# Factored DRO: Factored Distributionally Robust Policies for Contextual Bandits

**Tong Mu**
Stanford university
tongm@cs.stanford.edu

**Yash Chandak**
University of Massachusetts
ychandak@cs.umass.edu

**Tatsunori Hashimoto**
Stanford University
thashim@stanford.edu

**Emma Brunskill**
Stanford University
ebrun@cs.stanford.edu

## Abstract

While there has been extensive work on learning from offline data for contextual multi-armed bandit settings, existing methods typically assume there is no environment shift: that the learned policy will operate in the same environmental process as that of data collection. However, this assumption may limit the use of these methods for many practical situations where there may be distribution shifts. In this work we propose Factored Distributionally Robust Optimization (Factored-DRO)[1], which is able to separately handle distribution shifts in the context distribution and shifts in the reward generating process. Prior work that either ignores potential shifts in the context, or considers them jointly, can lead to performance that is too conservative, especially under certain forms of reward feedback. Our Factored-DRO objective mitigates this by considering the shifts separately, and our proposed estimators are consistent and converge asymptotically. We also introduce a practical algorithm and demonstrate promising empirical results in environments based on real-world datasets, such as voting outcomes and scene classification.

## 1 Introduction

In many real-world applications we often have access to historical data about past decisions and their associated outcomes. This data can be leveraged for *off-policy learning* to optimize for a decision making rule, called a policy, that is more effective for the given application than that used for data collection. Off-policy learning from historical data for contextual multi-armed bandit settings has been well studied, due in part to the myriad of potential applications; from personalized recommendation [Strehl et al., 2010] to optimizing medical treatment [Kim et al., 2011]. Prior methods often focus on the stationary setting, where the learned policy will be deployed in a setting with the same underlying data generation process that produces the observed contexts, and rewards upon intervention [Precup, 2000, Liu et al., 2018, Kallus and Uehara, 2020a,b, Liu et al., 2020]. However, in many settings, this assumption is not true and the distributions may shift.

As a motivating example, consider the problem of increasing political participation during elections using various interventions to encourage voting. In the past, different strategies have been implemented, ranging from reminding people of their own voting history, to reminding them of the voting history of their neighborhood, etc. [Gerber et al., 2008]. Leveraging this historical data, an important question is: *what strategy should be used during the next election to maximize political participation*? Addressing this question requires careful consideration of two potential sources of

[1] Code: https://github.com/StanfordAI4HI/FactoredDRO

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

distribution shift: **(a)** How much has the population demographic in the area changed since the past data was collected? (e.g. the percentage of single person households may have changed due to change and developments in the city.) **(b)** How much has the effectiveness of each strategy on each demographic group changed? (e.g. as technology develops, the effectiveness of mail-based interventions may change.) Such challenges are ubiquitous in real-world problems. For example, in automated healthcare, data from patients of one locality may be used to develop treatments for other localities where the effectiveness of treatments may differ [Finlayson et al., 2021]. Similarly, recommendation systems need to account for both the changing interests and the demographic shifts of their user base [Koh et al., 2021]. Developing a general procedure for addressing these challenges can provide a reliable solution for such applications where there may be high stakes in societal behavior, public health, or monetary assets.

Without considering potential shifts, there is a risk of using suboptimal strategies, however it is often difficult to predict exactly the shifts that will occur. Distributionally robust optimization formulations such as ours attempt to address this uncertainty over potential shifts by learning policies that attain good performance on all potential distributions within a user-defined set of distributions (denoted the *uncertainty set*). Prior methods for contextual multi-armed bandits [Si et al., 2020, Mo et al., 2021, Kallus et al., 2022] that aim to provide robustness against distribution shifts do not separate the different sources of shifts, and consider them in a joint formulation. These methods may be disadvantageous as they lack the flexibility to separate the sources. Consequently, we empirically find and show that some of these methods may produce degenerate results under binary reward feedback. This issue is theoretically proven to occur in the DRO literature in supervised learning [Hu et al., 2018]. Due to this, methods that better use structure to mitigate this issue have been an active area of research in the supervised learning DRO literature [Sagawa et al., 2019, Duchi et al., 2019]. Inspired by this, our insight is that better usage of structure is important, and there is often expert opinion about the maximum amount of distribution shift that may occur that can be used to achieve better performance on the deployment environments.

In this work we incorporate this insight in the framework of contextual bandits. We propose our Factored Distributionally Robust Optimization (FDRO) algorithm that uses past data to search for a policy that has the best worst-case performance, when the *context* (e.g., population demographic) or/and the *reward model* (e.g., influence of strategies on voting behavior) can be different from the ones during data collection. Towards this goal, we make three primary contributions:

- **I.** We present a factored distributionally robust optimization formulation for contextual bandits that provides end-users the flexibility to *separately* specify different amounts of potential shifts in the two main sources of uncertainty: the context distribution and the reward distribution. Our factored formulation can overcome some failure cases of non-factored prior methods Si et al. [2020].
- **II.** We establish theoretically that the proposed estimators provide convergence at a rate of $O\left(\sqrt{\log(n)/n}\right)$ in evaluation of any policy's worst-case performance under distribution shifts, as well as to a policy with the best worst-case performance.
- **III.** Building on our theoretical insights, we develop a practical algorithm, FDRO, that uses function approximation and a Taylor series expansion. We show that FDRO provides promising empirical results on several domains based on real-world datasets.

## 2 Related Works

Both off-policy learning for bandits and reinforcement learning, as well as distributionally robust optimization (DRO) for supervised learning and stochastic optimization have rich histories of research. To briefly summarize, methods for off-policy evaluation/learning predominantly assume that the deployment environment for the learned policy will be the same as the environment the historical data was collected in [Precup, 2000, Jiang and Li, 2016, Thomas and Brunskill, 2016a,b, Gelada and Bellemare, 2019, Liu et al., 2018, Imani et al., 2018, Yang et al., 2020, Jiang and Huang, 2020, Kallus and Uehara, 2020a,b, Liu et al., 2020]. In contrast, DRO methods for tackling distributional shifts have predominantly focused on settings like supervised learning, and do not consider decision making aspects present in (contextual) bandits [Bertsimas and Sim, 2004, Hu and Hong, 2013, Namkoong and Duchi, 2016, Shapiro, 2017, Duchi et al., 2016, Duchi and Namkoong, 2018, Hashimoto et al., 2018, Duchi et al., 2019, Sagawa et al., 2019, Oren et al., 2019, Michel et al., 2022].

In addition to the prior work already previously mentioned in Section 1, there have been a few recent papers investigating DRO for use in bandits and reinforcement learning. Some works use DRO to account for the mismatch between the true distributions (of the context and rewards) and the observed empirical distributions [Swaminathan and Joachims, 2015a,b, Faury et al., 2020, Sakhi et al., 2020]. These methods account for the potential errors of finite-data estimation, and, under infinite data, they converge to the best training distribution policy. However we are concerned with the setting when the test distribution may be different, and the best training-distribution policy may not be optimal.

There has also been other work that consider settings when different forms of information are available a priori. For example, in settings where knowledge about the distribution of covariates during evaluation [Kato et al., 2020], bounds on the ratio of train-eval covariate distribution [Hatt et al., 2021, Giguere et al., 2021], bounds on train-evaluation performance differences [Chandak et al., 2021], the causal directed acyclic graph [Subbaswamy et al., 2019, Singh et al., 2021], or when there is data through time that can be used to model potential non-stationarity [Thomas et al., 2017, Chandak et al., 2020, Xie et al., 2020, Qin and Russo, 2021] have been considered. While relevant, they look at complementary problems.

## 3 Preliminaries

In this work, we consider the batch setting where the goal is to learn from an existing dataset $\mathcal{D}$. We consider a setting with discrete actions, where $\mathcal{A}$ denotes the set of $k$ discrete actions, $A$. Let $\mathcal{X} \subset \mathbb{R}^d$ denote the set of contexts, $X$. A policy $\pi$ denotes a mapping from a context $X$ to a distribution over actions. For a given context $X$ and action $A$ we observe a stochastic reward $R$. We assume the dataset, $\mathcal{D}$, consists of $n$ independently collected datapoints, each a (context, action, reward) triplet: $\mathcal{D} = \{X_i, A_i, R_i\}_{i=1}^n$. For this dataset, we assume the actions are selected with some data collection policy $\pi_0$ where $A_i \sim \pi_0(\cdot|X_i)$. We assume there is an underlying context distribution $P_{x_0}$, and for each context-action pair there is an underlying reward distribution $P_{r_0|X,A}$. We additionally assume the observed contexts, $X_i$ are independent and identially distributed (i.i.d.) draws from $P_{x_0}$, and the observed rewards are i.i.d. draws from $P_{r_0|X_i,A_i}$. We assume $P_{x_0}$ is known while $P_{r_0|X_i,A_i}$ is unknown.

We consider using $\mathcal{D}$ to learn a policy $\pi$ that will achieve the highest worst-case-reward when the deployment environment may be shifted from the one of data collection. Given a fixed policy, there are two ways the environment can shift and change the distribution of future outcomes. First the context distribution can shift so the inputs we see are different. Second the reward generation model can shift, changing the reward distribution of each (context, action) pair. We assume the exact shift is not known, so we would like to learn a policy that would do well on all environments within a user specified set of distributions of the context distribution $P_x$ and reward distributions $P_{R|X,A}$. Formally, given $\mathcal{D}$ we aim to find the policy that optimizes the following distributionally robust objective:

$$\pi^*_{FDRO} \in \operatorname*{argmax}_{\pi \in \Pi} \inf_{P_x \in \mathbb{P}_{x_0}} \mathbb{E}_{P_x}\left[\mathbb{E}_\pi\left[\inf_{P_r \in \mathbb{P}_{r_0|X,A}} \mathbb{E}_{P_r}[R]\right]\right] \tag{1}$$

In Equation (1), the uncertainty sets $\mathbb{P}_{r_0|X,A}$ and $\mathbb{P}_{x_0}$ describe the set of possible distributions, $P_r$ and $P_x$, respectively. The optimization set $\mathbb{P}_{r_0|X,A}$ is conditioned on random variables $X \sim P_x$ and $A \sim \pi(\cdot|X)$. We term our objective *Factored-DRO* as we have two layers of distributionally robust optimization, first over reward shifts $P_{R|X,A}$, and another over context shifts $P_x$.

For theoretical analysis, we make the following assumptions common in prior literature [Si et al., 2020, Swaminathan and Joachims, 2015a, Kallus and Uehara, 2020a].

**Assumption 1.** *Assumptions on $\pi_0$ and the reward distribution*

*(i) Unconfounded: Rewards are independent with A conditioned on X: $(R_{X,A_1}, R_{X,A_2}...) \perp\!\!\!\perp A|X$*

*(ii) Bounded reward support: $0 \leq R \leq R_M$ for all rewards for a finite $R_M$*

*(iii) Coverage/Sufficient exploration: $\pi_0(A|X) \geq \epsilon_\pi$ for all $(X, A) \in \mathcal{X} \times \mathcal{A}$ for some $\epsilon_\pi > 0$*

*(iv) Positive density of rewards: Let $f_{R|X,A}$ be the pdf (continuous case) or $p_{R|X,A}$ be the pmf (discrete case) of the reward distribution $P_{R|X,A}$. Then assume, for all reward distributions involved, it holds that $f_{R|X,A} \geq \epsilon_r$ (continous) or $p_{R|X,A} \geq \epsilon_r$ (discrete) on the support of of R for all $(X, A) \in \mathcal{X} \times \mathcal{A}$ for some $\epsilon_r > 0$.*

We make an additional assumption, which we do not require in our practical algorithm:

**Assumption 2.** *Discrete contexts with sufficient mass: The context set $\mathcal{X}$ is a finite discrete set and the probability mass of each context is lower bounded with $P_{x_0}(x = X) \geq \epsilon_X$ for some $\epsilon_X > 0$*

**Note about model misspecification:** Context shifts are an issue when we have model misspecification and cannot model the best action for each context. In these cases, the learned model encodes the biases of the population distribution. Model misspecification can occur when the model is under-specified, for example, to reduce variance (bias-variance tradeoff), for interpretability, or due to unobserved variables. Misspecification also occurs when the model is over-specified, where it may overfit and not generalize (ex. neural networks [Sagawa et al., 2019]). The real world is often complex and difficult to model correctly, so throughout this work we assume the misspecified setting.

**Prior Work with Joint Shifts:** Recently Si et al. [2020] proposed an algorithm for solving a DRO formulation for contextual bandits that does not separate the context and reward shift. They consider the optimization problem: $\pi^* = \text{argmax}_\pi \inf_{P \in \mathbb{P}_{P_0}} \mathbb{E}_P[R]$, where $P$ encapsulates the joint distribution of contexts and rewards and $\mathbb{P}_{P_0} = \{P : D_{kl}(P||P_0) \leq \delta\}$.

**Degenerate Outcomes with binary rewards under the joint objective:** Unfortunately, the joint formulation may have a key limitation in the common setting of binary reward outcomes (ex, clicked or not-clicked[Chu et al., 2011], voted or did-not-votedGerber et al. [2008], etc). In supervised learning (SL), similar DRO formulations have been theoretically proven to fail with certain loss functions. Hu et al. [2018] prove that when features are continous, and 0-1 classification loss is used, the DRO objective is equivalent to the standard, non-robust optimization of the loss function on the training distribution. Intuitively, DRO can be seen as changing the weights of the datapoints based on the incurred loss, and increasing the radius parameters increases the weights on datapoints with larger incurred losses. However, under 0-1 loss, DRO upweights all missclassified examples equally, which does not change the decision boundary. Consequently, in SL, methods that can mitigate this issue by using structure, such as Duchi et al. [2019] or Sagawa et al. [2019], have been proposed. We observe this issue empirically for the joint formulation in the binary reward case (see Section 6).[2] Intuitively, the lower reward points for each context action pair are all upweighted equally so the learned policy is equivalent to that optimized for the training distribution. Our method, which trades-off adding an additional hyperparameter with better structure usage to separate out the shifts mitigates this issue.

## 4 Factored DRO for Policy Evaluation and Learning

In our work, we consider uncertainty sets defined on the KL radius from the data collection distributions. Due to the joint decomposition, we require the user to specify two KL-radius hyperparameters, $\delta_c$ and $\delta_x$, for rewards and contexts respectively as opposed to a single hyperparameter required in prior work. The uncertainty sets defined in Equation 1 are defined as $\mathbb{P}_{r_0|X,A} = \{P_r : D_{kl}(P_r, P_{r_0|X,A}) < \delta_c\}$ and $\mathbb{P}_{x_0} = \{P_x : D_{kl}(P_x, P_{x_0}) < \delta_x\}$.

We can decompose our DRO objective from Equation 1 into two steps, first a policy evaluation step which given a policy evaluates the worst case policy reward. Notice the policy evaluation step itself can be separated into two steps: a step considering the potential reward generation shift ($Q_{X,A}$) and a step over the potential context shift to arrive at the desired value ($V(\pi)$):

$$V(\pi) = \inf_{P_x \in \mathbb{P}_{x_0}} \mathbb{E}_{P_x} \left[ \mathbb{E}_\pi \left[ Q_{X,A} \right] \right] \quad \text{where} \quad Q_{X,A} = \inf_{P_r \in \mathbb{P}_{r_0|X,A}} \mathbb{E}_{R \sim P_r}[R] \tag{2}$$

Second, we have a policy optimization step over the worst case policy rewards:

$$\pi^*_{FDRO} = \underset{\pi}{\text{argmax}} \; V(\pi) \tag{3}$$

In this section, we present our estimators for policy evaluation and learning and our convergence guarantees. We give all result from prior work used and all full proofs in the appendix.

### 4.1 Policy Evaluation:

We start by discussing the policy evaluation step. Equation 2 as written is a difficult constrained optimization over a probability distribution space. Fortunately, prior work (such as Hu and Hong

[2]Note, Si et al. [2020] show good results in an environment (voting) which originally had binary rewards, but they transform the environment rewards to non-binary by shifting all rewards by a different offset per action.

[2013]) exists that uses methods from convex optimization to transform the problem to an equivalent but simpler convex optimization over a single scalar. They use convexity of the original objective and strong duality to derive an equivalent dual objective. We arrive at the following equivalent problem by applying this duality procedure twice, once for each layer.

**Theorem 1** (Strong Duality)**.** *The optimization problem in equation 2 is equivalent to solving:*

$$V(\pi) = -\min_{\alpha_x > 0} \left\{ \alpha_x \log \mathbb{E}_{X \sim P_{x_0}} \left[ \exp\left( \frac{\mathbb{E}_\pi \left[ -Q_{X,A} \right]}{\alpha_x} \right) + \alpha_x \delta_x \right] \right\}, \tag{4}$$

$$\textit{where } Q_{X,A} = -\min_{\alpha_c > 0} \left\{ \alpha_c \log \mathbb{E}_{R \sim P_{r_0 | X,A}} \left[ \exp\left( \frac{-R}{\alpha_c} \right) \right] + \alpha_c \delta_c \right\}. \tag{5}$$

*Proof Sketch* This follows from applying Theorem 1 of Hu and Hong [2013] twice.

We will use this easier to optimize dual form to derive estimators for policy evaluation. We will first consider the potential shift of the reward distribution of each (context, action) pair and calculate estimates of $Q_{X,A}$ of Equation 5. We will then use these values to perform the second DRO layer over context shifts to estimate $V(\pi)$ (Equation 4).

#### 4.1.1 First DRO, Reward Shift:

For theoretical analysis, we consider the setting when the context and action sets are both finite and discrete (Assumptions 1.2 and 2). Therefore we have a finite number of unique $(X, A)$ pairs, each which occurs with a non-zero probability. For a given $(X, A)$ pair, let $\mathcal{R}_{X,A}$ correspond to the set of the observed reward $R_i$ every time $(X, A)$ is observed in $\mathcal{D}$: $\mathcal{R}_{X,A} = \{R_i : (X_i, A_i, R_i) \in \mathcal{D} \text{ if } X_i = X, A_i = A\}$. Additionally, let $m_{xa} = p_0(x, a)n$ denote the expected number of datapoints in each $(X, A)$ bin and recall that $\delta_c$ denotes the user specified KL-radius of the ambiguity set.[3] For every unique $(X, A)$ we propose the following estimator for worst case context-action reward for $Q_{X,A}$ in equation 5:

$$\hat{Q}_{X,A} = -\min_{\alpha_c > 0} \left\{ \alpha_c \log \left( \frac{1}{m_{xa}} \sum_{R_j \in \mathcal{R}_{X,A}} \left[ \exp\left( \frac{-R_j}{\alpha_c} \right) \right] \right) + \alpha_c \delta_c \right\} \tag{6}$$

Note that the objective inside the minimization in equation 6 is convex in $\alpha_c$ for $\alpha_c > 0$, therefore many optimization methods (ex. gradient descent, bisection search) are able to achieve the global optimum when $\alpha_c \neq 0$. We give the gradients and a proof of convexity in the appendix. Additionally, note the $\hat{Q}_{X,A}$'s only depend on $\mathcal{D}$ and do not depend on $\pi$, therefore they only need to be computed once and can be used throughout the rest of the policy evaluation and optimization procedure.

#### 4.1.2 Second DRO, Context Shift:

With the $\hat{Q}_{X,A}$ values estimated above for each $(X, A)$ pair we can complete the policy evaluation. Let $\delta_x$ denote the user-specified KL-radius for context distribution shift, then with the computed $\hat{Q}_{X,A}$ for each $(X, A)$ pair, we can complete the policy evaluation and estimate $V(\pi)$ from equation 2.

$$\hat{V}(\pi) = -\min_{\alpha_x > 0} \left\{ \alpha_x \log \sum_{X \in \mathcal{X}} P_{x_0}(x) \exp\left( \frac{-\sum_{a \in \mathcal{A}} \pi(a|X_i) \hat{Q}_{X,a}}{\alpha_x} \right) + \alpha_x \delta_x \right\} \tag{7}$$

We can show that when $\delta_c > 0$ and $\alpha_c^* > 0$ policy evaluation converges at rate $O(\sqrt{\log(n)/n})$:

**Theorem 2** (Convergence of policy evaluation)**.** *For $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$, the following holds for any $\pi$ with probability of at least $1 - \delta$:*

$$\left| V(\pi) - \hat{V}(\pi) \right| \leq \mathcal{O}\left( c \sqrt{\frac{\log(n) \log\left( \frac{2|\mathcal{X}||\mathcal{A}|}{\delta} \right)}{n}} \right) \quad \textit{where } c = \frac{32(R_{max})^3}{\delta_c^2 \underline{\alpha}_c^2 \exp(\frac{-R_{max}}{\underline{\alpha}_c}) \underline{p}_0(x, a)} \tag{8}$$

[3]Note it is also possible to provide a different $\delta_{c,(X,A)}$ for every unique $(X, A)$, or a different $\delta_{c,A}$ per action, however this greatly increases the hyperparameter burden on the end-user.

*and $\underline{p}_0(x,a) = \min_{(x,a)} p_0(x,a) \geq \epsilon_\pi \epsilon_X$ is the minimum over probability of occurrence of (X,A) pairs, and $\underline{\alpha}_c$ is the minimum value for $\alpha_c$.*

*Proof sketch:* We utilize a uniform convergence result from stochastic optimization and local Lipschitz-continuity to prove the desired result.

Note the objective in equation 7 is also convex and easy to optimize for $\alpha_x > 0$.

### 4.2 Policy Learning

We now consider the optimization over policy of equation 3. Define the true robust optimal policy as $\pi^* = \pi^*_{FDRO} = \text{argmax}_\pi V(\pi)$, with true optimal robust value $V^* = V(\pi^*)$. Define the empirical robust optimal policy as $\hat{\pi}^* = \text{argmax}_\pi \hat{V}(\pi)$ with empirical robust optimal value $\hat{V}^* = V(\hat{\pi}^*)$. We additionally define the set of $\epsilon$-optimal policies $\Pi^*_\epsilon = \{\pi : V(\pi) \geq V^* - \epsilon\}$. We then have the following statement of the convergence of optimal policy and value:

**Theorem 3** (Convergence of Policy Learning)**.** *For $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$, the following holds with probability of at least $1-\delta$:*

$$\left|\hat{V}^* - V^*\right| \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{9}$$

*Equivalently this says*

$$P(\hat{\pi} \in \Pi^*_\epsilon) > 1 - \mathcal{O}\left(2|\mathcal{X}||\mathcal{A}|\exp\left(-\frac{n\epsilon^2}{\log(n)c^2}\right)\right) \tag{10}$$

While this policy optimization is generally non-convex, for many policy classes we can use gradient ascent with multiple restarts. To learn a policy, we first compute the $\hat{Q}_{X,A}$ values of Section 4.1.1, we then perform alternating steps of the evaluation step in Section 4.1.2 and the policy optimization in this section (Section 4.2).

## 5 Practical Algorithm with Function Approximation

Our estimator for solving for the worst case reward shift value, $\hat{Q}_{X,A}$ for each context action pair in our first DRO step (section 4.1.1) relies on having a sufficient number of datapoints per each $(X, A)$ to approximate $\mathbb{E}_{R\sim P_{R|X,A}}[\exp(-R/\alpha_c)] = 1/m_{xa}\sum_j \exp(-R_j/\alpha_c)$. However, in many settings, contexts may not be discrete and, in these cases, each continuous context may only be observed once. One approach can be to discretize the context space, however selecting the correct discretization is challenging. Too coarse a discretization introduces looseness and leads to overly conservative results, but too fine a discretization may not leave enough items in each bin to accurately estimate $\mathbb{E}_{R\sim P_{R|X,A}}[\exp(-R/\alpha_c)]$. To handle continuous and/or large contexts spaces, instead of binning we use a function approximation that may not have the same convergence guarantees. Fortunately, as we show in Section 6 it can be quite accurate and lead to better results than the baseline methods. We develop an empirically stable method for estimating $\mathbb{E}_{R\sim P_{R|X,A}}[\exp(-R/\alpha_c)]$ using a Taylor series approximation. The key feature of our method is that it does not require fitting the exponential dependence on $R$ and $1/\alpha_c$. We found alternate methods, such as fitting an approximator for $\mathbb{E}_{R\sim P_{R|X,A}}[\exp(-R/\alpha_c)]$ directly by fitting $\exp(-R_i/\alpha_c)$ using corresponding values of $X_i$, $A_i$ and multiple values of $\alpha_c$ value sampled from the relevant range, to be unstable at small values of $\alpha_c$ due to the exponential dependence.

### 5.1 Approximating $\mathbb{E}_{R\sim P_{R|X,A}}[\exp(-R/\alpha_c)]$:

We can use the Taylor series expansion, $\sum_{i=0}^{\infty}\frac{(1/\alpha_c)^i}{i!}\mathbb{E}[(-R)^i]$, to estimate an order $m$ approximation. We do this by fitting a multi-output function approximator, $\mathcal{F}$ to approximate the first $m$ moments. We train $\mathcal{F}$ such that the $j^{th}$ output predicts $(-R_i)^j$ from $(X_i, A_i)$. When trained on all

| | Simulated | Voting | Scene |
|---|---|---|---|
| $n_{total},\ (n_{train}, n_{test})$ | $40K, (20K, 20K)$ | $180K,\ (\sim 44K, \sim 12K)$ | $2.407K,\ (\sim 1.6K, \sim 0.8K)$ |
| $\lvert\mathcal{A}\rvert, n_{cgs}, n_{feats\ used}$ | 4, 4, 5 | 5, -, 2 | 6, 2, 294 (projected to 3) |
| Train % per cg | $(0.1, 0.4, 0.1, 0.4)$ | - | $(0.7, 0.3)$ |
| Test % per cg | Various | - | $(0.15, 0.85)$ |
| $\pi_0(a)$ | $(.1, .2, .4, .3)$ | $(.56, .11, .11, .11, .11)$ | $(.17, .17, .17, .17, .17, .17)$ |
| $\mathbb{E}_{train}[R\|cg, A]$ | $cg_{1,3}$:$(.35, .5, .2, .14)$<br>$cg_{2,4}$:$(.35, .2, .5, .14)$ | -<br>- | $cg_0$:$(.25, 0, 0, 0, .36, .34)$<br>$cg_1$:$(0, .3, .36, .35, .1, 0)$ |

Table 1: Environment/Experiment details. The first row lists the approximate number of points in the train and test splits as we randomize the exact splits across runs. 'cg' stands for *coarse group* as described in the text. The last three rows list the percentages/reward values per action in the tuple. Note for voting, we do not have coarse groups as we use a shift between cities, additionally we consider using data from only some of the cities so not all data is used. We use random projection to project the feature space of Scene to avoid ill-conditioned behavior.

the data, $\mathcal{F}[j]$ should model the $j^{th}$ moment $\mathbb{E}_{P_{R|X_i,A}}[(-R)^j]$. Let $\mathcal{D}_X$ represent the sequence of contexts in the dataset. Then for every context $X_i \in \mathcal{D}_X$ and every $a \in \mathcal{A}$ and any $\alpha_c$, we use our approximators $\mathcal{F}_j$ to estimate the desired value:

$$\hat{\mathbb{E}}_{P_{R|X_i,a}}\left[\exp\left(\frac{-R}{\alpha_c}\right)\right] = 1 + \frac{1}{\alpha_c}\mathcal{F}_1(X_i, a) + .. + \frac{1}{\alpha_c^m}\frac{\mathcal{F}_m(X_i, a)}{m!}, \quad \text{for any } a \in \mathcal{A}, \alpha_c > 0. \quad (11)$$

This approximation achieves better regularity as we are no longer fitting the dependence on $\alpha_c$. Practically, we find order $m = 5$ can often be quite accurate. Additionally, note that while the context is high dimenstional, the Taylor expansion is only for the scalar $\alpha_c$.

**Complete Algorithm:** With our estimator $\hat{\mathbb{E}}_{P_{R|X_i,A}}[\exp(-R/\alpha_c)]$ we proceed with our method by first calculating $n \times k$ values of $\hat{Q}_{X_i,a}$ for each context in the dataset and each action in the action set. To do this, we iterate through $(X_i, a) \in (\mathcal{D}_X \times \mathcal{A})$ and solve the following *convex* problem in $\alpha_c$:

$$\hat{Q}_{X_i,a} = -\min_{\alpha_c > 0}\{\alpha_c \log(\hat{\mathbb{E}}_{P_{R|X_i,a}}[\exp(-R/\alpha_c)]) + \alpha_c \delta_c\}. \quad (12)$$

We then carry out the procedure detailed in Section 4.1.2 onwards with these $\hat{Q}_{X_i,a}$. Algorithm 1 in the Appendix provides our full algorithm for policy evaluation and optimization, that considers both the discrete binned case (Section 4.2) and the function approximation case just discussed.

## 6 Experiments

We test and analyze our algorithm empirically in three settings, including two derived from real data. For all experiments, we use the practical algorithm discussed in Section 5. While our method works for any policy that can be optimized, we consider learning the parameters $\theta$ of a stochastic softmax linear contextual bandit policy of the form:

$$\pi_\theta(X, A) = \frac{\exp(\phi(X, A)^T\theta)}{\sum_{a \in \mathcal{A}} \exp(\phi(X, a)^T\theta)} \quad \text{where } \phi(X, A) = [X, OneHot(A)] \quad (13)$$

This policy $\pi_\theta$ is differentiable so we optimize $\theta$ using gradient descent. For the function approximators $\mathcal{F}$ discussed in section 5, we use exponential-kernel linear regression models.

We compare against three baselines:

(i) **baseline-IS**: The standard non-robust method for maximizing the reward calculated using importance sampling policy evaluation on the training data and does not consider the shift,

(ii) **baseline-POEM** [Swaminathan and Joachims, 2015a] POEM improves the baseline-IS method by adding variance regularization to account for finite sample error, and

(iii) **baseline-DRO** [Si et al., 2020] as mentioned throughout the text, considers the shift over the joint (context, reward) distribution as opposed to separating them.

### 6.1 Evaluation Domains

We first provide a brief overview of our evaluation settings and provide full descriptions in the appendix. In one setting, the voting dataset setting, we consider the case when data from one city is used to train a policy deployed in some of the other cities. In the two other settings, to construct example shifts we cluster the datapoints by one of the context covariates into what we denote as *Coarse Groups* (cg). We then consider settings where the training distribution may consist of a different composition of cgs than the deployment/testing distribution. For each environment we give details in Table 1, such as the average reward on the training set of each (cg, action) pair.

**Simulated Setting:** Our simulated setting has 4 cgs. To mimic the non-linear relationships that may occur when using real data, unknown to the designer, cg 1 and 3 ($cg_{1,3}$) are identical, as are cgs 2 and 4 ($cg_{2,4}$). Each context is composed of the cg and 4 other values sampled from uniform distributions. For each of the 4 actions and cg, each (cg, action) pair has a set of true reward parameters $\phi^*$. The linear combination of $\phi^*$ and the contexts determine the Bernoulli parameter of the observed reward.

**Scene Setting:** We additionally test in a setting derived from the multiclass-supervised learning Scene classification dataset from the LibSVM [Chang and Lin, 2011] repository. We use a slightly modified version of the supervised to bandit conversion method from prior literature [Swaminathan and Joachims, 2015a, Agarwal et al., 2014]. In this method we assume we do not observe the correct label directly. Instead each label is an action and selecting the correct action/label gives positive reward, while the wrong action/label gives negative reward. We use the true label to create cgs and consider the case where we do not have access to the coarse group at test time so it cannot be used in learning the policy. This illustrates a more challenging and realistic case where there is shift in the unobserved context features. To create a context distribution shift between train and test, we sampled different proportions of datapoints from each group to put in the train distribution.

**Voting Setting:** The voting dataset by Gerber et al. [2008] contains data collected from a randomized controlled trial-style study testing various actions (1 control and 4 treatments) to increase voter turnout. We consider a potential real shift in the dataset by using data from one city to train a policy deployed in some of the other cities. To illustrate the worst case, we chose the training city and the deployment cities such that there is a large context difference between the cities (details in appendix). We consider the recorded binary voting outcome as our reward. Because the non-control actions require additional effort of printing and mailing a letter compared to the control (do nothing), we slightly lower the non-control action rewards by 0.09 to simulate this effort cost.

### 6.2 Results

We use the training dataset to optimize the policies and evaluate on the test dataset. For the voting dataset we use importance sampling to evaluate the policies. For Simulated and Scene, we have the true rewards for all action and use them for evaluation. All the robust algorithms have robustness parameters. FDRO has two, the reward distribution uncertainty set radius, $\delta_c$, the context distribution radius, $\delta_x$. We expect many of our settings predominantly have context shifts. We leverage this information to set our FDRO parameter for reward shift to $\delta_c = 0.001$. In Figure 1f and the appendix we give performance at different values of $\delta_c$ and show performance is similar for a range of values. We show our main results in the top row of Figure 1 where we consider the effect of $\delta_x$. Baseline-DRO has a single parameter for the KL-radius over the joint context-reward distribution ambiguity set, $\delta$. Baseline-POEM also has a single parameter, the variance regularization strength, $\lambda$. Faury et al. [2020] prove that POEM can be equivalently viewed from a DRO formulation. So for comparison when reporting the results, we first translate $\lambda$ to the corresponding DRO-radius $\delta = n\lambda^2$. We demonstrate in all three settings, FDRO is able to outperform all the baselines.

In Figure 1a, we consider the case of testing an algorithm fit with a fixed $\delta_x$ in different populations. We plot the performance gain of each algorithm over baseline-IS as the test population is increasingly different. We made this environment especially challenging for FDRO by fixing $\delta_x$ for FDRO while solving for the **best** $\delta$ for *each* different test population for baseline-DRO and baseline-POEM. We see that as expected when the test context distribution is nearly identical to the training context distribution, the robust policy learned by FDRO does not perform as well (negative reward difference). However as the test distribution differs, we see FDRO starts performing much better than the non-robust policy. Also as predicted, in setting with binary rewards, the joint formulation of baseline-DRO suffers from the same issues of degeneration in learning the optimal train-distribution policy as in

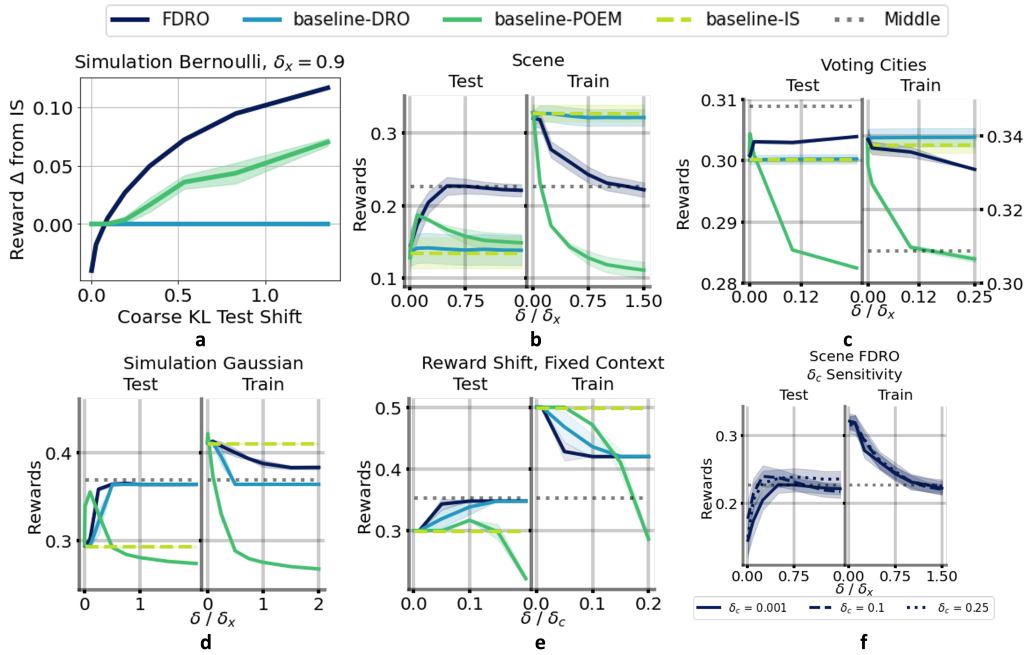


Figure 1: The dotted line 'Middle' serves as a constant horizontal reference across the two panels in plots (b-e). All plots are plotted with 95% confidence intervals shaded. Reported results and confidence intervals are averaged over 10 runs in all simulated domains, and 30 runs in Scene and Voting. **Top Row:** We plot our main results and demonstrate FDRO is able to learn more robust policies under context shift in all setting. (a) We vary the shift in the test distribution and plot results as the performance gain over baseline-IS. (b,c) We fix the train and test distributions and plot the performance at different $\delta/\delta_x$ hyperparameters in Scene and Voting. **Bottom Row:** We present discussion results: (d) Results from a continuous-reward domain: FDRO is able to outperform baselines (e) A domain where the test reward distribution shifts but context does not. (f) We show performance of FDRO at multiple values of $\delta_c$ for the Scene dataset.

the supervised DRO literature [Hu et al., 2018]. We additionally see that even with the advantage of using the best parameter per test population, baseline-POEM still does not perform as well.

In Figures 1b and c. We consider fixed training and test distributions and plot performance at different values of $\delta/\delta_x$ for the voting and scene settings. We see that with increasing $\delta_x$, FDRO can learn robust policies that perform well on the test set. Baseline-DRO suffers from the same issue previously discussed. The performance of baseline-POEM decreases on both distributions. As the variance regularization weight increases, POEM is incentivized to learn low variance policies, such as choosing low variance but low reward actions, or reducing the variance from importance sampling by choosing a policy close to the data-collection policy, which may be suboptimal. Figure 1c shows a real-world shift between cities, we can see if we would like to use data from one city to train a policy to deploy in other cities, it is important to consider distribution shift. Additionally for the case we consider with binary rewards it may be necessary to use FDRO.

### 6.3 Discussion

**Continuous Rewards and Reward Shift** We focus on the common bandit setting of binary rewards in the main result; however, we also show results for a continuous rewards setting in a simulations environment with Gaussian-distributioned rewards. In Figure 1d, we see that baseline-DRO is able to learn good polices, however FDRO with extra structure information was able to learn a better policy, with better train performance at the same test performance. Additionally, we explore reward shifts. We create a new simple environment (described in the appendix) where we can easily create a test environment with shifted rewards. We fix $\delta_x$ and sweep $\delta_c$ for FDRO and we see in Figure 1d, both FDRO and baseline-DRO are able to handle this this case well.

**When to use FDRO vs. baseline-DRO (equivalence of solutions)** One question is if the parameters can be set such that baseline-DRO and FDRO choose the same solution, and if not when one is beneficial over the other. To examine this question we can look at the KL-divergence chain rule decomposition:

$$D_{kl}(p_0(R,X)||p(R,X)) = D_{kl}(p_0(X)||p(X)) + E_{p_0(x)}[D_{kl}(p_0(R|X)||p(R|X))].$$

Recall $D_{kl}(p_0(R,X)||p(R,X)) = \delta$, $D_{kl}(p_0(X)||p(X)) = \delta_x$, $D_{kl}(p_0(R|X)||p(R|X)) = \delta_c$. We can see, when $\delta_c$ is the same for each $X$, we have $\delta = \delta_x + \delta_c$. When $\delta$ of baseline-DRO is set to $\delta_x + \delta_c$ baseline-DRO and FDRO consider distributions the same KL distance away, with FDRO placing additional constraints on the set of distributions searched. From the decomposition, we can see, with additional stipulations (details in appendix), we can set a family of $\delta_x$ and $\delta_c$ values for FDRO to achieve the same solution of baseline-DRO for each $\delta$ . However, for a given $\delta_x$ and $\delta_c$ it is not possible to find a $\delta$ for baseline-DRO that achieves the same solution. Baseline-DRO has less constraints and can, for example, place the shift "budget" ($\delta_x + \delta_c$) all in the rewards or all in the contexts if that is the worst case. This may cause the degenerate performance in the binary rewards case for baseline-DRO where reward shift dominates. FDRO can set limits on how much reward shift is considered and ensures context shift is considered, allows us to overcome this. This example informs us that generally, if the algorithm designer would like to ensure a certain amount of reward and context shift are considered, then it may be beneficial to use FDRO and invest the additional effort to decompose $\delta$ between $\delta_x$ and $\delta_c$.

**Setting the $\delta$ parameters and potential negative societal impacts** One limitation common to all DRO methods is that choosing the uncertainty set ($\delta$) may be difficult [Rahimian and Mehrotra, 2019]. Accounting for too large a shift may result in overly conservative policies. While DRO methods and FDRO are robust to shifts, if the shift is not as large as predicted, the policy learned using FDRO may be overly conservative which can potentially cause negative impact in terms of lower than necessary performance. To mitigate this risk, $\delta$ should be carefully set. Common approaches from prior work, which can apply to our method as well, is to use other similar datasets or sources to derive reasonable shifts. Additionally, empirically we see, our method often works for many values of $\delta$. For example in Figure 1a, we show while we may incur a small penalty by setting $\delta_x$ too large if the distribution does not shift, we can learn a policy that has much larger gains if the context distribution does shift. Similarly, in Figure 1f we show robustness to $\delta_c$. Another approach could be to utilize a human in the loop approach by examining the worst case distribution. The worst case distribution can be easily calculated for KL uncertainty sets and more details and demonstration are provided in the appendix.

## 7 Conclusion

In this work we consider the batch learning of contextual bandit policies when the deployment distribution may shift from that of data collection. We propose a distributionally robust optimization formulation that separates the reward and context shift. We demonstrated that our proposed estimators converge asymptotically and then developed a practical algorithm FDRO that is able to overcome some failure cases of prior methods and learn better performing robust policies.

## Acknowledgements

This work was supported in part by a Stanford Human-Centered AI Hoffman-Yee Grant.

## References

A. Agarwal, D. Hsu, S. Kale, J. Langford, L. Li, and R. Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*, pages 1638–1646. PMLR, 2014.

D. Bertsimas and M. Sim. The price of robustness. *Operations research*, 52(1):35–53, 2004.

S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Y. Chandak, G. Theocharous, S. Shankar, M. White, S. Mahadevan, and P. Thomas. Optimizing for the future in non-stationary mdps. In *International Conference on Machine Learning*, pages 1414–1425. PMLR, 2020.

Y. Chandak, S. Niekum, B. da Silva, E. Learned-Miller, E. Brunskill, and P. S. Thomas. Universal off-policy evaluation. *Advances in Neural Information Processing Systems*, 34, 2021.

C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

W. Chu, L. Li, L. Reyzin, and R. Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214. JMLR Workshop and Conference Proceedings, 2011.

J. Duchi and H. Namkoong. Learning models with uniform performance via distributionally robust optimization. *arXiv preprint arXiv:1810.08750*, 2018.

J. Duchi, P. Glynn, and H. Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *arXiv preprint arXiv:1610.03425*, 2016.

J. C. Duchi, T. Hashimoto, and H. Namkoong. Distributionally robust losses against mixture covariate shifts. *Under review*, 2, 2019.

L. Faury, U. Tanielian, E. Dohmatob, E. Smirnova, and F. Vasile. Distributionally robust counterfactual risk minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3850–3857, 2020.

S. G. Finlayson, A. Subbaswamy, K. Singh, J. Bowers, A. Kupke, J. Zittrain, I. S. Kohane, and S. Saria. The clinician and dataset shift in artificial intelligence. *The New England journal of medicine*, 385(3):283, 2021.

C. Gelada and M. G. Bellemare. Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3647–3655, 2019.

A. S. Gerber, D. P. Green, and C. W. Larimer. Social pressure and voter turnout: Evidence from a large-scale field experiment. *American political Science review*, 102(1):33–48, 2008.

S. Giguere, B. Metevier, Y. Brun, P. S. Thomas, S. Niekum, and B. C. da Silva. Fairness guarantees under demographic shift. In *International Conference on Learning Representations*, 2021.

T. Hashimoto, M. Srivastava, H. Namkoong, and P. Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR, 2018.

T. Hatt, D. Tschernutter, and S. Feuerriegel. Generalizing off-policy learning under sample selection bias. *arXiv preprint arXiv:2112.01387*, 2021.

W. Hu, G. Niu, I. Sato, and M. Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pages 2029–2037. PMLR, 2018.

Z. Hu and L. J. Hong. Kullback-leibler divergence constrained distributionally robust optimization. *Available at Optimization Online*, 2013.

E. Imani, E. Graves, and M. White. An off-policy policy gradient theorem using emphatic weightings. *Advances in Neural Information Processing Systems*, 31, 2018.

N. Jiang and J. Huang. Minimax confidence interval for off-policy evaluation and policy optimization. *arXiv preprint arXiv:2002.02081*, 2020.

N. Jiang and L. Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

N. Kallus and M. Uehara. Double reinforcement learning for efficient off-policy evaluation in markov decision processes. *Journal of Machine Learning Research*, 21(167):1–63, 2020a.

N. Kallus and M. Uehara. Statistically efficient off-policy policy gradients. In *International Conference on Machine Learning*, pages 5089–5100. PMLR, 2020b.

N. Kallus, X. Mao, K. Wang, and Z. Zhou. Doubly robust distributionally robust off-policy evaluation and learning. *arXiv preprint arXiv:2202.09667*, 2022.

M. Kato, M. Uehara, and S. Yasui. Off-policy evaluation and learning for external validity under a covariate shift. *arXiv preprint arXiv:2002.11642*, 2020.

E. S. Kim, R. S. Herbst, I. I. Wistuba, J. J. Lee, G. R. Blumenschein, A. Tsao, D. J. Stewart, M. E. Hicks, J. Erasmus, S. Gupta, et al. The battle trial: personalizing therapy for lung cancer. *Cancer discovery*, 1(1):44–53, 2011.

P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

Q. Liu, L. Li, Z. Tang, and D. Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. *arXiv preprint arXiv:1810.12429*, 2018.

Y. Liu, A. Swaminathan, A. Agarwal, and E. Brunskill. Off-policy policy gradient with stationary distribution correction. In *Uncertainty in Artificial Intelligence*, pages 1180–1190. PMLR, 2020.

P. Michel, T. Hashimoto, and G. Neubig. Distributionally robust models with parametric likelihood ratios. *arXiv preprint arXiv:2204.06340*, 2022.

W. Mo, Z. Qi, and Y. Liu. Learning optimal distributionally robust individualized treatment rules. *Journal of the American Statistical Association*, 116(534):659–674, 2021.

H. Namkoong and J. C. Duchi. Stochastic gradient methods for distributionally robust optimization with f-divergences. *Advances in neural information processing systems*, 29, 2016.

Y. Oren, S. Sagawa, T. B. Hashimoto, and P. Liang. Distributionally robust language modeling. *arXiv preprint arXiv:1909.02060*, 2019.

D. Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

C. Qin and D. Russo. On adaptivity and confounding in contextual bandit experiments. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

H. Rahimian and S. Mehrotra. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*, 2019.

S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

O. Sakhi, L. Faury, and F. Vasile. Improving offline contextual bandits with distributional robustness. *arXiv preprint arXiv:2011.06835*, 2020.

S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *COLT*, volume 2, page 5, 2009.

A. Shapiro. Distributionally robust stochastic programming. *SIAM Journal on Optimization*, 27(4): 2258–2275, 2017.

N. Si, F. Zhang, Z. Zhou, and J. Blanchet. Distributional robust batch contextual bandits. *arXiv preprint arXiv:2006.05630*, 2020.

H. Singh, S. Joshi, F. Doshi-Velez, and H. Lakkaraju. Learning under adversarial and interventional shifts. *arXiv preprint arXiv:2103.15933*, 2021.

A. Strehl, J. Langford, L. Li, and S. M. Kakade. Learning from logged implicit exploration data. *Advances in neural information processing systems*, 23, 2010.

A. Subbaswamy, P. Schulam, and S. Saria. Preventing failures due to dataset shift: Learning predictive models that transport. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3118–3127. PMLR, 2019.

A. Swaminathan and T. Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research*, 16(1):1731–1755, 2015a.

A. Swaminathan and T. Joachims. The self-normalized estimator for counterfactual learning. *advances in neural information processing systems*, 28, 2015b.

P. Thomas and E. Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148, 2016a.

P. S. Thomas and E. Brunskill. Magical policy search: Data efficient reinforcement learning with guarantees of global optimality. In *European workshop on reinforcement learning*, 2016b.

P. S. Thomas, G. Theocharous, M. Ghavamzadeh, I. Durugkar, and E. Brunskill. Predictive off-policy policy evaluation for nonstationary decision problems, with applications to digital marketing. In *Twenty-Ninth IAAI Conference*, 2017.

A. Xie, J. Harrison, and C. Finn. Deep reinforcement learning amidst lifelong non-stationarity. *arXiv preprint arXiv:2006.10701*, 2020.

M. Yang, O. Nachum, B. Dai, L. Li, and D. Schuurmans. Off-policy evaluation via the regularized lagrangian. *arXiv preprint arXiv:2007.03438*, 2020.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Section 6.3
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Section 6.3
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] Appendix Section B
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] `https://github.com/StanfordAI4HI/FactoredDRO`
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Appendix Section A.2
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] 95% confidence intervals over multiple runs
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Appendix Section A.2
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] Appendix Section A.2
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

**Algorithm 1:** Double DRO policy optimization

**Data:** Dataset $\mathcal{D} = \{X_i, A_i, R_i\}$, $\pi_0$, $\delta_c$ (reward/cost KL radius), $\delta_x$ (context KL radius)

**Result:** $\hat{\pi}^*_{dro}, \hat{V}^*, \hat{\alpha}^*_x, \hat{\alpha}^*_c (n{\times}k$ values)

`// First DRO (rewards) (this step only needs to be done once):`

1 **if** *Binning* **then**

2 $\quad R_{X,A} \leftarrow$ Bin values of $R_i$ by unique $(X, A) \in \mathcal{X}{\times}\mathcal{A}$ pairs.;

3 $\quad$ Solve for $\hat{Q}_{X,A}$ for every unique $(X, A)$ using Eq. 5;

4 **else**

$\quad$ `// Using function approximation`

5 $\quad \{\mathcal{F}\}_{i=1}^m \leftarrow$ Fit approximators to evaluate $\hat{\mathbb{E}}[\exp(-R/\alpha_c)]$ using the method in Sec. 5;

6 $\quad$ Solve for $\hat{Q}_{X_i,a}$ for every $X_i \in \mathcal{D}, a \in \mathcal{A}$ using Eq. 12 `//` $n{\times}k$ `total` $\hat{Q}_{X_i,a}$ `values`

`// Policy Evaluation and Learning`

7 $\pi \leftarrow$ initial $\pi$;

8 **while** $\pi$ *not converged* **do**

9 $\quad J(\alpha_x) = \alpha_x \log \frac{1}{n} \sum_i \exp\left(\frac{-\sum_{a\in\mathcal{A}} \pi(a|X_i)\hat{Q}_{X_i,a}}{\alpha_x}\right) + \alpha_x \delta_x$;

10 $\quad \hat{V}_\pi = -\min_{\alpha_x \geq 0} J(\alpha_x)$, $\hat{\alpha}^*_x = \text{argmin}_{\alpha_x \geq 0} J(\alpha_x)$`// Policy Evaluation`

11 $\quad$ Update($\pi$) (ex. for differentiable $\pi$, fix $\hat{\alpha}^*_x$ and perform gradient ascent.)

`// Note the optimization problem on lines 3, 6, and 10 are all convex and can be solved using gradient descent, bisection search, etc.`

# A Algorithm and Additional Experimental Details

## A.1 Complete Algorithm

We give the pseudocode of our algorithm in Algorithm 1

## A.2 Experiment Details

In this section we give additional details for our experiments not included in the main text. Each experiment is run on an internal cluster machine with 4CPU and 16GB of memory. A single run of each experiment runs between 10 minutes and 2 hours depending on dataset size.

### A.2.1 Simulated Experiments

As mentioned in the main text, our simulated setting has 4 cgs, where cgs 1 and 3 ($cg_{1,3}$), and cgs 2 and 4, ($cg_{2,4}$) are the same. In addition to the cgs, there is also 4 other context features, $X_i^f$ randomly sampled from a uniform distribution. The first 2 features are randomly sampled from Uniform($[0, 0.4]$) while the second 2 features are randomly sampled from Uniform($[0.3, 0.7]$). There are 4 actions and each $(cg, a)$ pair has a set of true reward generating parameters $\phi^*_{(1,3),a}$ and $\phi^*_{(2,4),a}$ of length 4, and the observed reward is $R_i =$ Bernoulli($\phi^{*T}_{X_i,a} X_i^f$). The $\phi^*$'s are:

$$\phi^*_{(1,3),a} = \begin{cases} a_0: & [0.5, 0.0, 0.5, 0.0] \\ a_1: & [1.0, 0.0, 0.0, 0.0] \\ a_2: & [0.0, 0.0, 1.0, 0.0] \\ a_3: & [0.1, 0.1, 0.1, 0.1] \end{cases}, \quad \phi^*_{(2,4),a} = \begin{cases} a_0: & [0.5, 0.0, 0.5, 0.0] \\ a_1: & [0.0, 0.0, 0.0, 1.0] \\ a_2: & [0.0, 1.0, 0.0, 0.0] \\ a_3: & [0.1, 0.1, 0.1, 0.1] \end{cases}$$

We use a misspecified context for the experiments and the full context used is the $cg$ concatenated with the other features $X_i = [cg, X_i^f]$.

For the Gaussian experiments, we use the same setup, but instead we first add gaussian noise to the rewards and then clip them to $[0, 1]$: $R_i = \text{clip}(\phi^{*T}_{X_i,a} X_i^f + \mathcal{N}(0, 0.1), \min = 0, \max = 1)$. For the plot in the main text, the training dataset population split is as listed in Table 1 while the test dataset population split between the coarse groups was shifted to $(0.4, 0.1, 0.4, 01)$.

Reported results are averaged over 10 runs where we average over the randomization of the dataset split, the initial policy parameters, and the gradient descent procedure.

### A.2.2 Voting Experiments

The voting dataset of Gerber et al. [2008] is licensed under the Creative Commons Attribution-Non Commercial-No Derivatives 3.0 license. The dataset contains 180002 datapoints, each a voter in a different household across the state of Michigan. The researchers designed one control, and 4 treatment actions that involved mailing the selected individuals a letter ahead of the 2006 Michigan primary election. The actions are:

- Nothing (control): No letter is sent
- Civic: A letter with "Do your civic duty"
- Hawthorne: A letter with "You are being studied"
- Self-History: A letter with the voter's past voting participation record as well as that of the other members of the household. The letter also mentioned a follow-up letter will be sent with the household's updated voting participation after the election
- Neighbors-History: A letter with the voting participation records of the individual, the other members of the individual's household, as well as the neighbors. The letter also mentioned a follow-up letter will be sent after the election with everyone's updated participation, so the individual's participation will be made known among the neighbors.

The data collection policy randomly sampled actions with probability $\frac{5}{9}$ for the control action and $\frac{1}{9}$ for the other actions. We use the binary indicator of voting outcomes as the reward. The neighbors action was the best action for the population, with participants under that action voting at a rate of around $10\%$ higher than those under the control action. However all actions except the control require effort and cost in the form of printing and mailing a letter. Therefore, for each non-control action, we induce an "effort" cost by randomly flipping $0.09n_a$ of the positive outcomes to 0, where $n_a$ for each action is the number of participants that experienced that action. This lowers the average reward for all non-control action by $0.09$. We choose this cost to induce heterogeneous effects within the dataset.

For the experiment in the main text we take data from one city to use to train a policy deployed on other cities. We choose the train and deployment cities to have a large context shift. For simplicity we approximate a large context shift by choosing one context feature to examine, the 'p2004' feature which is an indicator if the individual voted in the 2004 primary election. We chose the one city with the largest average 'p2004' to be the training city and the 35 cities with the smallest 'p2004' to be the test cities. For computation efficiency and speed, we use a random sample of $25\%$ of the training data.

We additionally provide in t appendix a experiment similar to that of the artificial shift in the Scene and Simulated datasets that uses all the data. We set the 'coarse group' using this p2004 features. From the subgroup that did vote in the 2004 primary election, we partition 0.8 of the dataset into the training set and the remaining 0.2 into the test set. Of the subgroup that did not vote, we partition 0.2 of the dataset into the training set and the remaining 0.8 into the test set. This gives the coarse group composition of (0.75, 0.25) in the training set and (0.15, 0.85) in the test set. While we choose this split, different splits where there is a shift between the training distribution and test distribution also work. We use the features of household size and gender as contexts. We illustrate the more difficult case where some variables, such as the coarse group variable (whether the participants voted in the 2004 primaries), are not available at test time.

We report results averaged across 30 runs.

### A.2.3 Scene Experiments

The Scene classification dataset from the LibSVM library Chang and Lin [2011] is a multiclass-supervised learning dataset. Datsets in the LibSVM library are licensed under the BSD 3-Clause. The features are processed features of the image and the labels correspond to various landscapes (such as 'mountains', etc). As mentioned in the main text, we use a supervised-to-bandit conversion method where we assume we do not observe the correct labels directly and instead each label is an action and selecting a correct action/label gives positive reward (+1) while a wrong action/label gives no

reward (0). We assume we only observe 1 action per datapoint and we randomly sample the actions we observe. We use the true labels to partition the dataset into coarse groups, we place datapoints that contain labels 1, 2, and 3 into one group ($cg_0$) and the others into another group ($cg_1$). As in the voting dataset we induce a train test split such that 0.1 of the datapoints in $cg_0$ and 0.6 of the datapoints in $cg_1$ are in the train dataset with the rest in the test dataset. As in the voting dataset, we assume this feature will not be available at test time.

Due the small dataset size, we use orthonormal random projection, $P \in \mathbb{R}^{d \times d_p}$ to project the dataset onto 3 dimensions. Let $d = 294$ be the original context feature length and $d_p = 3$ be the desired projected length. To create our random orthonormal projection matrix of size $\mathbb{R}^{d \times d_p}$ we first sample a matrix of size $\mathbb{R}^{d \times d_p}$ where each item is sampled from Uniform($[0, 1]$). We then take the singular value decomposition and calculate the left and right singular values $U \in \mathbb{R}^{d \times d_p}$ and $V \in \mathbb{R}^{d_p \times d_p}$. We then calculate: $P = UV$.

Our reported results are averaged over 30 runs where each run randomizes across the datapoints in the train-test splits, the initial policy parameters, the gradient descent steps during policy optimization, and the random generated random projection matrix.

#### A.2.4 Reward Shift Experiment

Our simple environment for reward shift has two unique contexts ($X_0$, $X_1$), with $X_0 = [0, 1]$ and $X_1 = [1, 0]$, and 3 actions. We assume the rewards are binary, but with different support for each ($X$, $a$) pair. The reward generation process is defined by a tuple of 3 values $(r_0, r_1, p)$ with $r_1 \geq r_0$ and $R_i = (r_1 - r_0)$Bernoulli(p) $+ r_0$.

$$X_0 = \begin{cases} a_0: & (0.0, 1.0, p) \\ a_1: & (0.3, 0.5, p) \\ a_2: & (0.1, 0.15, p) \end{cases}, \quad X_1 = \begin{cases} a_0: & (0.0, 1.0, p) \\ a_1: & (0.0, 1.0, p) \\ a_2: & (0.1, 0.15, p) \end{cases}$$

We induce a reward shift by setting $p = 0.7$ when generating the training data and $p = 0.5$ for the test data.

### A.3 Additional Results: Different dataset splits and hyperparameters

In Figure 2 we give some results from different dataset splits and hyperparameters. In Figure 2 (a) we give the plot similar to plots (b-e) in Figure 1 of the main text for the Simulated Bernoulli experiment. In these plots, we fix the training and test distribution and examine different values of the $\delta/\delta_x$ hyperparameter. We see FDRO can learn more robust policies that work well in the test set under context shift. In Figure 2(b) and (c), we show the result of running different dataset splits. Figure 2(b) considers a different training and test context distributions. It considers a shift of group distribution from $(0.63, 0.17)$ in the training set to $(0.3, 0.7)$ in the test set. Figure 2(b) considers a different number of groups. As opposed to 2 coarse groups, we now partition the dataset into 3 groups (with items with labels 0 and 1 in $cg_0$, the remaining items with labels 2, 3 in $cg_1$, and all the remaining items in $cg_2$). We see in both cases FDRO is able to learn more robust policies that work well on the test context distribution compared to baselines. In Figure 2(d) and (e) we test out different hyperparameters to generate plots similar to that of Figure 1(a) of the main text. Like in the main text plot, we fix the $\delta_x$ and training distribution and generate test datasets with different shifted group distributions. Also, as in the main text, while we fix $\delta_x$ for FDRO, we optimize the uncertainty parameter at each test distribution for the baselines. We plot the performance of each algorithm as the performance gain in the test dataset over baseline-IS. We see that for different $\delta_x$ hyperparameters, FDRO can still perform better than baselines when there is shift. Setting a lower value of $\delta_x$ leads to a smaller decrease in performance when there is very little shift; however, the benefits at large shifts is also smaller.

### A.4 Additional Discussion

We provide some additional points of discussion. In Figure 3a we plot performance at different values of $\delta_c$ for the Simulated Bernoulli case and show performance to be similar across parameters.

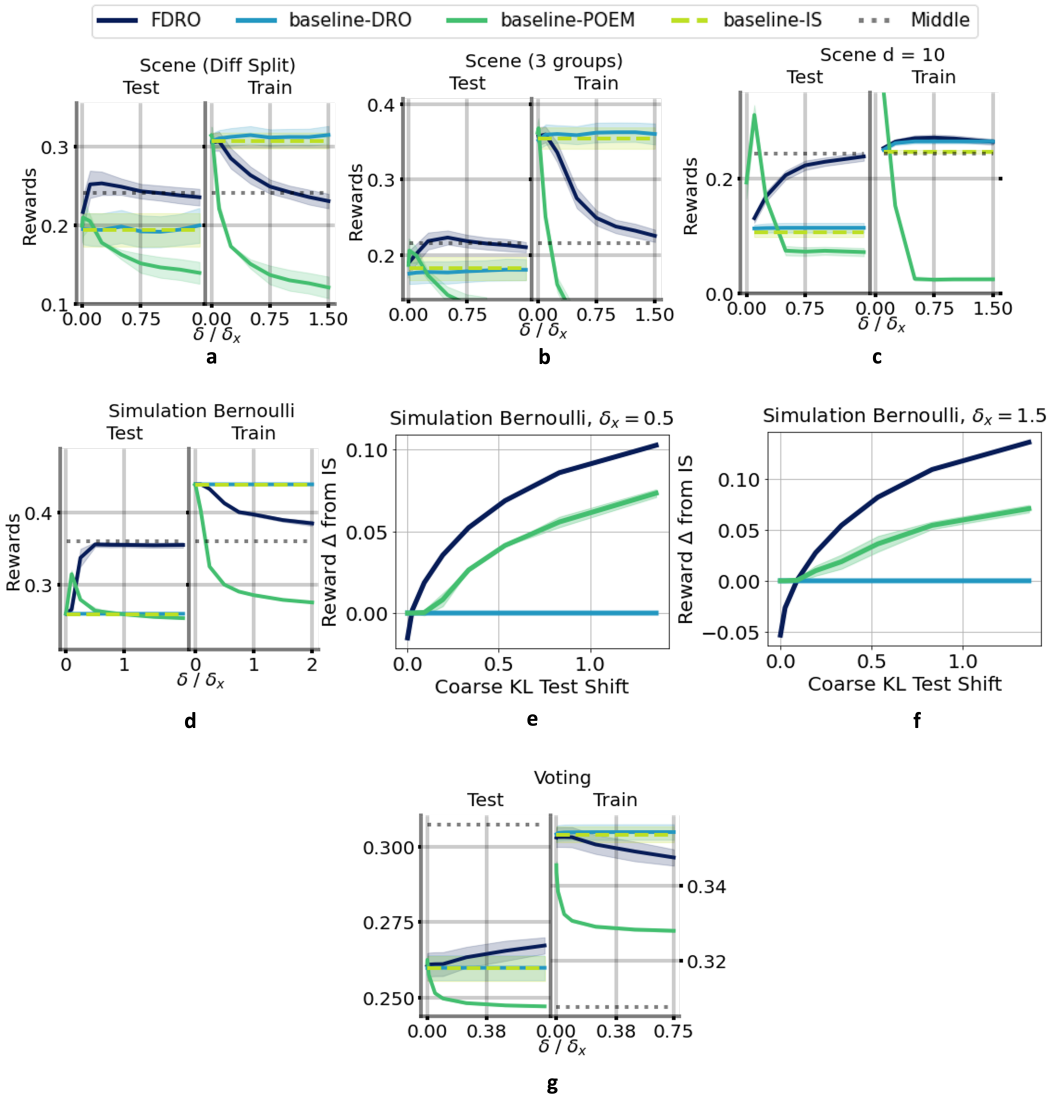


Figure 2: In these plots, we demonstrate FDRO can perform well even under different hyperparameter and dataset configurations. The dotted line 'Middle' serves as a constant horizontal reference across the two panels in plots (a-d, g). All plots are plotted with 95% confidence intervals shaded. Reported results and confidence intervals are averaged over 10 runs in all simulated domains and 30 runs in Scene and Voting. **Top Row:** (a-c) We fix the train and test distributions and plot the performance at different $\delta/\delta_x$ hyperparameters in the Scene domain. The plots show results from different configurations than that in the main text, including different dataset splits (a), different number of groups (b), and different number of features for random projection (c). We see FDRO is able to learn more robust policies that does well in the test distribution in all cases. **Middle Row:** (d-f) We give results for different configurations in the Simulated Bernoulli dataset. In (d) we give the two panel plot with the parameters described in the main text. In (e,f) We vary the shift in the test distribution and plot results as the performance gain over baseline-IS. **Bottom Row:** We give a plot for the voting dataset with an artifically induced shift.

Figure 3: We give additional results on hyperparameter sensitivity. The dotted line 'Middle' serves as a constant horizontal reference across the two panels in plots (a) we show sensitivity to $\delta_c$ by showing performance Simulation Bernoulli at multiple values of $\delta_c$ (b) The extracted worst case population of each $cg$ of our FDRO for different $\delta_x$. (c) We plot the reward for each action for the joint formulation (baseline-DRO) at different values of the dual parameter $\alpha$. As described in the text, the order between the actions never changes.

### A.4.1 Examining the Worst Case Context

The worst case distribution for DRO formulation with KL distance bounds is an softmax scaled weighting of the datapoints [Hu and Hong, 2013]. with a temperature determined by optimal dual variable $\alpha^*$. If we have discrete or easily discretized context features, we can examine the worst case distribution of that feature. We plot the worst case distributions of the $cg$ context feature of FDRO in Figure 3b in the Bernoulli simulated environment. For each $\delta$ value, the end user can examine the estimated performance on the training distribution as well as the worst-case distribution of some of the features to decide if the policy is reasonable.

Formally, from Hu et al. [2018], the worst case distribution is an exponentially/softmax reweighing of the points:

$$p_i = \frac{\exp((-\sum_{a\in\mathcal{A}} \hat{\pi}^*(a|X_i)\hat{Q}_{X,A})/\alpha_x^*)}{\sum_i \exp((-\sum_{a\in\mathcal{A}} \hat{\pi}^*(a|X_i)\hat{Q}_{X,A})/\alpha_x^*)} \tag{14}$$

where $\alpha_x^*$ is the optimal dual variable of $\hat{\pi}^*$ in Equation 7. Consider the $j^{th}$ feature in the context and assume this feature is either discrete, or easily discretized. We can calculate the worst case distribution over values the feature takes on as:

$$P(X[j] = x) = \sum p_i \mathbb{1}[X_i[j] = x] \tag{15}$$

### A.4.2 Utilizing unavailable features:

In two of the settings - voting and scene, we assumed there were context features in the train set that are unavailable at deployment. This situation occurs when those features are unavailable at deployment, or when they are sensitive features that should not be used for fairness considerations. Even without using it for the policy, FDRO is still able to better utilize this information in the first DRO step when calculating the worst case rewards $\hat{Q}_{X,A}$.

### A.4.3 Additional Discussion: Equivalence of Solutions

As mentioned in the main text, we can compare baseline-DRO and FDRO by examining the KL-divergence chain rule decomposition:

$$D_{kl}(p_0(R,X)||p(R,X)) = D_{kl}(p_0(X)||p(X)) + E_{P_{x_0}}[D_{kl}(p_0(R|X)||p(R|X))]$$

Recall $D_{kl}(p_0(R,X)||p(R,X)) = \delta$, $D_{kl}(p_0(X)||p(X)) = \delta_x$, $D_{kl}(p_0(R|X)||p(R|X)) = \delta_c$. This informs us that we have the following relation between the baseline-DRO parameter $\delta$ and the FDRO ones $\delta_x$ and $\delta_c$:

$$\delta = \delta_x + \sum_x P_{x_0}\delta_c$$

.

From this we can see that if we allow $\delta_c$ to vary with context $X$ and action $A$, we can set parameters of $\delta_x$ and $\delta_c(X, A)$ such that the FDRO will find the same solution as baseline-DRO. However this increase in parameter may be difficult for the algorithm designer. Therefore we expect FRDO to be particularly useful when there are independent shifts in the reward and context distributions, which is a common setting– such as shifting locations (for hospitals or social welfare solutions).

We give a detailed example of what the worst case shift in rewards look like for a single action, and how differs between the factored formulation (FDRO) and the joint formulation (baseline-DRO) Figure 4. In this example, we consider a simple two context, discrete reward setting and examine the worst case distributions for a single action. We compare with setting the single joint parameter to $\delta = 0.1$ and setting the context shift parameter of the factored formulation to $\delta_x = 0.1$, while allowing for very little reward shift $\delta_c = 0$. We can see the resulting worst case joint distribution is different under the two formulations. We can additionally see for the factored formulation to achieve the equivalent worst case distribution as the joint formulation, we need to set different reward shift radius parameters ($\delta_c$) for context $X_0$ and context $X_1$.

### A.4.4 Additional Details: Why Joint DRO May Not Work for Binary Rewards

In the main text we mentioned the formulation from prior work [Si et al., 2020] that jointly considers the context and reward distribution shift may lead to a degenerate result of finding the best policy on the training context distribution under binary reward feedback. This empirical observation is similar to a proven phenomena from the supervised learning literature in distributionally robust optimization [Hu et al., 2018] under continuous inputs. The intuition behind this phenomena is that DRO can be seen as weighing the datapoints based on the incurred loss/or incurred cost, where cost are negative rewards. Under binary reward feedback, the lower reward points for each context action pair are all upweighted equally and we empirically observe this leads to a learned policy equivalent to that optimal for the training context distribution. In this section we provide a simple example and additional details around this observation.

Consider a simplified version of the Simulated Bernoulli setting described above, however in this setting, let $cg$ be the only context feature. Additionally in this case there are only 2 actions $\{a_0, a_1\}$. $a_0$ achieves medium reward for all contexts (Bernoulli(0.4)), while $a_1$ achieves high reward for $cg_{1,3}$ (Bernoulli(0.5)) and low reward for $cg_{2,4}$ (Bernoulli(0.3)). Consider optimizing for the parameters, $\theta$, of a linear contextual bandit policy of the form $\text{argmax}_a \, \phi(x, a)^T \theta$ where $\phi(x, a)$ concatenates the context with a one hot encoding of the action. This representation attempts to fit a linear dependence on context when the relationship is very nonlinear. Due to this misspecification, this ends up being equivalent to optimizing a non-contextual bandit which chooses the single best action. Consider the case where $cg_{1,3}$ comprises a large majority (0.8) of the training context distribution, $P_{x,train}$. In this case, action $a_0$ is optimal for $P_{x,train}$. If we are certain only the context distribution will shift, as we expand the uncertainty set around $P_{x,train}$, we expect best worst-case action to eventually shift to the action that is good for both groups, $a_1$. However due to binary rewards, we empirically see the joint formulation upweights the lower rewards for each action equally. We demonstrate this in Figure 3(c), where we examine the re-weighted rewards for each action in the joint formulation, which the joint method uses with weighted importance sampling to optimize the policy. We look at the reward values under different values of the dual variable $\alpha$. Notice that for decreasing $\alpha$, while the re-weighted rewards are lower, the order of the benefit of the rewards does not change. Therefore in this example, there is no value of $\delta$ that will result in a policy different from that optimal for the training context distribution. Our method which decomposes context and reward generation shift mitigates this issue.

## B Proofs

Throughout the proofs, we will define a new variable $C$, or costs, for convenience, with $C = -R$ and $C_i = -R_i$ for every $i$.

### B.1 The Convexity and Derivatives of the Policy Evaluation steps

We first prove the convexity of the objectives inside the minimization of equations 6 and 7 and give their derivatives as mentioned in the main text. We first give this intermediate result:

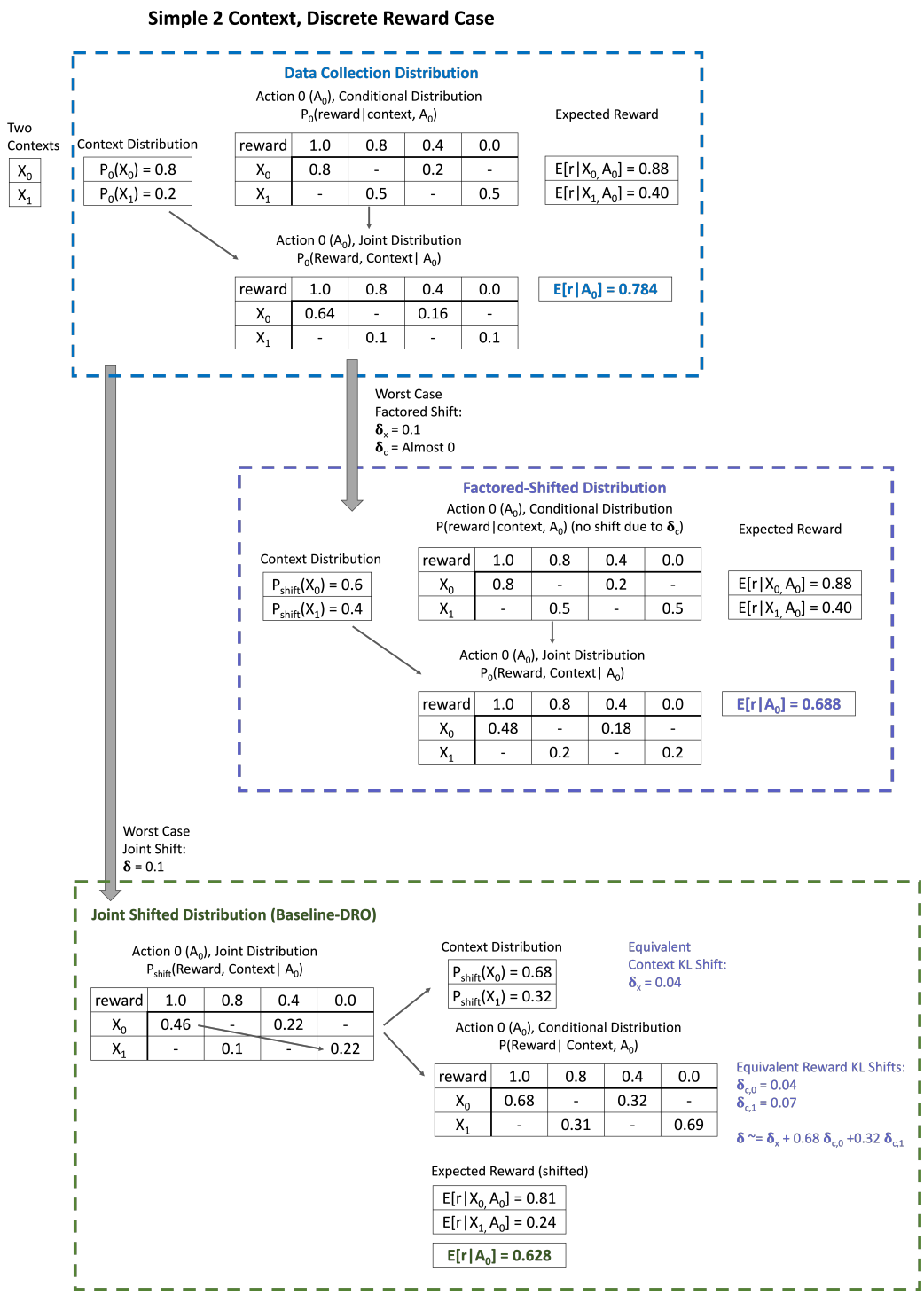


Figure 4: A detailed example of what the worst case shift in rewards look like for a single action. We choose a simple two context, discrete reward setting. We see that even under the same amount of KL shift ($\delta = 0.1$), the difference in results under the factored approach and the joint formulations are different. We also give the equivalent parameters for FDRO to be equivalent to the solutions of the joint formulation.

**Lemma 1** (Convexity of DRO formulation)**.** *Consider a set of $n$ values, $\{B_i\}$ for $i = 1, ..., n$ and function*

$$f(\alpha) = \alpha \log \left( \sum_i \frac{1}{n} \exp \left( \frac{B_i}{\alpha} \right) \right) + \alpha\delta$$

*Then $f(\alpha)$ is convex in $\alpha$ for $\alpha > 0$*

*Proof.*

$$\begin{aligned} f(\alpha) &= \alpha \log \left( \sum_i \frac{1}{n} \exp \left( \frac{B_i}{\alpha} \right) \right) + \alpha\delta \\ &= \alpha \log \left( \sum_i \exp \left( \frac{B_i}{\alpha} \right) \right) + \alpha\delta + \alpha \log \left( \frac{1}{n} \right) \end{aligned}$$

First recall a classic result from convex optimization that states a function $f(x, y) = yg(\frac{x}{y})$ is convex for $y > 0$ if an only if $g$ is convex. [Boyd et al., 2004]

Note the expression $g(B) = \log \left( \sum_i \exp \left( B_i \right) \right)$ takes on the form of a log-sum-exp which is convex. Therefore the function $f(B, \alpha) = \alpha g \left( \frac{B}{\alpha} \right) = \alpha \log \sum_i \left( \exp \left( \frac{B_i}{\alpha} \right) \right)$ is convex, and consequently convex in $\alpha$ when $\alpha > 0$.

The linear combination of convex function is still convex, so we have the desired result. □

Now we return to proving the convexity of equations 6 and 7. We first reproduce them here with new intermediate variables $\hat{Z}_{X,A}$ and $\hat{J}(\pi, \alpha_x)$ that represent the objective inside the optimization:

$$\hat{Z}_{X,A}(\alpha_c) = \alpha_c \log \left( \frac{1}{m_{xa}} \sum_{R_j \in \mathcal{R}_{X,A}} \left[ \exp \left( \frac{-R_j}{\alpha_c} \right) \right] \right) + \alpha_c \delta_c \tag{16}$$

$$\hat{J}(\pi, \alpha_x) = \alpha_x \log \sum_i \frac{1}{n} \exp \left( \frac{-\sum_{a \in \mathcal{A}} \pi(a|X_i) \hat{Q}_{X_i,a}}{\alpha_x} \right) + \alpha_x \delta_x \tag{17}$$

With these variables, we have:

$$\hat{Q}_{X,A} = -\min_{\alpha_c \geq 0} \left\{ \hat{Z}_{X,A}(\alpha_c) \right\} \quad \text{and } \hat{V}(\pi) = -\min_{\alpha_x \geq 0} \left\{ \hat{J}(\pi, \alpha_x) \right\} \tag{18}$$

We can directly apply Lemma 1 to $\hat{Z}_{X,A}(\alpha_c)$ and $\hat{J}(\pi, \alpha_x)$ as defined above with $B_j = R_j$ and $B_j = \sum_{a \in \mathcal{A}} \pi(a|X_j) \hat{Q}_{X_j,a}$ respectively to obtain both $\hat{Z}_{X,A}(\alpha_c)$ and $\hat{J}(\pi, \alpha_x)$ are convex.

We also give the gradient of $\hat{Z}_{X,A}(\alpha_c)$ and $\hat{J}(\pi, \alpha_x)$:

**Remark 1** (Gradient of DRO formulation)**.** *The gradient $\frac{\partial f(\alpha)}{\partial \alpha}$ of $f(\alpha)$ given in Lemma 1 is:*

$$\frac{\partial f(\alpha)}{\partial \alpha} = -\frac{\sum_i B_i \exp \left( \frac{B_i}{\alpha} \right)}{\alpha \sum_i \exp \left( \frac{B_i}{\alpha} \right)} + \log \left( \sum_i \exp \left( \frac{B_i}{\alpha} \right) \right) + \delta$$

We now prove the main lemmas and theorems from the main text.

### B.2 Collection of Useful Results From Prior Work

We first collect results from prior work we will use for proofs of Theorems 1, 2 and 3.

We will utilize Hoeffding's inequality:

**Lemma 2** (Hoeffding's Inequality)**.** *Let $Z_1, ...Z_n$ be independent bounded random variables with $Z_i \in [a, b]$ for all i, where $-\infty < a \leq b < \infty$. Then:*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t\right) \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right) \tag{19}$$

*and*

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \leq t\right) \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right) \tag{20}$$

*for all $t \geq 0$*

**Distributionally Robust Optimization with KL divergence [Hu and Hong, 2013]**

We also use a result from the DRO literature in stochastic optimization from Hu and Hong [2013]. The distributionally robust optimization formulation from stochastic optimization literature considers optimizing the model variables $\theta \in \Theta$ (for example, parameters of a logisitic regression model) to achieve the best worst-case future performance (ex. performance on the test set). The goal is to use the observed random values $\xi$ (ex. (input, label) pairs of the training data) to optimize a cost function $\mathcal{L}(\theta, \xi)$ (ex.classification loss) when evaluated in the unknown test distribution. The test distribution is assumed to be within an uncertainty set, $\mathbb{P}_{\xi_0}$. In this work, we focus on ambiguity sets defined as a bound on the Kullback-Leibler (KL) divergence from the data generation distribution $P_{\xi_0}$. The DRO optimization problem is:

$$\theta^* = \arg\min_{\theta\in\Theta} \max_{P_\xi \in \mathbb{P}_\xi} \mathbb{E}_{P_\xi}[\mathcal{L}(\theta, \xi)], \text{ where } \mathbb{P}_\xi = \{P_\xi : D_{kl}(P_\xi || P_0) \leq \delta\}$$

The inner maximization over $P_\xi$ is difficult to solve directly as it is a constrained optimization problem in probability space. However Hu and Hong [2013] show through strong duality that this can be transformed into a much easier unconstrained convex minimization over a scalar:

**Lemma 3** (Theorem 1 from Hu and Hong [2013]:)**.** *Assume $\mathcal{L}(\theta, \xi)$ takes on a finite set of values or is bounded almost surely. Then through strong duality, the inner maximization is equivalent to solving the following optimization problem:*

$$\max_{P_\xi \in \mathbb{P}_\xi} \mathbb{E}_{P_\xi}[\mathcal{L}(\theta, \xi)] = \min_{\alpha \geq 0} \alpha \log \mathbb{E}_{P_0}[\exp(\mathcal{L}(\theta, \xi)/\alpha)] + \alpha\delta$$

**Uniform Convergence of Stochastic Optimization [Shalev-Shwartz et al., 2009]**

We will also make use of the following uniform convergence result from Shalev-Shwartz et al. [2009] of the optimization of convex objectives.

Consider the following stochastic minimization problem:

$$w^* = \operatorname*{argmin}_{w\in\mathcal{W}} F(w) \tag{21}$$

Where $f(w; Z)$ denote the loss objective and $F(w) = \mathbb{E}_Z[f(w, Z)]$. Define the empirical objective and minimizer as $\hat{F}(w) = \frac{1}{n}\sum_{i=1}^{n} f(w; z_i)$ and $\hat{w} = \text{argmin}_w \hat{F}(w)$.

Then the following result guarantees uniform convergence of $F(w)$ over $w$:

**Lemma 4** (Uniform Convergence of Stochastic Convex Optimization (Theorem 5 of Shalev-Shwartz et al. [2009]))**.** *Let the following conditions hold:*

1. *$f(w, z)$ is convex in $w$*
2. *$\mathcal{W} \subset \mathbb{R}^d$ is bounded by $B$*
3. *$f(w, z)$ is L-Lipschitz w.r.t $w$*

*Then with probability of at least $1 - \delta$, for all $w \in \mathcal{W}$:*

$$\left|F(w) - \hat{F}(w)\right| \leq \mathcal{O}\left(LB\sqrt{\frac{d\log(n)\log(d/\delta)}{n}}\right) \tag{22}$$

We also use Si et al. [2020]'s result on the upper bound on $\alpha$:

**Lemma 5** (Lemma 11 of Si et al. [2020])**.** *Let variable B be bounded between 0 and M, then the optimal $\alpha$ in the objective:*

$$\inf_{\alpha \geq 0} \alpha \log \mathbb{E}_{P_B}[\exp(B/\alpha)] + \alpha\delta$$

*is upper bounded: $\alpha^* \leq M/\delta$.*

### B.3 Proof of Theorem 1

We first restate Theorem 1:

**Theorem 1** (Strong Duality)**.** *The optimization problem in equation 2 is equivalent to solving:*

$$V(\pi) = -\min_{\alpha_x > 0} \left\{ \alpha_x \log \mathbb{E}_{X \sim P_{x_0}} \left[ \exp\left( \frac{\mathbb{E}_\pi\left[-Q_{X,A}\right]}{\alpha_x} \right) + \alpha_x \delta_x \right] \right\}, \tag{4}$$

$$\textit{where } Q_{X,A} = -\min_{\alpha_c > 0} \left\{ \alpha_c \log \mathbb{E}_{R \sim P_{r_0|X,A}} \left[ \exp\left( \frac{-R}{\alpha_c} \right) \right] + \alpha_c \delta_c \right\}. \tag{5}$$

The proof of Theorem 1 directly follows from applying Lemma 3 twice. Notice that because rewards, R, are bounded $[0, R_M]$, the assumption of Lemma 3 is satisfied for $Q_{X,A}$. Similarly, $Q_{X,A}$ are also bounded $[0, R_M]$ so the assumption is also satisfied for $V(\pi)$.

### B.4 Policy Evaluation and Learning convergence

We first prove the probability of convergence for $\hat{Q}_{X,A}$ for each $(X, A)$ pair separately. We then apply the union bound to bound the probability they hold jointly for all $(X, A)$ pairs. We then use this result along with local Lipschitz continuity to prove the rest of the results.

We start with proving PAC uniform convergence for $\hat{Q}_{X,A}$:

**Lemma 6** (PAC convergence of $Q_{X,A}$ (first DRO layer))**.** *Given a $(X, A)$ pair, when $n \geq \frac{(\underline{p}_0(x,a))^2}{2\log(2/\delta)}$, the following result holds with probability at least $1 - \delta$ and all $\alpha_c \in [\underline{\alpha}_c/2, 2\bar{\alpha}_c]$:*

$$\left| Q_{X,A} - \hat{Q}_{X,A} \right| \leq \mathcal{O}\left( c\sqrt{\frac{\log(n)\log(2/\delta)}{n}} \right) \tag{23}$$

*where*

$$c = \frac{32(R_{max})^3}{\delta_c^2 \underline{\alpha}_c^2 \exp(-R_{max}/\underline{\alpha}_c)\underline{p}_0(x,a)} \tag{24}$$

*and $\underline{p}_0(x, a)$ is the lowest probability $(X, A)$ pair: $\underline{p}_0(x, a) = \min_{(x,a)} p_0(x, a)$, and $\underline{\alpha}_c$ is the minimum value for $\alpha_c$ and satisfies $\underline{\alpha}_c > 0$.*

*Proof* We first rewrite the objective in terms the total number of samples, $n$, across all state, action pairs, as opposed to the number of samples $m_{xa}$ for the $(X, A)$ pair:

$$\hat{Q}_{X,A} = -\min_{\alpha_c > 0} \left\{ \alpha_c \log \left( \frac{1}{m_{xa}} \sum_{R_j \in \mathcal{R}_{X,A}} \left[ \exp\left( \frac{-R_j}{\alpha_c} \right) \right] \right) + \alpha_c \delta_c \right\} \tag{25}$$

$$= -\min_{\alpha_c > 0} \left\{ \alpha_c \log \left( \frac{1}{p_0(x,a)n} \sum_{i=1}^{n} \left[ \mathbb{1}_{xa} \exp\left( \frac{-R_i}{\alpha_c} \right) \right] \right) + \alpha_c \delta_c \right\} \tag{26}$$

Where $\mathbb{1}_{xa}$ is the indicator function that is 1 if both $X_i = x$ and $A_i = a$, and 0 otherwise. We use the fact $\sum_{R_j \in \mathcal{R}_{X,A}} \left[ \exp\left( \frac{-R_j}{\alpha_c} \right) \right] = \sum_{i=1}^{n} \left[ \mathbb{1}_{xa} \exp\left( \frac{-R_i}{\alpha_c} \right) \right]$

Consider the value of $Q_{X,A}$ for a single $(X, A)$ pair. Consider the function $F_{xa}(\alpha_c) = \mathbb{E}[\mathbb{1}_{xa} \exp(C_i/\alpha_c)]$ with $f_{xa}(\alpha_c, C_i) = \mathbb{1}_{xa} \exp(C_i/\alpha_c)$. Then $f_{xa}(\alpha_c, C_i)$ is upper bounded by 1

and lower bounded by 0. Additionally note that $f_{xa}(\alpha_c, C_i)$ is locally L-Lipschitz with respect to $\alpha_c$ on $[\underline{\alpha}_c/2, 2\bar{\alpha}_c]$:

$$\frac{\partial f_{xa}(\alpha_c)}{\alpha_c} = -\frac{\mathbb{1}_{xa} C_i \exp(C_i/\alpha_c)}{\alpha_c^2} \tag{27}$$

$$\left|\frac{\partial f_{xa}(\alpha_c)}{\alpha_c}\right| \leq \frac{4R_{max}}{\underline{\alpha}_c^2} \tag{28}$$

Additionally, from Lemma 5, we have $\bar{\alpha}_c = R_{max}/\delta_c$. Therefore, we can directly apply Lemma 4 to $F_{xa}(\alpha_c)$ and $\hat{F}_{xa} = \frac{1}{n}\sum_{i=1}^n f_{xa}(\alpha_c, C_i)$ with $d = 1$, $B = 2\bar{\alpha}_c = 2R_{max}/\delta_c$ and $L = 4R_{max}/\underline{\alpha}_c^2$ and conclude uniform convergence in $\alpha_c$ on the range $[\underline{\alpha}_c/2, 2\bar{\alpha_c}]$: with probability $1 - \delta'$:

$$\sup_{\alpha_c \in [\underline{\alpha}_c/2, 2\bar{\alpha_c}]} \left|F_{xa}(\alpha_c) - \hat{F}_{xa}(\alpha_c)\right| \leq \mathcal{O}\left(\frac{8(R_{max})^2}{\delta_c \underline{\alpha}_c^2}\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right) \tag{29}$$

Now define, $G_{xa}(\alpha_c) = \alpha_c \log\left(\frac{F_{xa}(\alpha_c)}{p_0(x,a)}\right) + \alpha_c\delta_c$. Note that $Q_{X,A} = \min_{\alpha_c \geq 0} G_{xa}(\alpha_c)$

$$\sup_{\alpha_c}\left|G_{xa}(\alpha_c) - \hat{G}_{xa}(\alpha_c)\right| = \sup_{\alpha_c}\left|\alpha_c(\log(F(\alpha_c)) - \log(\hat{F}(\alpha_c)))\right| \tag{30}$$

$$\leq 2\bar{\alpha}_c L \sup_{\alpha_c}\left|F(\alpha_c) - \hat{F}(\alpha_c)\right| \tag{31}$$

$$\leq \mathcal{O}\left(2\bar{\alpha}_c \frac{8(R_{max})^2 L}{\delta_c \underline{\alpha}_c^2}\sqrt{\frac{\log(n)\log(\delta/2)}{n}}\right) \tag{32}$$

Where the second line utilizes the fact that log is locally L-Lipschitz and:

$$L = \frac{1}{\min\{\inf_{\alpha_c} \hat{F}(\alpha_c), \inf_{\alpha_c} F(\alpha_c)\}}$$

We now derive the quantity $L$. First note the expression $\exp(C_i/\alpha_c) \geq \exp(-R_{max}/\underline{\alpha}_c)$. With this we can write:

$$F_{xa}(\alpha_c) = p_0(x,a)\mathbb{E}[\exp(C_i/\alpha_c)|X_i = x, A_i = a] \tag{33}$$

$$\geq p_0(x,a)\exp(-R_{max}/\underline{\alpha}_c) \tag{34}$$

and (35)

$$\hat{F}_{xa}(\alpha_c) = \frac{1}{n}\sum_{i=1}^n \mathbb{1}_{xa}\exp(C_i/\alpha_c) \tag{36}$$

$$\geq \exp(-R_{max}/\underline{\alpha}_c)\frac{1}{n}\sum_{i=1}^n \mathbb{1}_{xa} \tag{37}$$

$$= \exp(-R_{max}/\underline{\alpha}_c)\hat{p}_0(x,a) \tag{38}$$

Where $\hat{p}_0(x,a) = \frac{1}{n}\sum_{i=1}^n \mathbb{1}_{xa}$ is the empirical probability of observing $(X, A)$ in the dataset. Now we need to lower bound $\hat{p}_0(x,a)$. Let us lower bound the probability it is less than half of the true

value $p_0(x,a)/2$:

$$P\left(\hat{p}_0(x,a) \leq \frac{p_0(x,a)}{2}\right) = P\left(-\hat{p}_0(x,a) \geq -\frac{p_0(x,a)}{2}\right) \tag{39}$$

$$= P\left(p_0(x,a) - \hat{p}_0(x,a) \geq p_0(x,a) - \frac{p_0(x,a)}{2}\right) \tag{40}$$

$$= P\left(p_0(x,a) - \hat{p}_0(x,a) \geq \frac{p_0(x,a)}{2}\right) \tag{41}$$

$$\overset{(1)}{\leq} \exp\left(-2n\left(\frac{p_0(x,a)}{2}\right)^2\right) \tag{42}$$

$$= \exp\left(-\frac{np_0(x,a)^2}{2}\right) \tag{43}$$

Where step (1) applies Hoeffding inequality on the binary random variables $\mathbb{1}_{xa}$ and with $t = \frac{p_0(x,a)}{2}$

We can now calculate the minimum number of timesteps such that $P\left(\hat{p}_0(x,a) \leq \frac{p_0(x,a)}{2}\right) \leq \delta'$:

$$\exp\left(-\frac{p_0(x,a)^2}{2n}\right) \leq \delta' \xrightarrow[\text{Solve for n}]{} n \geq \frac{2\log(1/\delta')}{p_0(x,a)^2} \tag{44}$$

Therefore for $n \geq \frac{2\log(1/\delta')}{p_0(x,a)^2}$, with probability $1-\delta$ we have $\hat{p}_0(x,a) \geq \frac{p_0(x,a)}{2}$. We therefore have:

$$L = \frac{2}{\exp\left(\frac{-R_{max}}{\underline{\alpha}_c}\right) p_0(x,a)} \geq \frac{2}{\exp\left(\frac{-R_{max}}{\underline{\alpha}_c}\right) \underline{p}_0(x,a)} \tag{45}$$

We can now put everything together for the constant from equation 32:

$$c = 2\bar{\alpha}_c \frac{8(R_{max})^2 L}{\delta_c \underline{\alpha}_c^2} = \frac{32(R_{max})^3}{\delta_c^2 \underline{\alpha}_c^2 \exp\left(\frac{-R_{max}}{\underline{\alpha}_c}\right) \underline{p}_0(x,a)} \tag{46}$$

Let $\hat{\alpha}_c = \text{argmin}_{\alpha_c} \hat{G}_{xa}(\alpha_c)$. Note that $\hat{Q}_{X,A} = \hat{G}_{xa}(\hat{\alpha}_c)$. Let $\alpha_c^* = \text{argmin}_{\alpha_c} G_{xa}(\alpha_c)$. Note that $Q_{X,A} = G_{xa}(\alpha_c^*)$.

It then follows:

$$G_{xa}(\hat{\alpha}_c) - \hat{G}_{xa}(\hat{\alpha}_c) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right) \tag{47}$$

$$G_{xa}(\hat{\alpha}_c) - \hat{G}_{xa}(\hat{\alpha}_c) + G_{xa}(\alpha_c^*) \leq G_{xa}(\alpha_c^*) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right) \tag{48}$$

$$G_{xa}(\alpha_c^*) - \hat{G}_{xa}(\hat{\alpha}_c) \leq G_{xa}(\alpha_c^*) - G_{xa}(\hat{\alpha}_c) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right) \tag{49}$$

$$Q_{X,A} - \hat{Q}_{X,A} \leq \underbrace{G_{xa}(\alpha_c^*) - G_{xa}(\hat{\alpha}_c)}_{\leq 0} + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right) \tag{50}$$

$$Q_{X,A} - \hat{Q}_{X,A} \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right) \tag{51}$$

Similarly we can start with $G_{xa}(\alpha_c^*) - \hat{G}_{xa}(\alpha_c^*) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right)$ and prove the other side $\hat{Q}_{X,A} - Q_{X,A} \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log(1/\delta')}{n}}\right)$.

By setting $\delta' = \delta/2$, and taking the union bound over the two sides, the desired lemma result follows. □

**Note about $\underline{\alpha}_c$:** In this work we only consider cases where $\alpha_c > 0$. $\alpha_c$ is strictly bounded away from 0 by some $\underline{\alpha}_c > 0$. This occurs either when rewards have continuous support, or, in the case of discrete reward support, when $\delta_c$ is small enough such that the uncertainty set does not include the distribution with all mass at a single support value.

Since we are considering the case of discrete, finite context and action sets, we can now take a union bound over all $(X, A)$ pairs and achieve joint uniform convergence over all $(X, A)$ pairs to arrive at the following result:

**Lemma 7** (Uniform convergence over all $(X, A)$)**.** *When $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$, the following holds with probability of at least $1-\delta$:*

$$\max_{(X,A)} \left| Q_{X,A} - \hat{Q}_{X,A} \right| \leq \mathcal{O}\left( c\sqrt{\frac{\log(n)\log\left(2|\mathcal{X}||\mathcal{A}|/\delta\right)}{n}} \right) \tag{52}$$

*Where*

$$c = \frac{32(R_{max})^3}{\delta_c^2 \underline{\alpha}_c^2 \exp\left(\frac{-R_{max}}{\underline{\alpha}_c}\right) \underline{p}_0(x,a)} \tag{53}$$

With this, we continue the proof, recall our goal is to optimize the distributionally robust objective:

$$\hat{V} = \min_{\pi, \alpha_x \geq 0} \left\{ \hat{J}(\pi, \alpha_x) \right\} \quad \text{and} \quad V(\pi) = \min_{\pi, \alpha_x \geq 0} \left\{ J(\pi, \alpha_x) \right\} \tag{54}$$

Where

$$\hat{J}(\pi, \alpha_x) = \alpha_x \log \mathbb{E}\left[ \exp\left( \frac{-\hat{Q}^{avg}_{X,\pi}}{\alpha_x} \right) \right] + \alpha_x \delta_x \tag{55}$$

$$J(\pi, \alpha_x) = \alpha_x \log \mathbb{E}\left[ \exp\left( \frac{-Q^{avg}_{X,\pi}}{\alpha_x} \right) \right] + \alpha_x \delta_x \tag{56}$$

And for a fixed policy $\pi$ and a context $X$

$$\hat{Q}^{avg}_{X,\pi} = \sum_a \pi(a|X) \hat{Q}_{X,a} \quad \text{and} \quad Q^{avg}_{X,\pi} = \sum_a \pi(a|X) Q_{X,a} \tag{57}$$

We will next show the following result which gives uniform convergence of $J(\pi, \alpha_x)$ in $\pi$ and $\alpha_x$.

**Lemma 8** (Uniform convergence of optimization objective)**.** *With probability of at least $1-\delta$ and with $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$:*

$$\sup_{\pi, \alpha_x} \left| \hat{J}(\pi, \alpha_x) - J(\pi, \alpha_x) \right| \leq \mathcal{O}\left( c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}} \right) \tag{58}$$

*Proof:* For all steps, assume $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$. Because $\sum_a \pi(a|X) = 1$, $Q^{avg}_{X,\pi}$ is a weighted average of $Q_{X,A}$.

$$\left| \hat{Q}^{avg}_{X,\pi} - Q^{avg}_{X,\pi} \right| \leq \max \left| Q_{X,A} - \hat{Q}_{X,A} \right| \leq \mathcal{O}\left( c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}} \right) \tag{59}$$

Additionally because Lemma 7 holds for simultaneously all $(X, A)$ pairs, and every $\hat{Q}^{avg}_{i,\pi}$ depends on the same set of $Q_{X,A}$, we have the above result holds with probability with $1-\delta$ for all $i$. Formally, with probability $1-\delta$:

$$\max_X \left|\hat{Q}^{avg}_{X,\pi} - Q^{avg}_{X,\pi}\right| \leq \max \left|Q_{X,A} - \hat{Q}_{X,A}\right| \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{60}$$

Then we have:

$$\begin{aligned}
\sup\left|\hat{J}(\pi,\alpha_x) - J(\pi,\alpha_x)\right| &= \alpha_x \sup\left|\log\mathbb{E}\left[\exp\left(\frac{-\hat{Q}^{avg}_{X,\pi}}{\alpha_x}\right)\right] - \log\mathbb{E}\left[\exp\left(\frac{-Q^{avg}_{X,\pi}}{\alpha_x}\right)\right]\right| && (61)\\
&= \alpha_x \sup\left|\left(\log\mathbb{E}\left[\frac{\exp\left(\frac{-\hat{Q}^{avg}_{i,\pi}}{\alpha_x}\right)}{\exp\left(\frac{-Q^{avg}_{i,\pi}}{\alpha_x}\right)}\right]\right)\right| && (62)\\
&= \alpha_x \sup\left|\left(\log\mathbb{E}\left[\exp\left(\frac{Q^{avg}_{i,\pi} - \hat{Q}^{avg}_{i,\pi})}{\alpha_x}\right)\right]\right)\right| && (63)\\
&\leq \alpha_x\left(\log\mathbb{E}\left[\exp\left(\frac{\sup_i\left|Q^{avg}_{i,\pi} - \hat{Q}^{avg}_{i,\pi}\right|}{\alpha_x}\right)\right]\right) && (64)\\
&\leq \alpha_x\left(\log\left(\exp\left(\frac{\mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right)}{\alpha_x}\right)\right)\right) && (65)\\
&= \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) && (66)
\end{aligned}$$

Which is the desired result. □

We can now show convergence in policy evaluation. We first rewrite the objectives, $\hat{V}(\pi)$ and $\hat{V}(\pi)$, using $J(\pi,\alpha_x)$ and $\hat{J}(\pi,\alpha_x)$. Given a fixed policy $\pi$:

$$\hat{V}(\pi) = \min_{\alpha_x \geq 0}\left\{\hat{J}(\pi,\alpha_x)\right\} \quad \text{and} \quad V(\pi) = \min_{\alpha_x \geq 0}\left\{J(\pi,\alpha_x)\right\} \tag{67}$$

We now restate the theorem:

**Theorem 2** (Convergence of policy evaluation)**.** *For $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$, the following holds for any $\pi$ with probability of at least $1-\delta$:*

$$\left|V(\pi) - \hat{V}(\pi)\right| \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \quad \textit{where} \;\; c = \frac{32(R_{max})^3}{\delta_c^2\underline{\alpha}_c^2\exp(\frac{-R_{max}}{\underline{\alpha}_c})\underline{p}_0(x,a)} \tag{8}$$

*and $\underline{p}_0(x,a) = \min_{(x,a)} p_0(x,a) \geq \epsilon_\pi\epsilon_X$ is the minimum over probability of occurrence of (X,A) pairs, and $\underline{\alpha}_c$ is the minimum value for $\alpha_c$.*

*Proof* For all steps, assume $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$. Define:

$$\hat{\alpha}_x = \operatorname*{argmin}_{\alpha_x \geq 0}\left\{\hat{J}(\pi,\alpha_x)\right\} \quad \text{and} \quad \alpha_x^* = \operatorname*{argmin}_{\alpha_x \geq 0}\left\{J(\pi,\alpha_x)\right\} \tag{68}$$

It then follows:

$$J(\pi, \hat{\alpha}_x) - \hat{J}(\pi, \hat{\alpha}_x) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{69}$$

$$J(\pi, \hat{\alpha}_x) - \hat{J}(\pi, \hat{\alpha}_x) + J(\pi, \alpha_x^*) \leq J(\pi, \alpha_x^*) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{70}$$

$$J(\pi, \alpha_x^*) - \hat{J}(\pi, \hat{\alpha}_x) \leq J(\pi, \alpha_x^*) - J(\pi, \hat{\alpha}_x) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{71}$$

$$V(\pi) - \hat{V}(\pi) \leq \underbrace{J(\pi, \alpha_x^*) - J(\pi, \hat{\alpha}_x)}_{\leq 0 \text{ by definition of } \alpha_x^*} + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{72}$$

$$V(\pi) - \hat{V}(\pi) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{73}$$

We can also show the other side:

$$\hat{J}(\pi, \alpha*_x) - J(\pi, \alpha_x^*) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{74}$$

$$\hat{J}(\pi, \alpha*_x) - J(\pi, \alpha_x^*) + \hat{J}(\pi, \hat{\alpha}_x) \leq \hat{J}(\pi, \hat{\alpha}_x) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{75}$$

$$\hat{J}(\pi, \hat{\alpha}_x) - J(\pi, \alpha_x^*) \leq \hat{J}(\pi, \hat{\alpha}_x) - \hat{J}(\pi, \alpha*_x) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{76}$$

$$\hat{V}(\pi) - V(\pi) \leq \underbrace{\hat{J}(\pi, \hat{\alpha}_x) - \hat{J}(\pi, \alpha*_x)}_{\leq 0 \text{ by definition of } \hat{\alpha}_x} + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{77}$$

$$\hat{V}(\pi) - V(\pi) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{78}$$

Combining these two shows the result gives the desired result. □

Now lastly consider the case of policy learning. Define

$$V^* = \min_{\pi, \alpha_x} J(\pi, \alpha_x) \quad \text{and} \quad \hat{V}^* = \min_{\pi, \alpha_x} \hat{J}(\pi, \alpha_x) \tag{79}$$

Additionally define optimal parameters:

$$(\pi^*, \alpha_x^*) = \underset{\pi,\alpha_x}{\operatorname{argmin}}\, J(\pi, \alpha_x) \quad \text{and} \quad (\hat{\pi}, \hat{\alpha}_x) = \underset{\pi,\alpha_x}{\operatorname{argmin}}\, \hat{J}(\pi, \alpha_x) \tag{80}$$

Finally, recall the the set of $\epsilon$-optimal policies $\Pi_\epsilon^* = \{\pi : V(\pi) \geq V^* - \epsilon\}$.

We first restate the theorem:

**Theorem 3** (Convergence of Policy Learning)**.** *For $n \geq \frac{(2\log(2|\mathcal{X}|\mathcal{A}|/\delta)}{(\underline{p}_0(x,a))^2}$, the following holds with probability of at least $1-\delta$:*

$$\left|\hat{V}^* - V^*\right| \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{9}$$

*Equivalently this says*

$$P(\hat{\pi} \in \Pi_\epsilon^*) > 1 - \mathcal{O}\left(2|\mathcal{X}||\mathcal{A}|\exp\left(-\frac{n\epsilon^2}{\log(n)c^2}\right)\right) \tag{10}$$

*Proof*: This proof is similar to the proof of policy evaluation of the previous theorem. The first side of the inequality

$$J(\hat{\pi}, \hat{\alpha}_x) - \hat{J}(\hat{\pi}, \hat{\alpha}_x) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{81}$$

$$J(\hat{\pi}, \hat{\alpha}_x) - \hat{J}(\hat{\pi}, \hat{\alpha}_x) + J(\pi^*, \alpha_x^*) \leq J(\pi *' \alpha_x^*) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{82}$$

$$J(\pi^*, \alpha_x^*) - \hat{J}(\hat{\pi}, \hat{\alpha}_x) \leq J(\pi^*, \alpha_x^*) - J(\hat{\pi}, \hat{\alpha}_x) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{83}$$

$$V^* - \hat{V}^* \leq \underbrace{J(\pi^*, \alpha_x^*) - J(\hat{\pi}, \hat{\alpha}_x)}_{\leq 0 \text{ by definition of } (\pi^*, \alpha_x^*)} + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{84}$$

$$V^* - \hat{V}^* \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{85}$$

The other side of the inequality:

$$\hat{J}(\pi^*, \alpha*_x) - J(\pi^*, \alpha_x^*) \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{86}$$

$$\hat{J}(\pi^*, \alpha*_x) - J(\pi^*, \alpha_x^*) + \hat{J}(\hat{\pi}, \hat{\alpha}_x) \leq \hat{J}(\hat{\pi}, \hat{\alpha}_x) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{87}$$

$$\hat{J}(\hat{\pi}, \hat{\alpha}_x) - J(\pi^*, \alpha_x^*) \leq \hat{J}(\hat{\pi}, \hat{\alpha}_x) - \hat{J}(\pi^*, \alpha*_x) + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{88}$$

$$\hat{V}^* - V^* \leq \underbrace{\hat{J}(\hat{\pi}, \hat{\alpha}_x) - \hat{J}(\pi^*, \alpha*_x)}_{\leq 0 \text{ by definition of } (\hat{\pi}, \hat{\alpha}_x)} + \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{89}$$

$$\hat{V}^* - V^* \leq \mathcal{O}\left(c\sqrt{\frac{\log(n)\log\left(\frac{2|\mathcal{X}||\mathcal{A}|}{\delta}\right)}{n}}\right) \tag{90}$$

Combining these two shows the result gives the desired result. □