Universiteit Leiden
The Netherlands

BSc Data Science and Artificial Intelligence

What's the Point of Diffusion Planning
in Offline Reinforcement Learning?

Fong Jing Ashley Lo

First supervisor and second supervisor:
Álvaro Serra-Gómez & Thomas Moerland

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

01/07/2025

Abstract

Diffusion-based planning has emerged as a promising approach for offline reinforcement learning, with theoretical advantages in trajectory stitching and long-horizon planning that should address fundamental challenges in learning from static datasets. However, a disconnect has emerged between the theoretical promise of these methods and their empirical performance on standard benchmarks. This thesis investigates if, when, and why diffusion planning provides value over simpler alternatives in offline reinforcement learning.

We conduct a comparative empirical evaluation of Diffuser, a representative diffusion planning method, against Recursive Skip-Step Planning (RSP) and Implicit Q-Learning (IQL) on AntMaze and Hopper environments from the D4RL benchmark. Our reproduction of Diffuser achieves 4-14$\times$ improvements over published results on navigation tasks, yet still underperforms alternatives significantly (27-28% vs 73-80% normalized returns on AntMaze). Conversely, Diffuser demonstrates competitive performance on continuous control tasks (70-107% on Hopper), suggesting domain-specific strengths in behavioral imitation rather than genuine planning challenges.

The computational overhead of diffusion planning presents substantial practical limitations, requiring 8 hours for training and 1.8 seconds for inference compared to 3-8 minutes for training and sub-millisecond latency for alternatives. This efficiency gap, combined with mixed empirical results, raises questions about the practical viability of current diffusion planning implementations.

Drawing on recent theoretical work by Clark and Shkurti, we examine how architectural limitations in current diffusion planning implementations may explain this empirical-theoretical disconnect. Clark and Shkurti identify that the standard Diffuser architecture violate locality and positional equivariance requirements necessary for effective trajectory composition, potentially causing models to resort to memorization rather than genuine planning. We note that effective trajectory stitching appears to require multi-scale temporal reasoning, achieved explicitly by hierarchical methods like RSP and implicitly by diffusion models through their coarse-to-fine denoising process.

We conclude that while diffusion planning possesses theoretical advantages for complex compositional tasks, the substantial computational overhead is as-of-yet unjustified by performance gains, and simpler alternatives often achieve superior or comparable results with dramatically reduced complexity. However, future advances fixing Diffuser's architectural constraints may unlock the potential of diffusion-based approaches for offline reinforcement learning.

Contents

1 Introduction

Learning to control complex systems from existing data has become increasingly useful across domains where active exploration is impractical or dangerous. The ability to extract effective policies from historical datasets offers a path to deployment without the risks and costs associated with trial-and-error learning in real environments. This paradigm, known as offline reinforcement learning, presents unique challenges that distinguish it from traditional online learning approaches [LKTF20, PMC23].

The fundamental difficulty in offline reinforcement learning lies in the distribution shift between demonstrated behaviors and the policies we wish to learn [LKTF20]. When an agent attempts actions significantly different from those in the training data, it may encounter states or situations never observed during training, leading to unpredictable and potentially catastrophic failures. This challenge is compounded by the need for trajectory stitching, the ability to combine segments from different demonstrated trajectories to synthesize novel, improved behaviors that were never explicitly shown in the dataset [HM22, ZZZ+23, LSZ+24]. Additionally, many real-world tasks require long-horizon planning, where early decisions significantly influence outcomes hundreds of steps into the future, making it difficult to learn effective policies from suboptimal or incomplete demonstrations [JLL21, PFM+25].

Recent advances in generative modeling, particularly denoising diffusion probabilistic models [HJA20], have greatly impacted fields ranging from computer vision to natural language processing [RBL+22, CTG+24, NZY+25]. These models excel at capturing complex, multimodal data distributions and generating high-quality samples through an iterative refinement process.

1.1 Planning with Diffusion

The Theoretical Promise Diffusion planning approaches like Diffuser [JDTL22] represent an innovative solution to many of the challenges that plague offline reinforcement learning. Unlike traditional methods that make myopic, single-step decisions, diffusion planners optimize entire trajectories simultaneously, potentially maintaining coherence across extended time horizons without the compounding errors that affect autoregressive approaches. The iterative denoising process that defines diffusion models seems naturally suited to the gradual refinement of rough trajectory sketches into detailed, executable plans.

Perhaps more appealingly, diffusion planning offers a theoretical framework for trajectory stitching [CS25]. The iterative denoising process, which enforces local temporal consistency at each step, could in theory allow the model to combine familiar trajectory segments from the training data in novel ways [JDTL22]. This capability would enable agents to discover paths between states that were never directly connected in the original demonstrations.

The Empirical Reality Despite these theoretical attractions, a disconnect has emerged between the promise of diffusion planning and its empirical performance. Initial applications to challenging offline reinforcement learning benchmarks such as D4RL `AntMaze` [FKN+20] have yielded mixed results [DYH+24, WNL+25]. This inconsistency is particularly striking given that diffusion planning should excel precisely on the tasks where it has struggled, environments requiring trajectory stitching and long-horizon reasoning.

The computational overhead of diffusion planning presents another practical concern [DHY+24]. Where traditional methods can generate actions in milliseconds, diffusion planners require iterative denoising processes that can take seconds per decision. This latency renders the approach impractical for real-time applications and raises questions about whether the theoretical benefits justify the computational costs. Alternative methods like Recursive Skip-Step Planning (RSP) [WNL+25], which achieve comparable or superior performance with orders of magnitude less computation, challenges the necessity of complex diffusion frameworks.

The Central Paradox This disconnect between theoretical promise and empirical reality presents a paradox in the field of offline reinforcement learning. **Why does diffusion planning, which seems ideally suited to address the core challenges of offline learning, sometimes underperform simpler alternatives? What factors determine when diffusion planning succeeds or fails?** Furthermore, what does this tell us about the relative value of complex generative modeling approaches versus more straightforward methods in sequential decision-making?

As researchers increasingly turn to offline reinforcement learning for real-world applications, understanding when sophisticated planning methods provide genuine value versus when simpler approaches suffice becomes crucial. The computational and implementation complexity of diffusion planning means that its adoption should be justified by clear performance advantages.

1.2 Research Objectives

This thesis seeks to understand what's the point of diffusion planning in offline reinforcement learning: if, when, and why it provides value over simpler alternatives. This work takes an exploratory and investigative approach, combining empirical evaluation and literature synthesis to develop a nuanced understanding of diffusion planning's role in the offline reinforcement learning. Specifically, our investigation centers on several key questions:

1. How does diffusion planner perform against established baseline and simpler alternative, on tasks that should favor their theoretical advantages?

2. Is the computational overhead of iterative denoising justified by performance gains?

3. What insights from recent literature explain the conditions under which diffusion planning works in practice?

1.3 Methodological Approach

To address these questions, we employ a methodology that combines empirical evaluation with literature analysis. The empirical component focuses on reproducing Diffuser [JDTL22] as the diffusion-based approach, and comparing its performance against representative alternatives from different offline reinforcement learning paradigms. This includes Recursive Skip-Step Planning (RSP) [WNL+25] as a hierarchical behavioral cloning method, and Implicit Q-Learning (IQL) [KNL21] as a temporal difference method.

The evaluation uses the `AntMaze` and `Hopper` environments from the D4RL benchmark [FKN+20]. Rather than attempting comprehensive coverage of all possible scenarios and environments, the investigation focuses on understanding the mechanisms underlying observed performance differences and identifying the factors that enable or hinder diffusion planning effectiveness.

The theoretical component draws on recent advances in understanding diffusion models' compositional capabilities, examining requirements that may be necessary for effective trajectory stitching. This analysis is primarily supported by recent findings from the broader literature on what makes diffusion planning work in practice.

1.4 Thesis Structure

The remainder of this thesis is organized as follows. Chapter 2 provides background on offline reinforcement learning paradigms and the fundamentals of diffusion planning, establishing the theoretical foundation for the subsequent investigation. Chapter 3 details the methodology and experimental design, explaining the rationale for the chosen approaches and evaluation framework. Chapter 4 presents the empirical results and performance analysis, documenting the observed performance disparities and computational trade-offs. Chapter 5 discusses theoretical explanations for these findings and their implications for the field, drawing connections between empirical observations and recent theoretical insights. Finally, Chapter 6 concludes with a synthesis of findings and directions for future work.

2 Background

2.1 Offline Reinforcement Learning

Offline reinforcement learning (RL), also known as batch RL, addresses the practical limitations of active environment interaction during training. Unlike traditional online RL, offline RL limits agents to learning from a predetermined collection of historical experiences, preventing any additional data acquisition through environmental exploration [LKTF20, PMC23]. This paradigm is particularly relevant in real-world applications where active data collection is costly, dangerous, or impractical, such as in robotics, healthcare, or autonomous driving.

2.1.1 The Offline RL Problem Setting

Offline RL builds upon the standard Markov Decision Process (MDP) framework, characterized by states $\mathcal{S}$, actions $\mathcal{A}$, transition probabilities $\mathcal{T}$, rewards $\mathcal{R}$, and discount factor γ [Bel57]. The learning goal, which is the same as online RL, is to find a policy π that optimizes expected cumulative discounted rewards [LKTF20].

$$\mathcal{J}(\pi) = \mathbb{E}_\pi \left[\sum_t \gamma^t r(s_t, a_t) \right] \tag{1}$$

What separates offline from online RL is the nature of available data. Rather than actively interacting with the environment, offline agents must work exclusively with a pre-collected dataset $\mathcal{D} = \{(s_t^i, a_t^i, s_{t+1}^i, r_t^i)\}_{i=1}^N$, generated through the actions of previous behavior policies π_b. The agent must perform policy optimization using only this fixed data, without the ability to sample new trajectories from the environment.

The fundamental challenge in offline RL stems from the distribution shift between the learned policy π and the behavior policy π_b [GCS+20, KZTL20, KNL21]. When the learned policy attempts actions significantly different from those in the dataset, it may encounter states or state-action pairs not observed in the training data, leading to potentially catastrophic failures during deployment. This distribution shift manifests in several critical ways: the learned policy might exploit errors in value function approximation, encounter unknown states where dynamics models fail, or generate actions far from the training distribution [LKTF20].

Another particularly important challenge in offline RL is trajectory stitching – the ability to compose segments of demonstrated trajectories to solve new tasks or achieve better performance than individual demonstrations [HM22, ZZZ+23, LSZ+24]. This compositional challenge appears frequently, where complex tasks often require combining simpler behaviors in novel ways. The difficulty of trajectory stitching stems from several issues in offline RL: distribution shift at trajectory intersection points, the challenge of identifying compatible trajectory segments, and error accumulation when combining multiple trajectory segments. How different algorithmic approaches handle this stitching problem becomes a key differentiator in their practical utility and directly relates to the core research question of this thesis.

Long horizon planning presents another challenge that offline RL shares with its online counterpart [JLL21]. In both settings, agents must learn to make decisions that balance immediate rewards against long-term consequences, often requiring credit assignment across hundreds or thousands of time steps. The challenge becomes particularly acute when optimal policies require initial actions that appear suboptimal in isolation but enable superior long-term outcomes. In offline RL, this challenge is compounded by the inability to explore alternative long-term strategies [PMC23] – if the behavior policy π_b was short-sighted or followed suboptimal long-term strategies, the agent must somehow learn to extend beyond these limitations using only the available data. Additionally, errors in value estimation tend to compound over longer horizons, making it difficult to distinguish between genuinely good long-term strategies and those that appear promising due to optimistic value function errors accumulated over time [KFS+19, KZTL20].

The offline nature changes how algorithms must approach policy optimization and value function estimation. While online RL algorithms can actively explore to reduce uncertainty and gather corrective data, offline RL must work within the constraints of the available dataset. This limitation has led to the development of specialized algorithms designed specifically for the offline setting, which can be broadly categorized into three paradigms.

2.1.2 Temporal Difference Approaches in Offline RL

Temporal difference-based (TD) methods in offline RL build upon the principles of the Bellman equation [Bel57, SB18]. These approaches learn value functions through temporal difference updates and derive improved policies through optimization of these learned value functions. The basic temporal difference update takes the form:

$$Q(s, a) \leftarrow r + \gamma \max_{a'} Q(s', a') \tag{2}$$

However, the offline setting introduces critical challenges for TD methods. The most significant issue is extrapolation error, where standard Q-learning tends to overestimate values for out-of-distribution actions due to the absence of corrective interactions [KZTL20, GCS+20]. This problem is compounded by bootstrapping error [KFS+19], where value estimates can accumulate and amplify errors through recursive updates, which is problematic with limited data coverage.

Another fundamental limitation of TD methods, particularly acute in offline RL, is their weakness in long horizon planning [PFM+25]. TD methods learn through local, single-step bootstrapping updates that propagate value information backwards one step at a time [SB18]. This learning process struggles with tasks requiring extended sequences of coordinated actions, as the error accumulation inherent in bootstrapping means that small estimation errors in individual Q-values can compound exponentially over long sequences, making it nearly impossible to reliably distinguish between genuinely superior long-term strategies and those that appear attractive due to accumulated optimistic bias [KFS+19]. In the offline setting, this weakness is amplified: if the behavior policy π_b exhibited poor long-term planning or the dataset lacks sufficient coverage of optimal long-horizon trajectories, TD methods have no mechanism to "bridge" these gaps in demonstrated behavior. The single-step nature of TD updates makes it difficult to establish connections between distant states and rewards, especially when intermediate steps in optimal long-horizon strategies are poorly represented in the training data [PFM+25].

To address these challenges, modern TD-based methods incorporate explicit constraints and regularization techniques. Conservative Q-Learning (CQL) introduces a penalty term that pushes down Q-values for out-of-distribution actions while maintaining accurate estimates for in-distribution data [KZTL20]. This conservative approach helps prevent the policy from exploiting erroneously optimistic value estimates. Another approach is Implicit Q-Learning (IQL), which learns conservative value functions through quantile regression, avoiding explicit policy optimization and thus reducing the impact of overestimation bias [KNL21].

2.1.3 Behavioral Cloning Approaches in Offline RL

Behavioral cloning (BC) approaches reframe offline RL as an imitation problem, bypassing many challenges associated with value function learning and bootstrapping. In its simplest form, BC treats offline RL as supervised learning from expert demonstrations [BS95]. BC directly learns a mapping from states to actions by minimizing a maximum likelihood loss:

$$\mathcal{L}_{\text{BC}}(\pi) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[-\log \pi(a|s)\right] \tag{3}$$

While conceptually simple and stable in training, BC suffers from compounding errors and distribution shift, as small deviations from the expert policy can lead to visiting states not seen during training [EUSL22, AMT+25].

Outcome-conditioned behavior cloning (OCBC) extends the BC framework by conditioning the policy on both the state and the target outcome, which could be return (the expected cumulative reward) or goals [EEKL21, BBB+22, EUSL22]. Instead of simply imitating actions, OCBC learns to generate actions that achieve specified target outcomes G:

$$\mathcal{L}_{\text{OCBC}}(\pi) = \mathbb{E}_{(s,a,G)\sim\mathcal{D}}\left[-\log \pi(a|s,G)\right] \tag{4}$$

The intuition is that by conditioning on the desired outcome, the policy can learn to discriminate between high-quality and low-quality actions in the dataset, potentially exceeding the performance of the behavior policy while maintaining stability benefits of supervised learning.

The main advantage BC approaches has over TD-based methods is its relative simplicity [EEKL21, BBB+22]. Direct supervision eliminates the need for complex value function approximation, leading to more stable training dynamics. The supervised learning framework is also well-understood and often more computationally efficient than TD-based methods.

Despite these advantages, BC methods face a significant limitation in trajectory stitching capabilities [BBB+22]. Since BC methods learn to imitate demonstrated behaviors directly, they can struggle to compose novel combinations of trajectory segments that were never explicitly demonstrated together. To illustrate this limitation concretely, Brandfonbrener et al. [BBB+22] demonstrated this limitation through a simple illustrative example (Figure 1). Consider an MDP where the optimal policy requires taking action a_1 in state s_0, followed by action a_0 in state s_2 to obtain the reward. However, the dataset contains only two types of trajectories: one starting from state s_0 that takes suboptimal action a_1 at s_2 (red), and another starting from s_1 that takes optimal action a_0 at s_2 (blue). While the data contains all the necessary components to construct an optimal policy from s_0, BC methods, even when conditioned on outcome, fail to stitch these segments together. The fundamental issue is that no trajectory in the dataset demonstrates the complete optimal sequence starting from s_0, so when conditioning on the optimal outcome, the policy becomes undefined. This example highlights how BC approaches are constrained by the specific trajectory combinations present in the training data, unlike value-based methods that can identify and combine optimal sub-behaviors even when they were never demonstrated together in complete sequences.

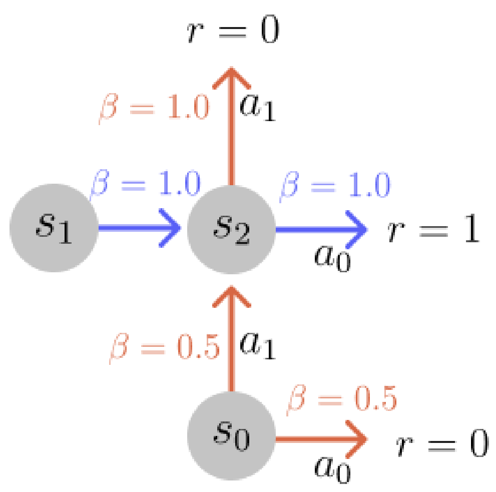


Figure 1: A toy example where BC approaches will fail to stitch. **Source: Brandfonbrener et al.** [BBB+22]

2.1.4 Model-Based Approaches in Offline RL

Model-based approaches represent a third paradigm that is orthogonal to the TD versus BC distinction [He23]. These methods learn explicit models of environment dynamics, which then serve as proxies for the true environment and can be used with various planning algorithms such as model predictive control (MPC) [GPM89]. Model-based methods can potentially be more sample efficient, as they can learn transition dynamics even from suboptimal trajectories [KRNJ20]. In principle, model-based approaches should be well-suited for trajectory stitching, as planning algorithms can compose novel sequences of actions using the learned dynamics model [HM22, ZZZ+23].

However, the offline setting introduces similar challenges for model-based approaches. The issue remains distribution shift [He23]: when the learned model is used for planning, it may encounter state-action pairs far from those in the training data, leading to compounding errors in multi-step predictions. This is particularly problematic because powerful trajectory optimizers tend to exploit inaccuracies in the learned dynamics model, finding adversarial trajectories that appear promising under the model but fail in reality [JDTL22]. The choice between model-based and model-free approaches in offline RL thus involves trade-offs between sample efficiency and robustness to distribution shift.

2.2 Fundamentals of Diffusion Planning

2.2.1 Denoising Diffusion Probabilistic Models

Denoising diffusion probabilistic models (DDPMs) represent a class of generative models that has achieved state-of-the-art results across various domains [CTG+24], including image generation [RBL+22], audio synthesis [KPH+20], and natural language processing [NZY+25]. Introduced by Sohl-Dickstein et al. [SDWMG15] and later refined by Ho et al. [HJA20], diffusion models learn to generate data by reversing a gradual noising process that systematically corrupts data into Gaussian noise.

The key insight of diffusion models is to decompose the complex problem of data generation into a series of simpler denoising steps. The process operates in two phases: a forward diffusion process that systematically corrupts data and a backward denoising process that learns to reverse this corruption.

Forward Process During the forward process, data points are gradually corrupted with Gaussian noise according to a variance schedule $\beta_1, \ldots, \beta_T \in (0, 1)$. Given a data point x_0 from the data distribution, the forward process produces a sequence of random variables $x_1, \ldots, x_T$ by:

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right) \tag{5}$$

A crucial property of this forward process is that it can be expressed in closed form for any timestep t using the reparameterization trick. By defining $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, we can directly sample x_t from x_0:

$$q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I\right) \tag{6}$$

This allows efficient sampling during training without iterating through all intermediate steps. As t increases, $\bar{\alpha}_t$ approaches zero, meaning that after sufficient steps (typically $T = 1000$ or more), the data x_T becomes virtually indistinguishable from pure Gaussian noise.

Backward Process The model learns to reverse the forward diffusion through a backward process. Unlike the forward process, which has a known analytical form, the true backward process $q(x_{t-1}|x_t)$ is intractable for complex data distributions. Therefore, a neural network p_θ is trained to approximate this reverse process:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right) \tag{7}$$

In practice, the covariance Σ_θ is often set to a fixed schedule $\sigma_t^2 I$, and the neural network focuses on predicting the mean $\mu_\theta(x_t, t)$. Ho et al. showed that instead of directly predicting the mean, it is more effective to train the network to predict the noise ϵ that was added at each step, leading to the training objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\epsilon,x_0}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right] \tag{8}$$

where $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ is the noisy version of x_0 at timestep t, and ϵ_θ is the neural network that predicts the added noise.

During generation, the model starts with pure Gaussian noise $x_T \sim \mathcal{N}(0, I)$ and iteratively applies the learned denoising steps. At each timestep t, the model predicts the noise $\hat{\epsilon} = \epsilon_\theta(x_t, t)$ and computes the denoised sample:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\epsilon}\right) + \sigma_t z \tag{9}$$

where $z \sim \mathcal{N}(0, I)$ is additional stochastic noise added for sample diversity.

Guided Sampling While unconditional diffusion models can generate high-quality samples, many applications require conditional generation where the output should satisfy certain constraints or follow specific guidance. Classifier guidance, introduced by Dhariwal and Nichol, uses the gradients of a pre-trained classifier to guide the reverse diffusion process toward samples that satisfy desired conditions [DN21].

The key insight is that the score function (gradient of the log probability) can be decomposed into unconditional and conditional components. During the reverse sampling process, the predicted noise is modified by incorporating gradients from a classifier $p_\phi(y|x_t)$ trained on noisy data points:

$$\tilde{\epsilon}_\theta(x_t, t, y) = \epsilon_\theta(x_t, t) - w\sqrt{1-\bar{\alpha}_t}\nabla x_t \log p_\phi(y|x_t) \tag{10}$$

where y represents the desired condition, w is a guidance weight controlling the strength of conditioning, and $\nabla_{x_t} \log p_\phi(y|x_t)$ is the gradient of the log-probability with respect to the noisy input. The classifier must be trained on noisy data points at various noise levels to provide meaningful gradients throughout the diffusion process.

2.2.2 Diffusion Models as Trajectory Optimizers

Recent work has demonstrated that diffusion models can be effectively adapted for trajectory optimization and planning. Diffuser, introduced by Janner et al. [JDTL22], presents a framework that integrates trajectory optimization with generative modeling. The key insight is to make planning and sampling nearly identical, eliminating the need for separate planning algorithms that often exploit imperfections in learned dynamics models, producing plans that resemble adversarial examples rather than realistic, high-return trajectories.

Control as Inference Framework Diffuser builds upon the control-as-inference framework [Lev18], which translates reinforcement learning into a probabilistic inference problem. In this perspective, optimal behavior corresponds to performing inference over trajectories, where optimality is represented as a binary random variable $\mathcal{O}_t$ associated with each timestep, with $p(\mathcal{O}_t = 1) = \exp(r(s_t, a_t))$. The reinforcement learning objective becomes sampling trajectories τ from the posterior distribution:

$$p(\tau|\mathcal{O}_{1:T}) \propto p(\tau) \prod_{t=1}^{T} p(\mathcal{O}_t|s_t, a_t) \tag{11}$$

Here, $p(\tau)$ serves as a trajectory prior representing realistic behavior (e.g., obeying dynamics), and the optimality terms bias sampling toward high-return trajectories. This transforms reinforcement learning from a sequential decision-making problem into a conditional sampling problem over trajectories that are both feasible and high-reward.

Diffuser implements the control-as-inference framework by learning the trajectory prior $p_\theta(\tau)$ through the diffusion process, then using guided sampling to condition on desired outcomes. During training, Diffuser learns to generate entire trajectories from noisy versions using the standard denoising diffusion objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{i,\epsilon,\tau^0}\left[\|\epsilon - \epsilon_\theta\left(\tau^i, i\right)\|^2\right] \tag{12}$$

where the model learns to reconstruct clean trajectories from noisy inputs by predicting the added noise ϵ at each diffusion timestep i. Then, during inference, we samples from perturbed distributions:

$$\tilde{p}_\theta(\tau) \propto p_\theta(\tau)h(\tau) \tag{13}$$

where $h(\tau)$ serves as a guidance function that biases sampling toward desirable outcomes, such as reaching a goal or maximizing reward. This formulation allows the same trained model to be reused across multiple tasks by simply changing $h(\tau)$.

2.2.3 Claimed Features and Advantages of Diffuser

The original Diffuser framework [JDTL22] proposes several key advantages over traditional planning approaches:

Long-horizon Planning and Anti-Causal Reasoning Janner et al. [JDTL22] claim that one of Diffuser's advantages lies in its ability to perform long-horizon planning without suffering from compounding errors that affect traditional single-step dynamics models. The authors claim that unlike autoregressive approaches that predict states sequentially and accumulate errors over time, Diffuser generates entire trajectories simultaneously. This enables the model to maintain coherent long-term plans even in sparse reward environments where conventional methods typically fail.They demonstrate this capability in maze navigation tasks, showing that Diffuser can generate feasible trajectories spanning hundreds of timesteps to reach distant goal locations.

Additionally, the authors contend that traditional autoregressive planning methods are inherently causal, predicting future states based solely on past information, whereas effective planning often requires anti-causal reasoning where current decisions depend on future objectives [JDTL22]. In essence, the planner can consider the future states when determining the very first action, a capability that eludes step-by-step sequential models. They propose that Diffuser's non-autoregressive nature enables goal-conditioned inference, where intermediate states are determined by both initial conditions and terminal goals – a capability they suggest is particularly valuable for goal-reaching tasks and constraint satisfaction problems.

Temporal Compositionality (Trajectory Stitching) Another advantage claimed by Janner et al. [JDTL22] is Diffuser's ability to exhibit temporal compositionality or trajectory stitching: the capacity to combine familiar trajectory segments in novel ways to generate out-of-distribution behaviors. The authors hypothesize that this stems from the iterative denoising process, which enforces global coherence through local consistency, potentially enabling Diffuser to "stitch" together in-distribution segments into previously unseen behavioral patterns. This property, if validated, would address one of the fundamental challenges in offline RL: learning to compose demonstrated behaviors in ways not explicitly shown in the training data.

Task Adaptability and Robustness The Diffuser framework proposes to separate the learned trajectory prior $p_\theta(\tau)$ from task-specific objectives, potentially enabling flexibility during test time [JDTL22]. The authors suggest that their framework allows for planning across multiple tasks using the same trained model by modifying the guidance function $h(\tau)$ during inference. They claim that a single trained model can be adapted to different reward functions, goal configurations, or constraint satisfaction problems without requiring retraining, which would offer significant practical advantages in multi-task scenarios if empirically validated.

Flexible Planning Horizons Unlike traditional neural network architectures where the planning horizon is typically fixed by architectural choices, Janner et al. [JDTL22] propose that Diffuser's planning horizon is determined solely by the dimensionality of the input noise that initializes the denoising process. This design choice theoretically enables dynamic adjustment of planning horizons at inference time, allowing the same model to generate both short-term reactive behaviors and long-term strategic plans as needed.

2.3 Related Work

2.3.1 Recent Work

Several works have built upon the foundational diffusion planning framework introduced by Janner et al. [JDTL22]. Decision Diffuser [ADG+22], developed near-concurrently with Diffuser, frames sequential decision-making as conditional generative modeling. Unlike Diffuser, which uses classifier-guided sampling to guide trajectory generation, Decision Diffuser employs classifier-free guidance to sample high-likelihood trajectories conditioned on returns, constraints, or skills. Additionally, Decision Diffuser diffuses only over states, and uses an inverse dynamics model to extract actions. This approach improves performance over diffusing both states and actions, particularly for tasks with high-frequency action [LHSL25].

One major development has been the exploration of hierarchical approaches that decompose complex planning problems into multi-level decision structures. Methods such as Hierarchical Diffusion for Offline Decision Making (HDMI) [LWJZ23], Hierarchical Diffuser (HD) [CDK+24], and DiffuserLite [DHY+24], have demonstrated that incorporating temporal hierarchy can substantially improve performance on long-horizon tasks. These approaches employ a multi-stage process where high-level planners generate abstract waypoints or subgoals, which are then refined by low-level planners to produce detailed action sequences. This hierarchical decomposition not only improves inference latency [DHY+24], but also enables better generalization across different task complexities and planning horizons [LHSL25].

A particularly influential contribution to the field has been the systematic empirical investigation conducted by Lu et al. in their work on Diffusion Veteran (DV) [LHSL25], which represents a comprehensive analysis of diffusion planning design choices to date. Through training and evaluating over 6,000 diffusion models across various architectural and algorithmic configurations, this work identified several counterintuitive findings that challenge conventional practices in diffusion planning. Key insights include the superiority of Transformer architectures over U-Net backbones for sequential planning tasks, the effectiveness of jump-step planning strategies that skip intermediate timesteps, and the finding that unconditional sampling with selection can outperform guided sampling methods when datasets contain sufficient high-quality demonstrations.

2.3.2 Diffusion Policy

While diffusion planners generate entire trajectory sequences to guide decision-making, diffusion policies represent an alternative approach where diffusion models directly parameterize the policy function itself [ZZH+23]. In this paradigm, the diffusion model learns to generate actions conditioned on the current state, effectively replacing traditional policy representations such as Gaussian policies in RL frameworks. Diffusion policies address the fundamental limitation of conventional unimodal policy parameterizations, which struggle to capture the multimodal action distributions often present in complex behavioral datasets [WHZ22]. This is particularly relevant in offline reinforcement learning settings where the dataset may contain diverse behavioral strategies that cannot be adequately represented by simple Gaussian distributions.

The key distinction from diffusion planning lies in the target of generation: while planners optimize entire trajectory sequences, diffusion policies focus solely on action generation given the current state. Representative works such as Diffusion-QL [WHZ22] integrate diffusion policies with Q-learning frameworks, demonstrating superior performance on datasets collected by multimodal behavior policies where traditional distance-based policy regularization approaches fail. Other significant contributions include Implicit Diffusion Q-Learning (IDQL) [HEKJ+23], which merges implicit Q-learning [KNL21] with diffusion policies within an actor-critic architecture. Additionally, Diffusion Actor-Critic (DAC) [FLZ+24] formulates policy iteration as a diffusion noise regression task, allowing for the direct parameterization of target policies as diffusion models, which is particularly advantageous for handling out-of-distribution actions in offline settings.

While diffusion policies offer enhanced expressiveness for policy learning, they fundamentally differ from the trajectory-level optimization approach employed by diffusion planners, making them orthogonal methodologies in the broader landscape of diffusion-based reinforcement learning.

2.3.3 Diffusion-Based Data Augmentation

A central challenge in offline RL is the limited presence of optimal trajectories in static datasets, which hinders the ability of agents to learn transitions to high-reward regions [PMC23]. Diffusion models have been leveraged for data augmentation in offline RL to address this issue. SynthER (Synthetic Experience Replay) [LBTPH23] utilizes diffusion models to generate synthetic experiences, effectively upsampling the agent's collected data. This method has proven effective in both offline and online RL settings, particularly in improving sample efficiency and policy performance by providing additional, high-quality training data. Similarly, DiffStitch [LSZ+24] introduces a diffusion-based data augmentation pipeline that generates stitching transitions between trajectories, connecting low-reward and high-reward segments to form globally optimal trajectories. This approach has demonstrated substantial enhancements in the performance of various offline RL methods, including TD-based methods like IQL.

2.4 Challenges of Diffusion Planning

2.4.1 Trajectory Stitching: Theory and Practice

A desired capability required for effective offline reinforcement learning is trajectory stitching: the ability to combine segments from different trajectories to synthesize novel, improved behaviors not explicitly present in the dataset [HM22, ZZZ+23]. This capability is crucial when dealing with suboptimal datasets where individual trajectories may be incomplete or suboptimal, but their composition could yield superior policies.

Theory for Stitching in Diffuser According to Janner et al. [JDTL22], Diffuser can perform trajectory stitching through its architectural design that enforces only local temporal consistency during each denoising step, rather than global trajectory coherence. Diffuser uses temporal convolutions with limited receptive fields, meaning each prediction can only "see" nearby timesteps in both past and future directions during a single denoising iteration. This constraint forces the model to generate trajectories by iteratively refining local consistency, where each denoising step ensures small trajectory segments are physically plausible based on their immediate temporal neighbors. Through the composition of many such local refinements across the full denoising process, global trajectory coherence emerges naturally. This mechanism enables trajectory stitching because the model can connect familiar subsequences from training data at their points of intersection, even when the complete trajectory was never observed during training.

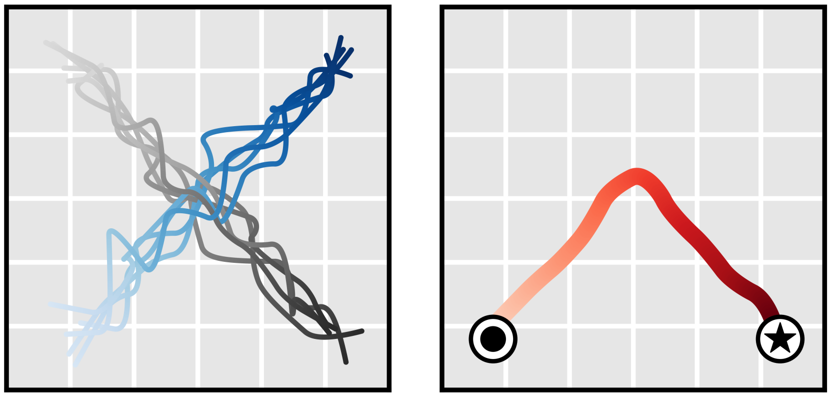

(a) Trajectory Data (b) Generated Plan

Figure 2: Janner et al. demonstrated that Diffuser can stitch together straight-line trajectories and generate V-shaped paths. **Source: Janner et al.** [JDTL22]

Janner et al. [JDTL22] provide some empirical evidence of trajectory stitching, where Diffuser, trained on straight-line trajectories, generates a V-shaped trajectory by combining two straight-line segments at their intersection (Fig. 1).

Diffusion Stitching in Practice Despite the theory, empirical evaluations reveal challenges in translating these capabilities to practical performance. Recent studies by Dong et al. [DYH+24] report that Diffuser achieve limited success on D4RL `AntMaze` environments, which are specifically designed to evaluate trajectory stitching capabilities through maze navigation tasks requiring composition of demonstrated path segments. The `AntMaze` benchmark is particularly relevant for evaluating stitching because optimal solutions often require combining partial trajectories that individually appear suboptimal but collectively lead to goal achievement. These empirical results suggest that while diffusion models possess the theoretical capacity for trajectory stitching, realizing this potential in practice may require additional algorithmic innovations beyond the basic diffusion framework.

2.4.2 Long-Horizon Planning: Alternative Approaches

While diffusion planning claimed advantages in generating coherent long-horizon plans, recent work has shown that comparable planning capabilities can be achieved through alternative approaches that offer significant computational advantages. Wang et al. demonstrate that effective long-horizon planning can be achieved using simple architectures through their recursive skip-step planning (RSP) approach [WNL+25]. RSP employs a hierarchical planning strategy where a lightweight network (a 2-layer MLP) recursively generates coarse-grained sub-goals, which are then executed by a goal-conditioned policy. This approach circumvents the step-wise error accumulation that affects fine-grained sequential modeling by operating at a higher level of temporal abstraction. These findings raise important questions about the necessity of complex diffusion frameworks for long-horizon planning tasks and highlight the value of exploring simpler, more efficient alternatives.

3 Research Methodology

3.1 Experimental Design Overview

In this research, we employ a comparative experimental framework to evaluate diffusion-based planning methods against alternative offline reinforcement learning approaches. The study is designed as a two-phase investigation that establishes baseline performance through reproduction of existing results, followed by a comparison against non-diffusion alternatives.

Phase I: Baseline Establishment In the first phase, we focus on reproducing published Diffuser results using standardized evaluation protocols. This reproduction phase serves multiple functions: it validates that reported performance can be achieved in our experimental environment, establishes reliable baseline metrics for subsequent comparisons, and ensures our experimental setup aligns with established benchmarks in the field. By demonstrating consistent reproduction of published results, we establish confidence in our experimental infrastructure and create a foundation for meaningful comparative analysis.

Phase II: Comparative Analysis In the second phase, we introduce two representative alternative approaches to offline RL: Recursive Skip-Step Planning (RSP) [WNL+25] and Implicit Q-Learning (IQL) [KNL21]. These methods were selected to represent fundamentally different paradigms within offline RL. RSP represents BC approaches with explicit hierarchical trajectory modeling, offering a direct alternative to diffusion-based trajectory generation. IQL represents a temporal difference method that learn conservative value functions, providing contrast to the BC paradigm.

3.2 Environment and Dataset Selection

3.2.1 D4RL Benchmark

The Datasets for Deep Data-Driven Reinforcement Learning (D4RL) benchmark serves as the evaluation framework for our study [FKN+20]. D4RL has become the standard benchmark for offline RL research, providing carefully curated datasets that enable fair and reproducible algorithm comparison. The benchmark's design addresses a critical challenge in offline RL evaluation: the variability introduced when different research groups generate their own datasets using different collection procedures, behavioral policies, and environment configurations.

D4RL's standardization extends beyond dataset provision to include consistent evaluation protocols, normalized scoring metrics, and reference implementations. This standardization is particularly valuable for comparative studies, as it eliminates potential confounding factors related to data collection methodology, evaluation procedures, or scoring normalization that could bias algorithmic comparisons.

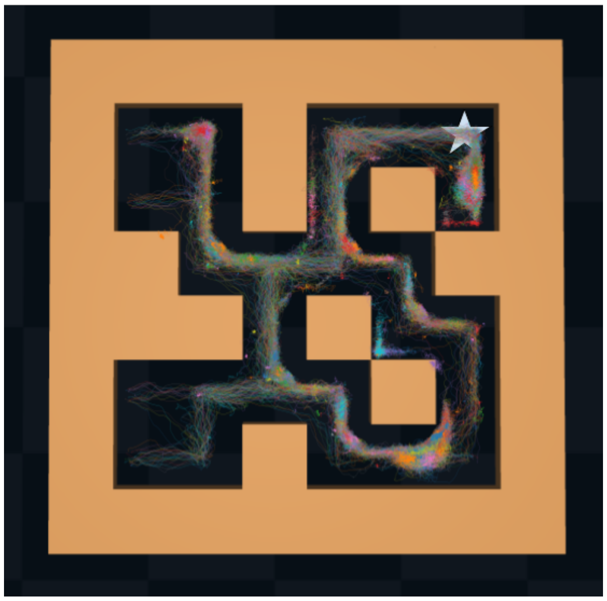
The benchmark encompasses multiple domains including navigation (`AntMaze`), continuous control (`MuJoCo locomotion`), and manipulation tasks, with datasets ranging from random exploration to expert demonstrations. This diversity enables evaluation across different offline learning scenarios, from challenging distribution shift settings to more favorable cases with high-quality demonstration data.

3.2.2 AntMaze Environment Specification

The `AntMaze` environment from the D4RL benchmark [FKN+20] combines continuous control with sparse reward navigation, creating a challenging testbed for long-horizon planning algorithms. The environment features a simulated quadruped ant agent navigating through maze-like structures to reach goal locations. The ant agent operates in an 8-dimensional continuous action space controlling joint torques, with observations including proprioceptive information (joint positions and velocities) and environmental coordinates.

Unlike simpler 2D navigation tasks, the ant agent must coordinate multiple joints through continuous torque control while navigating complex maze structures. The environment implements a sparse binary reward structure (0 or 1) that activates only upon successful goal achievement, eliminating intermediate reward signals that could guide simpler learning approaches. This sparse reward design requires agents to plan coherent action sequences spanning 100-200 timesteps without intermediate feedback. In our study, we focus on two `AntMaze` variants:

- `AntMaze Medium-Play:` Features trajectories generated from hand-picked initial positions to specific target locations, creating a controlled experimental setting with predetermined waypoints. Trajectories in this dataset achieves over 80% success rate, with failures primarily occurring when the ant agent becomes physically unstable. This dataset tests whether algorithms can reproduce demonstrated successful behaviors.
- `AntMaze Medium-Diverse:` Presents random goal locations throughout the maze, requiring navigation to diverse target positions from varied starting locations. This dataset challenges algorithms' capacity for trajectory stitching and generalization, as agents must learn to reach arbitrary goal locations rather than following memorized paths.


(a) `AntMaze Medium Diverse`

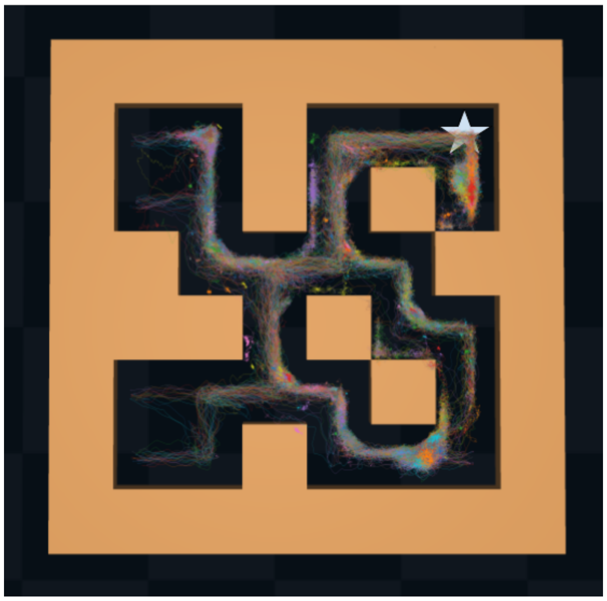

(b) `AntMaze Medium Play`

Figure 3: Trajectories in the `AntMaze Medium` dataset. **Source: Fu et al.** [FKN+20]

Both datasets utilize goal-reaching policies trained with SAC following waypoint-based navigation strategies, ensuring trajectories demonstrate feasible solutions while maintaining sufficient complexity for meaningful evaluation.

3.2.3 Hopper Environment Specification

The Hopper environment from the D4RL benchmark [FKN+20] presents a continuous control challenge where a simulated one-legged hopping robot must maintain forward locomotion while preserving balance and stability. The hopper agent operates in a 3-dimensional continuous action space controlling hip, thigh, and foot joint torques, with an 11-dimensional observation space encompassing joint angles, angular velocities, and center-of-mass information.

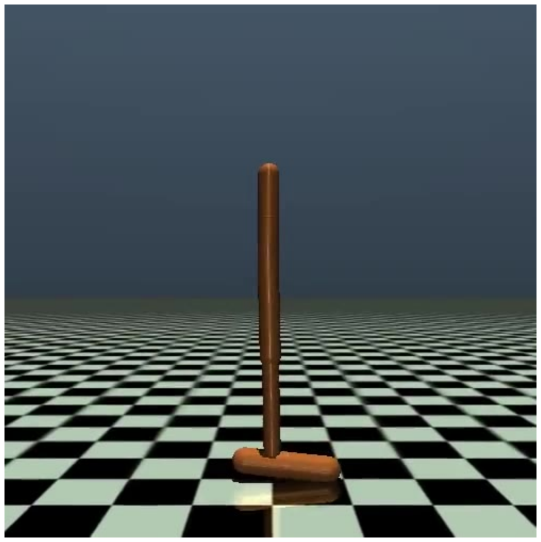


Figure 4: The Hopper robot. **Source: Fu et al.** [FKN+20]

The Hopper locomotion task requires precise coordination between torque control and dynamic balance, as the agent must generate forward momentum through rhythmic hopping motions while avoiding falls or backward movement. Unlike navigation tasks with explicit spatial goals, the Hopper environment implements a dense reward structure based on forward velocity, healthy bonus terms, and control costs, creating a multi-objective optimization problem where agents must balance speed, stability, and energy efficiency. In our study, we focus on two Hopper variants:

- Hopper Medium: Contains trajectories collected from a partially-trained SAC policy that achieves moderate performance levels. These trajectories exhibit a mix of successful hopping sequences and suboptimal behaviors, including occasional stumbles and inefficient gait patterns. The dataset provides a realistic representation of intermediate-skill demonstrations, testing whether algorithms can extract and improve upon partially successful strategies.

- Hopper Medium-Expert: Combines trajectories from both the medium-level policy and a highly-trained expert policy that demonstrates near-optimal hopping behavior. This mixed dataset challenges algorithms to distinguish between and learn from demonstrations of varying quality levels, requiring selective imitation of expert behaviors while avoiding the replication of suboptimal patterns from medium-quality trajectories.

Both datasets capture the fundamental challenges of dynamic locomotion control, where agents must learn temporally-extended policies that maintain consistent forward progress through coordinated multi-joint control strategies spanning hundreds of timesteps per episode.

3.2.4 Suitability of Environment

The `AntMaze` and `Hopper` environments provide complementary testbeds for comparing diffusion planning against alternatives, each highlighting distinct aspects of the planning challenge:

AntMaze Environment Characteristics:

1. **Long-Horizon Planning Requirements:** Navigation requires coordinated action sequences spanning 100-200 timesteps where early actions significantly influence goal reachability. The sparse reward structure provides no intermediate feedback during these extended sequences, testing each algorithm's ability to maintain coherent long-term planning.

2. **Trajectory Stitching Challenges:** The offline nature requires combining partial trajectory segments to construct novel paths to unseen goal configurations. Success requires "stitching" fragments of demonstrated behavior while maintaining physical plausibility, a capability that diffusion planners claim as a key advantage. While newer benchmarks explicitly designed to isolate trajectory stitching now exist [PFEL24], the `AntMaze` environment remains a standard and widely-used benchmark in the literature for evaluating these long-horizon and compositional planning capabilities.

3. **Computational Cost Justification:** The complexity of coordinating 8-DoF continuous control with long-horizon spatial reasoning provides meaningful context for evaluating whether diffusion planning's computational overhead yields proportional performance benefits.

Hopper Environment Characteristics:

1. **Dynamic Stability Planning:** Hopper locomotion demands maintaining dynamic equilibrium across extended sequences where each action affects future balance constraints. Unlike static navigation, the agent must plan actions that simultaneously achieve forward progress while satisfying complex stability requirements throughout the entire trajectory.

2. **Multi-Quality Data Integration:** The medium and medium-expert datasets test algorithms' ability to selectively learn from demonstrations of varying quality levels. This challenges diffusion planners to generate trajectories that capture expert-level coordination while avoiding the replication of suboptimal behaviors present in mixed-quality datasets.

3. **Dense Reward Temporal Credit Assignment:** The continuous reward signal tests whether diffusion planning's trajectory-level modeling provides advantages in environments where success depends on optimizing cumulative performance rather than reaching discrete goal states. This contrasts with `AntMaze`'s sparse rewards and evaluates planning approaches across different reward structures.

Together, these environments span the spectrum from sparse-reward spatial navigation to dense-reward dynamic control, providing comprehensive evaluation of diffusion planning's claimed advantages across diverse continuous control challenges.

3.3 Algorithm Descriptions and Implementations

Table 1: **Algorithm Comparison Summary**

	Diffuser	RSP	IQL
Paradigm	Trajectory-level generative modeling	Hierarchical behavioral cloning	Temporal difference
Architecture	1D U-Net	2-layer MLPs	2-layer MLPs
Planning Horizon	64	32, 1 level recursion	1, with value propagation
Key Advantage	Global trajectory coherence	Computational Efficiency	Proven foundation

3.3.1 Diffuser

Our Diffuser implementation utilizes CleanDiffuser, a modularized library designed specifically for diffusion models in decision-making tasks [DYH+24]. CleanDiffuser provides a framework that separates diffusion models, network architectures, and guided sampling methods into reusable components, enabling straightforward reproduction while maintaining consistency with original implementations.

Key advantages of CleanDiffuser include access to tuned hyperparameters and built-in handling of diffusion-specific mechanisms such as trajectory inpainting and guided sampling. This standardized implementation ensures our Diffuser baseline directly corresponds to published results. The only deviation is that we increased the batch size from 64 to 512 to better utilize GPU memory and parallelism on our hardware setup.

3.3.2 Recursive Skip-Step Planning (RSP)

RSP represents a hierarchical behavioral cloning approach that challenges the necessity of complex models for effective offline RL [WNL+25]. Instead of fine-grained sequential modeling, RSP employs recursive skip-step planning that predicts coarse-grained future sub-goals, using exponentially fewer planning steps than traditional approaches.

The core idea of RSP is its recursive skip-step planning scheme, which bypasses the fine-grained sequential modeling approach typically used by traditional planners. Instead of predicting immediate next states step-by-step, RSP recursively plans coarse-grained future sub-goals based on current state and target information. This approach uses exponentially fewer planning steps than traditional fine-grained sequence modeling methods, effectively circumventing the long-horizon compounding error accumulation that typically necessitates large, expressive models.

RSP augmenting the offline dataset with multi-horizon sub-goal information. For each transition, RSP creates skip-step horizons at exponentially decreasing intervals (e.g., $t + 32$, $t + 16$, $t + 8$ for 3-level recursion). The algorithm then trains separate dynamics models to predict sub-goals at each hierarchy level, conditioned on current state and higher-level predictions. The recursive conditioning stabilizes predictions by leveraging longer-horizon context, while the hierarchical structure enables effective long-horizon planning without the computational overhead of iterative denoising processes. Our RSP implementation uses the authors' official JAX-based codebase with pre-configured settings optimized for D4RL evaluation, ensuring consistency with published results.

3.3.3 Implicit Q-Learning (IQL)

IQL is a TD approach that addresses overestimation bias through expectile regression rather than standard Q-learning [KNL21]. Instead of directly learning Q-values and extracting policies, IQL learns state value functions using expectile regression, providing natural avoidance of overestimation on out-of-distribution actions.

The algorithm employs a three-step process: (1) learning value functions through expectile regression with pessimism parameter τ, (2) learning Q-functions using the learned values as targets, and (3) extracting policies through advantage-weighted regression. This procedure creates stable training that performs well across diverse offline datasets.

Unlike trajectory-level approaches, IQL's value learning enables trajectory stitching through local Bellman optimality rather than global trajectory modeling. The algorithm can identify high-value states from one trajectory and high-value actions from another, effectively combining optimal components from suboptimal demonstrations.

Our IQL implementation utilizes JAX-CORL [Nis24], a JAX implementation that provides significant computational speedups while maintaining algorithmic fidelity.

3.4 Performance Metrics and Computational Cost

Normalized Score Evaluation The primary performance metric employs D4RL's standardized normalized score, which transforms raw episode returns into a scale where 0 represents random policy performance and 100 represents expert policy performance:

$$\text{normalized score} = 100 \times \frac{\text{algorithm score} - \text{random score}}{\text{expert score} - \text{random score}} \tag{14}$$

This normalization enables meaningful comparison across different environments by accounting for inherent difficulty and reward scale variations. Each trained model's performance is assessed over 100 evaluation episodes to account for stochastic variation in policy execution. To account for training variability, we train each algorithm multiple times with different random seeds: Diffuser is trained 2 times (due to the prohibitive training cost per run), while RSP and IQL are trained 8 times each. Final reported performance represents the mean across these independent training runs, where each run is itself averaged over 100 evaluation episodes.

Computational Cost Analysis Beyond task performance, we evaluate computational efficiency through two key metrics:

- **Training Time:** Wall-clock time required for complete algorithm training, including data loading, forward/backward passes, and optimization steps, excluding one-time setup costs such as environment initialization.
- **Inference Latency:** Time required to generate action decisions during policy execution. For diffusion methods, this includes the complete iterative denoising process; for other methods, it captures policy evaluation or value function query time.

Hardware Specifications Experiments are conducted on NVIDIA A100 GPUs across different platforms. Diffuser training is performed on Vast.ai A100 instances, while IQL and RSP experiments are conducted on Google Colab A100 instances. While the GPU architecture is consistent, differences in system drivers, CPU performance, or memory bandwidth between platforms could introduce minor variations in wall-clock time measurements.

4 Experimental Results

Table 2: **Normalized returns, training time, and inference latency across different algorithms on `AntMaze` and `Hopper` tasks.** Original results are from published literature, while “Ours” indicates our reproduction experiments. Our results show mean performance over 2 seeds for Diffuser, and 8 seeds for RSP and IQL. Our results show mean performance over 2 seeds for Diffuser and 8 seeds for RSP and IQL. The lower seed count for Diffuser is due to its significant computational cost (8 hours per run), which limited the feasibility of more extensive runs and highlights the method’s practical efficiency challenges.

Environment	**Dataset**	**Diffuser** [DYH+24]	**Diffuser (Ours)**	**RSP** [WNL+25]	**RSP (Ours)**	**IQL** [KNL21]	**IQL (Ours)**
AntMaze	Medium-Play	6.7	27.2	91.4	73.3	71.2	78.3
	Medium-Diverse	2.0	28.5	92.7	75.0	70.0	79.8
” Hopper	Medium	89.5	70.8	63.4	61.6	66.3	57.8
	Medium-Expert	107.2	107.1	109.6	109.8	91.5	58.3

	Diffuser	RSP	IQL
Training Time	8 hours	3 minutes	8 minutes
Inference Latency	1.8 seconds	<0.001 seconds	<0.001 seconds

Our experimental evaluation across `AntMaze` and `Hopper` environments reveals substantial performance disparities between diffusion-based planning and traditional offline reinforcement learning approaches. However, reproduction discrepancies across multiple algorithm-environment combinations complicate direct comparison with published results and highlight the sensitivity of these methods to implementation details.

4.1 Performance Analysis

4.1.1 AntMaze: Navigation and Long-Horizon Planning

Our Diffuser results demonstrates fundamental limitations in sparse reward navigation tasks, despite reproduction improvements. Our implementation achieves normalized returns of 27.2 and 28.5 for `Medium-Play` and `Medium-Diverse` respectively, representing 4-14× improvements over published results (6.7 and 2.0) [DYH+24]. Since both our reproduction and the published results use the same CleanDiffuser repository, these discrepancies likely arise from differences in both random seed variance and the use of a larger batch size. A larger batch size, like the one we used (512), may contribute to more stable gradient estimates during training. This is particularly crucial for complex generative models like Diffuser, as it allows the model to learn a more accurate representation of the underlying trajectory data distribution, even if it does not solve the fundamental architectural flaws for planning. Nevertheless, Diffuser’s performance remains below competing methods. Even with these reproduction improvements, the consistent underperformance across all variants seems to indicate structural limitations in handling long-horizon credit assignment and goal-conditioned planning in navigation domains.

RSP exhibits strong performance (73.3 and 75.0) but underperforms published baselines (91.4 and 92.7) by approximately 18-19%. This reproduction gap may stem from subtle implementation variations not captured in the original methodology. Despite this underperformance relative to published results, RSP maintains clear superiority over Diffuser on both `Antmaze` tasks.

IQL consistently outperforms other methods across both `AntMaze` variants, achieving 78.3 and 79.8 returns, representing modest improvements over published baselines (71.2 and 70.0). Our implementation uses the CORL-JAX repository [Nis24] rather than the author's original version [KNL21], while maintaining identical hyperparameters from the literature. The improved performance may reflect optimizations in the CORL-JAX implementation, such as improved gradient estimation.

4.1.2 Hopper: Continuous Control Performance

The `Hopper` environment reveals markedly different algorithmic performance patterns and reproduction fidelity compared to `AntMaze`. Diffuser achieves competitive performance with decent reproduction fidelity, obtaining 70.8 and 107.1 on `Medium` and `Medium-Expert` respectively (compared to published 89.5 and 107.2). The near-perfect `Medium-Expert` reproduction (107.1 vs 107.2) and strong absolute performance suggest that continuous control domains with smooth reward landscapes and expert demonstrations better align with diffusion models' trajectory modeling capabilities.

RSP demonstrates excellent reproduction fidelity and strong performance on `Hopper` tasks, achieving 61.6 and 109.8 compared to published results of 63.4 and 109.6. The close alignment between our reproduction and original results suggests robust replicability across continuous control domains. Notably, RSP achieves the highest performance on the challenging `Medium-Expert` variant (109.8), indicating particular strength in leveraging high-quality demonstration data. The method's ability to maintain consistent performance across both medium-quality and expert datasets demonstrates its robustness to data quality variations in continuous control settings.

IQL shows significant underperformance on `Hopper` tasks, achieving only 57.8 and 58.3 returns compared to published baselines of 66.3 and 91.5. This represents a notable degradation of 13-36% across variants, with particularly severe impact on the `Medium-Expert` dataset. The performance drop on `Medium-Expert` (58.3 vs 91.5) suggests potential issues with our implementation's handling of high-quality demonstration data or value function estimation in continuous control domains. This reproduction failure highlights the sensitivity of value-based methods to implementation details.

4.2 Environment-Specific Insights

The contrast between `AntMaze` and `Hopper` results provides crucial insights into algorithmic strengths and domain-specific performance characteristics. The performance reversal between environments, where IQL excels in navigation but struggles in continuous control, while Diffuser shows the opposite pattern. This suggests fundamental differences in how these methods handle discrete versus continuous action spaces and sparse versus dense reward signals.

Navigation domains (`AntMaze`) appear to favor TD-based methods like IQL, which can effectively handle sparse rewards through learned value functions. The discrete nature of navigation decisions and the need for precise goal-conditioned planning align well with IQL's temporal difference learning approach.

Continuous control domains (`Hopper`) reveal diffusion models' relative strengths in trajectory modeling and action sequence generation. The smoother reward landscapes and continuous action spaces provide more favorable conditions for diffusion-based planning. RSP's strong performance across both domains demonstrates its versatility and computational efficiency.

4.3 Computational Efficiency Considerations

The computational cost analysis reveals Diffuser's most prominent limitation across all experimental conditions. The 8-hour training requirement and 1.8-second inference latency create substantial barriers to practical deployment, regardless of domain performance. This computational burden becomes particularly problematic in navigation tasks where rapid decision-making is essential, and even in continuous control where the method shows relative strength.

Both RSP and IQL maintain significant computational advantages across domains, with RSP completing training in 3 minutes and both achieving sub-millisecond inference times. These efficiency gains enable extensive hyperparameter exploration, ablation studies, and real-time deployment across diverse application scenarios. The computational efficiency of these simpler methods, combined with their competitive or superior performance, raises questions about the practical viability of diffusion-based approaches in their current form.

4.4 Reproduction Challenges and Methodological Implications

The substantial reproduction discrepancies observed across multiple algorithm-environment combinations highlight critical challenges in offline reinforcement learning research. The 4-14$\times$ improvement in Diffuser performance on `AntMaze`, the 18-19% degradation in RSP performance on `Antmaze`, and the IQL underperformance on `Hopper` collectively demonstrate the field's reproducibility challenges [KVYZ23, EHKS23]. The variability in reproduction success across different algorithm-environment pairs suggests that some methods may be more sensitive to implementation details than others, with implications for both research methodology and practical deployment.

5 Discussion

5.1 The Theoretical-Empirical Disconnect

The results presented in this work seem to reveal a paradox in diffusion-based planning for offline reinforcement learning. According to Janner et al.'s claims [JDTL22], Diffuser should excel precisely at the challenges that define successful offline RL: trajectory stitching across disparate data segments and long-horizon planning through complex state spaces. This theoretical promise should translate to superior performance in environments like `AntMaze`, where successful offline learning requires stitching together partial trajectories to form complete paths from arbitrary starting positions to goal locations. The sparse reward structure and long planning horizons characteristic of these environments should favor approaches capable of global trajectory optimization over traditional autoregressive, Markovian action selection. Diffusion models, with their established capacity to model complex multi-modal distributions and generate coherent long-sequence outputs, appear ideally positioned to address these fundamental challenges.

However, our empirical findings appear to contradict these theoretical expectations. Despite achieving 4-14$\times$ improvements over originally published results, Diffuser's performance on `AntMaze` variants remains around 27-28% normalized return – suboptimal behavior that falls short of the 70-90% returns achieved by alternative methods. This performance gap persists even with improved reproduction, suggesting potential systematic limitations.

Notably, this failure occurs precisely where diffusion planning should demonstrate its greatest theoretical advantages, while the method achieves competitive performance in domains where these advantages are less relevant. On `Hopper` tasks, Diffuser achieves 70-107% normalized returns, matching or approaching the performance of RSP. However, continuous control tasks like `Hopper` place limited demands on trajectory stitching capabilities. The motor control sequences required are relatively short-horizon, and the dense reward structure provides continuous guidance rather than requiring long-term credit assignment across sparse rewards. The success on `Hopper` demonstrates that diffusion models can effectively learn to reproduce behavioral patterns from demonstration data, but this capability does not translate to the more challenging planning scenarios that motivated the approach.

This disconnect is further illuminated by comparing the stitching mechanisms of the competing methods. IQL's success on `AntMaze` can be attributed to its TD-based approach, which enables trajectory stitching through local Bellman optimality [BBB+22]. It learns to identify high-value states and actions from disparate trajectories and combine them, effectively building an optimal path without ever needing to model a complete trajectory globally.

This theoretical-empirical disconnect suggests that the some assumptions supporting diffusion-based planning may not hold in practice, or that current architectural implementations fail to realize the approach's theoretical potential. The following section examine a possible explanation for this paradox, drawing on recent theoretical insights and architectural considerations that may illuminate why diffusion planning fails at its purported core competencies.

5.2 Theoretical Explanation for Empirical Failures

Very recent theoretical work by Clark and Shkurti provide a possible explanation for the empirical failures observed in our reproduction experiments [CS25]. Based on previous work on diffusion image generation models, Clark and Shkurti identify two critical architectural properties required for effective trajectory composition in diffusion planning: local receptive fields and positional equivariance:

1. **Local Receptiveness:** The locality requirement stipulates that during the denoising process, each state in a trajectory should primarily attend to a constrained neighborhood of nearby states rather than distant trajectory segments. Local receptiveness enables the model to compose trajectory fragments by preventing inappropriate dependencies between distant states that may belong to different sub-trajectories in the training data. Without this constraint, the model may learn spurious correlations between states that appear together in training trajectories but should not be causally linked during composition. Mathematically, they define a generative trajectory model as locally receptive when for any trajectory τ and state positions x_s within it, the denoising gradient estimate of the trajectory τ evaluated at state position x_s, is equal to the denoising gradient estimate of τ restricted to the immediate region Ω around x_s, evaluated at x_s, i.e. $M_t[\tau]\ (x_s) = M_t[\tau|\Omega_{x_s}](x_s)$.

2. **Positional Equivariance:** The model's denoising process should not depend on the absolute position of states within a sequence. This property enables the model to recognize that trajectory segments can be repositioned and recombined without losing their validity. Standard diffusion architectures frequently incorporate use downsampling and pooling operations that break this equivariance, constraining the model to reproduce trajectory segments only in their original temporal positions. Mathematically, a diffusion model $M_t[\tau]$ is positionally equivariant if for all position shifts U on trajectories, $M_t[U[\tau]] = U[M_t[\tau]]$.

Using this theoretical framework, Clark and Shkurti [CS25] provide an explanation why Diffuser fail at trajectory stitching despite the approach's conceptual promise. Janner et al.'s [JDTL22] 1D U-Net architecture's use of downsampling operations creates both non-local attention patterns and positional dependencies that prevent effective composition. When combined with training procedures that do not explicitly encourage compositional behavior, these architectural limitations result in models that resort to memorization rather than genuine trajectory synthesis. This analysis reveals a contradiction in Diffuser's design. While Janner et al. [JDTL22] emphasize how the local receptive fields of convolutions facilitate compositional learning, Diffuser's 1D U-Net's downsampling operations violate the locality and positional equivariant requirements identified by Clark and Shkurti [CS25]. The global receptive fields created by these operations could enable the model to memorize complete trajectories from the training distribution rather than learning to compose novel paths from trajectory fragments, precisely the opposite of the intended compositional behavior.

Clark and Shkurti provide direct empirical validation of their hypothesis [CS25]. In their work, they implement a diffusion planner using an architecture (Eq. net) that explicitly maintains locality and positional equivariance. Their experiments on toy datasets demonstrate that trajectory stitching does indeed work when these architectural constraints are respected, providing concrete evidence that the failure of current diffusion planning stems from architectural choices rather than fundamental limitations of the generative approach.

This theoretical framework provides a possible explanation. In AntMaze environments, where successful performance requires genuine trajectory composition to connect arbitrary start-goal pairs, Diffuser's architectural limitations might prevent effective learning despite the method's theoretical promises. The could explain the poor performance we observed (27-28% normalized return) even with improved reproduction.

Conversely, in Hopper tasks where trajectory composition is less critical, Diffuser is capable of competitive performance. The continuous control domain requires reproducing learned motor patterns rather than composing novel trajectories, aligning with what the architecture can actually accomplish rather than what it was designed to achieve.

5.3 The Hierarchical Advantage

Given the empirical failures of Diffuser and the success of methods like RSP, it is worth examining what potentially enables effective trajectory stitching for non-TD methods. Our insight is that successful trajectory composition may require reasoning at multiple temporal scales of abstraction, a capability that RSP achieves through explicit sub-goal planning, while diffusion planners possess through their inherent denoising process. RSP's design explicitly maintains its multi-scale reasoning, while the standard Diffuser implementation inadvertently destroy their implicit hierarchy through architectural choices that violate locality and positional equivariance.

Hierarchical approaches decompose trajectory planning into a hierarchy of decisions, where high-level planners generate coarse "plan sketches" outlining a trajectory's global structure, and low-level planners produce fine-grained actions to achieve each sub-goal [PSTQ21, LWJZ23, CDK+24]. At the highest abstraction level, these methods learn state reachability, determining whether two states can be connected, without being constrained by the specific trajectories observed in the training data. This abstraction enables the model to identify novel connections between states from distinct demonstration trajectories, facilitating compositional planning that transcends memorized paths.

Hierarchical decomposition reframes trajectory stitching as a problem of abstract state reachability rather than precise sequence matching. By operating at multiple temporal scales simultaneously, hierarchical methods can uncover connections between trajectory segments that remain invisible to single-scale approaches. Coarse-grained planning levels create "bridges" between disconnected segments by learning the underlying structure of the state space, while fine-grained levels ensure that these abstract connections translate to executable transitions.

This multi-scale abstraction enables two complementary roles that are both essential for effective stitching:

- **High-level planning learns "reachability":** At coarse temporal scales, the planner identifies which regions of the state space are accessible from one another, without committing to specific paths. For instance, a high-level planner might generate subgoals indicating the navigable regions of a maze, answering "can I reach that area?" rather than "how exactly do I get there?" This abstraction is crucial for stitching because it enables connections between states from different demonstration trajectories, even when direct paths between them are absent from the training data.

- **Low-level planning provides "feasibility":** At finer temporal scales, the planner executes concrete transitions between states, ensuring that abstract connections identified at higher levels are actually realizable. For example, a low-level policy generates precise actions to move between subgoals, filling in the gaps between high-level subgoals with concrete, executable steps.

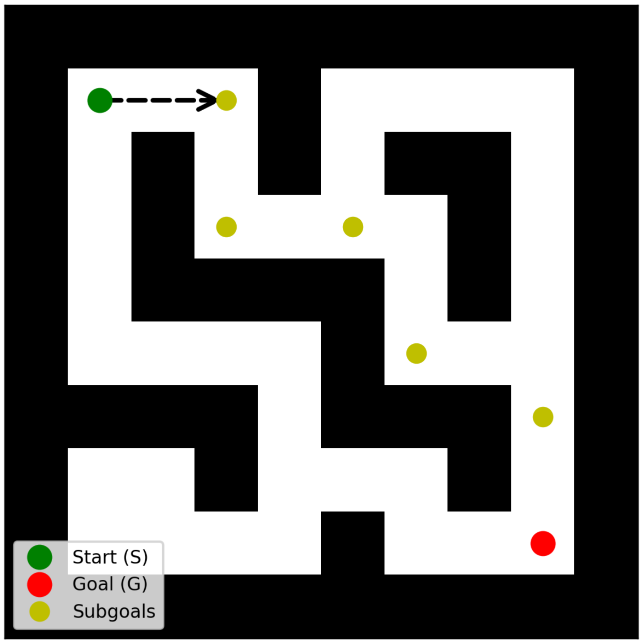


Figure 5: **Illustration of hierarchical planning decomposition.** The high-level planner identifies abstract subgoals (yellow dots) as key waypoints through the state space, while the low-level planner generates concrete, executable paths (dashed line) that navigate environmental constraints to connect these subgoals.

This separation of concerns is crucial: high-level planning identifies promising ways to connect trajectory segments, while low-level planning ensures their practical execution. Neither component suffices alone: high-level planning without executable refinement produces unrealizable plans, while low-level planning without global guidance remains trapped within the boundaries of demonstration trajectories. Hierarchical methods enable robust stitching by balancing global coherence with local precision.

Explicit Hierarchy in RSP RSP [WNL+25] uses explicit hierarchical planning through its recursive sub-goal prediction at exponentially decreasing horizons, forming a natural multi-scale abstraction hierarchy. The method's high-level dynamics model predicts distant future states (e.g., 32 steps ahead), capturing coarse-grained reachability patterns such as navigating around major obstacles in a maze environment. Subsequent hierarchical levels progressively refine these predictions at shorter horizons, gradually increasing the precision of the planning process. Finally, a goal-conditioned policy extracts concrete actions based on the current state and the complete hierarchy of sub-goals, ensuring feasibility by connecting abstract plans with executable transitions.

This explicit architecture directly addresses the core challenge of trajectory stitching: RSP's hierarchical structure enables it to identify connections between distant states (through high-level reachability planning) while ensuring these connections can be realized through concrete actions (via low-level policy execution). The recursive refinement process allows the method to bridge gaps between demonstration trajectories by planning at an appropriate level of abstraction.

Implicit Hierarchy in Diffusion Models While Janner et al.'s [JLL21] Diffuser struggles with trajectory stitching in practice, the diffusion denoising process contains an implicit hierarchical structure that should be conducive to compositional generation, if appropriately preserved through architectural design. This insight, building on Clark and Shkurti's theoretical framework [CS25], could help explain why diffusion planning should theoretically succeed at trajectory composition, and why current implementations fail to realize this potential.

Diffusion models generate trajectories by denoising a sample over T timesteps, transitioning from pure noise $\mathcal{N}(0, I)$ to a coherent trajectory $\tau = (s_0, a_0, ...)$. This denoising process exhibits a natural coarse-to-fine hierarchy that effectively parallels explicit hierarchical planning:

- Early denoising steps (high noise, $t \approx T$) capture global trajectory structure, determining the coarse shape and overall direction of a trajectory. In navigation tasks, these steps might establish whether a trajectory moves around the left or right side of a major obstacle, analogous to high-level sub-goal planning that determines reachability between distant state regions.
- Later denoising steps (low noise, $t \approx 0$) refine local trajectory details, ensuring precise state-action transitions within segments and smooth connections between waypoints. This resembles low-level policy execution that fills in concrete actions between abstract sub-goals.

This coarse-to-fine denoising hierarchy has been extensively validated in image diffusion, where early steps produce blurry global outlines and later steps sharpen fine-grained details [HJA20]. Our understanding, consistent with Clark and Shkurti's analysis [CS25], is that this implicit hierarchy could enable trajectory stitching in diffusion models, if the architecture preserves the locality and positional equivariance properties required for compositional generation.

5.4 When Is Computational Overhead Justified?

While our analysis demonstrates limitations in the standard Diffuser implementation, this does not entirely negate the potential value of the diffusion planners when implemented correctly. The question then becomes: under what circumstances might the substantial computational overhead of iterative denoising be justified compared to more efficient alternatives?

Our empirical findings, combined with broader evidence from the literature [DHY$^+$24, WNL$^+$25], suggest that diffusion planning's computational costs are unjustified for most practical offline RL scenarios *as of yet*. The success of simpler methods like RSP and IQL demonstrates that many offline RL tasks can be solved effectively without the architectural complexity of diffusion models. These approaches represent compelling alternatives that achieve comparable or superior performance while requiring dramatically reduced computational resources – RSP completes training in 3 minutes versus Diffuser's 8 hours, while both RSP and IQL achieve sub-millisecond inference compared to Diffuser's 1.8-second latency.

The computational efficiency gap becomes even more pronounced when considering practical deployment constraints. Real-time robotics applications, embedded systems, and scenarios requiring frequent replanning cannot accommodate the multi-step denoising process inherent to diffusion planning. In these contexts, methods like RSP and IQL provide practical advantages through their efficient inference procedures. Moreover, our results suggest that trajectory stitching, one of the primary theoretical motivations for diffusion planning, can be achieved more reliably and efficiently through hierarchical approaches like RSP, which directly addresses the offline RL composition problem without requiring expensive iterative generation.

However, diffusion planning may retain value in specific scenarios that explicitly require its unique generative capabilities. Tasks involving highly multimodal action distributions, where multiple distinct behavioral patterns must be captured and selectively sampled during inference, could potentially benefit from diffusion models' ability to represent complex probability distributions. For instance, manipulation tasks requiring the agent to choose between fundamentally different approach strategies (e.g., reaching around versus over obstacles) might justify the computational expense if the multimodal nature of optimal policies cannot be captured by simpler methods.

Similarly, applications requiring fine-grained trajectory generation with smooth interpolation between different behavioral modes, such as human-robot interaction scenarios where the robot must seamlessly blend between different social behaviors, might warrant diffusion models' sophisticated generative capabilities. Creative applications in robotics, where generating diverse and novel behaviors is valued over efficiency, could also benefit from the rich representational capacity of diffusion models.

The key insight is that diffusion planning's value proposition depends on whether the specific task requires capabilities that simpler methods cannot provide. Our study reveals that for some canonical offline RL benchmarks like navigation (`AntMaze`) and locomotion (`Hopper`), these unique capabilities are not essential, and in some cases, may even be counterproductive. The sparse reward navigation tasks that should theoretically favor diffusion planning's trajectory stitching abilities are better served by direct hierarchical decomposition approaches.

The evidence from our investigation and the broader literature [WNL+25] suggests practitioners should approach diffusion planning with appropriate skepticism. The combination of empirical failures on key benchmarks, substantial computational overhead with existing implementations, and the demonstrated effectiveness of simpler alternatives suggests that the default choice should be efficient methods like RSP or IQL. However, as diffusion methods mature and denoising efficiency improves, there may be increasing opportunities for their application to genuinely long-horizon planning problems where the computational trade-off becomes favorable.

6 Conclusion

This thesis investigated a timely question in offline reinforcement learning: **what is the point of diffusion planning,** when simpler, more efficient alternatives often achieve superior performance? Through empirical evaluation across `AntMaze` and `Hopper` environments, combined with theoretical analysis drawing on recent advances in understanding diffusion-based planning, our work reveals some disconnect between the theoretical promise and practical reality of diffusion planning approaches.

6.1 Key Findings

Our investigation yields several insights for diffusion-based planning in offline RL:

Empirical Performance Gaps Despite achieving 4-14× improvements over published Diffuser results on `AntMaze` tasks, our reproduction still demonstrates fundamental limitations in the domains where diffusion planning should theoretically excel. With normalized returns of only 27-28% on navigation tasks requiring trajectory stitching and long-horizon planning, Diffuser underperforms both hierarchical behavioral cloning methods like RSP (73-75%) and temporal difference approaches like IQL (78-80%), suggesting potential systematic rather than implementation-specific limitations.

Domain-Specific Performance Patterns The contrasting performance across environments reveals important insights about algorithmic strengths. While Diffuser achieves competitive performance on continuous control tasks like `Hopper` (70-107% normalized returns), it underperforms where its theoretical advantages should be most pronounced: in sparse reward navigation requiring trajectory composition (`AntMaze`). This reversal suggests that the standard diffusion planning implementation may be better suited for behavioral imitation than genuine planning challenges. Our findings suggest that the value of sophisticated planning approaches depends crucially on task characteristics. Navigation tasks requiring genuine composition benefit from explicit hierarchical decomposition, while continuous control tasks may be adequately served by behavioral imitation capabilities that even suboptimal diffusion implementations can provide.

Computational Efficiency Concerns The computational overhead of diffusion planning presents a significant practical limitation. With 8-hour training requirements and 1.8-second inference latency compared to RSP's 3-minute training and sub-millisecond inference, the computational costs appear to be unjustified. This efficiency gap becomes particularly problematic in real-time applications where rapid decision-making is essential.

Architectural Limitations Clark and Shkurti's recent theoretical work [CS25] identified that current Diffuser implementations violate the locality and positional equivariance, requirements necessary for effective trajectory composition. Diffuser's 1D U-Net architecture's downsampling operations create global dependencies that encourage memorization rather than genuine compositional learning, potentially explaining why diffusion planning fails at its purported core competency of trajectory stitching.

The Hierarchy Advantage We note that effective trajectory stitching may require multi-scale temporal reasoning, a capability that both successful methods (RSP) and architecturally sound diffusion approaches share. RSP achieves this through explicit hierarchical decomposition, while diffusion models possess implicit coarse-to-fine hierarchy through their denoising process.

6.2 Practical Implications

For practitioners and researchers in offline reinforcement learning, our findings suggest several important considerations:

Default Method Selection Given the substantial computational overhead and mixed empirical results, simpler methods like RSP or IQL should be the default choice for most offline RL applications. The burden of proof should fall on demonstrating that diffusion planning's unique capabilities are actually required for the specific task at hand.

Computational Trade-offs The efficiency advantages of simpler methods – orders of magnitude faster training and inference – enable extensive experimentation, hyperparameter exploration, and real-time deployment that remain impractical with current diffusion approaches. These practical considerations often outweigh theoretical sophistication in applied settings.

6.3 Addressing the Central Question

Returning to our central question – **what's the point of diffusion planning in offline reinforcement learning?** — our study suggests a nuanced answer. While diffusion planning possesses theoretical advantages that could be valuable for complex compositional tasks, the standard implementation fail to realize this potential possibly due to architectural limitations. The computational overhead is as-of-yet unjustified by performance gains, and simpler alternatives often achieve superior or comparable results with dramatically reduced complexity.

However, this should not be interpreted as a wholesale rejection of diffusion-based approaches. Rather, our findings highlight the importance of architectural design choices and suggest that future advances in diffusion planning must address fundamental limitations rather than simply scaling existing methods. The theoretical framework for effective trajectory composition exists, but its practical implementation remains an open challenge.

6.4 Limitations and Future Work

Our investigation suffers from several significant limitations that constrain the generalizability and conclusiveness of our findings. These limitations highlight important areas for future research and more comprehensive evaluation.

Experimental Scope Our evaluation focuses on a limited set of environments from the D4RL benchmark, focusing on `AntMaze-Medium` and `Hopper-Medium` as our test environments. This represents an narrow slice of the offline RL problem space and provides only some limited evidence to support broad claims about diffusion planning's utility across diverse domains. Even within the navigation domain, our evaluation lacks the depth required for conclusive findings. While `AntMaze-Medium` is purported to require trajectory stitching and long-horizon reasoning, this connection is not definitively established through our experimental design. We did not conduct controlled experiments that explicitly isolate stitching requirements from other task characteristics, making it impossible to attribute performance differences specifically to stitching capabilities or long-horizon reasoning.

We evaluate only the standard Diffuser implementation rather than exploring alternative architectures that might better preserve the compositional properties required for effective planning. Investigating architectures that explicitly maintain locality and positional equivariance could yield different conclusions about diffusion planning's potential.

Moreover, we did not conduct extensive hyperparameter optimizations, which would require computational resources beyond the scope of a bachelor thesis. Our reproduction challenges across multiple algorithms highlight the sensitivity of offline RL methods to implementation details. Future work should include more systematic hyperparameter exploration and sensitivity analysis to better understand the robustness of different approaches.

The scope of our experimental evaluation reflects the practical constraints of an individual bachelor thesis project, though future work with greater resources could provide more comprehensive validation across multiple environments and algorithmic variants.

Limited Training Runs for Diffuser Another crucial methodological limitation stems from computational constraints that limited Diffuser to only 2 independent training runs, compared to 8 for RSP and IQL. While each trained model is thoroughly evaluated over 100 episodes, having only 2 training seeds for Diffuser reduces our confidence in the stability of training outcomes and limits statistical power for detecting true performance differences between algorithms.

Standard practice in RL evaluation typically requires at least independent training runs to account for training variability [Moe]. The high variance in offline RL training means that performance differences we observe between 2-seed and 8-seed averages may reflect training luck rather than algorithmic superiority.

This constraint was unfortunate, yet unavoidable, given Diffuser's 8-hour training requirement and the individual nature of bachelor thesis research. Training 8 Diffuser models would have required 64+ hours of compute time, beyond resource constraints. However, this asymmetry in training samples limits our ability to make definitive comparative claims about algorithmic performance.

Benchmark Inadequacy The D4RL benchmark suite, while widely used, has known limitations in testing the full spectrum of offline RL capabilities. Many D4RL tasks may not actually require the sophisticated planning capabilities that diffusion models are designed to provide, making performance comparisons potentially misleading. This reliance on older benchmark environments represents a significant methodological limitation. The recently proposed OGBench [PFEL24], which includes `AntMaze-Stitch` environments specifically designed to test trajectory stitching capabilities, would provide more rigorous evaluation criteria for compositional planning methods. Our failure to evaluate on these purpose-built stitching benchmarks limits our conclusions to the environments that we have tested, and future work should evaluate on purpose-built benchmarks like OGBench that explicitly test trajectory stitching capabilities.

Replicability and Sensitivity to Implementation Another significant limitation of this investigation is the challenge encountered in precisely replicating the published results for the evaluated algorithms, which highlights a broader issue of reproducibility in the field [KVYZ23, EHKS23]. The experimental results revealed performance discrepancies between our implementations and the original papers across all three methods, complicating direct comparisons.

- **Diffuser:** Our implementation on `AntMaze` tasks achieved normalized returns that were 4 to 14 times higher than the published results we aimed to reproduce. This improvement is likely attributable to differences in hyperparameters such as batch size, which can provide more stable gradient estimates during training.

- **RSP:** In contrast, our RSP implementation underperformed the original paper's benchmarks on `AntMaze` by approximately 18-19%, which may stem from subtle implementation variations not captured in the original methodology.

- **IQL**: The IQL implementation showed a notable performance degradation of 13-36% on the `Hopper` tasks compared to its published baselines, with a particularly severe impact on the `Medium-Expert` dataset. This suggests a high sensitivity to implementation details or the specific code repository used.

This variability in reproduction success introduces a degree of uncertainty into our findings. While the thesis draws conclusions based on the performance observed in our controlled experimental setup, the instability of these baselines means that our results should be interpreted with caution. The sensitivity of these algorithms to minor implementation details makes it difficult to definitively claim superiority of one method over another without more extensive validation.

This issue reflects a known challenge within reinforcement learning research and has methodological implications for the field [KVYZ23, EHKS23]. Future work should therefore include more systematic hyperparameter optimization and sensitivity analyses to better understand the robustness of these different approaches. Conducting such extensive exploration was beyond the computational resources available for this bachelor thesis, but is essential for building a more reliable understanding of when and why certain offline RL algorithms succeed or fail.

6.5 Final Thoughts

This thesis contributes to a more nuanced understanding of diffusion planning's role in offline reinforcement learning by demonstrating that theoretical complexity might not automatically translate to practical value. The disconnect between Diffuser's conceptual appeal and empirical performance highlights the importance of rigorous, replicable evaluation and the need to question assumptions about the necessity of complex methods for sequential decision-making tasks.

As the field continues to develop increasingly sophisticated approaches to offline RL, our findings suggest that solving the fundamental challenges of the domain: trajectory stitching, long-horizon planning, and distribution shift may not require large expressive models, echoing the opinion of Wang et al. [WNL+25]. The most valuable contributions may come not from developing more complex generative models, but from understanding when and why simpler approaches suffice, and identifying the specific scenarios where computational complexity is truly justified by performance gains.

However, our findings should be interpreted within the context of current benchmark limitations and computational constraints. Following the spirit of the "bitter lesson" in machine learning [Sut19] – that methods leveraging computation ultimately prevail over those exploiting human knowledge – it is plausible that as we encounter increasingly complex real-world tasks beyond current benchmark capabilities, the sophisticated modeling capacity of diffusion planners may become essential. Tasks requiring genuine multimodal reasoning, complex temporal dependencies, or rich compositional behavior may eventually demand the expressiveness that current simple methods cannot provide.

Moreover, the computational overhead that currently limits diffusion planning's practical adoption may become less relevant as hardware capabilities continue to advance [TGM22]. The dramatic improvements in compute per unit cost, driven by ongoing developments in specialized AI hardware, suggest that what appears computationally prohibitive today may become routine tomorrow . As inference costs decrease and parallel processing capabilities expand, the 1.8-second latency that currently renders diffusion planning impractical for real-time applications may shrink to acceptable levels, potentially shifting the cost-benefit analysis in favor of more expressive methods.

The question of diffusion planning's value in offline RL thus remains open, contingent on both innovations that can bridge the gap between theoretical promise and practical performance, and the evolution of computational resources and task complexity. Until such advances materialize, practitioners are perhaps better served by the efficiency and reliability of simpler methods, while researchers continue to explore the conditions under which the advantages of diffusion planning can be realized in practice.

References

[ADG$^+$22] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[AMT$^+$25] Shan An, Ziyu Meng, Chao Tang, Yuning Zhou, Tengyu Liu, Fangqiang Ding, Shufang Zhang, Yao Mu, Ran Song, Wei Zhang, et al. Dexterous manipulation through imitation learning: A survey. *arXiv preprint arXiv:2504.03515*, 2025.

[BBB$^+$22] David Brandfonbrener, Alberto Bietti, Jacob Buckman, Romain Laroche, and Joan Bruna. When does return-conditioned supervised learning work for offline reinforcement learning? *Advances in Neural Information Processing Systems*, 35:1542–1553, 2022.

[Bel57] Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957.

[BS95] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine intelligence 15*, pages 103–129, 1995.

[CDK$^+$24] Chang Chen, Fei Deng, Kenji Kawaguchi, Caglar Gulcehre, and Sungjin Ahn. Simple hierarchical planning with diffusion. *arXiv preprint arXiv:2401.02644*, 2024.

[CS25] Quentin Clark and Florian Shkurti. What do you need for diverse trajectory stitching in diffusion planning? *arXiv preprint arXiv:2505.18083*, 2025.

[CTG$^+$24] Hanqun Cao, Cheng Tan, Zhangyang Gao, Yilun Xu, Guangyong Chen, Pheng-Ann Heng, and Stan Z Li. A survey on generative diffusion models. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[DHY$^+$24] Zibin Dong, Jianye Hao, Yifu Yuan, Fei Ni, Yitian Wang, Pengyi Li, and Yan Zheng. Diffuserlite: Towards real-time diffusion planning. *Advances in Neural Information Processing Systems*, 37:122556–122583, 2024.

[DN21] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[DYH$^+$24] 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.

[EEKL21] Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? *arXiv preprint arXiv:2112.10751*, 2021.

[EHKS23] Eric Eaton, Marcel Hussing, Michael Kearns, and Jessica Sorrell. Replicable reinforcement learning. *Advances in Neural Information Processing Systems*, 36:15172–15185, 2023.

[EUSL22] Benjamin Eysenbach, Soumith Udatha, Russ R Salakhutdinov, and Sergey Levine. Imitating past successes can be very suboptimal. *Advances in Neural Information Processing Systems*, 35:6047–6059, 2022.

[FKN$^+$20] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

[FLZ$^+$24] Linjiajie Fang, Ruoxue Liu, Jing Zhang, Wenjia Wang, and Bing-Yi Jing. Diffusion actor-critic: Formulating constrained policy iteration as diffusion noise regression for offline reinforcement learning. *arXiv preprint arXiv:2405.20555*, 2024.

[GCS$^+$20] Caglar Gulcehre, Sergio Gómez Colmenarejo, Jakub Sygnowski, Thomas Paine, Konrad Zolna, Yutian Chen, Matthew Hoffman, Razvan Pascanu, Nando de Freitas, et al. Addressing extrapolation error in deep offline reinforcement learning. 2020.

[GPM89] Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

[He23] Haoyang He. A survey on offline model-based reinforcement learning. *arXiv preprint arXiv:2305.03360*, 2023.

[HEKJ$^+$23] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[HM22] Charles A Hepburn and Giovanni Montana. Model-based trajectory stitching for improved offline reinforcement learning. *arXiv preprint arXiv:2211.11603*, 2022.

[JDTL22] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[JLL21] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

[KFS$^+$19] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

[KNL21] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[KPH$^+$20] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

[KRNJ20] 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.

[KVYZ23] Amin Karbasi, Grigoris Velegkas, Lin Yang, and Felix Zhou. Replicability in reinforcement learning. *Advances in Neural Information Processing Systems*, 36:74702–74735, 2023.

[KZTL20] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

[LBTPH23] Cong Lu, Philip Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36:46323–46344, 2023.

[Lev18] Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

[LHSL25] Haofei Lu, Dongqi Han, Yifei Shen, and Dongsheng Li. What makes a good diffusion planner for decision making? In *The Thirteenth International Conference on Learning Representations*, 2025.

[LKTF20] 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.

[LSZ$^+$24] Guanghe Li, Yixiang Shan, Zhengbang Zhu, Ting Long, and Weinan Zhang. Diffstitch: Boosting offline reinforcement learning with diffusion-based trajectory stitching. *arXiv preprint arXiv:2402.02439*, 2024.

[LWJZ23] Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In *International Conference on Machine Learning*, pages 20035–20064. PMLR, 2023.

[Moe] Thomas Moerland. Statistical practice.

[Nis24] Soichiro Nishimori. Jax-corl: Clean sigle-file implementations of offline rl algorithms in jax. 2024.

[NZY$^+$25] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

[PFEL24] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. *arXiv preprint arXiv:2410.20092*, 2024.

[PFM$^+$25] Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes rl scalable. *arXiv preprint arXiv:2506.04168*, 2025.

[PMC23] Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[PSTQ21] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys (CSUR)*, 54(5):1–35, 2021.

[RBL+22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[SDWMG15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[Sut19] Richard Sutton. The bitter lesson. *Incomplete Ideas (blog)*, 13(1):38, 2019.

[TGM22] Neil C Thompson, Shuning Ge, and Gabriel F Manso. The importance of (exponentially more) computing power. *arXiv preprint arXiv:2206.14007*, 2022.

[WHZ22] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[WNL+25] Guan Wang, Haoyi Niu, Jianxiong Li, Li Jiang, Jianming Hu, and Xianyuan Zhan. Are expressive models truly necessary for offline rl? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 21062–21070, 2025.

[ZZH+23] 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.

[ZZZ+23] Zhaoyi Zhou, Chuning Zhu, Runlong Zhou, Qiwen Cui, Abhishek Gupta, and Simon Shaolei Du. Free from bellman completeness: Trajectory stitching via model-based return-conditioned supervised learning. *arXiv preprint arXiv:2310.19308*, 2023.