Safe Offline Reinforcement Learning with Real-Time Budget Constraints

Qian Lin [1] **Bo Tang** [2 *] **Zifan Wu** [1 *] **Chao Yu** [1] **Shangqin Mao** [2] **Qianlong Xie** [2] **Xingxing Wang** [2] **Dong Wang** [2]

Abstract

Aiming at promoting the safe real-world deployment of Reinforcement Learning (RL), research on safe RL has made significant progress in recent years. However, most existing works in the literature still focus on the online setting where risky violations of the safety budget are likely to be incurred during training. Besides, in many real-world applications, the learned policy is required to respond to dynamically determined safety budgets (i.e., constraint threshold) in real time. In this paper, we target at the above *real-time budget constraint problem* under the offline setting, and propose **T**rajectory-based **RE**al-time **B**udget **I**nference (**TREBI**) as a novel solution that approaches this problem from the perspective of trajectory distribution. Theoretically, we prove an error bound of the estimation on the episodic reward and cost under the offline setting and thus provide a performance guarantee for TREBI. Empirical results on a wide range of simulation tasks and a real-world large-scale advertising application demonstrate the capability of TREBI in solving real-time budget constraint problems under offline settings.

1. Introduction

In recent years, Reinforcement Learning (RL) (Sutton & Barto, 2018) has achieved great successes in solving complex decision-making problems, such as games (Silver et al., 2016; Vinyals et al., 2019), robotics (Peng et al., 2018; Hanna et al., 2021) and recommendation systems (Zheng et al., 2018). However, the concern of safety still remains a major challenge preventing many real-world deployments of RL, which motivates the research of safe RL (Xu et al., 2022b; Gu et al., 2022). Briefly speaking, safe RL aims to learn a policy that maximizes the long-term reward while satisfying certain constraints. Many safe RL approaches have been proposed in the past few years (Achiam et al., 2017; Zhang et al., 2020; Sootla et al., 2022; Liu et al., 2022a). Unfortunately, most existing approaches only target at the online setting, where potentially risky constraint violations can be incurred during interactions with the real environment. As a kind of data-driven methods, offline RL (Levine et al., 2020) aims to derive a policy from offline data without further real-world exploration, and thus is particularly suitable for safety-critical applications. Despite the recent progress in the offline RL literature (Fujimoto et al., 2019; Kumar et al., 2020; Fujimoto & Gu, 2021), however, there are still limited works focusing on attaining a safe policy under the offline setting.

Moreover, many real-world scenarios require the deployed policy to respond to dynamically determined budgets (i.e., constraint threshold) in real time, rather than simply satisfy the same budget fixed throughout both the training and the deployment phase. In such a *real-time budget constraint problem*, it is often impossible to retrain a policy from scratch for each unseen budget during the deployment of the learned policy. Taking the advertising scenario for example, different advertisers set different budgets in real time, and the advertising system needs to make the optimal decision on the allocation of display channels under different budgets. Also, in the autonomous driving scenario, the road conditions could change rapidly, imposing time-varying constraints on the speed, acceleration or safe distance of the vehicles, hence the policy should be able to quickly adapt to such changes without further time-consuming retraining.

In this paper, we provide a novel offline RL solution, i.e., the **T**rajectory-based **RE**al-time **B**udget **I**nference (**TREBI**), to the above real-time budget constraint problem. The key idea of TREBI lies in approaching the constrained policy optimization problem from the perspective of trajectory distribution. This can not only yield the derivation of the optimal trajectory distribution w.r.t. a certain budget and a given offline dataset, but also enable strict constraint satisfaction (i.e., with probability one in theory) in the trajectory level, which is unlike the rough constraint satisfaction (i.e., expectation over trajectories) in most of previous studies. Note that this is of great significance in real-life applications of RL, particularly for safety-critical domains where only one single constraint violation could possibly lead to catas-

[*]Equal contribution [1]Sun Yat-Sen University [2]Meituan. Correspondence to: Chao Yu <yuchao3@mail.sysu.edu.cn>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

trophic consequences. Moreover, TREBI requires only a mild assumption regarding the quality of offline data. Thus, it is able to learn a safe policy even from a dataset collected by an unconstrained policy.

To model the optimal trajectory distribution w.r.t. a certain budget, we adopt the Diffuser (Janner et al., 2022), a trajectory optimization method based on the diffusion model (Sohl-Dickstein et al., 2015; Ho et al., 2020), as the backbone for TREBI. Specifically, the trajectory distribution of the behavior policy is approximated using the offline dataset in the learning phase, and then adaptive responses to the real-time budgets are achieved by employing a budget-related trajectory planning in the inference phase. Theoretically, we prove an error bound of the offline estimation on the episodic reward and cost, which in turn yields a performance guarantee for TREBI. Empirically, we demonstrate that TREBI can solve the offline real-time budget constraint problem effectively compared to the existing safe offline RL baselines both in simulation and real-world tasks. The capability to utilize datasets with various levels of quality is also verified in the experiments.

2. Related Work

Offline RL Offline RL refers to the problem of learning RL policies from static datasets. The main issue of offline RL is the distribution shift between the state-action pairs in the dataset and those induced by the learned policy (Fujimoto et al., 2019). To address this issue, model-free algorithms generally propose to regularize the learned policy to stay close to the behavior policy (Fujimoto et al., 2019; Kumar et al., 2019; Wu et al., 2019), learn conservative Q-values (Kumar et al., 2020; Lyu et al., 2022), while many model-based algorithms adopt uncertainty estimation (Yu et al., 2020; Kidambi et al., 2020) or conservative value function estimation (Yu et al., 2021; Rigter et al., 2022).

Safe RL One of the most commonly adopted paradigms for solving safe RL problems is alternating between optimizing the policy and updating the Lagrange dual variables, which is referred to as the primal-dual approaches (Chow et al., 2017; Achiam et al., 2017; Tessler et al., 2019; Ding et al., 2020; Stooke et al., 2020; As et al., 2022). However, this kind of approaches can be sensitive to the initialization and the learning rate of Lagrange multipliers. On the contrary, primal approaches (Xu et al., 2021; Sootla et al., 2022) sidestep this issue by excluding the dual variables from the optimization framework. Falling into this category, TREBI also does not need to optimize any dual variables.

Safe Offline RL Despite the significant progress of offline RL in recent years, works focusing on the safe offline setting are still limited (Le et al., 2019; Zhang et al., 2021; Xu et al., 2022a; Lee et al., 2022). CBPL (Le et al., 2019) solves the primal-dual problem from the game-theoretic perspective where the training and evaluation processes are performed alternatively and Lagrange multipliers are updated to narrow the dual gap. CPQ (Xu et al., 2022a) applies pessimistic estimation of cost value functions for out-of-distribution actions, and updates the reward value function as well as the policy only w.r.t. safe actions. Rather than using a policy or value functions like most of previous works, COptiDICE (Lee et al., 2022) directly optimizes the state-action stationary distribution and then extracts the policy from the stationary distribution by importance-weighted behavioral cloning. BCORLE(λ) optimizes the policy under offline BCQ (Fujimoto et al., 2019) and augments the state with Lagrangian multiplier. An concurrent work on safe offline RL, CDT (Liu et al., 2023), also focuses on the same problem setting as ours, where the constraint threshold is dynamically determined. It combines the decision transformer (Chen et al., 2021) with data augmentation and solve the problem from a perspective of multi-objective optimization. As a safe offline method, TREBI differs from the above methods mainly in terms of solving the real-time budget constraint problem in the trajectory level with strict constraint satisfaction.

Trajectory Optimization with Planning Model-based planning methods (Camacho & Alba, 2013; Botev et al., 2013; Nagabandi et al., 2020) have been widely developed to solve the trajectory optimization problem. Such methods are flexible in switching the optimization objective dynamically during planning, and thus suitable for solving the real-time budget constraint problem. PETS (Chua et al., 2018) incorporates uncertainty into the dynamics model by using an ensemble of bootstrapped models and compute the optimal trajectory by CEM (Botev et al., 2013). RCE (Liu et al., 2020), as a model-based planning method for safe RL problems, utilizes a bootstrapped ensemble of neural networks and applies CEM to maximize the expected returns of safe action sequences. It takes into account the model uncertainty and achieves robustness under this uncertainty. However, using single-step dynamics to autoregressively generate trajectories may suffer from the compounding rollout errors of long-term predictions (Janner et al., 2019), which would further lead to tremendous estimation error of value functions. In the safe RL context, such an error can have a detrimental effect on the constraint satisfaction. On the contrary, TREBI performs planning by generating the whole trajectories simultaneously and thus is able to avoid the compounding error issue.

3. Preliminaries

3.1. Problem Setting

The problem of safe RL with a budget can be formulated as a Constrained Markov Decision Process (CMDP), which

is represented by a tuple $(\mathcal{S}, \mathcal{A}, T, r, c, \gamma, b)$, where $\mathcal{S}$ is the state space; $\mathcal{A}$ is the action space; $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function, i.e., $s_{t+1} \sim T(\cdot|s_t, a_t)$; $r : \mathcal{S} \times \mathcal{A} \to [-R_{\max}, R_{\max}]$ and $c : \mathcal{S} \times \mathcal{A} \to [0, C_{\max}]$ denote the reward and cost function, bounded by $R_{\max}$ and $C_{\max}$, respectively; $\gamma \in [0, 1]$ is the discount factor for future reward and cost; and b is the safety budget. A policy $\pi : \mathcal{S} \to P(\mathcal{A})$ maps the states to a probability distribution over actions.

A trajectory τ is composed of a sequence of state-action pairs: $\tau = \{\tau_{s_0}, \tau_{a_0}, \tau_{s_1}, \tau_{a_1}, ..., \tau_{s_L}, \tau_{a_L}\}$, where L is the maximum episode length. The cumulative reward and cumulative cost of trajectory τ are denoted as $R(\tau) = \sum_{t=0}^{L} \gamma^t r(\tau_{s_t}, \tau_{a_t})$ and $C(\tau) = \sum_{t=0}^{L} \gamma^t c(\tau_{s_t}, \tau_{a_t})$, respectively. The policy can be inferred from the distribution of trajectory $p(\tau)$ since the actions can be determined by: $\pi(a|s) \propto \int p(\tau|\tau_{s_0} = s, \tau_{a_0} = a) d\tau$. In other words, the trajectory distribution can be seen as a special policy, since the output action a w.r.t. the input state s can be obtained by adopting the first action of the trajectory sampled from the conditioned distribution $p(\tau|\tau_{s_0} = s)$.

In the context of safe offline RL, we assume no access to online interactions with the environment and offline dataset $\mathcal{D} = \{(s, a, s', r, c)\}$ is the only data available for training. The dataset $\mathcal{D}$ is generated from a set of behavior policies jointly, which might violate the budget constraints. Let $\pi_\beta(a|s) = \frac{N(s,a)}{N(s)}$ denote the empirical behavior policy and $\hat{T}(s'|s, a) = \frac{N(s,a,s')}{N(s,a)}$ the empirical probability function, where $N(\cdot)$ is the count of the corresponding tuple observed in $\mathcal{D}$. Let p_{π_β} denote the distribution of trajectories generated by executing π_β in $\hat{T}$. The goal of safe offline RL with strict constraint satisfaction is to learn a policy π from $\mathcal{D}$ that maximizes the expected cumulative reward while satisfying the cumulative cost in the trajectory level:

$$\max_{\pi} \mathbb{E}_{\tau \sim \pi, T}[R(\tau)]$$
$$\text{s.t. } \forall \tau \sim \pi, T \quad C(\tau) \leq b.$$

where $\tau \sim \pi, T$ denotes that τ is generated by executing π in T.

3.2. Real-time Budget Constraint

The above safe RL problem with a fixed budget can be extended to a real-time budget constraint problem by taking dynamically determined budgets into account. The set of candidate budgets b is denoted as B which is bound and can contain finite or infinite elements. The optimization objective for any given budget $b \in B$ is then given as follows:

$$\max_{\pi_b} \mathbb{E}_{\tau \sim \pi_b, T}[R(\tau)]$$
$$\text{s.t. } \forall \tau \sim \pi_b, T \quad C(\tau) \leq b.$$

where $\pi_b = \pi(\cdot|s, b)$ denotes the policy under the given safety budget b. To solve the real-time budget problem, a simple and intuitive idea is to discretize the range of budgets and train a single policy for each discrete value of budgets with the off-the-shelf fixed-budget algorithms (Xu et al., 2022a; Lee et al., 2022). However, it is intractable to determine the appropriate granularity of budget division without guidance of prior knowledge. A small granularity leads to a large number of candidate policies to train, which is computationally expensive, while a large granularity incurs the difficulty of precisely covering all possible budgets. Safe RL with state augmentation (Zhang et al., 2021; Sootla et al., 2022) is another solution, where budget-related information is injected into the state definition to assist policy in dealing with different budget constraints. However, the growth of dataset size and state dimension caused by state augmentation may lead to scalability issues and extra computational cost during policy training. Our solution is to introduce planning based on the trajectory optimization to provide a flexibility for dynamic switching of constrained optimization objectives during the real-time inference process.

3.3. Diffuser

In diffusion probalitistic models (Sohl-Dickstein et al., 2015), by iteratively adding noise to the data through an forward diffusion process $q(\tau^i|\tau^{i-1})$, the raw data τ^0 in the offline dataset is converted into τ^N that approximately conforms to a noise distribution (e.g., the standard Gaussian). The denoising process $p(\tau^{i-1}|\tau^i)$, as the reverse of $q(\tau^i|\tau^{i-1})$, can be utilized to progressively rebuild the data distribution from the noise distribution. The reverse process is often parameterized as Gaussian, i.e., $p(\tau^{i-1}|\tau^i) = \mathcal{N}(\tau^{i-1}; \mu_\theta(\tau^i, i), \Sigma^i)$, while the forward process is pre-determined.

Trajectory optimization in Diffuser is divided into two phases. During training, a denoising model $p_\theta(\tau^{i-1}|\tau^i)$ is trained to model the trajectory distribution of behavior policy p_{π_β} by minimizing a variational bound on the negative log likelihood: $\min_\theta -E_{\tau \sim D}[\log p_\theta(\tau)]$, i.e., minimizing the KL divergence between p_θ and p_{π_β}. Meanwhile, a function $h(\tau)$ is learned to indicate the optimization objective for τ. For instance, to maximize the cumulative reward of generated trajectories, $h(\tau)$ can be set as the exponential of reward value estimation fitted on the offline trajectories. During inference, the generated trajectories are guided for planning to achieve the optimization objective. Specifically, provided that $\log h(\tau)$ is smooth enough (Sohl-Dickstein et al., 2015), we can sample trajectories from $\tilde{p}_\theta(\tau) = p_\theta(\tau)h(\tau)$ by modifying the original reverse process of diffusion model to:

$$\begin{aligned}\tilde{p}_\theta(\tau^{i-1}|\tau^i) &\propto p_\theta(\tau^{i-1}|\tau^i)h(\tau^i) \\ &\approx \mathcal{N}(\tau^{i-1}; \mu_\theta(\tau^i, i) + \Sigma^i g, \Sigma^i), \qquad (1)\end{aligned}$$

where τ^i denotes the trajectory obtained in the ith step of reverse process and $g = \nabla_{\tau^i} \log h(\tau^i)$.

4. Trajectory-based REal-time Budget Inference (TREBI)

In this section, we first introduce a trajectory perspective for the safe offline RL in Subsection 4.1, and then provide a practical algorithm to this problem in Subsection 4.2, followed by key theoretical analysis in Subsection 4.3.

4.1. A Trajectory Optimization Perspective of Safe Offline RL

In essence, trajectory optimization methods (Chen et al., 2021; Janner et al., 2021; Furuta et al., 2021; Janner et al., 2022) fall into the model-based RL category by simultaneously incorporating the dynamics and the policy into one single optimization objective. In the context of offline RL, the model-based objective of policy π can be written as maximizing the expected return under the empirical dynamics $\hat{T}$, i.e., $\max_\pi \mathbb{E}_{\tau\sim\pi,\hat{T}}[R(\tau)]$. To alleviate the distribution shift issue, a common practice in offline RL is to constrain the divergence between the target policy and the behavior policy (Kumar et al., 2019; Wu et al., 2019; Fujimoto & Gu, 2021), which is referred to as the *in-distribution constraint* in this paper. From the perspective of trajectory optimization, the in-distribution constraint can be expressed in the following form: $D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \leq \epsilon$, where $q(\tau)$is target trajectory distribution and ϵ is an approximately chosen divergence threshold. Then, the policy optimization problem can be converted to the form of trajectory optimization as follows:

$$\max_{q(\tau)} \mathbb{E}_{\tau\sim q(\tau)}[R(\tau)], \tag{2}$$

$$\text{s.t. } D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \leq \epsilon. \tag{3}$$

It is worth noting that such a conversion is equivalent only when the dynamics induced by the optimized q is consistent with $\hat{T}$. In environments with deterministic dynamics, this condition is satisfied since the inconsistency between $\hat{T}$ and the dynamics induced by the optimized q would result in $p_{\pi_\beta}(\tau) = 0$ and $q(\tau) \neq 0$ simultaneously for some τ, which would in turn lead to an infinite KL Divergence in the left hand side of Eq. (3) and thus violates the in-distribution constraint. In environments with probabilistic dynamics, such a conversion is not strictly equivalent, but the in-distribution constraint can still induce a trade-off between estimating some parts of the transition dynamics inaccurately (which would increase $D_{KL}(q(\tau)||p_{\pi_\beta}(\tau))$) and optimizing the expected return of the trajectory distribution (i.e., $\mathbb{E}_{\tau\sim q(\tau)}[R(\tau)]$). More theoretical analysis will be discussed in Subsection 4.3.

One of the advantages of optimizing the policy in the trajectory level is the capability of enabling strict constraint satisfaction, i.e., realizing the per-trajectory constraint with probability one in theory, which is in contrast to the constraint of the expectation over trajectories in most of previous studies. Formally, for any given budget b, the per-trajectory safety constraint can be formulated as $\int_{\{\tau|C(\tau)\leq b\}} q(\tau)d\tau = 1$, where the integral is taken over all constraint-satisfied trajectories. For brevity, we slightly abuse the notation $\int_{C(\tau)\leq b}$ to represent $\int_{\{\tau|C(\tau)\leq b\}}$ in the following. Then, the offline constrained trajectory optimization problem can be given as follows:

$$\max_{q(\tau)} \mathbb{E}_{\tau\sim q(\tau)}[R(\tau)] \tag{4}$$

$$\text{s.t.} \int_{C(\tau)\leq b} q(\tau)d\tau = 1 \tag{5}$$

$$D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \leq \epsilon. \tag{6}$$

While the closed-form solution satisfying both Eq. (5) and Eq. (6) does not necessarily exist, we show that the such a solution can be achieved provided a mild condition is met:

Theorem 4.1. *If the trajectory distribution of the behavior policy satisfies:*

$$\int_{C(\tau)\leq b} p_{\pi_\beta}(\tau)d\tau \geq e^{-\epsilon}, \tag{7}$$

then the optimal trajectory distribution for problem (4)(5)(6) *exists and takes the following form:*

$$q_b^*(\tau) = \begin{cases} p_{\pi_\beta}(\tau)\exp(\alpha R(\tau))/Z & \text{if } C(\tau) \leq b; \\ 0 & \text{otherwise}, \end{cases} \tag{8}$$

where α is a constant depending on ϵ and b, and $Z = \int_{C(\tau)\leq b} p_{\pi_\beta}(\tau)\exp(\alpha R(\tau))d\tau$ is a constant normalizer to make sure that $q_b^*(\tau)$ is a valid distribution.

Proof. Please refer to Appendix A.1, Theorem A.1. □

Note that condition (7) in Theorem 4.1 states that the probability of the constraint-satisfied trajectories generated by the behavior policy should not be too small, which implies that there exist a sufficient number of constraint-satisfied trajectories in dataset. This condition can be easily satisfied for most of possible budgets since the behavior policy usually consists of multiple policies which generate the trajectories with diverse cost. Then, Theorem 4.1 shows that the constrained optimal trajectory distribution q_b^* can be simply obtained by zeroing the probability of unsafe trajectories and redistributing the probability of the remaining trajectories based on their returns.

A Probabilistic Inference Perspective The result similar to Eq. (8) can be also derived from the perspective of probabilistic inference (Levine, 2018). Specifically, an optimal

variable O is introduced to denote whether a given trajectory is optimal. Following the probabilistic graphical models in (Levine, 2018), the likelihood of being optimal given trajectory τ can be represented as the exponential of the discounted cumulative reward: $P(O=1|\tau) \propto \exp(\alpha' R(\tau))$, where α' is a temperature parameter. Since the posterior we are concerned with (i.e., $p_{\pi_\beta}(\tau|O=1)$) is intractable, an auxiliary trajectory distribution $q(\tau)$ is introduced to serve as an approximation, and the discrepancy between $q(\tau)$ and $p_{\pi_\beta}(\tau|O=1)$ can be derived as (see Appendix A.2 for the proof):

$$\begin{aligned} &D_{KL}(q(\tau)||p_{\pi_\beta}(\tau|O=1)) \\ =&\log P_{\pi_\beta}(O=1) - J(q(\tau)), \end{aligned}$$

where $J(q(\tau)) = -D_{KL}(q(\tau)||p_1(\tau))$ is the evidence lower bound (ELBO) and $p_1(\tau) \propto p_{\pi_\beta}(\tau)\exp(\alpha' R(\tau))$. Note that $\log P_{\pi_\beta}(O=1)$ is independent of $q(\tau)$, hence the discrepancy between $q(\tau)$ and $p_{\pi_\beta}(\tau|O=1)$ can be minimized by optimizing the ELBO.

Constraining $q(\tau)$ within a feasible distribution family to satisfy the safety budget with probability one, the optimization problem becomes the following form:

$$\max_{q(\tau)\in\Pi_b} \quad -D_{KL}(q(\tau)||p_1(\tau)) \tag{9}$$

$$\Pi_b = \{q(\tau) : \int_{C(\tau)\le b} q(\tau)d\tau = 1\}. \tag{10}$$

The solution of Eq. (9)(10) has the same form of (8) (see Appendix A.2 for the detailed derivation), except that here α' (corresponding to α in Eq. (8)) is a tunable hyper-parameter and does not vary w.r.t. budget b and divergence threshold ϵ. The explanation of the difference between α and α' is also provided in Appendix A.2.

4.2. A Practical Algorithm

To obtain the optimal trajectory distribution in Theorem 4.1, we adopt Diffuser (Janner et al., 2022), a recently proposed trajectory optimization framework that has demonstrated significant scalability to long-horizon trajectories, as the backbone of our algorithm. The Diffuser model first approximates the trajectory distribution in the dateset, and then performs optimization over the approximate distribution via a guided planning process. Recall that this planning process requires the target optimal distribution to be in the form of $p_\theta(\tau)h(\tau)$. Thus we rewrite $q_b^*(\tau)$ in Eq. (8) as:

$$q_b^*(\tau) = p_{\pi_\beta}(\tau)h_b(\tau), \tag{11}$$

where $h_b(\tau) \propto \exp(\alpha\tilde{R}_b(\tau))$ and

$$\tilde{R}_b(\tau) = \begin{cases} R(\tau) & \text{if } C(\tau)\le b; \\ -\infty & \text{otherwise.} \end{cases} \tag{12}$$

Algorithm 1 **T**rajectory-based **RE**al-time **B**udget **I**nference

Require: Diffuser model μ_θ, parameter α, covariances Σ^i, budget b, hyper-parameter n;

```
z_0 = b;
for t = 0, ..., T do
    Observe state s; initialize trajectories τ^N ~ N(0, I);
    for i = N, ..., 1 do
        if C(τ^i) ≤ z_t then
            g = α∇R(τ^i);
        else
            g = α(∇R(τ^i) − n · ∇C(τ^i));
        end if
        μ ← μ_θ(τ^i);
        τ^{i−1} ~ N(μ + Σ^i g, Σ^i);
        τ^{i−1} ← s;
    end for
    Execute the first action τ_{a_0} and get cost c_t;
    z_{t+1} = (z_t − c_t)/γ.
end for
```

The form of distribution product in Eq. (11) enables the Diffuser-like two-phase trajectory optimization, which consists of first approximating $p_{\pi_\beta}(\tau)$ and then performing guided planning over the approximate distribution via $h_b(\tau)$. Note that an infinite $\tilde{R}_b(\tau)$ would lead to value mutations of $h_b(\tau)$ and may thus do harm to the guiding process of the diffusion model, as mentioned in Subsection 3.3. Therefore, we introduce an alternative trajectory distribution to $q_b^*(\tau)$:

$$q_{b,n}^*(\tau) = p_{\pi_\beta}(\tau)h_{b,n}(\tau), \tag{13}$$

where $h_{b,n}(\tau) \propto \exp(\alpha\tilde{R}_{b,n}(\tau))$, and

$$\tilde{R}_{b,n}(\tau) = R(\tau) - n\cdot\mathbb{I}(C(\tau)>b)\cdot(C(\tau)-b). \tag{14}$$

Intuitively, the hyper-parameter n controls the "conservativeness" of the planning process, meaning that a larger n induces a smaller probability of constraint violation. Formally,

$$\lim_{n\to\infty}\int_{C(\tau)>b} q_{b,n}^*(\tau)d\tau = 0. \tag{15}$$

To sample from the approximate optimal distribution $q_{b,n}^*$, we adapt the guided planning of Diffuser (i.e., the modified reverse process (1)) to the constrained setting by replacing $h(\tau)$ and g respectively with $h_{b,n}(\tau)$ and g_b, where

$$\begin{aligned} g_b &= \nabla_{\tau^i}\log h_{b,n}(\tau^i) \\ &= \begin{cases} \alpha\nabla_{\tau^i}R(\tau^i) & \text{if } C(\tau^i)\le b; \\ \alpha(\nabla_{\tau^i}R(\tau^i) - n\nabla_{\tau^i}C(\tau^i)) & \text{otherwise.} \end{cases} \end{aligned} \tag{16}$$

By assigning different budget values to b in Eq. (16) during inference, the trajectories satisfying the corresponding

budget constraint can be sampled from the trained model without time-consuming retraining, which is the major advantage of our approach in addressing the real-time budget constraint problem. In addition, in practice the reward and the cost value function (i.e., R and C in Eq. (16)) are often unknown and need to be approximated. To reduce the long-horizon approximation error, the trajectory optimization are re-performed at some predetermined time intervals [1] during a long-horizon environment episode, based on the current state and the remaining budget. Specifically, at the tth time step, the remaining trajectory τ is regenerated by taking s_t as the first state τ_{s_0} and $z_t = (b - \sum_{k=0}^{t-1} \gamma^k c(s_k, a_k))/\gamma^t$ as a scaled version of the remaining safety budget. Formally, the following time-varying constraint should be satisfied during each trajectory optimization:

$$\begin{aligned} & C(\tau) \leq z_t \\ \text{where} \quad & \tau_{s_0} = s_t, \\ & z_0 = b, \\ & z_t = (z_{t-1} - c(s_{t-1}, a_{t-1}))/\gamma. \end{aligned}$$

Keeping aware of the remaining budget, TREBI can adjust its "conservativeness" flexibly with the time steps to better ensure constraint satisfaction. The pseudo code of TREBI is presented in Algorithm 1.

4.3. Theoretical Analysis

In this subsection, we derive a performance guarantee for the reward maximization and constraint satisfaction of our optimal trajectory distribution in Proposition 4.1, and provide analysis and interpretations of the theoretical results. To begin with, we present an error bound of the offline estimation on the episodic reward and cost, which serves as a useful lemma in the derivation of our conclusion:

Lemma 4.2. *Let π be a policy derived from $q(\tau)$. Denote $J_{T_q}(\pi) = \mathbb{E}_{\tau\sim q(\tau)}[R(\tau)]$ as the expected return of $q(\tau)$ and $J_T(\pi)$ as the expected return of π under the real environment T. Let $C(\tau)$ and $\hat{C}(\tau)$ be the episodic cost of trajectory τ under the T and empirical dynamics $\hat{T}$, respectively. With probability at least $1-\delta$, the gap between $J_{T_q}(\pi)$ and $J_T(\pi)$ is bounded by:*

$$\begin{aligned} |J_{T_q}(\pi) - J_T(\pi)| \leq & \frac{2R_m}{1-\gamma}\Bigg[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} + \\ & \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L} \frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\Bigg], \end{aligned} \tag{17}$$

where the first term $(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}}$ vanishes in deterministic environment, and for each feasible trajectory τ,

$$|C(\tau) - \hat{C}(\tau)| \leq \sum_{t=0}^{L} \frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}}, \tag{18}$$

where $C_{T,\delta}, C_{c,\delta}$ are constants depending on the concentration properties of $T(s'|s,a), c(s,a)$, respectively, and $\delta \in (0,1)$. $N(s,a)$ is the counts for each state-action pair (s,a) in the offline dataset.

Proof. Please refer to Appendix A.3, Lemma A.3. □

According to Eq. (17), the estimation error of the expected return is controlled mainly by two terms: $(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}}$ and $\mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L} \frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]$. The first term is caused by the inconsistency between $\hat{T}$ and the dynamics induced by the optimized $q(\tau)$. As described in Subsection 4.1, this inconsistency only exists in environment with probabilistic dynamics. Lemma 4.2 theoretically show that such an inconsistency in probabilistic dynamics can be controlled by ϵ. Thus, applying a strict in-distribution constraint can reduce the dynamics inconsistency in the probabilistic case, but at a cost of limiting the parameter search space for trajectory optimization. The second term represents the uncertainty of dynamics on the trajectory generated by π and $\hat{T}$. This uncertainty generally decreases as the size of dataset increases. Besides, this term can be further replaced by $\frac{2(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi_\beta,\hat{T}}\left[\sum_{t=0}^{L} \frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]$ (see Appendix A.3 for the proof), where the two terms respectively represent the proximity of π to behavior policy π_β and the uncertainty of π_β. Thus, by adopting a stricter in-distribution constraint, the error caused by dynamics uncertainty can be reduced toward the uncertainty of behavior policy, which can be regarded approximately as a lower bound of the feasible policy's uncertainty.

As for the trajectory cost estimation, the bound in Eq. (18) shows that the more in-distribution state-action pairs the trajectory τ contains, the more accurate its cumulative cost estimation will be. Therefore, employing the constraint in Eq. (3) induces better constraint satisfaction. Having Lemma 4.2, the performance guarantee for the optimal trajectory distribution $q_b^*(\tau)$ can be easily derived as:

Proposition 4.3. *Denoting $\pi_{q_b^*}$ as the policy induced by the constrained optimal trajectory distribution q_b^*, for any policy π derived from the trajectory distribution that simultaneously satisfies the in-distribution constraint (6) and the safety constraint (5), the following inequality holds with probability at least $1-\delta$:*

$$\begin{aligned} J_T(\pi_{q_b^*}) \geq J_T(\pi) - & \frac{4R_m}{1-\gamma}\Bigg[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} \\ & + \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L} \frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\Bigg], \end{aligned} \tag{19}$$

where the first term $(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}}$ vanishes in environ-

[1] The ablation study of the time interval is shown in Appendix E.

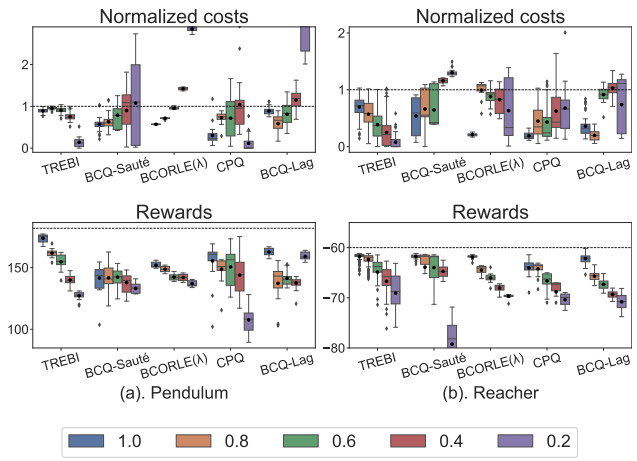


Figure 1. The results on Pendulum and Reacher with regard to the normalized episodic costs (the top row) and the episodic rewards (the bottom row), under the constraints w.r.t. five different budgets (bars with different colors). The constraint w.r.t. a budget is satisfied only when the value of the normalized cost is less than one. The dashed lines in the cost plot and the reward plot indicate the normalized episodic cost threshold and the expected episodic reward of the converged unconstrained policy trained online, respectively. Note that for the convenience of display, some of the boxes are not completely shown in the figure, e.g., the reward of BCQ-Sauté on Reacher with budget 0.2.

ments with deterministic dynamics. For each trajectory τ generated from q_b^*, the episodic cost of τ is bounded with probability at least $1-\delta$ as follow:

$$C(\tau) \leq b + \sum_{t=0}^{L} \frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}}. \tag{20}$$

Proof. Please refer to Appendix A.3, Proposition A.7. □

In the case of deterministic dynamics, $\pi^*_{q_b}$ can be almost as good as any policy derived from the trajectory distribution that satisfies both the in-distribution constraint and the safety constraint, provided that uncertainty of dataset is small, which generally corresponds to having sufficient samples in the datase. In the case of probabilistic dynamics, the performance gap between $\pi^*_{q_b}$ and other feasible policies can be controlled by the in-distribution constraint. Eq. (20) suggests that when the trajectories generated from $q_b^*(\tau)$ are in-distribution, the upper bound of the cumulative cost is more likely to be controlled.

5. Experiments

5.1. Simulation Experiments

Environments and Datasets We first validate TREBI's capability in solving real-time budget constraint problems on two OpenAI Gym tasks with additional safety constraints (Pendulum swing-up and Reacher) (Sootla et al., 2022), three MuJoCo tasks (Hopper-v2, HalfCheetah-v2, Walker2d-v2) (Todorov et al., 2012) with speed limit (Zhang et al., 2020; Yang et al., 2022) and two Bullet-Safety-Gym tasks (SafetyCarCircle-v0, SafetyBallReach-v0) (Gronauer, 2022). For Pendulum swing-up, Reacher and two Bullet-Safety-Gym tasks, we train an unconstrained SAC policy online and take its replay buffer as the offline dataset. For MuJoCo tasks, we test on three types of datasets in the D4RL benchmark (Fu et al., 2020), i.e., medium, medium-replay and medium-expert, to verify TREBI's capability of utilizing datasets with various levels of quality. Note that the offline data in our experiments are collected by *unconstrained* behavior policies. More environmental introduction and implementation details can be found in Appendix B.

Baselines We compare TREBI [2] with: 1) BCQ-Lagrangian, which combines BCQ (Fujimoto et al., 2019) with the Lagrangian approach; 2) CPQ (Xu et al., 2022a), which considers out-of-distribution actions as unsafe and updates the reward critic using only safe state-action pairs; 3) BCORLE(λ) (Zhang et al., 2021), which merges Lagrangian multiplier λ into the state and finds an optimal value of λ for a given budget by bisection search; 4) BCQ-Sauté, which combines BCQ with Sauté (Sootla et al., 2022) by augmenting the state space with the remaining safety budget and assigning large penalty values to the reward when the remaining safety budget is less than 0. Note that BCORLE(λ) and BCQ-Sauté have the potential to handle different budgets, as the most of possible budgets have been taken into account during policy training by state augmentation.

Evaluation Protocols The approaches are evaluated on real-time budgets varying across 5 different values, i.e., $0.2, 0.4, 0.6, 0.8$ and 1.0 times of the maximum budget respectively[3]. The maximum budgets for each tasks are detailed in Appendix D. We normalize the episodic cost of a trajectory by dividing it with the corresponding budget, resulting in a unified constraint budget of one. We adopt the box-and-whisker plots to show the results of constraint satisfaction in the trajectory level. Concretely, the box shows the median, q_3 and q_1 quartiles of the distributions, and whiskers depict the error bounds $1.5(q_3 - q_1)$ and outliers, while the mean is marked by a black dot. Such indicators serve as crucial criteria for the validation of constraint satisfaction with probability one. Note that among the baselines, BCQ-Lagrangian and CPQ are only applicable to fix-budget settings by design, while BCORLE(λ) and BCQ-Sauté can handle different safety budgets without retraining. Therefore, BCQ-Lagrangian and CPQ are retrained for each

[2] Implementation details can be found in Appendix D.

[3] In Appendix E, we present the results of additional experiments where the ratios to the maximum budget are sampled from interval $[0, 1]$ rather than 5 fixed values.

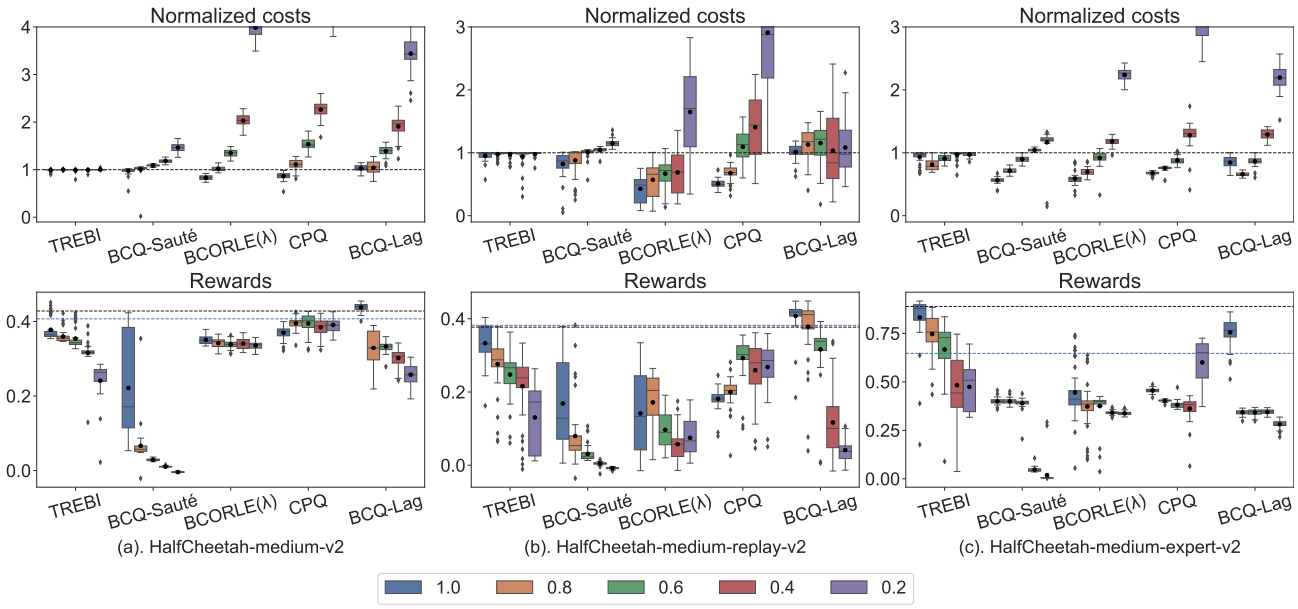


Figure 2. Results on HalfCheetah using three different types of datasets. The black dashed lines and the blue dashed lines in the reward plots indicate the expected episodic rewards of the unconstrained Diffuser and BCQ respectively.

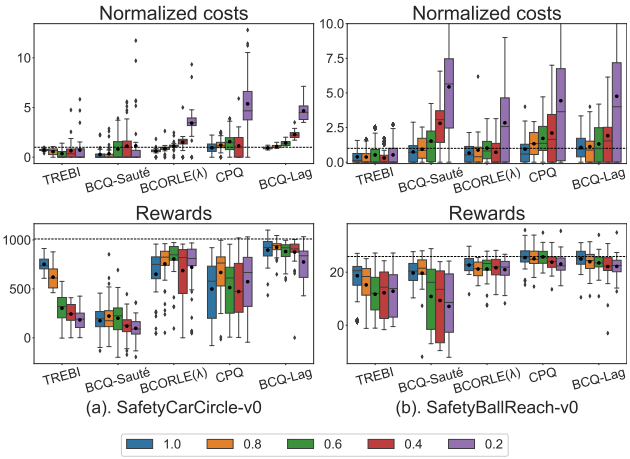


Figure 3. The results on SafetyCarCircle and SafetyBallReach. The dashed lines in reward plots indicate the expected episodic reward of the behavior policy in the final iteration of online training.

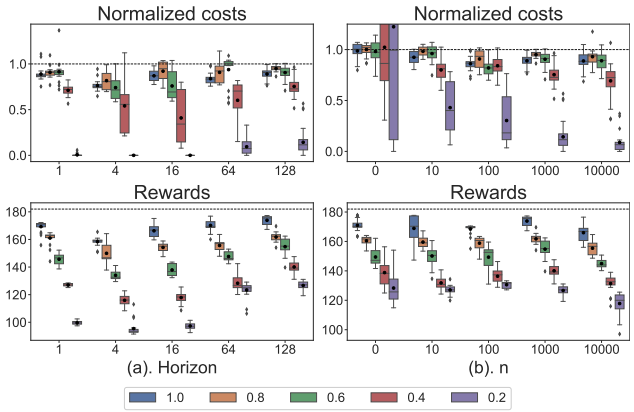


Figure 4. Ablation studies on Pendulum.

budget before they are evaluated, while BCORLE(λ), BCQ-Sauté and TREBI are trained only once before evaluation for each budget.

Comparative Evaluations Figures 1, 2 and 3 show the results for Pendulum, Reacher, HalfCheetah and two Bullet-SafetyGym tasks. Due to space limit, the results on other MuJoCo tasks are placed in Appendix E. Overall, TREBI incurs less constraint violation compared to the baselines and achieves the competitive performance on the episodic reward. Importantly, TREBI achieves a high level of per-trajectory constraint satisfaction on most of the tasks and budgets, which is consistent with our theoretical setting. In contrast, the baselines not only show large variances in the episodic costs, but also suffer from a high probability of serious constraint violation especially when the budget is low. Moreover, the consistent superior performance across the three different types of MuJoCo datasets also serves as a proof for TREBI's capability in utilizing data with various levels of quality.

Ablation 1 It is often impractical to sample full-length episodes in planning due to the computational complexity. Hence, following the implementation of Diffuser, we sample and optimize the trajectories within only a fixed horizon. The effect induced by different horizon settings are given

Table 1. Results in the oCPM advertising bidding task.

Algorithm	Relative Improvement	Violation rate
TREBI	**7.15%**	**14.69%**
BCORLE(λ)	1.55%	18.66%
BCQ-Sauté	-2.55%	21.49%
CPQ	-0.76%	20.75%
BCQ-Lag	0.00%	22.94%

in Figure 4(a) , showing a positive trend (higher reward & better constraint satisfaction) as the horizon increases.

Ablation 2 In Eq. (14), the parameter n is introduced to avoid numerical issues of the guiding process. Figure 4(b) provides the ablation result of n. Roughly speaking, when n varies within a relatively low-value range, a larger n leads to better constraint satisfaction and higher rewards. However, when n comes to an extremely large value, e.g., 10000, the above two metrics instead become worse, which verifies that a large n can degrade the smoothness of the guiding distribution $h(\tau)$ and thus has a detrimental effect on the modified reverse process (Sohl-Dickstein et al., 2015).

5.2. Real-world Environments

We further evaluate TREBI in a real-world advertising bidding scenario w.r.t. the optimized cost-per-mille (oCPM).

Environment In the advertising bidding scenario, a sequence of advertisements (ads) slots is going to be presented to users browsing the site, and the advertisers would bid for these slots to display their advertisements. Advertisers assign the target ROI (return on investment) and their maximum total payments for the advertisements. Then, the policy trained by the auto-bidding algorithm helps the advertisers bid for each slot. The advertisers consume their budgets and obtain the revenue only when their ads are successfully displayed, and the policy needs to ensure the average ROI to be higher than the assigned target. Thus, the objective is maximizing the summation of the payments of all advertisers while satisfying the ROI constraint. In this scenario, a major challenge is to respond to the ROI targets set by each advertiser, which would change once in a day. Due to the high risk of online training, obtaining a safe policy using merely offline data is promising. Appendix C provides more detailed information of this scenario.

Dataset and Baselines The offline dataset is collected from a real-world ads bidding platform, including 1938151000 bidding records of 42376 advertisers sampled from a one-month bidding history. The baselines adopted in this experiment are the same as those in Subsection 5.2.

Evaluation Protocol We deploy the algorithms on the real-world ad bidding platform for 10 days, conducting evaluations on the 485904618 bidding records randomly sampled from 29660 advertisers. As explained in Subsection 5.1, BCORLE(λ) and BCQ-Sauté are trained only once by augmenting the samples w.r.t. the real-time budgets. As for BCQ-Lagrangian and CPQ, since the budget varies from 20 to around 2000 while the online deployment requires real-time responses to the budgets, it is impractical to retrain these two baselines on each budget. Thus, we train BCQ-Lagrangian and CPQ only once using the mean of budget in the dataset as the fixed budget.

Comparative Evaluation Table 1 reports the total pay improvement relative to BCQ-Lagrangian and the violation rate, which is the proportion of trajectories violating ROI constraints during evaluation. Due to the extremely large scale of the bidding platform and the dramatic variation of the real-time budgets, none of the algorithms can achieve the 100% constraint satisfaction. However, compared to the baselines, TREBI achieves a higher episodic reward and generates fewer trajectories violating the constraint, which demonstrates the potential of TREBI to solve the real-time budget constraint problem in real-world applications.

6. Conclusion

In this paper, we proposed TREBI, a novel solution to the real-time budget constraint problem under the offline setting. We view the safe offline RL problem from a trajectory optimization perspective and solve it by a planning method, which enables the strict constraint satisfaction in the trajectory level as well as efficient utilization of unconstrained offline dataset. We provide an analysis on the performance guarantee for TREBI theoretically, and verify its capability empirically in addressing the real-time budget constraint problems in a wide range of simulation and real-world tasks.

Inheriting from the Diffuser method, our method would be more computationally expensive during inference than non-planning algorithms, due to the slow iterative trajectory generation of the diffusion model. In this regard, several improvements can be made to reduce the time complexity, such as warm-starting the diffusion process (Janner et al., 2022) and reducing the control frequency (referred to Appendix E). In the future, we plan to study the technology for accelerating diffusion planning and explore other trajectory optimization frameworks to further improve the efficiency of TREBI.

Acknowledgements

We gratefully acknowledge support from the National Natural Science Foundation of China (No. 62076259), the Fundamental and Applicational Research Funds of Guangdong province (No. 2023A1515012946), and the Fundamental Research Funds for the Central Universities-Sun Yat-sen University. This research was supported by Meituan.

References

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained policy optimization. In *International Conference on Machine Learning*, pp. 22–31. PMLR, 2017.

As, Y., Usmanova, I., Curi, S., and Krause, A. Constrained policy optimization via bayesian world models. *arXiv preprint arXiv:2201.09802*, 2022.

Auer, P., Jaksch, T., and Ortner, R. Near-optimal regret bounds for reinforcement learning. *Advances in Neural Information Processing Systems*, 21, 2008.

Botev, Z. I., Kroese, D. P., Rubinstein, R. Y., and L'Ecuyer, P. The cross-entropy method for optimization. In *Handbook of Statistics*, volume 31, pp. 35–59. Elsevier, 2013.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Camacho, E. F. and Alba, C. B. *Model predictive control*. Springer science & business media, 2013.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, 34:15084–15097, 2021.

Chow, Y., Ghavamzadeh, M., Janson, L., and Pavone, M. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

Chua, K., Calandra, R., McAllister, R., and Levine, S. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in Neural Information Processing Systems*, 31, 2018.

De Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134:19–67, 2005.

Ding, D., Zhang, K., Basar, T., and Jovanovic, M. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems*, 33:8378–8390, 2020.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pp. 2052–2062. PMLR, 2019.

Furuta, H., Matsuo, Y., and Gu, S. S. Generalized decision transformer for offline hindsight information matching. In *International Conference on Learning Representations*, 2021.

Gronauer, S. Bullet-safety-gym: A framework for constrained reinforcement learning. 2022.

Gu, S., Yang, L., Du, Y., Chen, G., Walter, F., Wang, J., Yang, Y., and Knoll, A. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022.

Hanna, J. P., Desai, S., Karnan, H., Warnell, G., and Stone, P. Grounded action transformation for sim-to-real reinforcement learning. *Machine Learning*, 110(9):2469–2499, 2021.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Janner, M., Fu, J., Zhang, M., and Levine, S. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Janner, M., Li, Q., and Levine, S. Offline reinforcement learning as one big sequence modeling problem. *Advances in Neural Information Processing Systems*, 34: 1273–1286, 2021.

Janner, M., Du, Y., Tenenbaum, J., and Levine, S. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pp. 9902–9915. PMLR, 2022.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 21810–21823, 2020.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020.

Le, H., Voloshin, C., and Yue, Y. Batch policy learning under constraints. In *International Conference on Machine Learning*, pp. 3703–3712. PMLR, 2019.

Lee, J., Paduraru, C., Mankowitz, D. J., Heess, N., Precup, D., Kim, K.-E., and Guez, A. Coptidice: Offline constrained reinforcement learning via stationary distribution correction estimation. In *International Conference on Learning Representations*, 2022.

Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Liu, Z., Zhou, H., Chen, B., Zhong, S., Hebert, M., and Zhao, D. Constrained model-based reinforcement learning with robust cross-entropy method. *arXiv preprint arXiv:2010.07968*, 2020.

Liu, Z., Cen, Z., Isenbaev, V., Liu, W., Wu, S., Li, B., and Zhao, D. Constrained variational policy optimization for safe reinforcement learning. In *International Conference on Machine Learning*, pp. 13644–13668. PMLR, 2022a.

Liu, Z., Guo, Z., Cen, Z., Zhang, H., Tan, J., Li, B., and Zhao, D. On the robustness of safe reinforcement learning under observational perturbations. *arXiv preprint arXiv:2205.14691*, 2022b.

Liu, Z., Guo, Z., Yao, Y., Cen, Z., Yu, W., Zhang, T., and Zhao, D. Constrained decision transformer for offline safe reinforcement learning. *arXiv preprint arXiv:2302.07351*, 2023.

Lyu, J., Ma, X., Li, X., and Lu, Z. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

Nagabandi, A., Konolige, K., Levine, S., and Kumar, V. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pp. 1101–1112. PMLR, 2020.

Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. In *International Conference on Robotics and Automation*, pp. 3803–3810. IEEE, 2018.

Rigter, M., Lacerda, B., and Hawes, N. Rambo-rl: Robust adversarial model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2022.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Sootla, A., Cowen-Rivers, A. I., Jafferjee, T., Wang, Z., Mguni, D. H., Wang, J., and Ammar, H. Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *International Conference on Machine Learning*, pp. 20423–20443. PMLR, 2022.

Stooke, A., Achiam, J., and Abbeel, P. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pp. 9133–9143. PMLR, 2020.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Tessler, C., Mankowitz, D. J., and Mannor, S. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2019.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wu, Y., Tucker, G., and Nachum, O. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Xu, H., Zhan, X., and Zhu, X. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8753–8760, 2022a.

Xu, M., Liu, Z., Huang, P., Ding, W., Cen, Z., Li, B., and Zhao, D. Trustworthy reinforcement learning against intrinsic vulnerabilities: Robustness, safety, and generalizability. *arXiv preprint arXiv:2209.08025*, 2022b.

Xu, T., Liang, Y., and Lan, G. Crpo: A new approach for safe reinforcement learning with convergence guarantee. In *International Conference on Machine Learning*, pp. 11480–11491. PMLR, 2021.

Yang, L., Ji, J., Dai, J., Zhang, Y., Li, P., and Pan, G. Cup: A conservative update policy algorithm for safe reinforcement learning. *arXiv preprint arXiv:2202.07565*, 2022.

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J. Y., Levine, S., Finn, C., and Ma, T. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. Combo: Conservative offline model-based policy optimization. *Advances in Neural Information Processing Systems*, 34:28954–28967, 2021.

Zhang, Y., Vuong, Q., and Ross, K. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems*, 33:15338–15349, 2020.

Zhang, Y., Tang, B., Yang, Q., An, D., Tang, H., Xi, C., Li, X., and Xiong, F. Bcorle (λ): An offline reinforcement learning and evaluation framework for coupons allocation in e-commerce market. *Advances in Neural Information Processing Systems*, 34:20410–20422, 2021.

Zheng, G., Zhang, F., Zheng, Z., Xiang, Y., Yuan, N. J., Xie, X., and Li, Z. Drn: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pp. 167–176, 2018.

A. Proofs and Derivations

A.1. Proof of Theorem 4.1

Theorem A.1 (4.1 in the main paper)**.** *If $\int_{C(\tau)\leq b} p_{\pi_\beta}(\tau)d\tau \geq e^{-\epsilon}$ holds, there exists an optimal trajectory distribution for problem*

$$\max_{q(\tau)} \mathbb{E}_{\tau\sim q(\tau)}[R(\tau)] \tag{21}$$

$$\text{s.t.} \int_{C(\tau)\leq b} q(\tau)d\tau = 1 \tag{22}$$

$$D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \leq \epsilon, \tag{23}$$

which has the form of

$$q_b^*(\tau) = \begin{cases} p_{\pi_\beta}(\tau)\exp(\alpha R(\tau))/Z & \text{if } C(\tau)\leq b; \\ 0 & \text{otherwise}, \end{cases} \tag{24}$$

where α depends on ϵ and b, and $Z = \int_{C(\tau)\leq b} p_{\pi_\beta}(\tau)\exp(\alpha R(\tau))d\tau$ is a constant normalizer to make sure that $q_b^*(\tau)$ is a valid distribution.

Before presenting the proof of Theorem 4.1, we first prove a lemma that will be used later:

Lemma A.2. *Assuming that x is a continuous variable, $p(x)$ and $q(x)$ are functions w.r.t. x, $\mathcal{A}\subseteq\mathcal{X}$ where $\mathcal{X}$ is the domain of x by definition, the optimization problem*

$$\min_q J(q) = \int_{\mathcal{A}} q(x)\log\frac{q(x)}{p(x)}, \tag{25}$$

$$\text{s.t.} \int_{\mathcal{A}} q(x) = 1. \tag{26}$$

have the solution: $q^*(x) = \frac{p(x)}{\int_{\mathcal{A}} p(x)dx}$.

Proof. We first construct the Lagrange function as:

$$L(x,q,\dot{q}) = \int_{\mathcal{A}} q(x)\log\frac{q(x)}{p(x)} + \lambda q(x).$$

It is obvious that the original problem has a minimum point q^*. Thus, there exists a constant multiplier λ such that q^* satisfies the Euler-Lagrange Equation:

$$\frac{\partial L}{\partial q} - \frac{d}{dx}\frac{\partial L}{\partial \dot{q}} = 0,$$

which yields $\frac{\partial L}{\partial q} = 0$ and thus

$$\frac{\partial L}{\partial q} = \log\frac{q(x)}{p(x)} + 1 + \lambda = 0$$
$$\implies q(x) = p(x)\cdot\exp(-(1+\lambda)).$$

Since $\int_{\mathcal{A}} q(x)dx = 1$, we can easily get $\exp(1+\lambda) = \int_{\mathcal{A}} p(x)dx$. As a result, the minimum point can be expressed by $q^*(x) = \frac{p(x)}{\int_{\mathcal{A}} p(x)dx}$. In the case when x is a discrete variable, we can easily get a similar solution, i.e., $p^*(x) = \frac{p(x)}{\sum_{x\in A} q(x)dx}$. □

Now we are ready to prove Theorem 4.1:

Proof. We first assume that $\int_{C(\tau)\leq b} q(\tau)d\tau$ is arbitrarily closed to 1 rather than directly equals 1 in our derivation, by which we avoid the occurrence of $p(\tau)=0$ in the calculation of the KL divergence. We denote $\int_{C(\tau)\leq b} q(\tau)d\tau$ as P_c. When $P_c \to 1$, the probability of constrain-violating trajectory is tend to 0. Thus,

$$\lim_{P_c\to 1}\int_{C(\tau)>b} q(\tau)\log\frac{q(\tau)}{p_{\pi_\beta}(\tau)}d\tau = \int_{C(\tau)>b}\lim_{q(\tau)\to 0} q(\tau)\log q(\tau) - q(\tau)\log p_{\pi_\beta}(\tau) = 0.$$

When $P_c \to 1$, the divergence of distribution is

$$\begin{aligned}
&\lim_{P_c\to 1} D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \\
&= \lim_{P_c\to 1}\left(\int_{C(\tau)>b} q(\tau)\log\frac{q(\tau)}{p_{\pi_\beta}(\tau)}d\tau + \int_{C(\tau)\leq b} q(\tau)\log\frac{q(\tau)}{p_{\pi_\beta}(\tau)}d\tau\right) \\
&= \int_{C(\tau)\leq b} q(\tau)\log\frac{q(\tau)}{p_{\pi_\beta}(\tau)}d\tau.
\end{aligned}$$

Then, we study the existence condition of the solution to problem (21)(22)(23). We first derive the trajectory distribution closest to the behavior distribution only under the safety constraint Eq. (22), i.e.,

$$\begin{aligned}
&\min_{q(\tau)} D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \\
&\quad = \int_{C(\tau)\leq b} q(\tau)\log\frac{q(\tau)}{p_{\pi_\beta}(\tau)}d\tau \\
&\quad \text{s.t.} \int_{C(\tau)\leq b} q(\tau)d\tau = 1
\end{aligned}$$

Replacing $p(x), q(x), \mathcal{A}$ with $p_{\pi_\beta}(\tau), q(\tau), \{\tau|C(\tau)\leq b\}$, respectively, we apply Lemma A.2 to get the optimal distribution $q^*(\tau) = \frac{p_{\pi_\beta}}{\int_{C(\tau)\leq b} p_{\pi_\beta} d\tau}$, which is the closest distribution to $p_{\pi_\beta}(\tau)$ under the safety constraint. There exists a distribution $q(\tau)$ satisfying both the safety constraint (22) and the in-distribution constraint (23) only when the divergence between $p_{\pi_\beta}(\tau)$ and the closest safe distribution $q^*(\tau)$ is less than ϵ:

$$\begin{aligned}
D_{KL}(q^*(\tau)||p_{\pi_\beta}(\tau)) &= \int_{C(\tau)\leq b} q(\tau)\log\frac{p_{\pi_\beta}(\tau)}{p_{\pi_\beta}(\tau)\int_{C(\tau)\leq b} p_{\pi_\beta} d\tau}d\tau \\
&= -\log\int_{C(\tau)\leq b} p_{\pi_\beta} d\tau \leq \epsilon,
\end{aligned}$$

which results in:

$$\int_{C(\tau)\leq b} p_{\pi_\beta} d\tau \geq e^{-\epsilon}. \tag{27}$$

Next, to solve the optimization problem (21)(22)(23), we construct the Lagrange function as:

$$\begin{aligned}
L(\tau, q, \dot{q}) =& \mathbb{E}_{\tau\sim q(\tau)}[R(\tau)] - \lambda\left(\int_{C(\tau)\leq b} q(\tau)d\tau - 1\right) - \mu\left(D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) - \epsilon\right) \\
=& \int_{\tau\sim q(\tau)}\left[R(\tau) - \mu\left(\log q(\tau) - \log p_{\pi_\beta}(\tau)\right)\right]d\tau - \lambda\int_{C(\tau)\leq b} q(\tau)d\tau + \lambda + \mu\epsilon.
\end{aligned}$$

If the inequality (27) holds strictly (i.e., the Slater's condition holds), we get $\frac{\partial L}{\partial q} = 0$ from Karush-Kuhn-Tucker (KKT) conditions, i.e.,

$$\frac{\partial L}{\partial q} = R(\tau) - \mu(\log q(\tau) + 1 - \log p_{\pi_\beta}(\tau)) - \lambda = 0. \tag{28}$$

Provided that $R(\tau)$ is not a constant function, we have $\mu \neq 0$ in Eq. (28). Solving Eq. (28) yields:

$$q^*(\tau) = \begin{cases} \frac{p_{\pi_\beta}(\tau)\exp(\alpha R(\tau))}{Z} & \text{if } C(\tau)\leq b; \\ 0 & \text{otherwise,} \end{cases} \tag{29}$$

where $Z = \exp(1 + \frac{\lambda}{\mu})$ and $\alpha = 1/\mu$. To exclude the Lagrangian multipliers λ and μ from Z and α, we introduce the KKT conditions:

$$\mu \left(D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) - \epsilon \right) = 0, \tag{30}$$

$$\int_{C(\tau)\leq b} q(\tau) d\tau = 1. \tag{31}$$

Solving Eq. (29)(30)(31), we finally obtain the relationship between a, Z and ϵ, b:

$$Z = \int_{C(\tau)\leq b} p_{\pi_\beta}(\tau) \exp(\alpha R(\tau)) d\tau,$$

$$\alpha \int_{C(\tau)\leq b} q^* R(\tau) - \log Z = \epsilon.$$

□

A.2. The Probabilistic Inference Perspective

The derivation of ELBO Recall that the objective of probabilistic inference is to minimize the discrepancy between the auxiliary trajectory distribution $q(\tau)$ and the optimal trajectory distribution of the behavior policy $p_{\pi_\beta}(\tau|O=1)$:

$$\begin{aligned}
& D_{KL}(q(\tau)||p_{\pi_\beta}(\tau|O=1)) \\
=& \mathbb{E}_{\tau\sim q}[\log q(\tau)] - \mathbb{E}_{\tau\sim q}[\log p_{\pi_\beta}(\tau|O=1)] \\
=& \mathbb{E}_{\tau\sim q}[\log q(\tau)] - \mathbb{E}_{\tau\sim q}[\log p_{\pi_\beta}(\tau, O=1)] + \mathbb{E}_{\tau\sim q}[\log p_{\pi_\beta}(O=1)] \\
=& \mathbb{E}_{\tau\sim q}[\log q(\tau)] - \mathbb{E}_{\tau\sim q}\left[\log[p_{\pi_\beta}(O=1|\tau) p_{\pi_\beta}(\tau)]\right] + \log p_{\pi_\beta}(O=1) \\
=& \mathbb{E}_{\tau\sim q}[\log q(\tau) - \log\left(p_{\pi_\beta}(\tau)\exp(a' R(\tau))/Z'\right)] + \log p_{\pi_\beta}(O=1) \\
=& -\left(-D_{KL}(q(\tau)||p_1(\tau))\right) + \log p_{\pi_\beta}(O=1) \\
=& -J(q(\tau)) + \log P_{\pi_\beta}(O=1),
\end{aligned} \tag{32}$$

where $J(q(\tau)) = -D_{KL}(q(\tau)||p_1(\tau))$, $p_1(\tau) = p_{\pi_\beta}(\tau)\exp(\alpha' R(\tau))/Z'$ and Z' is a constant normalizer. Since $\log p_{\pi_\beta}(O=1)$ is independent of $q(\tau)$, minimizing the discrepancy between $q(\tau)$ and $p_{\pi_\beta}(\tau|O=1)$ is equivalent to maximizing $J(q(\tau))$.

Trajectory optimization from the probabilistic inference perspective Recall the optimization problem derived from the probabilistic inference perspective:

$$\max_{q(\tau)} J(q(\tau)) = -D_{KL}(q(\tau)||p_1(\tau))$$

$$\Pi_b = \{q(\tau) : \int_{C(\tau)\leq b} q(\tau) d\tau = 1\},$$

where $p_1(\tau) \propto p_{\pi_\beta}(\tau)\exp(\alpha' R(\tau))$. By directly applying Lemma A.2, we replace $\mathcal{A}, p(x)$ with $\{\tau|C(\tau)\leq b\}, p_1(\tau)$ respectively and obtain

$$q^*(\tau) = \begin{cases} \frac{p_{\pi_\beta}(\tau)\exp(\alpha' R(\tau))}{Z} & \text{if } C(\tau)\leq b; \\ 0 & \text{otherwise,} \end{cases}$$

which has the same form as q^* in Eq. (29) except that α' is a preset hyper-parameter and does not change with budget b and divergence threshold ϵ.

Difference between α and α' In Theorem 4.1, α controls the degree of policy improvement over the behavior policy, and is determined by divergence threshold ϵ and budget b. However, in Eq. (33) which is derived from the probabilistic inference perspective, α' is a preset hyper-parameter independent of ϵ and b. This is because the in-distribution constraint does not appear in the probabilistic inference process. Without the in-distribution constraint, α', which determines the degree of optimality and the distribution's deviation from the behavior policy, can be arbitrarily controlled. Since in Theorem 4.1 the exact dependence of α w.r.t a certain budget is hard to derive analytically, we set α to a fixed value in pratice for different budgets, which can still yield good performances in various tasks, as shown in the empirical results.

A.3. Proof of Lemma 4.2 and Proposition 4.3

First, we denote $J_T(\pi)$ as the expected return of π under the environment with transition dynamics T. Then, we denote $\pi_\beta, \hat{T}, \hat{r}, \hat{c}$ as empirical behavior policy, transition dynamics, reward function and cost function, respectively, which are all estimated from the offline dataset. Let T, r, c be the transition dynamics, reward and cost in real environment. Let π and T_q be the learned policy and transition dynamics induced from the trajectory distribution $q(\tau)$ by:

$$\begin{aligned} \pi(a|s) &\propto \int q(\tau|\tau_{s_0} = s, \tau_{a_0} = a) d\tau \\ T_q(s'|s,a) &\propto \int q(\tau|\tau_{s_0} = s, \tau_{a_0} = a, \tau_{s_1} = s') d\tau. \end{aligned} \tag{33}$$

For the sake of simplicity in theoretical derivations, we will ignore the sampling error of the reward function and only consider the sampling error of the cost function, which is critical in the constrained problem. However, the sampling error of the reward function can also be easily incorporated into the theoretical results. Besides, we also ignore the difference in initial state distributions $p(s_0)$ across different MDPs and assume the empirical cost function $\hat{c}$ can be accurately modeled from offline dataset.

Lemma A.3 (4.2 in the main paper)**.** *Let π be a policy derived from $q(\tau)$. Denote $J_{T_q}(\pi) = \mathbb{E}_{\tau\sim q(\tau)}[R(\tau)]$ as the expected return of $q(\tau)$ and $J_T(\pi)$ as the expected return of π under the real environment T. Let $C(\tau)$ and $\hat{C}(\tau)$ be the episodic cost of trajectory τ under the T and empirical dynamics $\hat{T}$, respectively. With probability at least $1-\delta$, the gap between $J_{T_q}(\pi)$ and $J_T(\pi)$ is bounded by:*

$$|J_{T_q}(\pi) - J_T(\pi)| \leq \frac{2R_m}{1-\gamma}\left[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L} \frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right],$$

where the first term $(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}}$ vanishes in deterministic environment, and for each feasible trajectory τ,

$$|C(\tau) - \hat{C}(\tau)| \leq \sum_{t=0}^{L} \frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}},$$

where $C_{T,\delta}, C_{c,\delta}$ are constants depending on the concentration properties of $T(s'|s,a), c(s,a)$, respectively, and $\delta \in (0,1)$. $N(s,a)$ is the counts for each state-action pair (s,a) in the offline dataset.

Before presenting the proof of Lemma A.3, we first introduce the assumption of concentration properties from (Auer et al., 2008; Kumar et al., 2020) and a modified lemma from (Janner et al., 2019). We introduce the assumption of concentration properties for the transition dynamics, and cost function:

Assumption A.4. $\forall s, a \in \mathcal{D}$, the following relationships hold with probability at least $1-\delta$:

$$\begin{aligned} \frac{1}{2}\|T(s'|s,a) - \hat{T}(s'|s,a)\|_1 &\leq \frac{C_{T,\delta}}{\sqrt{N(s,a)}}, \\ |c(s,a) - \hat{c}(s,a)| &\leq \frac{C_{c,\delta}}{\sqrt{N(s,a)}}, \end{aligned} \tag{34}$$

where $C_{T,\delta}, C_{c,\delta}$ are constants dependent on T, c respectively and $\delta \in (0,1)$.

Then we provide a lemma modified from Lemma B.3 in (Janner et al., 2019).

Lemma A.5. *The return gap between π_1 in environment T_1 and π_2 in environment T_2 can be bounded as:*

$$\begin{aligned} &|J_{T_1}(\pi) - J_{T_2}(\pi)| \\ =&\left|\mathbb{E}_{\pi,T_1}[\sum_t \gamma^t r(s,a)] - \mathbb{E}_{\pi,T_2}[\sum_t \gamma^t r(s,a)]\right| \\ \leq& R_m \sum_t \sum_{s,a} \gamma^t |T_1^t(s,a) - T_2^t(s,a)| \\ \leq& 2R_m \sum_t \gamma^t \sum_{t'=0}^{t} \left(\mathbb{E}_{s,a\sim p_1^{t'}(s,a)}\left[D_{TV}(T_1(\cdot|s,a)||T_2(\cdot|s,a))\right] + \mathbb{E}_{s\sim p_1^{t'}(s)}\left[D_{TV}(\pi_1(\cdot|s)||\pi_2(\cdot|s))\right]\right) \end{aligned} \tag{35}$$

where $p_1^{t'}$ and $p_2^{t'}$ are state/state-action visitation distributions w.r.t T_1 and T_2, respectively.

Lemma A.6. *Suppose π_1 and T_1 are induced by trajectory distribution $p_1(\tau)$ and π_2 and T_2 are induced by trajectory distribution $p_2(\tau)$, if $D_{KL}(p_1(\tau)||p_2(\tau)) \leq \epsilon$, we can obtain:*

$$\begin{aligned} \sum_t^L \left[\mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{TV}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right]\right] &\leq \sqrt{\frac{L\epsilon}{2}} \\ \sum_t^L \left[\mathbb{E}_{s\sim p_1^t(s)}\left[D_{TV}(\pi_1(\cdot|s)||\pi_2(\cdot|s))\right]\right] &\leq \sqrt{\frac{L\epsilon}{2}} \end{aligned} \tag{36}$$

where L is the maximum length of trajectory.

Proof.

$$\begin{aligned} \epsilon &\geq D_{KL}(p_1(\tau)||p_2(\tau)) \\ &= \sum_\tau p(\tau)\log\frac{p_1(s_0)\prod_{t=0}^L \pi_1(a_t|s_t)p_1(s_{t+1}|s_t,a_t)}{p_2(s_0)\prod_{t=0}^L \pi_2(a_t|s_t)p_2(s_{t+1}|s_t,a_t)} \\ &\geq \sum_t^L \mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{KL}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right] + \sum_t^L \mathbb{E}_{s\sim p_1^t(s)}\left[D_{KL}(\pi_1(\cdot|s)||\pi_2(\cdot|s))\right] \\ &\geq \sum_t^L \mathbb{E}_{s,a\sim p_1^t(s,a)}\left[2D_{TV}^2(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right] + \sum_t^L \mathbb{E}_{s\sim p_1^t(s)}\left[2D_{TV}^2(\pi_1(\cdot|s)||\pi_2(\cdot|s))\right] \\ &\geq 2\sum_t^L \left[\mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{TV}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right]\right]^2 + 2\sum_t^L \left[\mathbb{E}_{s\sim p_1^t(s)}\left[D_{TV}(\pi_1(\cdot|s)||\pi_2(\cdot|s))\right]\right]^2 \end{aligned} \tag{37}$$

Thus,

$$\begin{aligned} \epsilon &\geq 2\sum_t^L \left[\mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{TV}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right]\right]^2 \\ &\geq 2L\frac{1}{L}\sum_t^L \left[\mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{TV}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right]\right]^2 \\ &\geq 2L\left[\frac{1}{L}\sum_t^L \mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{TV}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right]\right]^2 \end{aligned} \tag{38}$$

Then, we have $\sum_t^L \left[\mathbb{E}_{s,a\sim p_1^t(s,a)}\left[D_{TV}(p_1(\cdot|s,a)||p_2(\cdot|s,a))\right]\right] \leq \sqrt{\frac{L\epsilon}{2}}$. Similarly, we can obtain $\sum_t^L \left[\mathbb{E}_{s\sim p_1^t(s)}\left[D_{TV}(\pi_1(\cdot|s)||\pi_2(\cdot|s))\right]\right] \leq \sqrt{\frac{L\epsilon}{2}}$. □

Now we are ready to present the proof of Lemma A.3:

Proof. First, we consider the return gap of π between learned dynamics T_q and empirical dynamics $\hat{T}$. As discussed in Section 4.1, the T_q and $\hat{T}$ is consistent in deterministic dynamics, which means

$$|J_{T_q}(\pi) - J_{\hat{T}}(\pi)| = 0. \tag{39}$$

In the case of probabilistic dynamics, since we constraint the trajectory divergence between empirical behavior and target

policy ($D_{KL}(q(\tau)||p_{\pi_\beta}(\tau)) \leq \epsilon$), we apply A.6 and A.5 and get

$$
\begin{aligned}
&|J_{T_q}(\pi) - J_{\hat{T}}(\pi)| \\
=&\left|\mathbb{E}_{\pi,T_q}[\sum_{t=0}^{L}\gamma^t \hat{r}(s,a)] - \mathbb{E}_{\pi,\hat{T}}[\sum_{t=0}^{L}\gamma^t \hat{r}(s,a)]\right| \\
\leq& 2R_m \sum_{t=0}^{L}\gamma^t \sum_{t'=0}^{t}\left(\mathbb{E}_{s,a\sim p_{\hat{T}}^{t'}(s,a)}\left[D_{TV}(T_q(\cdot|s,a)||\hat{T}(\cdot|s,a))\right]\right) \qquad (40)\\
\leq& 2R_m \sum_{t=0}^{L}\gamma^t \sqrt{\frac{L\epsilon}{2}} = \frac{2R_m(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}}. \qquad (41)
\end{aligned}
$$

Then, we consider the return gap of π between $\hat{T}$ and T:

$$
\begin{aligned}
&|J_T(\pi) - J_{\hat{T}}(\pi)| \\
=&\left|\mathbb{E}_{\pi,T}[\sum_{t=0}^{L}\gamma^t r(s,a)] - \mathbb{E}_{\pi,\hat{T}}[\sum_{t=0}^{L}\gamma^t r(s,a)]\right| \qquad (42)\\
\leq& 2R_m \sum_{t=0}^{L}\gamma^t \sum_{t'=0}^{t}\left(\mathbb{E}_{s,a\sim p_{\hat{T}}^{t'}(s,a)}\left[D_{TV}(T(\cdot|s,a)||\hat{T}(\cdot|s,a))\right]\right) \qquad (43)\\
=&\frac{2R_m}{1-\gamma}\sum_{t'}^{L}\gamma^{t'}\mathbb{E}_{s,a\sim p_{\hat{T}}^{t'}(s,a)}\left[D_{TV}(T(\cdot|s,a)||\hat{T}(\cdot|s,a))\right] \qquad (44)\\
\leq& \frac{2R_m}{1-\gamma}\mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]. \qquad (45)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
&|J_{T_q}(\pi) - J_T(\pi)| \qquad (46)\\
\leq& |J_{T_q}(\pi) - J_{\hat{T}}(\pi)| + |J_{\hat{T}}(\pi) - J_T(\pi)| \qquad (47)\\
\leq& \frac{2R_m}{1-\gamma}\left[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right].
\end{aligned}
$$

where the first term vanishes in deterministic dynamics.

Then, we further prove $|J_T(\pi) - J_{\hat{T}}(\pi)|$ can be bound by $\frac{2R_m}{1-\gamma}\left[\frac{2(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi_\beta,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right]$. After denoting the uncertainty reward $u(s,a) = D_{TV}(T(\cdot|s,a)||\hat{T}(\cdot|s,a))$ which is less than 1, the term $\sum_{t'}^{L}\gamma^{t'}\mathbb{E}_{s,a\sim p_{\hat{T}}^{t'}(s,a)}\left[D_{TV}(T(\cdot|s,a)||\hat{T}(\cdot|s,a))\right]$ can be seen as the cumulative uncertainty reward of π and $\hat{T}$, and we denote it as $U_{\hat{T}}(\pi)$ for brevity. Therefore, we can bound the uncertainty return gap between behavior policy π_β and learned policy π:

$$
\begin{aligned}
&|U_{\hat{T}}(\pi) - U_{\hat{T}}(\pi_\beta)| \\
=&\left|\mathbb{E}_{\pi,\hat{T}}[\sum_{t=0}^{L}\gamma^t u(s_t,a_t)] - \mathbb{E}_{\pi_\beta,\hat{T}}[\sum_{t=0}^{L}\gamma^t u(s_t,a_t)]\right| \\
\leq& 2\sum_{t=0}^{L}\gamma^t \sum_{t'=0}^{t}\left(\mathbb{E}_{s\sim p_1^t(s)}\left[D_{TV}(\pi(\cdot|s)||\pi_\beta(\cdot|s))\right]\right) \qquad (48)\\
\leq& 2\sum_{t=0}^{L}\gamma^t \sqrt{\frac{L\epsilon}{2}} = \frac{2(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}}. \qquad (49)
\end{aligned}
$$

After upper bounding $U_{\hat{T}}(\pi)$ by $\frac{2(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}} + U_{\hat{T}}(\pi_\beta)$, we can obtain the return gap of π between $\hat{T}$ and T.

$$
\begin{aligned}
&|J_{\hat{T}}(\pi) - J_T(\pi)| && (50)\\
\leq &\frac{2R_m}{1-\gamma} U_{\hat{T}}(\pi) && (51)\\
\leq &\frac{2R_m}{1-\gamma}\left[\frac{2(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}} + U_{\hat{T}}(\pi_\beta)\right]. && (52)
\end{aligned}
$$

Then, there is:

$$
\begin{aligned}
&|J_{T_q}(\pi) - J_T(\pi)| && (53)\\
\leq &|J_{T_q}(\pi) - J_{\hat{T}}(\pi)| + |J_{\hat{T}}(\pi) - J_T(\pi)| && (54)\\
\leq &\frac{2R_m(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}} + \frac{2R_m}{1-\gamma}\left[\frac{2(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi_\beta,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right].
\end{aligned}
$$

where the term $\frac{2R_m(1-\gamma^{L+1})}{1-\gamma}\sqrt{\frac{L\epsilon}{2}}$ vanishes in deterministic dynamics. For the error bound of the cost, we note that the probability of all constraint-violating trajectories ($\hat{C}(\tau) > b$) is zero for π on both $\hat{T}$ and T_q, since the empirical cost can be accurately modeled even in T_q under our assumption. Therefore, the error only comes from the cost function divergence between the empirical and real environments. For each trajectory τ, we utilize the concentration properties to obtain that

$$
|C(\tau) - \hat{C}(\tau)| \leq \sum_t \gamma^t |c(\tau_{s_t}, \tau_{a_t}) - \hat{c}(\tau_{s_t}, \tau_{a_t})| \leq \sum_{t=0}^{L}\frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}}.
$$

Finally, we prove the proposition A.3. □

Proposition A.7 (4.3 in the main paper)**.** *Denoting $\pi_{q_b^*}$ as the policy induced by the constrained optimal trajectory distribution q_b^*, for any policy π derived from the trajectory distribution that simultaneously satisfies the in-distribution constraint* (6) *and safety constraint* (5)*, the following inequality holds with probability at least $1-\delta$:*

$$
J_T(\pi_{q_b^*}) \geq J_T(\pi) - \frac{4R_m}{1-\gamma}\left[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right] \tag{55}
$$

where the first term $(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}}$ vanishes in deterministic environment. And for each trajectory τ generated from q_b^*, the episodic cost of τ is bounded with probability at least $1-\delta$ as follow:

$$
C(\tau) \leq b + \sum_{t=0}^{L}\frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}}. \tag{56}
$$

Proof. Let $\sigma_1 = \frac{2R_m}{1-\gamma}\left[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right]$. Recall that q_b^* is the optimal distribution, while $\pi_{q_b^*}$ enjoys the theoretical property of q_b^*. Hence, for any π derived from the trajectory distribution that satisfies both the in-distribution constraint and safety constraint, we have $J_{T_q}(\pi_{q_b^*}) \geq J_{T_q}(\pi)$. Then we utilize the Lemma A.3 to obtain

$$
\begin{aligned}
J_T(\pi_{q_b^*}) \geq &J_{T_q}(\pi_{q_b^*}) - \sigma_1 \geq J_{T_q}(\pi) - \sigma_1 \geq J_T(\pi) - 2\sigma_1\\
\geq &J_T(\pi) - \frac{4R_m}{1-\gamma}\left[(1-\gamma^{L+1})\sqrt{\frac{L\epsilon}{2}} + \mathbb{E}_{\pi,\hat{T}}\left[\sum_{t=0}^{L}\frac{\gamma^t C_{T,\delta}}{\sqrt{N(s_t,a_t)}}\right]\right].
\end{aligned}
$$

Besides, $\hat{C}(\tau) \leq b$ is true for any trajectory τ generated from q_b^*, thus, by using the Lemma A.3 we can obtain

$$
C(\tau) \leq \hat{C}(\tau) + \sum_{t=0}^{L}\frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}} \leq b + \sum_{t=0}^{L}\frac{\gamma^t C_{c,\delta}}{\sqrt{N(\tau_{s_t}, \tau_{a_t})}}.
$$

□

B. Details of Simulation Environments and Datasets

Pendulum and Reacher In the Pendulum swing-up task, the agent tries to keep the pendulum upright and balanced under the constraint of keeping away from unsafe angles. In the Reacher task, the robotic arm needs to move towards the goal while avoiding the unsafe region. The implementation of the Pendulum swing-up task follows Sauté (Sootla et al., 2022) without additional modification. The single Pendulum swing-up is taken from the classic control library in the Open AI Gym (Brockman et al., 2016). The cost function is defined as:

$$c = \begin{cases} 1 - \frac{|\theta - \delta|}{50} & \text{if } -25 \leq \theta \leq 75, \\ 0 & \text{otherwise}, \end{cases}$$

where θ is the angle of the pole deviation from the upright position and $\delta = 25$. Such a cost function is designed to create a trade-off between keeping upright and staying away from the unsafe degree i.e., δ. The cost function of Reacher is designed as:

$$c = \begin{cases} 100 - 200\|x - x_{\text{unsafe}}\|_2 & \text{if } \|x - x_{\text{unsafe}}\|_2 \leq 0.5 \\ 0 & \text{otherwise}, \end{cases}$$

where x_{unsafe} is set to $(0.5, 0.5, 0)$, following the implementation of Sauté. Refer to (Sootla et al., 2022) for more details. To collect the offline dataset, we train an unconstrained SAC policy online until convergence and take its replay buffer as the offline dataset, which contains $2e4$ samples.

MuJoCo Environments In MuJoCo tasks (Todorov et al., 2012), following the constraint setting of previous works (Zhang et al., 2020; Yang et al., 2022), the goal of the agent is to control the robot to keep it balanced and go forward with the cost of the moving speed. We use the MuJoCo environments provided by OpenAI Gym (Brockman et al., 2016) and refer to FOCOPS (Zhang et al., 2020) to design the cost function: $c = |v|$, where v is the agent's velocity. We utilize three types of offline datasets (medium, medium-replay and medium version) from D4RL (Fu et al., 2020) for each environment.

Bullet-Safety-Gym Environments Bullet-Safety-Gym (Gronauer, 2022) contains a set of safe RL tasks that are widely used to assess safe RL algorithms. In the SafetyCarCircle task, the goal of a car agent is to move on a circle while remaining in the safety zone. The cost in this task is defined as $c = \mathbb{I}(|x| > x_{\text{lim}})$. In the SafetyBallReach task, a ball agent is controlled to move towards a goal while avoiding collisions with obstacles. To collect the offline dataset, we train an unconstrained SAC policy online for $1e6$ steps (one training step for one data point) and take its replay buffer as the offline dataset, which contains $1e6$ samples.

Relationship between reward and cost As shown by our experimental results, the reward drops when enforcing stricter constraints. This is because there exists various degrees of conflict between the task objective and the safety constraint in the above environments (e.g., in MuJoCo tasks, the cost function is also an important component of reward function). As a result, the optimal solution would probably be located at some edge area of the feasible policy space. When the constraint is set stricter, the edge of feasible space will be pushed inward and the reward of the optimal policy is likely to shrink. In other words, these tasks have a non-empty tempting policy class which lies in outside of the feasible policy space and has higher reward than the optimal policy, as formally defined as Temptation in the mentioned work (Liu et al., 2022b).

C. Real-world Advertising Bidding Scenario

Problem Formulation In the real-world oCPM advertising bidding scenario, advertiser s provides the target ROI Roi_s and the maximum total pay $P_{\max}$ at the beginning of each day. At time step t, the advertising system bids $pay_{s,t}$ for advertiser s. If this bid is successful, the advertiser expends $pay_{s,t}$ and obtains revenue $gmv_{s,t}$, i.e., the gross merchandise value, from the displayed ads. In this process, the system has to ensure that the average conversion is not less than the target Roi_s, i.e.,

$$\frac{\sum_{t=0}^{T} gmv_{s,t}}{P_{\max}} > Roi_s \iff \sum_{t=0}^{T} -gmv_{s,t} < -Roi_s \cdot P_{\max}. \tag{57}$$

The optimization problem in this scenario can be described as:

$$\begin{aligned} &\max \sum_{t=0}^{T} pay_{s,t} \\ \text{s.t.} \ &\sum_{t=0}^{T} -gmv_{s,t} < -Roi_s \cdot P_{\max}, \forall s \end{aligned} \tag{58}$$

We formulate this problem as a Markov Decision Process (MDP).

- A trajectory contains the record of one advertiser's successful bidding within one day.
- The state contains the user context features, ads context features, cumulative revenue, cumulative pay, maximum total pay, target ROI, predicted click-through rate $pCTR$, predicted conversion rate $pCVR$ and predicted gross merchandise volume $pGMV$ of one advertiser.
- The action is to output a weight $k \in [0.5, 1.5]$, which controls the bidding by formula $bid = k \cdot \frac{pCTR \cdot pCVR \cdot pGMV}{Roi_s}$.
- The reward is $pay_{s,t}$ for a successful bid and 0 otherwise.
- The cost is $-gmv_{s,t}$ for a successful bid and 0 otherwise.
- The constraint budget is $-Roi_s \cdot P_{\max}$.

Deployment Details In this task, the cost and the constraint budget are negative, which means that the constraint is violated at the starting stage of bidding and can only be satisfied over time. Although Sauté assumes that the cost is positive, it still works in this scenario. We modify Sauté by augmenting the state with the value $Roi_s \cdot P_{\max} - \sum_{t=0}^{T} gmv_{s,t}$ and adding an extra high reward to the state with $Roi_s \cdot P_{\max} - \sum_{t=0}^{T} gmv_{s,t} < 0$. In this way, the agent is encouraged to reach the target cumulative gmv and maximize the total pay. Besides, during evaluation, each advertiser switches among policies trained by different algorithms, in order to eliminate the differences between advertisers and make the online experiment fairer.

About the Data-collecting Policy The dataset used in our experiment is collected by a set of constrained policies jointly. The policy set includes BCORLE(λ) (Zhang et al., 2021) (60% advertisers), PID (Stooke et al., 2020) (20% advertisers) and CEM (De Boer et al., 2005) (20% advertisers).

D. Implementation Details

Implementation Details of TREBI The overall implementation of TREBI is based on Diffuser (Janner et al., 2022) and can be found at `https://github.com/qianlin04/Safe-offline-RL-with-diffusion-model`. The inference of TREBI requires three components that should be obtained during the training stage. **The first component** is a U-Net style neural network, which mainly consists of one-dimensional temporal convolutions. It models the mean $\mu_\theta(\tau^i, i)$ of Gaussian distribution $p_\theta = \mathcal{N}(\tau^{i-1}; \mu_\theta(\tau^i, i), \Sigma)$ in the reverse process. The input of this network is the trajectory τ^{i-1} obtained in the previous denoising steps, which can be seen as a special "image" where 1) the length of the trajectory is analogous to the size of the image and 2) the dimension of the concatenation of the state and the action is analogous to the channel of the image. **The second and third components** are respectively the reward and the cost value estimation, i.e., $R(\tau)$ and $C(\tau)$, which form the objective function $h_{b,n}(\tau)$ in Eq. (13). They are modeled by neural networks that take τ^{i-1} as the input and output the value estimation. These two components also contain U-Net style networks similar to the first component. The loss of the reward/cost value estimation is the MSE loss between the prediction of a trajectory sampled from the dataset and the cumulative reward/cost of this trajectory. In other words, the value estimation is learned by using the Monte Carlo method. In addition, during planning multiple trajectories rather than one single trajectory are generated randomly in the reverse process, and the one with the highest cumulative reward among constraint-satisfying trajectories is selected to generate the action for real-environment execution.

Table 2. Maximum budgets for the used datasets.

Dataset	Max budget
Pendulum	60
Reacher	14
HalfCheetah-medium-v2	414
HalfCheetah-medium-replay-v2	384
HalfCheetah-medium-expert-v2	731
Hopper-medium-v2	130
Hopper-medium-replay-v2	140
Hopper-medium-expert-v2	169
Walker2d-medium-v2	160
Walker2d-medium-replay-v2	160
Walker2d-medium-expert-v2	234
SafetyCarCircle-v0	150
SafetyBallReach-v0	35

Implementation Details of Baselines As for the baselines, we implement BCQ-Lagrangian, BCORLE(λ) and BCQ-Sauté by introducing the Lagrange multiplier and the two kinds of state augmentations respectively on the open-source BCQ implementation. Since there is currently no publicly available CPQ implementation, we reproduce CPQ and follow the hyper-parameters setting mentioned in paper (Xu et al., 2022a), unless the learning rate of α is set to 0.01 and the latent space threshold is set to the fixed percentile 0.75 of the latent KL loss of the whole batch dataset. For BCORLE(λ), the maximum lambda is set to 20 for all tasks. For BCQ-Sauté, the maximum of initial budget is set to the maximum budget w.r.t. for each task and the penalty for the states violating the constraint is set to 1000. For all actor-critic methods, the learning rate is $1e-5$ for the actor and $1e-3$ for the critic. For all the BCQ-based methods, we fine-tune the perturbation range Φ among $\{0.15, 0.10, 0.05, 0.015\}$.

Hyper-parameter Settings In all simulation environments, we evaluate each algorithm on 60 different trajectories for each possible budget. All algorithms are trained for $1e6$ iterations and the results are obtained on 3 different random seeds. The time interval of regenerating the trajectory for decision making (i.e., control frequency) is set to 1 for all tasks. The cost discount factor is set to $\gamma = 1$ for Pendulum, Reacher and two Bullet-Safety-Gym task, and $\gamma = 0.99$ for all MuJoCo tasks. The max episode length is set to 200 for Pendulum, 50 for Reacher, 1000 for all MuJoCo tasks, 500 for SafetyCarCircle-v0 and 250 for SafetyBallReach-v0. For Pendulum, Reacher and two Bullet-Safety-Gym tasks, the maximum budget is set to the expected episodic cost of the unconstrained behavior policy. For MuJoCo tasks, the maximum budget is set to the expected episodic cost induced by Diffuser's unconstrained planning. We provide the maximum budgets for all environments in Table 2. The hyper-parameter n in Eq. (16) is set to 1000 for Pendulum, Reacher, the Ads bidding task and 100 for all MuJoCo tasks and Bullet-Safety-Gym tasks. The length of trajectory is set to 128 for Pendulum and the Ads bidding task, and 32 for Reacher, all the MuJoCo tasks and Bullet-Safety-Gym tasks. The hyper-parameter α in Eq. (16) is set to 0.1 for all tasks.

E. Additional Results

Results on Hopper and Walker2d We present the performance of TREBI on Hopper and Walker2d datasets in Figure 5 and 6 respectively. For all MuJoCo tasks with a budget ratio of $1.0, 0.8, 0.6$, TREBI achieves nearly perfect constraint satisfaction in the trajectory level. In some tasks with a low budget ratio of $0.4, 0.2$, none of the evaluated approaches can achieve constraint satisfaction even in the sense of expectation. One possible cause can be that the datasets of these tasks lack the trajectory samples that satisfy such extremely low budget constraints, hence satisfying the in-distribution constraint makes it almost impossible to learn safe policies for these low budgets. Besides, TREBI demonstrates better low-budget constraint satisfaction on medium-replay datasets rather than medium and medium-expert datasets. Note that the overall returns of the three types of datasets satisfies: medium-replay $<$ medium $<$ medium-expert. While the reward and cost in MuJoCo tasks (and actually in many real-world tasks) have an approximately positive correlation, there are less low-cost trajectories in the medium and medium-expert datasets than in the medium-replay datasets, leading to a difficulty to learn safe policies for low-budget constraints. Therefore, these empirical results agree with the condition of the dataset's quality in Theorem 4.1.

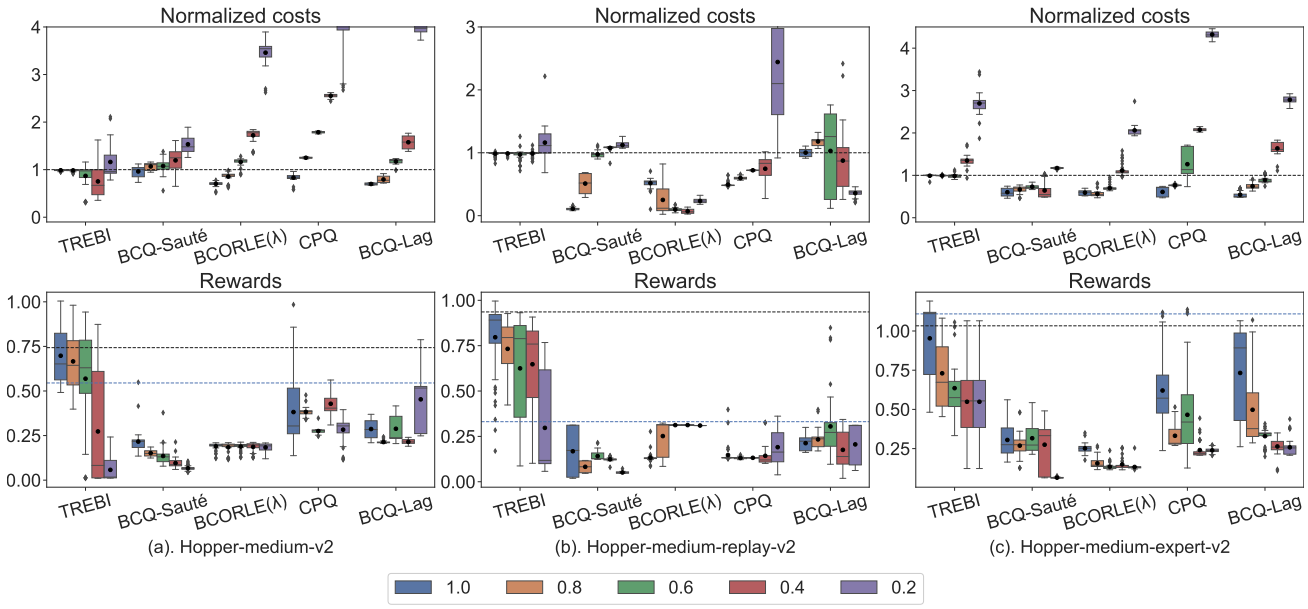


Figure 5. Results on Hopper tasks.

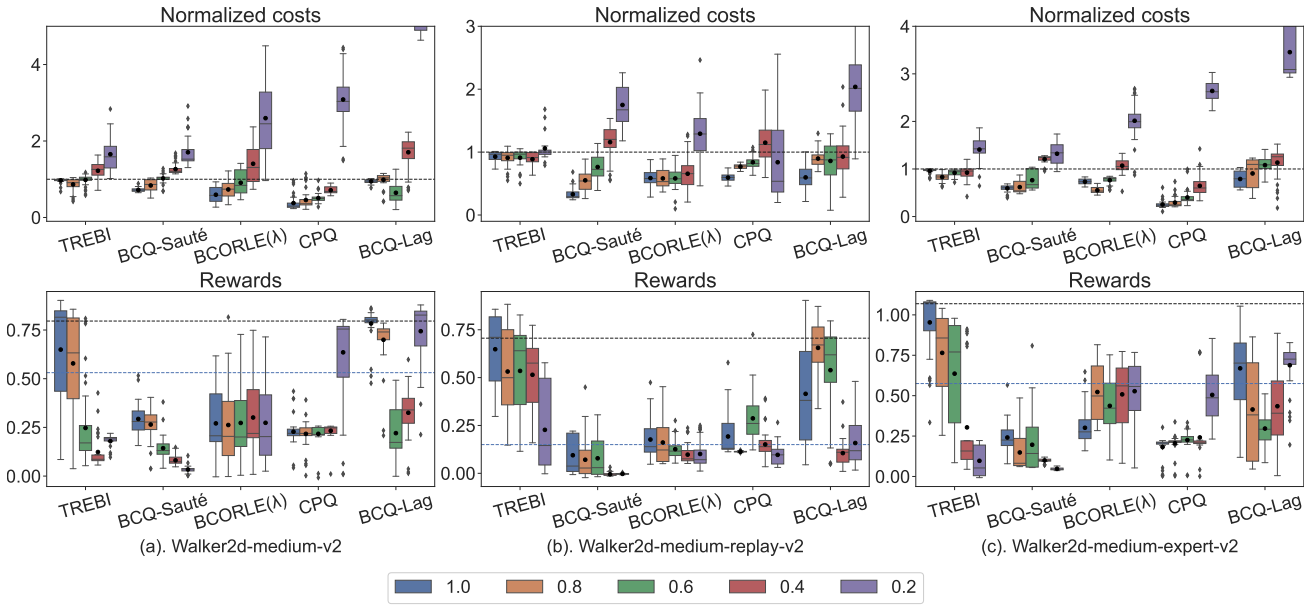


Figure 6. Results on Walker2d tasks.

Control Frequency In our experiments, the planning process is performed at each step to ensure the best performance. Here in Figure 7, we provide an empirical analysis of the control frequency (i.e., the number of step intervals for replanning) in halfcheetah-medium-replay. We can see that decreasing the control frequency from per-1-step to per-4-step leads to slight reward declines and insignificant constraint violations in most budget settings. Thus it is possible to reduce the overall computational costs by decreasing the control frequency without too much performance degradation and constraint violation.

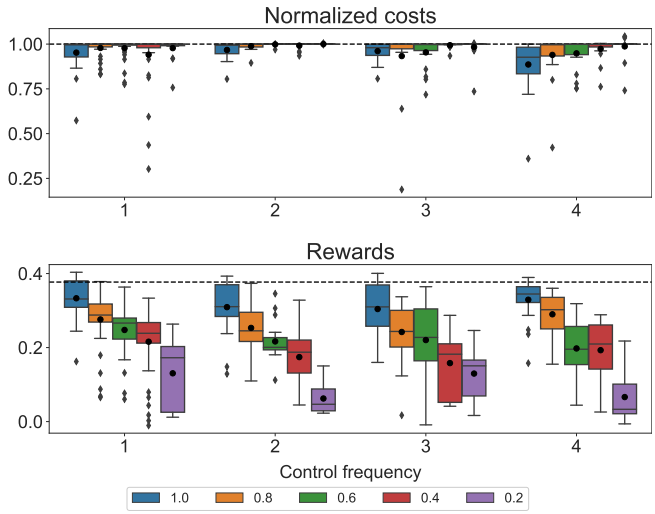
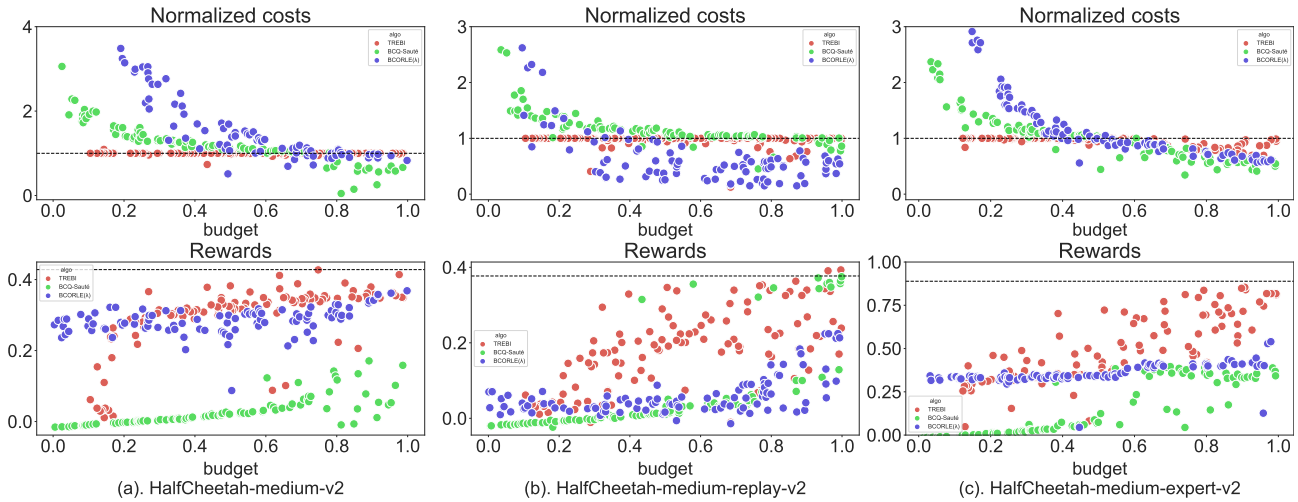


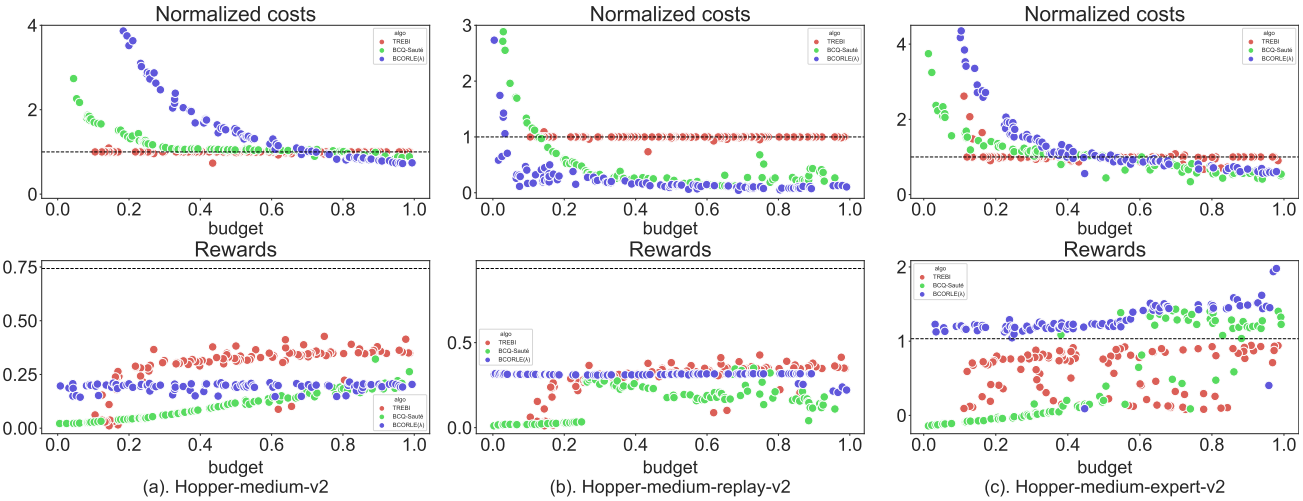
Figure 7. Results on HalfCheetah-medium-replay tasks with different constrol frequency.

Budgets ratio sampled from $[0, 1]$ To further validate the effectiveness of TREBI under randomly selected budgets, we do additional experiments in MuJoCo tasks and Bullet-Safety-Gym tasks where the ratios to maximum budgets are uniformly sampled from interval $[0, 1]$. For each algorithm, we sample 100 safety budgets and demonstrate the performances under these budgets. Note that the fixed-budget baselines BCQ-Lagrangian and CPQ are not evaluated under this setting since they have to retrain for each new budget, which leads to an enormous amount of computation. The results in Figures 8-11 demonstrate that TREBI performs consistently well across different levels of budgets in various environments.



(a). HalfCheetah-medium-v2 (b). HalfCheetah-medium-replay-v2 (c). HalfCheetah-medium-expert-v2

Figure 8. Results on HalfCheetah tasks with random budgets sampled from $[0, 1]$. The x-axis of a sample represents the given budget, which is set to x times the maximum budget, and the y-axis represents the cost and reward under the given budget.



(a). Hopper-medium-v2 (b). Hopper-medium-replay-v2 (c). Hopper-medium-expert-v2

Figure 9. Results on Hopper tasks with random budgets sampled from $[0, 1]$.

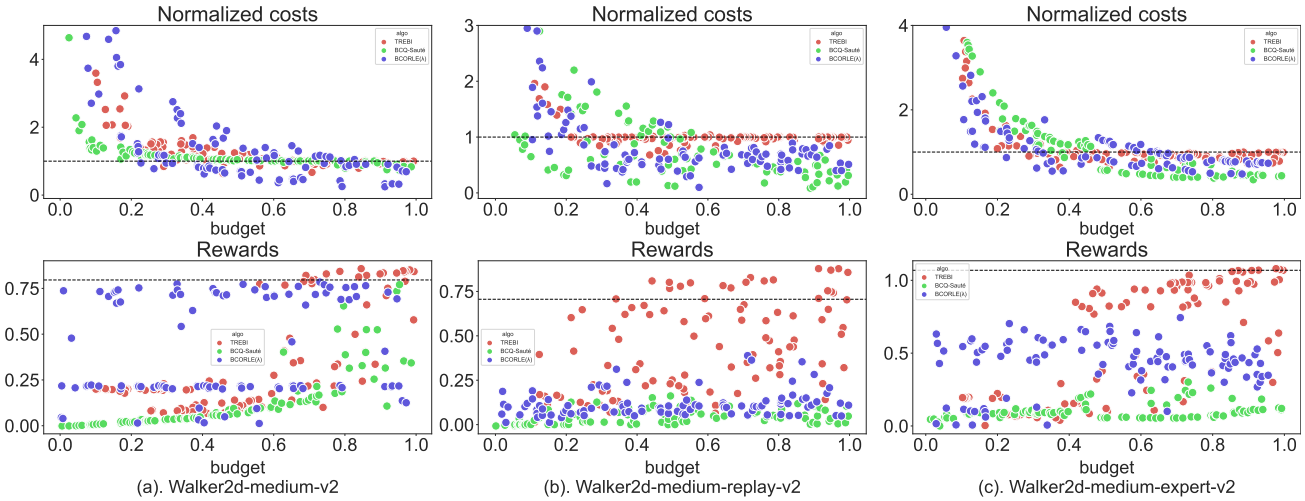


Figure 10. Results on Walker2d tasks with random budgets sampled from $[0, 1]$.

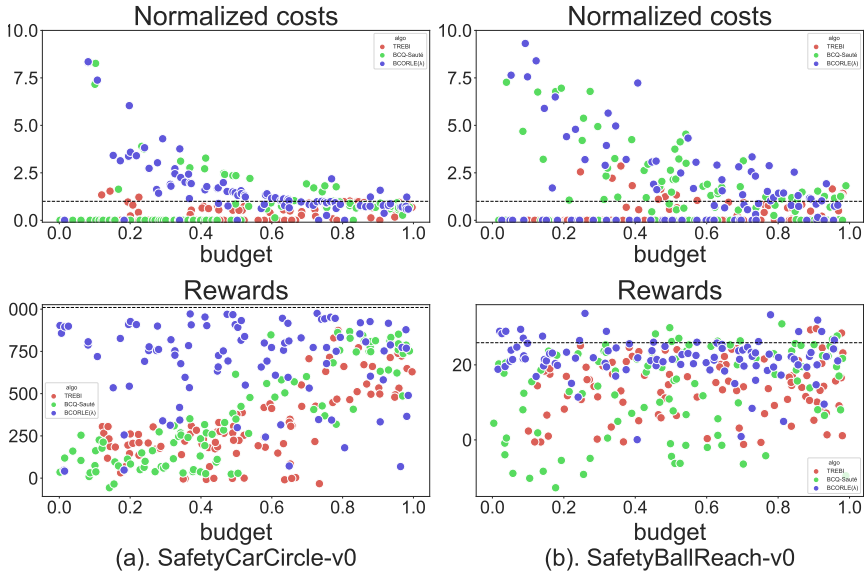


Figure 11. Results on Bullet-Safety-Gym tasks with random budgets sampled from $[0, 1]$.