DDPG (TD3) [9] algorithm. The total training time is set to 50000 steps, and we checkpoint policies every 5000 steps. The behavior policy is set to the target policy π_3. The complete list of hyper-parameters is provided in Table III.

Description	Value
number of hidden layers in actor and critic	2
number of neurons per layer in actor and critic	256
hidden activation function	ReLU
output activation function	$\tanh$
Gaussian noise for exploration	0.1
noise added to target policy during critic update	0.2
target noise clipping	0.5
frequency of delayed policy updates	2
moving average of target θ'	0.005
learning rate of Adam optimizer	0.0003
batch size	256
replay buffer size	1000000

TABLE III
HYPER-PARAMETERS FOR TRAINING TARGET POLICIES ON OPENAI GYM DOMAINS.

c) Bounded Action Space: Since the action spaces for all domains are compact bounded intervals, we need to restrict the action space of the policy networks during evaluation. We accomplish this by applying the $\tanh$ transformation to each Gaussian action distribution and then scaling the result to the required range. Note that this transformation constrains the action probability distribution of all policies to a bounded range, and thus satisfies the requirement of Assumption C.3.

APPENDIX G
BASELINES

The following model-free baseline methods were chosen for empirical comparison with STITCH-OPE:

a) Fitted Q-Evaluation (FQE): [26] evaluates a target policy π by estimating its Q-value function $Q_\theta(s,a)$ using a neural network. The loss function for θ is

$$\mathcal{L}_{FQE}(\theta) = \mathbb{E}_{\substack{(s,a,r,s')\sim\mathcal{D}_\beta, \\ a'\sim\pi(\cdot|s')}} \left[\left(Q_\theta(s,a) - r - \gamma Q_\theta(s',a')\right)^2 \right].$$

We follow [38, 25] and learn a target Q-network $Q_{\theta'}(s,a)$ in parallel for added stability. We use the AdamW algorithm [34] for optimizing the loss function in a minibatched setting, with gradient clipping applied to limit the norm of each gradient update to 1. The complete list of hyper-parameters used is provided in Table IV.

Description	Hopper	Walker	HalfCheetah	Pendulum	Acrobot
number of hidden layers	2	2	2	2	2
number of neurons per layer	500	500	500	256	100
hidden activation function	sigmoid	sigmoid	sigmoid	sigmoid	sigmoid
learning rate of AdamW optimizer	0.001	0.003	0.00003	0.003	0.001
moving average of target θ'	0.05	0.05	0.001	0.005	0.05
training epochs (passes over data set)	100	50	70	100	200
batch size	512	256	256	128	512

TABLE IV
HYPER-PARAMETERS FOR FITTED Q-EVALUATION (FQE).

b) Doubly Robust (DR): [21, 49] leverages both importance sampling and value function estimation to construct a combined estimate that is accurate when either one of the individual estimates is correct. First, we define an estimate $\hat{Q}(s,a)$ of the Q-value function of policy π, and let $\hat{V}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[\hat{Q}(s,a)]$ be the corresponding value estimate. We also define $\rho_t = \frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}$ as the policy ratio at step t. Then, the DR estimator is defined recursively as

$$V_{DR}^{t+1} = \hat{V}(s_t) + \rho_t \left(r_t + \gamma V_{DR}^t - \hat{Q}(s_t,a_t) \right),$$

such that the policy value estimate $\hat{J}_{DR}(\pi) = V^0_{DR}$. We parameterize both $\hat{Q}(s,a)$ and $\hat{V}(s)$ as MLPs and train them using AdamW in a mini-batched setting. Similar to FQE, we also update a target value network to improve convergence. The full list of hyper-parameters is provided in Table V.

Description	Hopper	Walker	HalfCheetah	Pendulum	Acrobot
number of hidden layers	2	2	2	2	2
number of neurons per layer	500	500	500	256	100
hidden activation function	sigmoid	sigmoid	sigmoid	sigmoid	sigmoid
learning rate of AdamW optimizer	0.0003	0.003	0.003	0.003	0.00003
moving average of target θ'	0.05	0.05	0.05	0.05	0.001
training epochs (passes over data set)	50	50	50	100	100
batch size	32	256	512	256	128

TABLE V
HYPER-PARAMETERS FOR DOUBLY ROBUST (DR) ESTIMATION.

c) Importance Sampling (IS): [43] evaluates the target policy by importance weighting the full trajectory returns in the behavior dataset, i.e.

$$\hat{J}_{IS}(\pi) = \mathbb{E}_{\tau\sim p_\beta}\left[\left(\prod_{t=0}^{T-1}\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}\right)\sum_{t=0}^{T-1}\gamma^t R(s_t,a_t)\right].$$

It requires access to the target and behavior policy probabilities in order to compute the weighting. Specifically, we use the *per-decision* variant of IS (PDIS), i.e.

$$\hat{J}_{PDIS}(\pi) = \mathbb{E}_{\tau\sim p_\beta}\left[\sum_{t=0}^{T-1}\gamma^t\left(\prod_{u=0}^{t}\frac{\pi(a_u|s_u)}{\beta(a_u|s_u)}\right)R(s_t,a_t)\right],$$

which has lower variance than IS.

d) Density Ratio Estimation (DRE): [39] estimates the ratio $w(s,a) = d^\pi(s,a)/d^\beta(s,a)$ of the discounted state-action occupancies of the target policy π relative to the behavior policy β. The *discounted state-action occupancy* of a policy $\mu \in \{\beta, \pi\}$ is defined as

$$d^\mu(s,a) = \lim_{T\to\infty}\frac{\sum_{t=0}^{T}\gamma^t p(s_t = s,\, a_t = a \,|\, \mu)}{\sum_{t=0}^{T}\gamma^t},$$

where $p(s_t = s,\, a_t = a \,|\, \mu)$ indicates the probability of sampling state-action pair (s,a) from μ at time step t. We also tested the variants of DICE [54] but found their performance to be unsatisfactory, so they have been omitted from the study. The target policy value is estimated as

$$\hat{J}(\pi) = \frac{1}{1-\gamma}\mathbb{E}_{(s,a,r)\sim\mathcal{D}_\beta}[w(s,a)\cdot r].$$

$w(s,a)$ is parameterized as a feedforward neural network and its parameters are trained using Adam in a mini-batched setting. Fixed hyper-parameters necessary to reproduce the experiment are listed in Table VI. Additionally, since the method requires a kernel function to be specified, we use a Gaussian kernel $k(x,x') = \exp\left(-\eta\|x-x'\|^2\right)$, where x and x' are concatenations of the (standardized) state and action vectors. Since this requires setting a kernel bandwidth $\eta > 0$ which affects the overall performance significantly, we run this baseline for different values $\eta \in \{0.01, 0.1, 1, 10, 100\}$ and report the best performing result (according to log-RMSE).

Description	Value
number of hidden layers of $w(s,a)$	2
number of neurons per layer of $w(s,a)$	256
hidden activation function	Leaky ReLU
output activation function	SoftPlus
learning rate of Adam optimizer	0.001
training epochs (passes over the data set)	20 (D4RL), 200 (Gym)
batch size	512

TABLE VI
HYPER-PARAMETERS FOR DENSITY RATIO ESTIMATION (DRE) [39].

The following model-based baseline methods were also chosen for empirical comparison with STITCH-OPE. They were chosen to determine the benefits of STITCH-OPE compared to fully autoregressive sampling, i.e. $w = 1$, and non-autoregressive sampling, i.e. $w = T$.

e) Model-Based (MB): [21, 52] consists of learning dynamics $\hat{P}(s'|s,a)$, reward function $\hat{R}(s,a)$ and termination function $\hat{D}(s)$ trained on the behavior dataset to directly approximate the data-generating distribution of the target policy, $p_\pi(\tau)$. $\hat{P}$ directly predicts the next state s' given the current state s and action a. Both $\hat{P}$ and $\hat{R}$ can be found by solving a standard nonlinear regression problem, and $\hat{D}$ can be found by solving a binary classification problem trained on termination flags in the behavior dataset. We parameterize all functions as nonlinear MLPs and obtain their optimal parameters using Adam in a mini-batched setting. Once we obtain their optimal parameters, we estimate the target policy return by generating 50 length-T rollouts from the estimated model, and average their empirical cumulative returns. The necessary hyper-parameters are described in Table VII.

Description	Value
number of hidden layers	3
number of neurons per layer	500
hidden activation function	ReLU
learning rate of Adam optimizer	0.0003
training epochs (passes over data set)	100
batch size	1024

TABLE VII
HYPER-PARAMETERS FOR MODEL-BASED (MB) ESTIMATION.

f) Policy-Guided Diffusion (PGD): [18] takes a generative approach by simulating target policy trajectories using a guided diffusion model. We follow the original implementation by training a diffusion model on the behavior data, using the official implementation located at https://github.com/EmptyJackson/policy-guided-diffusion (MIT licensed). We then generate 50 full-length trajectories from the model using guided diffusion [20] with the guidance function $g_{simple}(\tau) = \nabla_\tau \sum_t \log \pi(a_t|s_t)$, using which we estimate the empirical return of the target policy. All hyper-parameters for training the diffusion models are fixed as per the original paper and codebase (see Appendix A therein for details). However, we found that the policy guidance coefficient α and guidance normalization both have significant effects on performance, thus we ran PGD for different choices of $\alpha \in \{0.001, 0.01, 0.1, 1.0, 10, 100, 1000\}$ with and without guidance normalization, and report the best performing result (according to log-RMSE).

APPENDIX H
METRICS

Let $\pi_1, \dots \pi_{10}$ be the target policies, $\hat{J}_1(\pi_i), \hat{J}_2(\pi_i), \dots \hat{J}_5(\pi_i)$ be the estimates of the target policy values across the 5 seeds, and $J(\pi_1), \dots J(\pi_{10})$ be the target policy values estimated using 300 rollouts collected by running the target policies in the environments.

The following metrics were used to quantify and compare the performance of STITCH-OPE and all metrics:

a) Log Root Mean Squared Error (LogRMSE): This is defined as the log root mean squared error using the estimates $\hat{J}_j(\pi_1), \dots \hat{J}_j(\pi_{10})$ and the ground truth returns $J(\pi_1), \dots \dots J(\pi_{10})$, averaged across seeds $j = 1 \dots 5$. Mathematically,

$$\frac{1}{5}\sum_{j=1}^{5} \log \sqrt{\frac{1}{10}\sum_{i=1}^{10}(\hat{J}_j(\pi_i) - J(\pi_i))^2}.$$

b) Spearman (Rank) Correlation: This is defined as the Spearman correlation [47] between the estimates $\hat{J}_j(\pi_1), \dots \hat{J}_j(\pi_{10})$ and the ground truth returns $J(\pi_1), \dots \dots J(\pi_{10})$, averaged across seeds $j = 1 \dots 5$.

c) Regret@1: This is defined as the absolute difference in return between the best policy selected using the baseline policy returns $\hat{J}_j(\pi_i)$ and the policy selected according to the ground truth estimates $J(\pi_i)$, averaged across seeds $j = 1 \dots 5$, i.e:

$$\frac{1}{5}\sum_{j=1}^{5} \left| J(\pi_{i_j^{max}}) - \max_{i=1\dots10} J(\pi_i) \right|, \qquad \text{where} \quad i_j^{max} = \text{argmax}_{i=1\dots10}\, \hat{J}_j(\pi_i).$$

d) Normalization: In order to compare metrics consistently across environments, we follow [16] and use the normalized policy values:

$$\frac{\hat{J}_j(\pi_i) - V_{min}}{V_{max} - V_{min}}, \qquad \text{where} \quad V_{min} = \min_i J(\pi_i), \quad V_{max} = \max_i J(\pi_i),$$

where V_{min} and V_{max} are the minimum and maximum target policy values, respectively.

e) Error Bars: All tables and figures report error bars defined as +/- one standard error, i.e. $\hat{\sigma}/\sqrt{n}$ where $\hat{\sigma}$ is the empirical standard deviation of each metric value across seeds and n is the number of seeds (fixed to 5 for all experiments).

APPENDIX I
STITCH-OPE TRAINING AND HYPER-PARAMETER DETAILS

We follow the configuration used in [20] for training the diffusion model, including architecture, optimizer, and noise schedule. Specifically, we parameterize the diffusion process ϵ as a UNet architecture with residual connections [46], trained with a cosine learning rate schedule [35]. The list of training hyper-parameters is provided in Table VIII. The reward predictor $\hat{R}(s, a)$ is a two-layer MLP with ReLU activations and 32 neurons per hidden layer, and is trained using Adam with a learning rate of 0.001 and batch size of 64.

Description	Value
diffusion architecture	UNet
learning rate of Adam optimizer	0.0003
training epochs (passes over the data set)	150
batch size	128
training steps per epoch	5000 (D4RL), 2000 (Gym)
guidance coefficient for π, i.e. α	0.5 (D4RL), 0.1 (Gym)
guidance coefficient ratio for β , i.e. $\frac{\lambda}{\alpha}$	0.5 (D4RL), 1 (Gym)
window size of sub-trajectories, i.e. w	8 (D4RL), 16 (Gym)

TABLE VIII
HYPER-PARAMETERS FOR STITCH-OPE.

a) Guidance Coefficients: For Gym domains, we use $\alpha = \lambda = 1$, corresponding to the theoretically justified guidance function in Equation 6, assuming low distribution shift. For D4RL tasks, we use tempered values $\alpha = 0.5$ and $\lambda = 0.25$ to improve sample stability and regularization, which we found empirically helpful in higher-dimensional settings.

b) Sub-Trajectory Length: We use $w = 16$ for Gym domains and $w = 8$ for most D4RL tasks. For HalfCheetah, we reduce to $w = 4$ due to the environment's fast dynamics, which caused degradation in stitching fidelity with longer sub-trajectories.

APPENDIX J
DIFFUSION POLICY TRAINING AND EVALUATION

We follow [53] and parameterize each target policy π'_i, $i = 1 \dots 10$ as a conditional diffusion model $\epsilon_{\phi_i}(a^k, k|s)$, whose parameters ϕ_i are learned by optimizing the behavior cloning objective.

$$\mathcal{L}(\phi_i) = \mathbb{E}_{k,\, \epsilon\sim\mathcal{N}(0,I),\, s\sim\mathcal{D}_\beta,\, a\sim\pi_i(\cdot|s)} \left[\|\epsilon - \epsilon_{\phi_i}(a^k, k|s)\|^2 \right].$$

In order to use the fine-tuned $\epsilon_{\phi_i}(a^k, k|s)$ as a guidance function for off-policy evaluation in STITCH-OPE, we use the following equivalence between score-based models and denoising diffusion [10] (extended trivially to the conditional setting)

$$\nabla_a \log \pi'_i(a|s)|_{a=a^k} = -\frac{\epsilon_{\phi_i}(a^k, k|s)}{\sigma_k}.$$

Specifically, this expression cannot be calculated at $k = 0$ since $\sigma_0 = 0$ using the standard parameterization of diffusion models, so we approximate it at $k = 1$ and use the resulting gradient in STITCH-OPE.

We implement the diffusion model using the CleanDiffuser package [11] with official repository at https://github.com/CleanDiffuserTeam/CleanDiffuser (Apache 2.0 licensed). To train the diffusion policies, we first generate rollouts from each of the pre-trained target policies in D4RL [16], and then minimize the behavior cloning objective $\mathcal{L}(\phi_i)$ above to obtain the diffusion policy parameters. The list of relevant hyper-parameters is provided in Table IX.

Description	Value
embedding dimension	64
hidden layer dimension	256
learning rate	0.0003
diffusion time steps	32
EMA rate	0.9999
total training steps	10000
number of transitions to generate for each dataset	1000000
training batch size	256

TABLE IX
HYPER-PARAMETERS FOR TRAINING DIFFUSION POLICIES.

		FQE	DRE	MBR	PGD	**Ours**
Log RMSE ↓	Hopper	-0.21 ± 0.01	-0.38 ± 0.00	-1.56 ± 0.02	-0.89 ± 0.00	**-1.65 ± 0.01**
	Walker2d	-0.59 ± 0.01	-0.49 ± 0.00	-0.81 ± 0.01	-0.50 ± 0.00	**-1.20 ± 0.01**
	HalfCheetah	-0.19 ± 0.00	**-1.19 ± 0.00**	-0.24 ± 0.01	-0.96 ± 0.00	-0.50 ± 0.00
Rank Corr. ↑	Hopper	0.35 ± 0.06	0.35 ± 0.04	0.68 ± 0.02	0.45 ± 0.00	**0.81 ± 0.01**
	Walker2d	0.03 ± 0.04	0.45 ± 0.03	0.47 ± 0.02	**0.52 ± 0.01**	0.46 ± 0.09
	HalfCheetah	0.59 ± 0.01	0.80 ± 0.03	0.75 ± 0.05	0.46 ± 0.06	**0.81 ± 0.02**
Regret@1 ↓	Hopper	0.06 ± 0.03	0.41 ± 0.22	0.18 ± 0.00	**¡0.01 ± 0.00**	**¡0.01 ± 0.00**
	Walker2d	0.24 ± 0.02	0.59 ± 0.13	0.17 ± 0.02	0.23 ± 0.00	**0.03 ± 0.00**
	HalfCheetah	**¡0.01 ± 0.00**	**¡0.01 ± 0.00**	0.03 ± 0.01	0.02 ± 0.01	0.02 ± 0.01

TABLE X

COMPARISON OF OPE METHODS ACROSS ENVIRONMENTS WHEN THE TARGET POLICY IS A DIFFUSION POLICY; ANY REGRET SHOWN AS ¡0.01 IS NONZERO BUT ROUNDS TO ZERO AT TWO DECIMALS.

APPENDIX K
ADDITIONAL EXPERIMENTS

The complete breakdown of performance across each environment is provided in Table XI. This includes separate evaluations for HalfCheetah, Hopper, Walker2d, Pendulum, and Acrobot across all evaluated metrics (LogRMSE, Spearman Correlation, and Regret@1). These results support the claim that STITCH-OPE consistently outperforms existing baselines across tasks of varying complexity and dynamics.

		FQE	DR	IS	DRE	MB	PGD	**Ours**
Log RMSE ↓	Hopper	-0.42 ± 0.03	-0.57 ± 0.02	-0.48 ± 0.01	-0.42 ± 0.00	-1.70 ± 0.04	-1.22 ± 0.02	**-2.33 ± 0.02**
	Walker2d	-0.48 ± 0.01	-1.25 ± 0.08	-0.71 ± 0.01	-0.45 ± 0.00	-0.88 ± 0.01	-0.32 ± 0.01	**-1.33 ± 0.01**
	HalfCheetah	-0.05 ± 0.00	0.01 ± 0.01	-0.84 ± 0.02	-1.19 ± 0.00	-0.37 ± 0.00	**-1.47 ± 0.00**	-0.85 ± 0.01
	Pendulum	-0.58 ± 0.00	-1.02 ± 0.04	-0.15 ± 0.00	-0.58 ± 0.00	-0.43 ± 0.01	-0.91 ± 0.01	**-2.34 ± 0.07**
	Acrobot	-0.14 ± 0.00	-0.49 ± 0.06	-1.00 ± 0.01	0.20 ± 0.01	-1.54 ± 0.02	-0.13 ± 0.01	**-2.02 ± 0.05**
Rank Corr. ↑	Hopper	0.17 ± 0.05	0.69 ± 0.06	-0.06 ± 0.13	-0.09 ± 0.14	0.52 ± 0.03	0.36 ± 0.09	**0.76 ± 0.02**
	Walker2d	0.41 ± 0.05	0.50 ± 0.02	0.51 ± 0.11	0.42 ± 0.05	**0.65 ± 0.04**	-0.07 ± 0.10	0.63 ± 0.03
	HalfCheetah	-0.03 ± 0.06	-0.48 ± 0.07	0.57 ± 0.06	0.80 ± 0.02	0.32 ± 0.03	0.50 ± 0.00	**0.87 ± 0.01**
	Pendulum	0.89 ± 0.03	0.72 ± 0.07	-0.60 ± 0.00	-0.40 ± 0.15	0.84 ± 0.06	0.54 ± 0.02	**0.96 ± 0.02**
	Acrobot	0.75 ± 0.02	0.63 ± 0.08	0.52 ± 0.01	0.01 ± 0.12	0.53 ± 0.11	0.43 ± 0.14	**0.82 ± 0.04**
Regret@1 ↓	Hopper	0.13 ± 0.03	0.05 ± 0.02	0.13 ± 0.02	0.27 ± 0.17	0.04 ± 0.03	**0.04 ± 0.01**	0.11 ± 0.04
	Walker2d	0.23 ± 0.04	0.12 ± 0.00	0.09 ± 0.06	0.11 ± 0.00	0.05 ± 0.04	0.32 ± 0.16	**¡0.01 ± 0.00**
	HalfCheetah	0.36 ± 0.00	0.37 ± 0.00	0.03 ± 0.01	**¡0.01 ± 0.00**	0.32 ± 0.03	0.10 ± 0.00	0.08 ± 0.01
	Pendulum	0.03 ± 0.03	0.08 ± 0.03	0.98 ± 0.00	0.85 ± 0.13	0.07 ± 0.03	0.13 ± 0.00	**¡0.01 ± 0.01**
	Acrobot	0.04 ± 0.01	**¡0.01 ± 0.00**	**¡0.01 ± 0.00**	0.28 ±0.06	0.10 ± 0.06	0.22 ± 0.06	**0.01 ± 0.01**

TABLE XI

COMPARISON OF OPE METHODS ACROSS ENVIRONMENTS. ERROR BARS REPRESENT ± ONE STANDARD ERROR ACROSS 5 SEEDS; ANY REGRET SHOWN AS ¡0.01 IS NONZERO BUT ROUNDS TO ZERO AT TWO DECIMALS.

A. Sensitivity to Guidance Coefficients

We evaluate STITCH-OPE across different choices of the guidance coefficients α and λ, and plot the resulting trends in Figure 4 for Hopper and Figure 5 for Walker2D. Each plot is generated by applying bicubic interpolation to the grid evaluations of the Spearman correlation and LogRMSE. The optimal coefficient values of α and λ remain consistent across environments. The optimal balance for off-policy evaluation is attained by assigning a moderate coefficient for the target policy score α (i.e. $\alpha < 1$) and a smaller but positive coefficient to the behavior policy score, i.e. $0 < \lambda < \alpha$.

B. Trajectory Visualizations

We visualize and compare trajectories generated by the guided and unguided versions of STITCH-OPE and Policy-Guided Diffusion (PGD) [18] against both random and optimal policies. These visualizations highlight differences in the quality of generated trajectories, alignment with target policies, and generalization capabilities across various environments. As shown in Figures 6 and 8, STITCH-OPE closely mimics the target policy behavior. On the other hand, PGD performs poorly, significantly overestimating the performance of the random policy. Figure 7 further demonstrates that STITCH-OPE maintains consistent and robust behavior across policy settings.

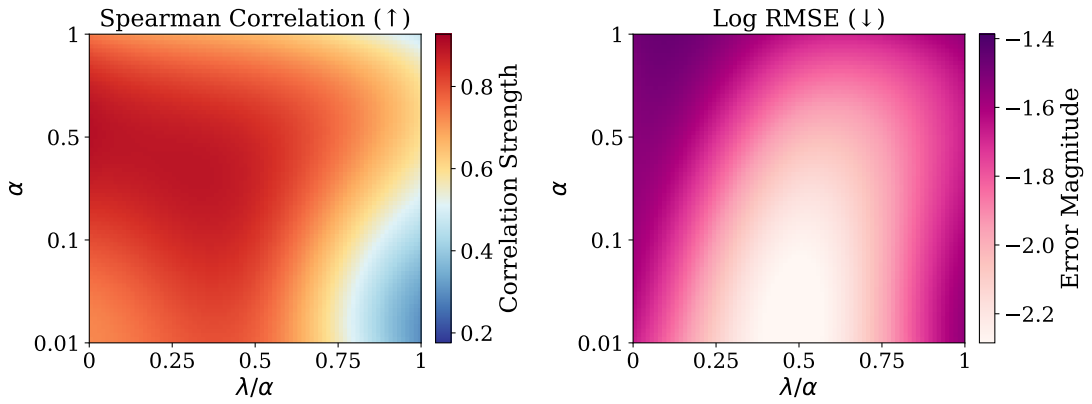

Fig. 4. Smoothed performance landscape for Hopper. **Left:** Spearman correlation is largest around $\alpha \in [0.1, 0.5]$, $\lambda \leq 0.5\alpha$. **Right:** The LogRMSE is smallest around $\alpha \in [0.01, 0.5]$, $\lambda \in [0.25\alpha, 0.75\alpha]$. These results confirm the optimal range of λ is $0 < \lambda < \alpha$.

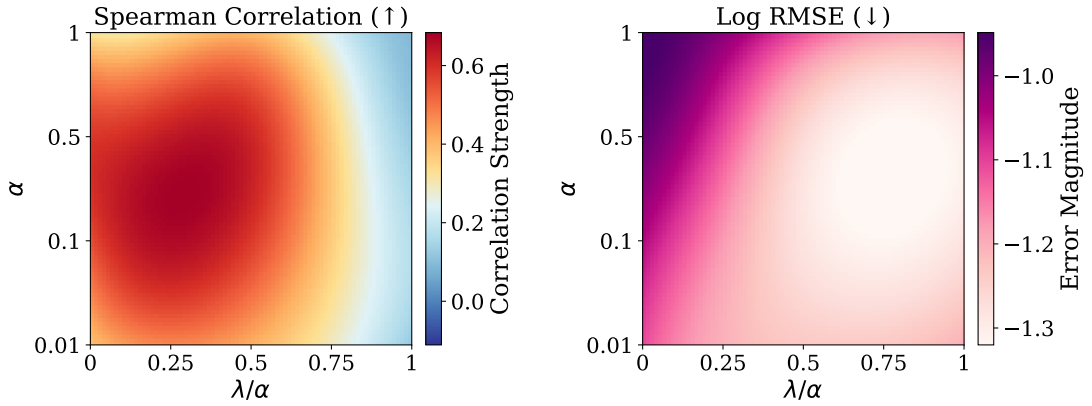

Fig. 5. Smoothed performance landscape for Walker2d. Results are generally consistent with Hopper. **Left:** Spearman correlation is largest around $\alpha \in [0.1, 0.5]$, $\lambda \approx 0.25\alpha$. **Right:** The LogRMSE is smallest around $\alpha \in [0.1, 0.5]$, $\lambda \approx 0.75\alpha$. These results confirm the optimal range of λ is $0 < \lambda < \alpha$.

APPENDIX L
COMPUTING RESOURCES

a) Hardware and Software: All experiments were conducted on a local workstation running Ubuntu 20.04 LTS and Python 3.9, with the following hardware:

- 2× NVIDIA RTX 3090 GPUs (24 GB each)
- Intel(R) Core(TM) i9-9820X CPU @ 3.30GHz (10 cores / 20 threads)
- 128 GB RAM.

b) Runtime: Each full training of a diffusion model for a D4RL task took approximately 20 hours to complete, depending on environment complexity and rollout length. Each OpenAI Gym task took approximately 5 hours. Each evaluation for a D4RL environment took around 18 hours in total (across all 5 seeds) to complete, and each OpenAI Gym environment took around 6 hours to complete.

APPENDIX M
RELATED WORK

Off-policy evaluation plays a critical role in offline reinforcement learning, enabling the evaluation of policies without directly interacting with the environment. OPE has been studied across a wide range of different domains including robotics [23], healthcare [40, 44, 42] and recommender systems [12, 48]. Relevant work includes model-free and model-based OPE approaches, including recent generative methods in offline RL.

a) Model-Free Methods: Model-free methods, such as Importance Sampling (IS) and per-decision Importance Sampling (PDIS) [43] reweight trajectories (or single-step transitions) from the behavior policy to approximate returns under a target policy. However, this class of methods suffers from the so-called "curse of horizon", in which the variance grows exponentially in the length of the trajectory [30, 33]. Doubly Robust (DR) methods [21, 49, 13] further combine estimation of value functions

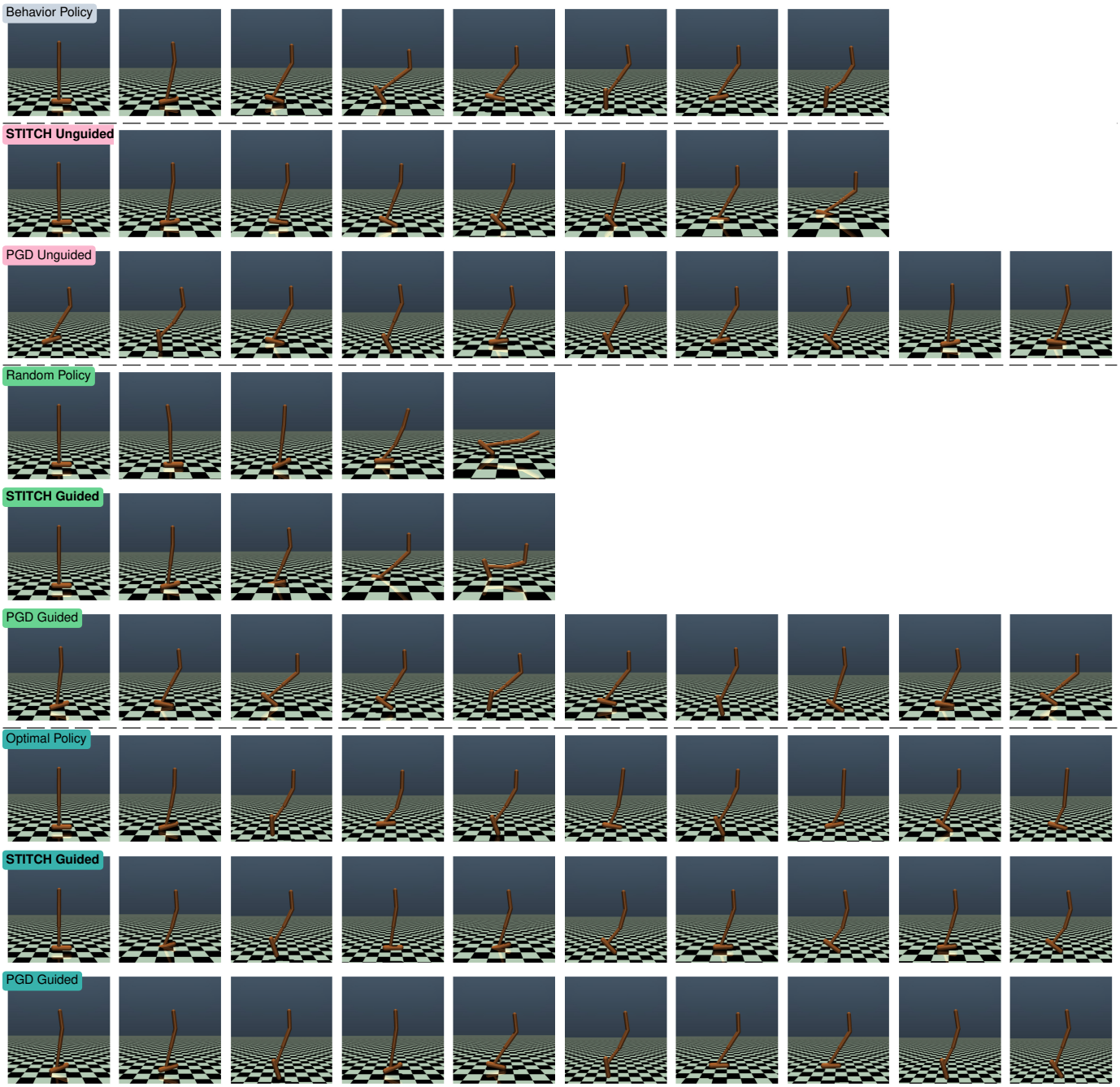

Fig. 6. Trajectory visualizations in the Hopper environment. Both STITCH-OPE and PGD track the optimal policy. PGD significantly overestimates the performance of the random policy, while STITCH-OPE correctly models both the state trajectory and the termination.

with importance weights, reducing the overall variance. Distribution-correction methods (DICE) [41, 54, 58] and their variants [30, 39] try to mitigate the curse-of-horizon by performing importance sampling from the stationary distribution of the underlying MDP. However, these methods perform relatively poorly on high-dimensional long-horizon tasks [16].

b) Model-Based Methods: Model-based OPE methods estimate the target policy value by learning approximate transition and reward models from offline data and simulating trajectories under the target policy [21, 24]. These methods have shown strong empirical performance, especially in continuous control domains [50, 57], but they often suffer from compounding errors during rollouts, which can lead to biased estimates in high-dimensional or long-horizon settings [22, 19].

c) Offline Diffusion: Inspired by the recent performance of diffusion models across many areas of machine learning [17, 10], a new stream of reinforcement learning has emerged which leverages diffusion models trained on behavior data

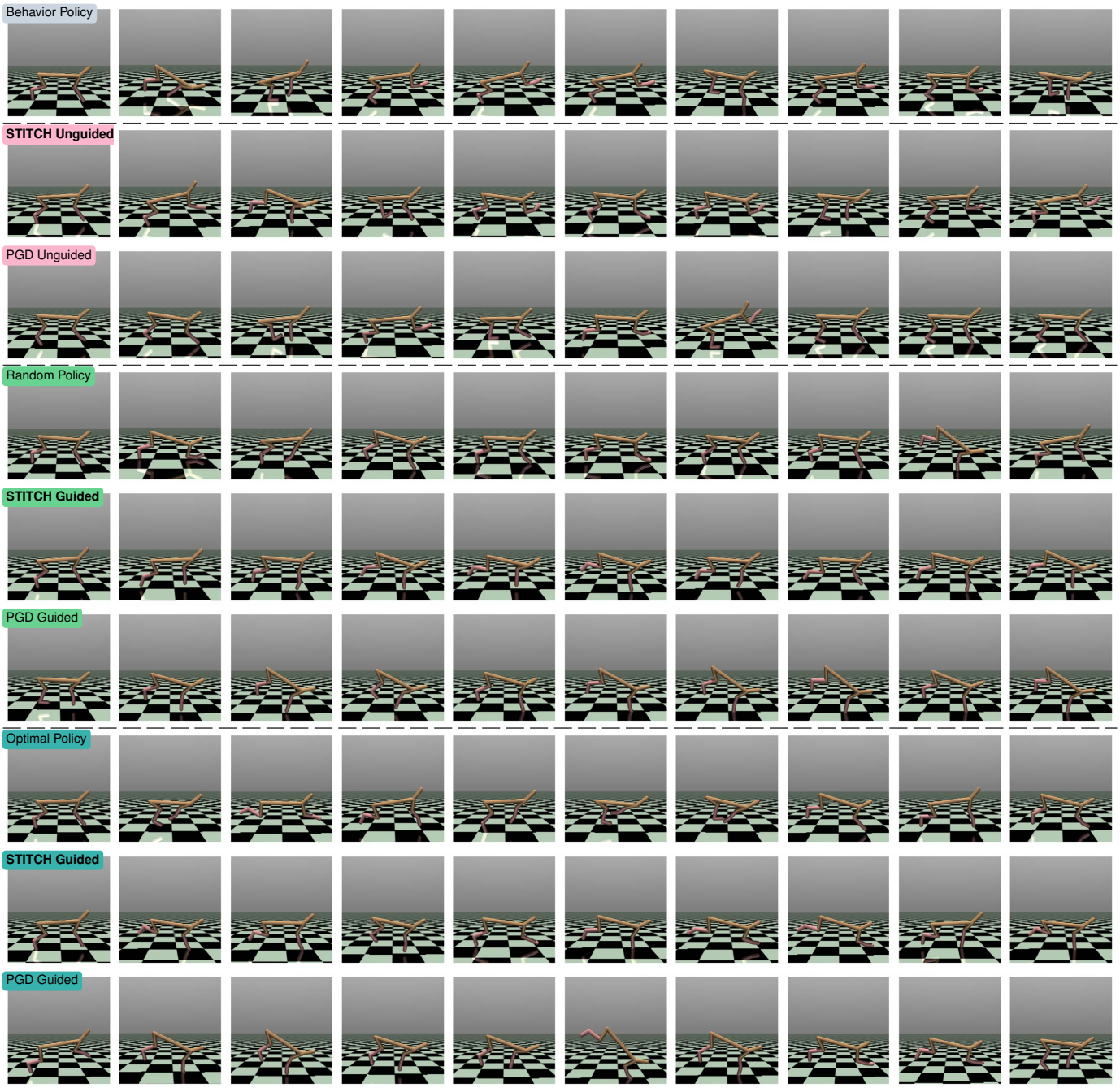

Fig. 7. Trajectory visualizations in the HalfCheetah environment. STITCH-OPE and PGD both demonstrate consistent behavior across all policy types, highlighting their robust generalization on this task.

[36, 60]. [20, 1] train diffusion models on behavior data that can be guided to achieve new goals. [18, 45] apply guided diffusion to offline policy optimization by setting the guidance function to be the score of the learned policy, while [59] applies guided diffusion to satisfy added safety constraints. Unlike STITCH-OPE, these works do not use negative guidance nor stitching, which we found leads to unstable policy values when applied directly for offline policy evaluation over a long-horizon. [37] applies DICE to estimate the stationary distribution of the underlying MDP, which is used as a guidance function to correct the policy distribution shift for offline policy optimization. Unlike STITCH-OPE, this work is not directly applicable to offline policy evaluation. Finally, [29] introduces a variant of trajectory stitching for augmenting behavior, but does not apply it for offline policy evaluation. To the best of our knowledge, STITCH-OPE is the first work to apply diffusion models to evaluate policies on offline data.

Fig. 8. Trajectory visualizations in the Walker2d environment. STITCH-OPE effectively imitates both random and optimal policies. As for the Hopper environment, PGD struggles to correctly imitate the random policy, significantly overestimating its performance.