Non-Adaptive Experimental Design under Misspecifications for Contextual Bandits

Jian Vora
Department of Computer Science, Stanford University
jianv@cs.stanford.edu

Abstract

There are often many advantages to using simple function classes to represent the value function: such classes may have a lower sample complexity, and be more interpretable. However, there is a possibility that assuming such function classes accurately represent the true value function, the realizability assumption may be particularly harmful, even if in the end the goal is to fit a function in this class. This raises the question of how misspecification should impact adaptive data gathering in the pure exploration setting, where the goal is to gather a dataset in order to learn a policy of high performance at the end of the sampling period. In this work, we show that *soft* exploration schemes are a good tradeoff between robustness to misspecification and statistical efficiency.

1 Introduction

A stochastic linear contextual bandit problem is defined as a tuple $< \mathcal{S}, \mathcal{A}, \mu, r >$ where $\mathcal{S}$ is the state space form which a state s a drawn following a distribution μ. $r(s,a) = \phi(s,a)^T\theta^* + \eta$ is the reward function where η has zero mean and is 1-subgaussian. In the above formulation, the parameter θ^* is unknown but the $d-$dimensional features ϕ are known for all state-action pairs (assume a discrete state and action space, but very large, potentially exponential in d). The above model of having contexts or states in the standard stochastic bandit formulation has a lot of applications as this allows us to take *personalized* decisions. For example, contextual bandits have been used in online advertisements, where the click rate can be thought of as the reward and the features could be prior information of the user (age, demographics, past interaction history). Other applications could be in testing the response of patients to new drugs or portfolio management.

The objective here could be to minimize cumulative regret or simple regret. If π^* is the optimal policy, for minimizing the cumulative regret, we want to take actions $a_t = \pi_t(s_t)$ in an online manner that minimizes the following,

$$\mathbb{E}_{s_t \sim \mu} \sum_{t=1}^{T} (\max_a r(s_t, a) - r(s_t, \pi_t(s_t)) \tag{1}$$

Note that we care about expectation over the state contexts and not the worst-case regret which is usually the more studied setting. In the offline setting, we are given offline state contexts $C = \{s_1, s_2, \ldots, s_M\}$, we want to design an exploration policy π_e to gather a dataset $\mathcal{D}$ of the form $\{s_i, a_i, r_i\}_{i=1}^N$ to learn a policy $\hat{\pi}$ from it and that is should not be *very far* from the optimal policy π^* with regards to cumulative reward, i.e., ensure that simple regret is bounded by a constant ϵ:

$$\mathbb{E}_{s \sim \mu}(\max_a \phi(s,a)\theta^* - \phi(s, \hat{\pi}(s))\theta^*) \leq \epsilon \tag{2}$$

In doing the above task, we want to minimize the number of online samples N to be collected. The above problem of exploration in contextual bandits in an offline manner and then estimating

Preprint. Under review.

the unknown parameter was first introduced in [16]. They also provided a reward-free LinUCB [1] algorithm to design the exploration policy π_e which tries to take actions that have a high variance in the feature space. Intuitively, we want to get good coverage of the rewards in a fixed number of experiments that we want to run. Concretely, the policy components of the mixture are obtained as:

$$\pi_m(s) : \arg\max_a \|\phi(s,a)\|_{\Sigma_m^{-1}} \tag{3}$$

Here, Σ_m is the empirical covariance matrix obtained up to the mth iteration. For large state and action spaces, the result was that the number of samples in the offline and online data should be of the order $O(d^2/\epsilon^2)$ where d is the embedding dimensionality. The Sampler and Planner algorithm (SP) as introduced in [16] is shown below:

Algorithm 1 PLANNER (Reward-free LINUCB)

Input: Contexts $\mathcal{C} = \{s_1, \ldots, s_M\}$, reg. λ_{reg}
$\Sigma_1 = \lambda_{reg} I$
$m = 1$
for $m = 1, 2, \ldots M$ **do**
  if $\det(\Sigma_m) > 2\det(\Sigma_{\underline{m}})$ or $m = 1$ **then**
    $\underline{m} \leftarrow m$
    $\Sigma_{\underline{m}} \leftarrow \Sigma_m$
  end if
  Define $\pi_m : s \mapsto \arg\max_{a \in \mathcal{A}_s} \|\phi(s,a)\|_{\Sigma_{\underline{m}}^{-1}}$
  $\Sigma_{m+1} = \Sigma_m + \alpha\phi_m\phi_m^\top$; $\phi_m = \phi(s_m, \pi_m(s_m))$
end for
return policy mixture π_{mix} of $\{\pi_1, \ldots, \pi_M\}$

Algorithm 2 SAMPLER

Input: $\pi_{mix} = \{\pi_1, \ldots, \pi_M\}$, reg. λ_{reg}
Set $\mathcal{D}' = \emptyset$
for $n = 1, 2, \ldots N$ **do**
  Receive context $s'_n \sim \mu$
  Sample $m \in [M]$ uniformly at random
  Select action $a'_n = \pi_m(s'_n)$
  Receive feedback reward r'_n
  Store feedback $\mathcal{D}' = \mathcal{D}' \cup \{s'_n, a'_n, r'_n\}$
end for
return dataset $\mathcal{D}'$

Figure 1: π_e from [16]

The policy $\hat{\pi}$ is obtained as follows:

1. Perform rollouts of π_e obtained by Eq. 3 to collect an offline dataset $\mathcal{D} = \{s_i, \pi_e(s_i), r_i\}_{i=1}^N$.
2. Fit an estimator $\hat{\theta}$ on the dataset $\mathcal{D}$ using an MLE approach assuming the noise η to be a zero-mean gaussian.
3. $\hat{\pi}(s) = \text{argmax}_a \phi(s,a)^T\hat{\theta}$

Such a method of designing offline policies to collect data in a non-adaptive fashion can have a lot of applications. A fixed data collection strategy is practically desirable (1) whenever multiple agents collect data asynchronously and communication to update their policy is difficult or impossible, and (2) whenever changing the policy requires a significant overhead either in the engineering infrastructure. [16] has already solve the experimental design problem for the realizable case. In this work, we shall explore how to perform experimental design under reward misspecification.

2 Experimental Design Under Misspecification

Say, the reward model is misspecified, i.e. it is not a linear function of the features of the form $r(s,a) = \phi(s,a)^T\theta^* + \eta$. Getting away with the linearity assumption is crucial if we want to use the above exploration ideas for real-world problems, where misspecification is ubiquitous, especially when the features ϕ are not expressive enough. This raises the following questions:

1. What should be a good exploration policy π_e in order to *guarantee* an ϵ-suboptimal regret? For some intuition, if the perturbation is arbitrarily bad, one would guess π_e to be uniformly random would probably be the best solution. But if the perturbation is structured and the model is *not very far* from the linear model, we can perhaps do something between greedily following Eq. 3 and following a random policy.
2. Given any exploration policy π_e and a dataset $\mathcal{D}$ collected by performing rollouts, what would be a good policy $\hat{\pi}$? Now we cannot just rely on estimating the parameter $\hat{\theta}$ as something of the following form may not even exist. Even if the perturbation ensures that the model is *close* to linear, even then the MLE estimate of $\hat{\theta}$ would not be the least-square solution as now the effective noise is signal-dependent.

3 Relevant Work

3.1 Offline exploration for linear contextual bandits

[16] provided a sampler-planner algorithm to design offline experiments in order to minimize regret with the minimal size of the dataset $\mathcal{D}$. [17] studied the batch regret setting where data is collected in batched of B by an exploration policy repeatedly. The setting considered in [16] can be considered as a special case of [17] where there is only a single exploration stage. The main differences are in the way they use the offline M contexts available and the policy is decided only finally after iteration over all the available contexts is completed. [12] dealt with a similar problem and provides sample complexity bounds independent on ϵ but at the expense of a very large polynomial in the dimensionality d ($\mathcal{O}(d^{16})$) which is not desirable for practical purposes.

Agnostic Contextual Bandits: One line of work in contextual bandits is to not assume that the rewards come from a linear model but to assume a huge class of functions for a policy and perform a search in that space. Such methods usually have very poor regret guarantees as opposed to the $d\sqrt{T}$ regret bound for linear contextual bandits.

3.2 Misspecifications in the reward function

A lot of reward functions do not follow the linear approximation and design of experiments following the approximation can be provably suboptimal in these cases. Lots of prior works have explored contextual bandits under different misspecification models, but most of them deal with the online setting where the goal is to minimize cummulative regret. [9, 8] try to adapt to misspecified contextual bandits using offline regression oracles. [3] shows the that any misspecified model will suffer from a linear regret and the regret in the online setting will be of the form $\mathcal{O}(d\sqrt{T} + \epsilon T\sqrt{d})$ where ϵ is the max distance between the linear approximator and the reward function class. Similar results were also shown in [10, 5]. There have been recent work in offline linear contextual bandits which use ℓ_p estimators to get a good estimate for the underling parameter [11]. [4] deal with the problem of online learning in misspecified linear stochastic multi-armed bandit problems. [7] proposes a misspecification model which has some structure in the sense that it is additive and only state-dependent. [14] deals with misspecification for MDPs instead of contextual bandits and in the online setting.

3.3 How to learn the policy $\hat{\pi}$ after collecting the dataset $\mathcal{D}$?

One interesting question to ask would be how to get a good policy after the dataset $\mathcal{D}$ is collected? It surely depends on the kind of misspecification model but simply minimizing the MSE loss should not be a good idea. [7] gave a method to estimate the parameter θ under a bias term and some classic statistics and econometric paper deal with fitting a linear model under misspecification such as [15].

4 Misspecification Models

To start off with, we consider only additive perturbations in the reward model of the form,

$$r(s, a) = \phi(s, a)^T \theta^* + \epsilon(s, a) + \eta \tag{4}$$

where $\eta \sim \mathcal{N}(0, 1)$ and ϵ is the perturbation, which can depend in general on both states and actions. Clearly, if ϵ is not constrained, then the above model can represent any reward function by simply taking $\epsilon(s, a) = f(s, a) - \phi(s, a)^T \theta^*$ for any arbitrary function f which could represent all reward functions. However, we are mainly concerned with the case when the *reward is mostly linear and is misspecified only slightly but even this slightly could make the regret provably sub-optimal*.

4.1 When ϵ is only state-dependent

Consider the following special form of reward function,

$$r(s, a) = \phi(s, a)^T \theta^* + \epsilon(s) + \eta \tag{5}$$

Note that for the above perturbation, the optimal policy is independent of the perturbation ϵ. For a simplistic case of bandits without any context, this is just equivalent of having a non-zero mean

noise and all we have to do is to handle the bias introduced by ϵ for the estimation of the parameter θ^*. One very naive way to handle the above model when the state-space is finite with cardinality $|\mathcal{S}|$, then one algorithm will be of the following form,

---

NAIVE ALGORITHM 1:

1. Initialize $\tilde{\phi}(s,a) = \text{Concat}(\phi(s,a), \text{one-hot}(s))$ where one-hot(s) is a one-hot vector of length $|\mathcal{S}|$ with 1 being on the index of the state s.
2. Run the algorithm in [16] using $\tilde{\phi}(s,a)$.

The correctness of the above algorithm can be proved very easily and we can recover both (θ^*, ϵ) from solving the least squared minimization problem. One can quickly note that the guarantees from [16] can be adapted quickly and the number of both offline and online samples needed would be of the order $\mathcal{O}\left(\frac{(d+|\mathcal{S}|)^2}{\epsilon^2}\right)$ which is clearly not desirable. This is because we are also estimating the perturbations exactly which is not needed as the decision making does not depend on ϵ. One can even notice that this approach shall work even for perturbations when ϵ is a function of both states and actions. There the effective dimensionality will increase by $|\mathcal{S} \times \mathcal{A}|$.

We shall now see another approach which can potentially be better and is based on performing an unbiased version of linear regression as shown in [7]. Note that the quantity which we want to minimise is the following:

$$\mathbb{E}_{s\sim\mu}[\max_a \phi(s,a)\theta^* + \epsilon(s) - \phi(s,\hat{\pi}(s))\theta* - \epsilon(s)] \leq \epsilon \implies \mathbb{E}_{s\sim\mu}[\max_a \phi(s,a)\theta^* - \phi(s,\hat{\pi}(s))\theta*] \leq \epsilon \tag{6}$$

The above is essentially the same condition we needed for the realizable case and thus the exploration policy should be the same which tries to maximise for variance of the feature space while exploring. Thus, given the fact that π_e will be the same, we need to figure out how to estimate the parameter θ^*. This could be done by following some old econometric papers which try to learn a linear model under misspecifications [15, 7].

Action Centering to fit the Least Squares Estimator

One way to remove the bias effect is to center the features with respect to actions and then fit a least squares estimator on the transformed features as shown in [7]. Concretely, once we have collected a dataset using π_e which will be the same as sampler-planner (SP) [16], one does the following to obtain $\hat{\theta}$, an estimator of θ^*.

$$\mu(s) = \mathbb{E}_{a\sim\pi}[\phi(s,a)] \tag{7}$$

$$\tau = \lambda I + \sum_i (\phi(s_i,a_i) - \mu(s_i))(\phi(s_i,a_i) - \mu(s_i))^T \tag{8}$$

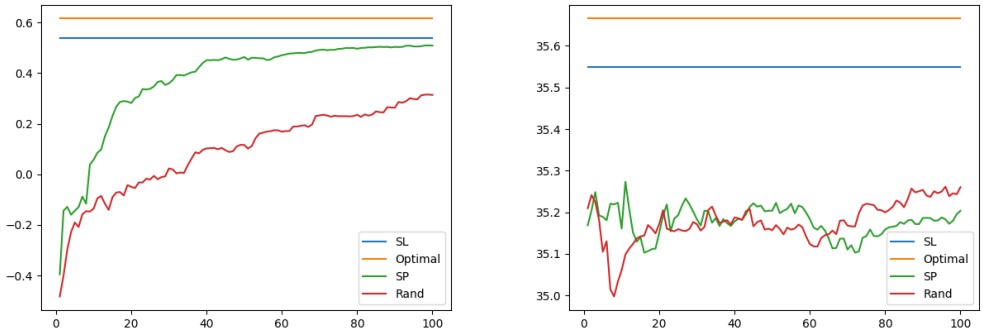
$$\hat{\theta} = \tau^{-1} \sum_i (\phi(s_i,a_i) - \mu(s_i)) r_i \tag{9}$$

This is a consistent estimator of the underlying parameter θ when the only perturbation is of the form of state dependent biases, check Lemma 5 and Lemma 6 in [7]. The results from implementing the above method (using π_e from [16] as the exploration policy and [7] as an estimator for $\hat{\theta}$). Figure 5 shows the results of estimating the unknown parameter θ^* under large and small bias values. We can clearly see that the above method fails when the perturbations are large and even though they do not affect decision making, they make it harder to learn the underlying parameter from the offline dataset.

A softmax policy and performing gradient descent to estimate θ^*

One good way to get rid of the additional bias term is to learn a softmax policy $\hat{\pi}$ from the collected dataset $\mathcal{D}$. Specifically, estimation of θ can be done by performing a maximum likelihood estimation. Note that here the additional term does not come into picture as it simply cancels out in the calculation of the partition funciton.

$$\pi(a|s) \propto \exp(r(s,a)) \propto \exp(\phi(s,a)^T\theta + \epsilon(s)) \propto \exp(\phi(s,a)^T\theta) \tag{10}$$



(a) Performing action centering estimation for estimating θ for small perturbations

(b) Performing action centering estimation for estimating θ for large perturbations

Figure 2: Results for only state-bias perturbations in two cases of the magnitude of misspecifications. Y-axis: reward of the learnt policy, X-axis: Number of online samples collected in the dataset to learn the policy. SL: Supervised learning baseline (fit an estimator with $N = 10000$ samples)

The parameter θ can be obtained by performing gradient descent, if the loss is L (log-likelihood) then the gradient is given as,

$$L = \log(\pi(a|s)) = \phi(s,a)^T\theta - \log \sum_i \exp(\phi(s,a_i)^T\theta) \tag{11}$$

$$\frac{\partial L}{\partial \theta} = \phi(s,a) - \frac{\sum_i \exp(\phi(s,a_i)^T\theta)\phi(s,a_i)}{\sum_i \exp(\phi(s,a_i)^T\theta)} \tag{12}$$

$$\frac{\partial L}{\partial \theta} = \phi(s,a) - \frac{\sum_i \exp(\phi(s,a_i)^T\theta + \epsilon(s))\phi(s,a_i)}{\sum_i \exp(\phi(s,a_i)^T\theta + \epsilon(s))} \tag{13}$$

$$\frac{\partial L}{\partial \theta} \approx \phi(s,a) - \frac{\sum_i \exp(\phi(s,a_i)^T\theta^* + \epsilon(s))\phi(s,a_i)}{\sum_i \exp(\phi(s,a_i)^T\theta^* + \epsilon(s))} \approx \phi(s,a) - \frac{\sum_i \exp(r(s,a_i))\phi(s,a_i)}{\sum_i \exp(r(s,a_i))} \tag{14}$$

Consider the following algorithms (left columns) for learning the parameter θ from biased rewards. The right columns indicate the MSE value (lower the better) between that estimation $\hat{\theta}$ and the true parameter θ^*.

Algorithm	MSE Value
Ignoring the bias and performing MSE (Upper Bound)	25.67
Action centering to fit the least square estimator	10.32
Use an unbiased estimator of the reward difference [6]	11.87
A softmax policy and performing gradient descent to get θ	4.57
Querying all states and fit a θ on the difference of rewards (Lower Bound)	1.93

Table 1: Various approaches to estimate θ^* under bias-only perturbation

In the above table, we see how different algorithms perform for estimating the underlying parameter in the presence of a state-dependent bias term. We note that if have rewards from the same state for two different actions, then we can estimate θ^* by performing least square estimation on the reward difference which will cancel out the bias terms. This is the lower bound but it is not feasible because. We see that the gradient descent approach based on the softmax policy does the best for estimation under a state-dependent bias.

In the subsequent subsection, we shall see a more interesting case, where ϵ can be action dependent which will change the data collecting policy π_e as well over the problem of estimating $\hat{\pi}$ from the collected offline dataset.

4.2 When ϵ can be action-dependent

[4] introduced a perturbation that was sparse. We notice that just having a sparse perturbation can provably lead to suboptimal regret for the offline contextual bandit case. So now the perturbation term is dependent on the action as well and it can be described as $r(s,a) = \phi(s,a)^T\theta^* + \epsilon(s,a) + \eta$ where $\epsilon(s,a)$ is $s-$sparse.

Theorem (Large Sparse Deviations). When the reward function is provided as the one with large sparse perturbations, the sampler-planner algorithm [16] will *provably achieve a linear regret* for a suitable set of features ϕ even for a 1-sparse perturbation. As an extension, any exploration policy not having full support will achieve provably linear regret.

Proof Sketch: The sampler-planner algorithm just aggressively samples a few actions if an adversary designs the features ϕ accordingly. In such a case, sampler-planner (SP) will not have a full support over the action space. Just adding a single perturbation that assigns a large reward to that unexplored action. Let a_e be the unexplored action by the sampler method given features ϕ. Consider the reward function,

$$r(s,a) = \phi(s,a)^T\theta^* + c\mathbb{I}(a = a_e) + \eta \tag{15}$$

Clearly, the offline dataset $\mathcal{D}$ will not have any samples having the action a_e and the corresponding high reward. One can then show that the planner algorithm will have a linear regret proportional to T. This motivates the need of another exploration policy as even a very simple deviation from the linear approximation can be extremely bad for sampler-planner.

Before going into the design of a *good* exploration policy under misspecification, how to learn the policy, even for a *any* (perhaps random) exploration policy? We can solve the optimization problem which can be posed as following:

$$\hat{\theta}, \hat{\epsilon} = \text{argmin}_{\theta,\epsilon} \sum_i (r_i - \phi(s_i,a_i)^T\theta - \epsilon(s_i,a_i))^2 + \lambda\|\epsilon(s,a)\|_1 \tag{16}$$

How many free parameters does the above problem have? There are a total of $d + |\mathcal{S}||\mathcal{A}|$. The above can be solved using an alternating minimization algorithm. On fixing $\epsilon(s,a)$, ϕ can be updated by solving a simple least squares problem, and on fixing ϕ, $\epsilon(s,a)$ can be updated by solving a LASSO problem using standard solvers.

Figure 3 gives a good empirical evidence of the theorem on large sparse deviations under which the sampler-planner algorithm will fail due to poor exploration. We implemented the above algorithm to jointly recover θ and ϵ (which was 1-spare giving a reward of 100 to action number 10). The features ϕ were chosen so that sampler-planner (SP) [16] would only explore action indices $1, 3, 4, 7$. Thus SP would never explore action 10 to see a large reward of 100 and hence the recovery algorithm will never be able to learn the correct ϵ. The policy value curves are as follows:

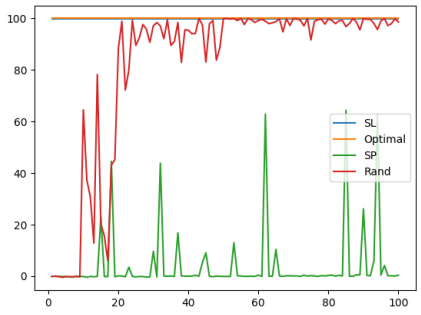


Figure 3: Performance of exploration schemes under a 1-sparse deviation. It is hard for Sampler-Planner to explore that sparse reward action and hence leads to sub-optimal policy value compared to random exploration. Y-axis: reward of the learnt policy, X-axis: Number of online samples collected in the dataset to learn the policy. SL: Supervised learning baseline.

5 Exploration under Reward Misspecification

In order to design a new sampling policy π_e which is robust to misspecification, we take note of the following points:

1. Most of these bandit methods essentially perform optimism under the face of uncertainty by calculating upper confidence bounds.
2. Knowing some structure of the reward affects the exploration bonus.
3. In vanilla UCB, without any knowledge of the reward structure, the action-taking strategy is performing $\left[\text{argmax}_a \hat{r}_{a,t} + \sqrt{\frac{2\log(t)}{n_{a,t}}}\right]$. The reward-free version of this would try to keep the sampling of each arm equal times and thus this is equivalent to random exploration as we have no structure on reward.
4. On the other extreme, we have linUCB which works under the realizability assumption. The action-taking strategy is performing $\left[\text{argmax}_a \hat{r}_{s,a,t} + \alpha\sqrt{\|\phi(s,a)\|_{\Sigma_t^{-1}}}\right]$. Thus here, we have ellipsoid confidence bounds and the reward-free version of this is essentially sampler-planner.

Now, we don't want to stay at either of the extremes. We still want to leverage the structure but not be overconfident on the confidence bound, and hence can use a *perturbed* version of the same. We shall formalize this now. In the subsequent sub-section, we take a slight detour to take a look at extreme valued theorem.

5.1 Extreme Value Theorem

The extreme value theorem tells us that the maximum of i.i.d. samples from exponentially tailed distributions will asymptotically converge to the Gumbel distribution $\mathcal{G}(\mu, \beta)$, which has pdf $p(x) = \exp(-(z + e^{-z}))$ where $z = (x - \mu)/\beta$ with location parameter μ and scale parameter β.

McFadden-Rust model:[13] An MDP following the standard Bellman equations with stochasticity in the rewards due to unobserved state variables will satisfy the soft-Bellman equations over the observed state with actual rewards $\bar{r}(s, a)$, given two conditions:

1. Additive separability (AS): observed rewards have additive i.i.d. Gumbel noise, i.e. $r(s, a) = \bar{r}(s, a) + \eta(s, a)$, with actual rewards $\bar{r}(s, a)$ and i.i.d. noise $\eta(s, a) \sim \mathcal{G}(0, \beta)$.
2. Conditional Independence (CI): the noise $\eta(s, a)$ in a given state-action pair is conditionally independent of that in any other state-action pair.

Now let us revisit the reward-free linUCB algorithm. We want to perturb the overconfident confidence bound to ensure decent exploration as well. Hence, we want to do the following:

$$\text{argmax}_a(\|\phi(s,a)\|_{\Sigma^{-1}} + \eta(s,a)) \tag{17}$$

The Gumbel-Max trick shows that for i.i.d. $\epsilon_i \sim \mathcal{G}(0, \beta)$ added to a set $\{x_1, ..., x_n\} \in \mathbb{R}$, $\max_i(x_i + \epsilon_i) \sim \mathcal{G}(\beta \log \sum_i \exp(x_i/\beta), \beta)$, and $\text{argmax}(x_i + \epsilon_i) \sim \text{softmax}(x_i/\beta)$. Now, if we taken $\eta(s, a) \sim \mathcal{G}(0, \beta)$, then we simply get the following,

$$\text{argmax}_a(\|\phi(s,a)\|_{\Sigma^{-1}} + \eta(s,a)) \sim \text{softmax}\left(\frac{\|\phi(s,a)\|_{\Sigma_m^{-1}}}{\beta}\right) \tag{18}$$

Note that the above is not too restrictive as we can always make the action-dependent perturbation to be zero mean by solving for the bias which was handled earlier and does not affect the exploration policy (which we shall see motivated from the above theory).

$$\eta(s,a) = \mathbb{E}_a\eta(s,a) + [\eta(s,a) - \mathbb{E}_a\eta(s,a)] \tag{19}$$

In the subsequent subsections, we shall look into two different exploration schemes π_e under misspecification which shall help in dealing with sparse deviations which was introduced in the previous sections. Both of them ensure more exploration than the SP algorithm and come with a tunable parameter which interpolates between random exploration and the SP [16] exploration policy.

5.2 Soft Sampler Planner (SSP)

We shall change lines 9 and 10 in the sampler-planner algorithm which was described in Fig. 1. The changes for to get the SSP algorithm shall be as follows:

$$\text{(Line 9)}\pi_m := \text{softmax}\left(\frac{\|\phi(s,a)\|_{\Sigma_m^{-1}}}{\beta}\right)$$

$$\text{(Line 10)}\Sigma_{m+1} = \Sigma_m + \alpha\mathbb{E}_{\pi_m}(\phi(s_m, \pi_m(s_m)\phi(s_m, \pi_m(s_m)^T)$$

In the above formulation, β is an important hyperparameter to be set which smoothly interpolates between doing random exploration ($\beta = \infty$) and taking the argmax and being very aggressive on the realizability assumption ($\beta = 0$). SSP is motivated from the above formulation where gumbel noise is added to the confidence bounds for taking an action, which converts the argmax to a softmax operation. Choosing β carefully ensures decent exploration and complete coverage of the action space which shall overcome the sparse deviation misspecifcaiton issue.

5.3 Explore Sampler Planner (ESP)

We shall change lines 9 and 10 in the sampler-planner algorithm which was described in Fig. 1. The changes for to get the SSP algorithm shall be as follows:

$$\text{(Line 9)}\pi_m := \text{argmax}\left(\|\phi(s,a)\|_{\Sigma_m^{-1}} + \frac{\beta}{N_a}\right), N_a = \text{number of times action a is taken}$$

$$\text{(Line 10)}\Sigma_{m+1} = \Sigma_m + \alpha\phi(s_m, a_m)\phi(s_m, a_m)^T; a_m = \pi_m(s_m)$$

$\beta = 0$ recover the sampler-planner (SP) algorithm and a very large β effectively ensures a random exploration. This is simply motivated by the fact of constructing an effective confidence bound as a linear combination of the confidence bounds of the linUCB and the vanilla UCB algorithms respectively.

5.4 Simulation Results

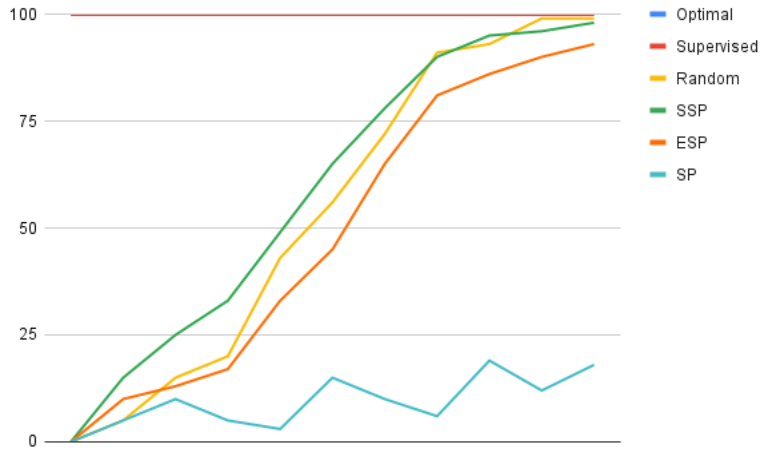


Figure 4: Comparisions of various experimental design techniques under sparse deviations from linearity

In the above figure, the features was selected in the same manner as done in [16] and the perturbation chosen was a reward of 100 on action number 10 which shall not be explore in the sampler-planner (SP) algorithm leading to lower reward of a policy learnt by its exploration. The recovery is θ, ϵ is done by solving the alternate minimization lasso problem. The state space was 3 and the action space was 10. For SSP, $\beta = 0.1$ and for ESP, $\beta = 10$ were the choice of the hyperparameters. We can see that SSP algorithm does the best of the possible solutions in this particular setting of sparse perturbation. More experimental results are needed across different settings (perhaps on real-world datasets) to analyze this result more rigorously.

5.5 Theory and Future Work

Given the fact the SSP algorithm does the best, we analyze the algorithm in some more detail and ask the following question:

Q: How (worse) does SSP perform compared to SP in the realizable case?
As we are in the realizable case, we still care about $\max_a \|\phi(s,a)\|_{\Sigma^{-1}}$ which would help us bound the regret. The proof in the paper proceeds by bounding the offline uncertainty and then showing that the online uncertainty would not be very far from the offline uncertainty.

$$\Sigma_m = \alpha \sum_{i=0}^{m} \phi(s_i,a_i)\phi(s_i,a_i)^T + \lambda_{reg} I; a_i = \pi_i(s_i), s_i \sim \mu$$

$$U_m := \max_a \|\phi(s_m,a)\|_{\Sigma_m^{-1}}$$

$$\mathbb{E}_M U_M \leq \frac{1}{M}\sum_m \mathbb{E}_m U_m \approx\leq \frac{1}{M}\sum_m U_m \approx\leq \sqrt{\frac{d}{\alpha M}}$$

For SSP, with small enough β, we get the order complexity

$$O\left(\frac{d^2(1+KL(p||q))}{\epsilon^2}\right); p =_a \|\phi(s,a)\|_{\Sigma^{-1}}, q = \text{softmax}_a \|\phi(s,a)\|_{\Sigma^{-1}}/\beta$$

$$KL(p||q) = \log \sum_a \exp \|\phi(s,a)\|_{\Sigma^{-1}} - \max_a \|\phi(s,a)\|_{\Sigma^{-1}} \leq \log(A) = O(d)$$

$$\max(x_1,x_2,\ldots,x_n) \leq \log \sum_i \exp(x_i) \leq \max(x_1,x_2,\ldots,x_n) + \log(n)$$

Hence we need offline contexts of the order $O\left(\frac{d^2(1+d))}{\epsilon^2}\right)$. This is particularly interesting as it looks like a tradeoff between the sample complexity and robustness to misspecification – soft exploration schemes are robuster to misspecification but suffer a larger sample complexity. A principled way to study when to use SSP can be formulated as follows:

$$r(s,a) = \phi(s,a)^T\theta^* + \epsilon(s,a) + \eta$$

Consider the following reward functions (expected reward gain under an adversarial perturbation for a given policy π),

$$J(\pi) = \min_{\epsilon \in \mathcal{E}} \left(\mathbb{E}_{s\sim\mu}\left[\phi(s,\pi(s))^T\theta^* + \epsilon(s,a)\right]\right) \tag{20}$$

The above formulation is inspired from this paper [2]. It essentially shows for online RL that policies trained by adding the entropy term in the RL objective are robust to reward and dynamics perturbations. We can use similar analysis for our case as well. The above loss function is very generic and one needs to characterize the set $\mathcal{E}$ to limit what kind of errors are we considering (as clearly if $\mathcal{E}$ is very large, then it seems kind of hopeless and a random exploration might be the best option). Let the policy learnt using sampler-planner be π_{SP} and the with the soft-sampler-planner be π_{SSP},

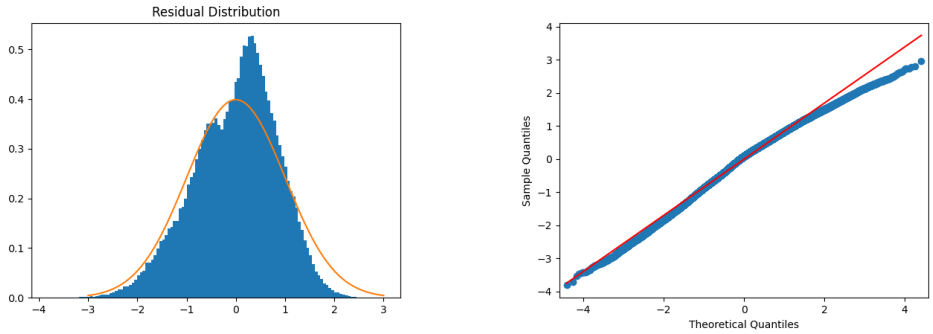
$$\text{Characterize } \mathcal{E} \text{ s.t. } J(\pi_{SSP}) > J(\pi_{SP})$$

Example: Speaking intuitively, $\mathcal{E}$ = sparse weights on actions not in the support of π_{SP} satisfies the above inequality. This a venue for potential future work in trying to understand the SSP algorithm in more detail. On a theoretical front, other avenues of future work could be:

1. Pose this simply as a bayesian optimization problem but try to perform it robustly. Choose a good utility function $I(\theta, D)$ seems to be wasteful if we just want to minimize simple regret. The most obvious thing would be to consider $[\theta, \epsilon]$ as a joint vector which we want to get information about and put constraints on ϵ.
2. Other option could be to just change the utility function and add an entropy of the exploration policy in it like soft-RL algorithms.
3. Real-world reward distributions are multimodal and not a single mode gaussian even if we model the mean vector to be very non-linear. This would require some more data analysis, some of which is done in the next subsection.

5.6 Real World Deviation Analysis

Starting with the Yahoo learning to rank dataset, we fit a linear model on the rankings then features which were 300-dimensional and visualized the distribution of the residuals vs a gaussian to see that they are indeed pretty close.



(a) Blue: Histogram of residuals after fitting a linear model to the Yahoo ranking dataset and Yellow: Gaussian with mean 0 and variance as that of the residual This might explain why the Sampler-Planner algorithm did decently good on this particular dataset

(b) QQ-plot to compare quantiles of the residual to a standard gaussian, for the same distribution, the blue line should fall exactly on the red line, here we see some tail deviation from a gaussian which could be fine as we have taken the noise to be a subgaussian

Figure 5: Visual Analysis of residual distribution **300**-dimensional features

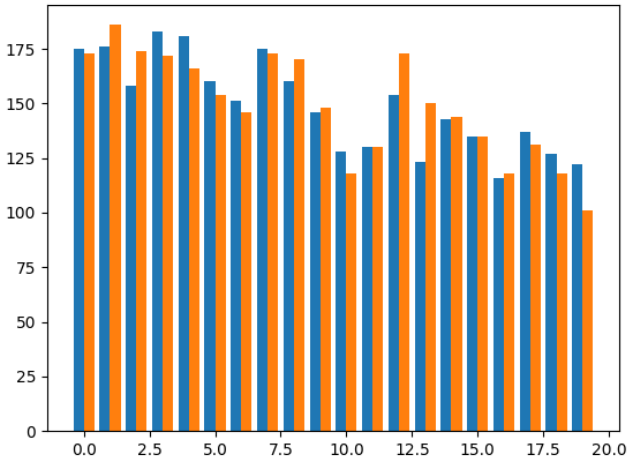


Figure 6: Distribution of actions taken by a random policy (blue) vs a sampler-planner policy (yellow) indicating the empirical counts are not very far

Future work would analyze more real-world datasets to find the real type of misspecification present in those datasets.

6 Conclusion

In this work, we dealt with the problem of experimental design under reward misspecification and show the *soft* exploration schemes lead to an interesting tradeoff between statistical efficiency and robustness to misspecifications.

References

[1] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 208–214, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[2] Benjamin Eysenbach and Sergey Levine. Maximum entropy rl (provably) solves some robust rl problems. In *International Conference on Learning Representations*, 2021.

[3] Dylan J. Foster, Claudio Gentile, Mehryar Mohri, and Julian Zimmert. Adapting to misspecification in contextual bandits. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2022. Curran Associates Inc.

[4] Avishek Ghosh, Sayak Ray Chowdhury, and Aditya Gopalan. Misspecified linear bandits. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[5] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In Jacob Abernethy and Shivani Agarwal, editors, *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 2137–2143. PMLR, 09–12 Jul 2020.

[6] Johannes Kirschner and Andreas Krause. Bias-robust bayesian optimization via dueling bandits. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5595–5605. PMLR, 18–24 Jul 2021.

[7] Akshay Krishnamurthy, Zhiwei Steven Wu, and Vasilis Syrgkanis. Semiparametric contextual bandits. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2776–2785. PMLR, 10–15 Jul 2018.

[8] Sanath Kumar Krishnamurthy, Vitor Hadad, and Susan Athey. Adapting to misspecification in contextual bandits with offline regression oracles. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5805–5814. PMLR, 18–24 Jul 2021.

[9] Sanath Kumar Krishnamurthy, Vitor Hadad, and Susan Athey. Tractable contextual bandits beyond realizability. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 1423–1431. PMLR, 13–15 Apr 2021.

[10] Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. Learning with good feature representations in bandits and in rl with a generative model. In *International Conference on Machine Learning*, pages 5662–5670. PMLR, 2020.

[11] Gene Li, Cong Ma, and Nathan Srebro. Pessimism for offline linear contextual bandits using l-p confidence sets. *arXiv preprint arXiv:2205.10671*, 2022.

[12] Yufei Ruan, Jiaqi Yang, and Yuan Zhou. Linear bandits with limited adaptivity and learning distributional optimal design. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 74–87, 2021.

[13] John Rust. Structural estimation of markov decision processes. In R. F. Engle and D. McFadden, editors, *Handbook of Econometrics*, volume 4, chapter 51, pages 3081–3143. Elsevier, 1 edition, 1986.

[14] Andrew Wagenmaker and Kevin Jamieson. Instance-dependent near-optimal policy identification in linear mdps via online experiment design. *arXiv preprint arXiv:2207.02575*, 2022.

[15] Halbert White. Consequences and detection of misspecified nonlinear regression models halbert white*. *Journal of the American Statistical Association*, 76, 04 2001.

[16] Andrea Zanette, Kefan Dong, Jonathan N Lee, and Emma Brunskill. Design of experiments for stochastic contextual linear bandits. *Advances in Neural Information Processing Systems*, 34:22720–22731, 2021.

[17] Zihan Zhang, Xiangyang Ji, and Yuan Zhou. Almost optimal batch-regret tradeoff for batch linear contextual bandits. *arXiv preprint arXiv:2110.08057*, 2021.