Warm-Starting Contextual Bandits under Latent Reward Scaling

Bastian Oetomo*, R. Malinga Perera†§, Renata Borovica-Gajic*, Benjamin I. P. Rubinstein*
*School of Computing and Information Systems, The University of Melbourne
†Amazon Web Services (AWS)
bastian.oetomo@unimelb.edu.au, wperera@amazon.com, renata.borovica@unimelb.edu.au, brubinstein@unimelb.edu.au

***Abstract*—Multi-armed bandits have long been known to enjoy optimal long-term performance, with sub-linear cumulative regret bounds standard. Recent developments take the performance of early rounds into consideration by 'warm-starting' bandits via incorporating pre-existing information into initialisation. Unfortunately, existing warm-start approaches are brittle to differences in the reward distributions between pre-training and deployment phases. This paper considers one such contextual bandit setting, where the same linear relationship relates contexts and rewards in pre-training and deployment phases, but only up to (unknown) constant scaling. A probabilistic model is proposed to capture this novel transfer learning problem, and a simple algorithm is derived as a *maximum a posteriori* point estimate. We present a regret bound for our method, with empirical evaluation across a range of datasets and against several cold- and warm-start baselines. A real-world motivated experiment on database index selection demonstrates non-linear modelling via neural network feature embeddings.**

***Index Terms*—multi-armed bandits, warm-start, pre-training**

I. Introduction

Multi-armed bandits (MABs) [1][2][3] have been studied and applied widely thanks to sub-linear cumulative regret bounds and practical algorithms. Compared to the more general framework of reinforcement learning, a contextual MAB acts based on current state to maximise a reward that may depend on given state and action taken. However specific to the MAB setting, state is independent of previous states and actions. This seemingly minor difference eliminates the need for MAB learners to plan, and provides for the simplest setting for balancing exploration-exploitation while offering significant practical utility. Many applications of sequential decision-making under uncertainty use bandit learners, from news [4], movie [5], crowd-sourcing recommendation [6], strategic playing [7], software testing [8], to database index selection [9], [10], [11], query optimisation [12], and adaptive online query plan processing [13].

Although long-term cumulative regret remains an important performance metric of bandit learners, short-term performance has emerged as an important consideration in practice. Were a bandit deployed to optimise database system performance, an administrator may not be willing to suffer terrible system performance over several dozen rounds as the bandit approximates a performant policy especially if each round spans a day. Known as the *cold-start problem* [14] in recommendation systems, the poor early performance of bandit learners originates from the demand of balancing *exploitation* with *exploration* necessary to attain its long-term sub-linear regret guarantee.

The problem in warm-starting bandits bears similarity to the conservative bandit problem [15]. For non-contextual bandit problems, an approach was proposed by [16]. Many methods have also been proposed for the contextual variant. For example, [17] addressed the cold-start problem via metric learning and graph regularisation while [18] employed an additional bandit to help each arm choose the user passively in an item-user problem setting. Another warm-starting approach is to group bandit arms into clusters, where arms coming from the same cluster intuitively behave similarly. This realisation was used extensively by [19], [20], [21], [22] and [23]. In some cases, historical data containing similar information to the bandit rounds exist. These data can be used to *pre-train* the bandit prior to its deployment. This pre-existing dataset is the core of the recent development of warm-starting bandits by [23], [24], [25] and [26], where the last two notably take the possibility of prior misspecification into account. This development mirrors parallel research in deep learning where pre-trained large language models [27], models in speech recognition [28] and vision [29][30], are now the norm.

Despite positive progress towards warm-starting bandits with the availability of pre-training data, any kind of reward distribution drift between pre-training and deployment phases presents a major challenge to today's methods. For example, units for rewards being mismatched can lead to counterproductive warm start, where performance may be inferior to cold start. This may occur even when the reward distribution, and its dependence on context, is otherwise unchanged, since training a value function to predict two very different scales of rewards may fail spectacularly. As an example, the rewards in adaptive query processing within databases [13] measure total elapsed processing time, which varies from machine to machine with an unknown scaling factor. Therefore, using a dataset obtained from one machine to pre-train a bandit destined for a different machine would likely lead to very poor short-term performance.

Building on the warm-start framework of [26], we develop a practical algorithm to address exactly this challenge of rescaling rewards, with an accompanying regret bound. Experiments following [25] and [26] and on synthetic data demonstrate

§R. Malinga Perera worked as a PhD candidate at the University of Melbourne. This paper describes work performed at the University of Melbourne and is not associated with Amazon.

state-of-the-art performance. The real-world experiment to automate index selection in a highly non-linear database system with our algorithm demonstrates its potential when LinUCB is applied on top of a neural network which also supplies the bandit's initial weights.

II. Background: Contextual Linear Bandits

The contextual multi-armed bandit problem is a sequential decision-making task. At round $t = 1, 2, \ldots, T$ the bandit:

1) observes k actions (or *arms*), along with their *context vectors* $\boldsymbol{x}_t(i) \in \mathbb{R}^d$ for each arm $i \in [k] = \{1, \ldots, k\}$;
2) selects an action (or *pulls* an arm) $i_t \in [k]$; and
3) observes scalar $R_{i_t}(t)$, a random reward for the chosen arm i_t at time t, sampling from a stationary distribution on the arm, conditionally on context. Rewards are independently drawn over rounds.

Notably an arm's rewards are only observed by pulling that arm. The goal of the MAB is to choose a sequence of arms maximising the *expected cumulative reward*, equivalent to minimising the *cumulative regret* at round T, formally defined as:

$$Reg(T) = \sum_{t=1}^{T} \mathbb{E}[R_{i_t^\star}(t) \mid \boldsymbol{x}_t(i_t^\star)] - \mathbb{E}[R_{i_t}(t) \mid \boldsymbol{x}_t(i_t)] \ ,$$

where $i_t^\star \in \arg\max_{i\in[k]} \mathbb{E}[R_i(t) \mid \boldsymbol{x}_t(i)]$, which is an arm yielding the maximum expected reward at round t given the contexts are known. One important metric for a bandit algorithm is its long-term performance, which is typically sub-linear in T, implying that the average regret $Reg(T)/T$ approaches zero as T approaches infinity—a property known as *Hannan consistency*.

A linear contextual bandit problem is a contextual bandit setting in which the (expected) reward is linear in the context vector. Specifically, we model the reward as a random variable satisfying

$$r_t(i) = R_i(t) \mid \boldsymbol{x}_t(i) = \boldsymbol{\theta}_\star(i)^T \boldsymbol{x}_t(i) + \epsilon_t(i) \ ,$$

where $\boldsymbol{\theta}_\star(i) \in \mathbb{R}^d$ is a latent variable vector for each arm i and $\epsilon_t(i)$ is conditionally R-subgaussian noise.

Since the reward model is linear, it is natural to apply regularised empirical risk minimisation to estimate the latent variable via (linear) ridge regression: $\hat{\boldsymbol{\theta}}_t = \boldsymbol{V}_t^{-1}\boldsymbol{b}_t$, where

$$\boldsymbol{V}_t = \lambda \boldsymbol{I} + \sum_{s=1}^{t-1} \boldsymbol{x}_s(i_s)\boldsymbol{x}_s^T(i_s) \ , \quad \boldsymbol{b}_t = \sum_{s=1}^{t-1} \boldsymbol{x}_s(i_s) r_t(i_s) \ ,$$

with $\lambda > 0$ being the regularisation hyperparameter. We discuss three algorithms considered in this paper, built on this parameter estimate. These span the standard approaches to (cold-start) linear contextual bandit learning.

a) ϵ-Greedy: This relatively crude approach explores by pulling an arm uniformly at random with small probability ϵ and otherwise greedily exploits by pulling any arm in $\arg\max_{i\in[k]} \hat{\boldsymbol{\theta}}_t^T \boldsymbol{x}_t(i_t)$ uniformly at random with probability $1 - \epsilon$.

b) LinUCB: Linear contextual UCB [4] follows the upper confidence bound (UCB) principle of pulling an arm that maximises an optimistic value function: added to a maximum likelihood estimate of an arm's reward is an upper confidence bound which effectively provides an exploration boost to under-explored arms. Formally we pull $i_t \in \arg\max_{i\in[k]} \hat{\boldsymbol{\theta}}_t^T \boldsymbol{x}_t(i) + \rho R\sqrt{\boldsymbol{x}_t(i)^T \boldsymbol{V}_t^{-1} \boldsymbol{x}_t(i)}$. This provides a more nuanced trade-off between exploration and exploitation.

c) LinTS: The linear Thompson sampler [31][32] employs the general approach of Thompson sampling, whereby a Bayesian posterior estimates future rewards, and to act, we simply draw a parameter from the posterior then pull greedily. The randomness introduced by the sampling ensures sufficient exploration for long-term optimality. We may introduce a random perturbation term to our current parameter estimate, to implement LinTS: $\tilde{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_t + \beta_t(\delta')\boldsymbol{V}_t^{-1/2}\boldsymbol{\eta}_t$, where $\beta(\delta)$ is scalar and $\boldsymbol{\eta}_t$ is a random perturbation. We choose an arm with the highest value of $\tilde{\boldsymbol{\theta}}^T \boldsymbol{x}_t(i)$. [32] has shown that with probability at least $1 - \delta$, it is sufficient for the distribution of $\boldsymbol{\eta}_t$ to follow a certain concentration and anti-concentration property, to yield a sub-linear regret bound. Meanwhile, the scalar $\beta_t(\delta)$ is chosen such that

$$\beta_t(\delta) = R\sqrt{2\log\frac{\det(\boldsymbol{V}_t)^{1/2}\det(\boldsymbol{V}_1)^{-1/2}}{\delta}} + \sqrt{\lambda}S \ ,$$

with S being the upper bound of the true parameter's ℓ_2-norm $\|\boldsymbol{\theta}_\star\|$.

III. Warm Starting Linear Bandits with Latent Reward Scaling

Suppose that we have a pre-training phase prior to deployment. This pre-training setting might involve bandit interaction (for example training a bandit to optimally configure a development machine), or it might offer a non-interactive supervised regression dataset. Irrespectively, a true (optimal) weight parameter $\boldsymbol{\mu}_\star \in \mathbb{R}^d$ minimises risk when predicting rewards in this pre-training phase. This parameter is unobserved. After pre-training is complete, we have our estimate of $\boldsymbol{\mu}_\star$, given by $\hat{\boldsymbol{\mu}}$.

In the next subsection we will expand on how this pre-trained estimate may be used to warm start the deployment phase of contextual bandit learning. Subsequently we will detail our rescaled rewards setting and present an algorithm for warm starting under rescaled rewards. We will then present a regret bound for our approach.

A. Warm-Start Basic Model

We follow [26] who assume that the optimal (but unobservable) deployment weight parameter $\boldsymbol{\theta}_\star$ is related to the true pre-training weight $\boldsymbol{\mu}_\star$ via $\boldsymbol{\theta}_\star = \boldsymbol{\mu}_\star + \bar{\boldsymbol{\delta}}_\star$, where $\bar{\boldsymbol{\delta}}_\star$ is the true concept drift between the two datasets. A Bayesian perspective suggests employing random variables $\boldsymbol{\theta} = \boldsymbol{\mu} + \bar{\boldsymbol{\delta}}$ for all latent parameters. The basic principle of the [26] approach is that one could model the drift instead of the full weight parameter $\boldsymbol{\theta}$. Any randomness we might have had in $\boldsymbol{\mu}$ (due to using a posterior to summarise the pre-training phase) can be absorbed into the second term without loss of generality, and so the new drift is modelled around $\hat{\boldsymbol{\mu}}$, taking: $\boldsymbol{\theta} = \hat{\boldsymbol{\mu}} + \boldsymbol{\delta}$.

Since $\hat{\boldsymbol{\mu}}$ is deterministic (as we condition on it), the problem of estimating $\boldsymbol{\theta}$ reduces to inferring $\boldsymbol{\delta}$. By our reward model assumption, we have

$$y_t(i) = r_t(i) - \hat{\boldsymbol{\mu}}^T \boldsymbol{x}_t(i) = \boldsymbol{\delta}_\star^T \boldsymbol{x}_t(i) + \epsilon_t(i) \ ,$$

which can be estimated using ridge regression, this time with $\boldsymbol{\delta}$ as the weight. The point estimate is $\hat{\boldsymbol{\delta}}_t = \boldsymbol{V}_t^{-1}\boldsymbol{b}_t$, where

$$\boldsymbol{V}_t = R^2 \boldsymbol{V}_1 + \sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i^T, \quad \boldsymbol{b}_t = \sum_{i=1}^{t-1} y_i \boldsymbol{x}_i \ ,$$

and $\boldsymbol{V}_1 = R^2(\boldsymbol{\Sigma}_\mu + \alpha^{-1}\boldsymbol{I}_d)^{-1}$ as parameter of similarity between datasets α and covariance of pre-training dataset's weight $\boldsymbol{\Sigma}_{\boldsymbol{\mu}}$.

B. Rescaling Rewards via Recalibration

We now introduce this paper's reward distribution shift setting, where pre-training phase rewards are rescaled in the deployment phase. Specifically, we assume that there exists some unknown scalar κ which scales pre-training rewards as: $\boldsymbol{\theta} = \kappa\boldsymbol{\mu} + \boldsymbol{\delta}$. Following the above basic warm-start setting, we model $\boldsymbol{\delta}$ as having a Gaussian distribution.

Now given an intermediate *recalibration* phase between pre-training and deployment, assume we observe a supervised set of contexts and rewards $(\boldsymbol{x}_1, r_1), \dots, (\boldsymbol{x}_n, r_n)$.[1] A maximum likelihood estimate (MLE) of κ can then be found by minimising $\sum_{i=1}^n (r_i - (\kappa\hat{\boldsymbol{\mu}} + \boldsymbol{\delta})^T \boldsymbol{x}_i)^2$. To address short-length pre-training phases we may regularise the drift parameter with a penalty on the ℓ_2-norm leading to the following minimisation objective

$$L(\kappa, \boldsymbol{\delta}) = \sum_{i=1}^n (r_i - (\kappa\hat{\boldsymbol{\mu}} + \boldsymbol{\delta})^T \boldsymbol{x}_i)^2 + \lambda\|\boldsymbol{\delta}\|_2^2 \ .$$

To simplify our derivation's notation, we introduce matrices $\boldsymbol{r} = [r_1, \dots, r_n]^T$ and $\boldsymbol{X} = [\boldsymbol{x}_1^T, \dots, \boldsymbol{x}_n^T]^T$, after which we may rewrite the objective function as

$$L(\kappa, \boldsymbol{\delta}) \quad = \quad \|\boldsymbol{r} - \boldsymbol{X}(\kappa\hat{\boldsymbol{\mu}} + \boldsymbol{\delta})\|_2^2 + \lambda\|\boldsymbol{\delta}\|_2^2 \ . \qquad (1)$$

Algorithm 1 solves this minimisation problem, to estimate how to rescale a pre-trained $\hat{\boldsymbol{\mu}}$ using a recalibration dataset $\boldsymbol{X}, \boldsymbol{r}$. This rescaling serves as a bridge between reward distribution shifted pre-training/deployment phases and current warm-start methods such as [26] which operate with standard contextual bandits. Algorithm 2 describes how these methods are composed, and is discussed following the next lemma.

Lemma 1. *Algorithm 1 produces rescaling estimates* $\hat{\boldsymbol{\delta}}_t, \hat{\kappa}_t$ *that exactly estimate the MAP for log-posterior Equation* (1).

[1]This recalibration phase need not be distinct. It could be a small number of rounds at deployment if the user is willing to run the policy parametrised by $\hat{\boldsymbol{\mu}}$ in parallel. Our approach is agnostic to the specific source of this data.

Algorithm 1 Rescale Pre-trained Rewards

```
Require: recalibration training set $\boldsymbol{X}, \boldsymbol{r}$, pre-trained parameters $\hat{\boldsymbol{\mu}}$, regularisation hyperparameter $\lambda > 0$
1: $\boldsymbol{V}_t^{-1} \leftarrow (\lambda I + \boldsymbol{X}^T\boldsymbol{X})^{-1}$
2: $d \leftarrow \|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}$            // $\|\boldsymbol{c}\|_{\boldsymbol{A}}$ denotes $\sqrt{\boldsymbol{c}^T\boldsymbol{A}\boldsymbol{c}}$
3: if $d = 0$ then
4:     $\hat{\kappa}_t \leftarrow 1$
5: else
6:     $\hat{\kappa}_t \leftarrow \frac{1}{d}\langle\hat{\boldsymbol{\mu}}, \boldsymbol{X}^T\boldsymbol{r}\rangle_{\boldsymbol{V}_t^{-1}}$            // $\langle\boldsymbol{u},\boldsymbol{v}\rangle_{\boldsymbol{A}}$ denotes $\boldsymbol{u}^T\boldsymbol{A}\boldsymbol{v}$
7: end if
8: $\hat{\boldsymbol{\delta}}_t \leftarrow \boldsymbol{V}_t^{-1}\boldsymbol{X}^T(\boldsymbol{r} - \kappa\boldsymbol{X}\hat{\boldsymbol{\mu}})$
9: return $\hat{\kappa}_t, \hat{\boldsymbol{\delta}}_t$
```

Proof. We find the extrema by taking the partial derivatives with respect to κ and $\boldsymbol{\delta}$ and set them to zero, noting minimality is assured by convexity, yielding:

$$\begin{aligned}
\hat{\boldsymbol{\delta}}_t &= (\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{r} - \hat{\kappa}_t\boldsymbol{X}\hat{\boldsymbol{\mu}}) \\
\hat{\kappa}_t &= \frac{1}{b-a}\hat{\boldsymbol{\mu}}^T(\boldsymbol{I} - \boldsymbol{X}^T\boldsymbol{X}(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1})\boldsymbol{X}^T\boldsymbol{r} \ ,
\end{aligned}$$

where we have defined $a = \|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{X}^T\boldsymbol{X}(\lambda\boldsymbol{I}+\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}}$ and $b = \|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{X}^T\boldsymbol{X}}$. Now consider $\boldsymbol{M} = \boldsymbol{X}^T\boldsymbol{X}(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}$, thus:

$$\begin{aligned}
\boldsymbol{M} &= \boldsymbol{X}^T\boldsymbol{X}(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I} - \lambda\boldsymbol{I}) \\
&= \boldsymbol{X}^T\boldsymbol{X} - (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I} - \lambda\boldsymbol{I})(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\lambda \\
&= \boldsymbol{X}^T\boldsymbol{X} - \lambda\boldsymbol{I} + \lambda^2(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1} \ ,
\end{aligned}$$

which leads to a simpler expression of $b - a$ as follows:

$$\begin{aligned}
b - a &= \hat{\boldsymbol{\mu}}^T\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}^T\boldsymbol{M}\hat{\boldsymbol{\mu}} \\
&= \hat{\boldsymbol{\mu}}^T\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\mu}} - \\
&\qquad \hat{\boldsymbol{\mu}}^T(\boldsymbol{X}^T\boldsymbol{X} - \lambda\boldsymbol{I} + \lambda^2(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1})\hat{\boldsymbol{\mu}} \\
&= \lambda\hat{\boldsymbol{\mu}}^T(\boldsymbol{I} - \lambda(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1})\hat{\boldsymbol{\mu}} \\
&= \lambda\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}} \ .
\end{aligned}$$

Now, the numerator for $\hat{\kappa}_t$ can be simplified further:

$$\begin{aligned}
(b-a)\hat{\kappa}_t &= \hat{\boldsymbol{\mu}}^T(\boldsymbol{I} - \boldsymbol{X}^T\boldsymbol{X}(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1})\boldsymbol{X}^T\boldsymbol{r} \\
&= \hat{\boldsymbol{\mu}}^T(\boldsymbol{I} - (\boldsymbol{I} - \lambda(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1})\boldsymbol{X}^T\boldsymbol{r} \\
&= \lambda\hat{\boldsymbol{\mu}}^T(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{r} \\
&= \lambda\langle\hat{\boldsymbol{\mu}}, \boldsymbol{X}^T\boldsymbol{r}\rangle_{\boldsymbol{V}_t^{-1}} \ ,
\end{aligned}$$

yielding a much simpler expression for $\hat{\kappa}_t$:

$$\hat{\kappa}_t = \frac{\langle\hat{\boldsymbol{\mu}}, \boldsymbol{X}^T\boldsymbol{r}\rangle_{\boldsymbol{V}_t^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}$$

as required. □

C. Warm Start Under Rescaled Rewards

Algorithm 2 outlines the key stages of warm-started bandit learning under rescaled rewards, bringing together our algorithmic contribution. First a pre-training oracle $\omega(\cdot)$ is called to produce an estimate $\hat{\boldsymbol{\mu}}$. This oracle might involve an interactive bandit learning session, supervised regression on batch context and reward data, or manually estimated parameters coming from a domain expert. Our approach is agnostic to the specifics of this first oracle. Second, a recalibration phase produces recalibration data via a call to $r(\cdot)$—again we are agnostic to how this phase operates but foresee either a small separate source of destination (deployment) environment contexts and rewards or operation of the pre-trained parameters as a policy for a number of rounds of deployment without any need to update the bandit during that period. Third, Algorithm 1 is run to rescale the obtained $\hat{\boldsymbol{\mu}}$ with recalibration data. Finally, the deployment phase can begin by initialising any standard bandit like ϵ-greedy, LinUCB or LinThompson with the pre-trained and rescaled policy instead of a "blank" initial policy, parameterised by $\kappa\hat{\boldsymbol{\mu}} + \delta$.

Estimating Warm-Start Covariance While Algorithm 2 deploys bandits requiring only a point estimate for weights $\boldsymbol{\theta}_\star$, LinUCB and LinThompson require a covariance estimate also [26]. Assuming a deterministic model $\boldsymbol{\theta} = \kappa\boldsymbol{\mu}$, we may: obtain a covariance estimate $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\mu}}$ alongside $\hat{\boldsymbol{\mu}}$ from pretraining line 1; discard $\boldsymbol{\delta}$ estimated in line 3; and replace line 4 with Deploy $m(\kappa\hat{\boldsymbol{\mu}}, \kappa^2\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\mu}})$. The rescaled covariance estimate follows immediately from the definition of covariance under a random variable's linear transformation. Alternatively, we can absorb all this randomness into the hyperparameter λ and call line 3 and line 4 every round before we stop updating κ.

Algorithm 2 Warm-Start MAB Under Latent Reward Scaling

```
Require: warm-start oracle $\omega(\cdot)$, recalibration oracle $r(\cdot)$,
    deployment MAB $m(\boldsymbol{\mu}_0)$ requiring an initial parameter
    vector, hyperparameter $\lambda > 0$
1: $\hat{\boldsymbol{\mu}} \leftarrow \omega(\cdot)$
2: $\boldsymbol{X}, \boldsymbol{r} \leftarrow r(\cdot)$
3: $\boldsymbol{\delta}, \kappa \leftarrow Rescale(\boldsymbol{X}, \boldsymbol{r}, \hat{\boldsymbol{\mu}}, \lambda)$        // Algorithm 1
4: Deploy $m(\kappa\hat{\boldsymbol{\mu}} + \delta)$
```

D. Regret Analysis

We state a cumulative regret bound for the deployment phase of Algorithm 2 when used with a LinThompson learner. Its proof is a direct consequence of [26] and [32] by providing valid initialisations for both $\hat{\boldsymbol{\mu}}', \hat{\boldsymbol{\Sigma}}'_{\boldsymbol{\mu}}$.

Theorem 2. *Consider Algorithm 2 with $m(\kappa\hat{\boldsymbol{\mu}}, \kappa^2\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\mu}})$ deploying LinThompson. If the deployment phase contexts $\boldsymbol{x}$, concept drift $\boldsymbol{\delta}$, and noise $\{\epsilon_t\}_t$ satisfy all of:*

1) $\|\boldsymbol{x}\| \le 1$ *for all* $\boldsymbol{x} \in \mathcal{X}$;
2) *there exists constant* $S \in \mathbb{R}^+$ *such that* $\|\boldsymbol{\delta}\| \le S$;
3) *Letting $\mathcal{F}_1$ be the information on prior knowledge, $\{\epsilon_t\}_t$ is conditionally R-subgaussian process and is a martingale difference sequence with $\mathcal{F}^x_t = (\mathcal{F}_1, \sigma(\boldsymbol{x}_1, r_1, \cdots, r_{t-1}, \boldsymbol{x}_t))$ as its filtration; and*
4) *LinThompson samples its perturbation random vectors $\boldsymbol{\eta}$ from a standard normal,*

then for δ the regret bound's confidence, $\delta' = \delta/(4T)$, and $\beta_t(\delta), \gamma_t(\delta)$ respectively a confidence region radius for ridge regression and a scaled version thereof, and $\boldsymbol{V}_t$ a scaled precision matrix within LinThompson, each defined in full in [26], with probability at least $1-\delta$, the cumulative regret of deployed LinThompson can be decomposed as $Reg(T) = R^{TS}(T) + R^{RLS}(T)$ with upper bounds:

$$R^{TS}(T) \le \frac{4\gamma_T(\delta')}{p}\left(\sqrt{2T\log\frac{\det(\boldsymbol{V}_{t+1})}{\det(R^2\boldsymbol{V}_1)}} + \sqrt{\frac{8T}{\lambda_{min}(R^2\boldsymbol{V}_1)}\log\frac{4}{\delta}}\right)$$

$$R^{RLS}(T) \le (\beta_T(\delta') + \gamma_T(\delta'))\sqrt{2T\log\frac{\det(\boldsymbol{V}_{t+1})}{\det(R^2\boldsymbol{V}_1)}} .$$

As suggested in the previous section, we may update κ and $\boldsymbol{\delta}$ every round and absorb the randomness we have in $\hat{\boldsymbol{\mu}}$ into λ. This approach requires further analysis since there is randomness in the estimator for κ. We next present a confidence ellipsoid bound, which plays a crucial role in the regret bound of many bandit algorithms.

Theorem 3. *For a given initial guess $\hat{\boldsymbol{\mu}}$, with randomness absorbed into λ, consider calling Algorithm 1 for every round in the calibration phase, until round n_c. With $\boldsymbol{V_t} = \lambda\boldsymbol{I} + \sum_{i=1}^{t-1}\boldsymbol{x}_i\boldsymbol{x}_i^T$, if all the context vectors $\boldsymbol{x}_i$ and the noise $\{\epsilon_t\}_t$ satisfy all of the following conditions:*

- *Letting $\mathcal{F}_1$ be the information on prior knowledge, $\{\epsilon_t\}_t$ is a $\mathcal{F}^x_t$-measurable conditionally R-subgaussian process and is a martingale difference sequence with $\mathcal{F}^x_t = (\mathcal{F}_1, \sigma(\boldsymbol{x}_1, r_1, \cdots, r_{t-1}, \boldsymbol{x}_t))$ as its filtration;*
- $\|\boldsymbol{x}\| \le L$ *for all* $\boldsymbol{x} \in \mathcal{X}$; *and*
- $\boldsymbol{x}$ *is* $\mathcal{F}^x_{t-1}$*–measurable,*

then for any $\delta \in (0,1)$, we have with probability at least $1-\delta$:

$$\begin{aligned}\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} &\le R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{(t-1)L^2}{\lambda d}\right)} \\ &+ \lambda\phi_k R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{(k-1)L^2}{\lambda d}\right)} \\ &+ \lambda\left\|\boldsymbol{\theta}_\star - \operatorname{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_k^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_k^{-1}} ,\end{aligned}$$

hold simultaneously for $t \ge 2$, where we have defined:

$$k = \min\{t, n_c + 1\}$$

$$\phi_k = \min\left\{\frac{1}{\lambda_{min}(\boldsymbol{X}_{k-1}^T\boldsymbol{X}_{k-1})}, \frac{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_2^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2 - \lambda\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_2^{-1}}}\right\}$$

$$\operatorname{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{A}}\boldsymbol{\theta}_\star = \frac{\langle\hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star\rangle_{\boldsymbol{A}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{A}}}\hat{\boldsymbol{\mu}} .$$

Proof. We now detail the derivation of the confidence bound. We begin by rewriting our estimator $\hat{\boldsymbol{\theta}}_t$, following [33] closely:

$$\boldsymbol{X}_t = \begin{bmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_t^T \end{bmatrix}, \quad \boldsymbol{r}_t = \begin{bmatrix} r_1 \\ \vdots \\ r_t \end{bmatrix}, \quad \boldsymbol{\epsilon}_t = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_t \end{bmatrix} .$$

To avoid clutter, we further abbreviate $\boldsymbol{X} = \boldsymbol{X}_{t-1}$, $\boldsymbol{r} = \boldsymbol{r}_{t-1}$ and $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}_{t-1}$. With these notations, our estimator for the parameter at round t can be written with initial scaled precision matrix $\lambda \boldsymbol{I}$ as $\boldsymbol{V}$ as follows:

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_t &= \hat{\kappa}_t \,\hat{\boldsymbol{\mu}} + \hat{\boldsymbol{\delta}}_t \\
&= \hat{\kappa}_t \,\hat{\boldsymbol{\mu}} + (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{r} - \hat{\kappa}_t \boldsymbol{X}\,\hat{\boldsymbol{\mu}}) \\
&= \hat{\kappa}_t \,\hat{\boldsymbol{\mu}} + (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{r} - \\
&\qquad (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{V} - \boldsymbol{V})\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} \\
&= (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{V}\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} + \boldsymbol{X}^T\boldsymbol{r}) ,
\end{aligned}$$

which can be thought of as a weighted average between $\hat{\kappa}_t \,\hat{\boldsymbol{\mu}}$ and $\boldsymbol{X}^T\boldsymbol{r}$. We next expand this by using $\boldsymbol{r} = \boldsymbol{X}\,\boldsymbol{\theta}_\star + \boldsymbol{\epsilon}$:

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_t &= (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{V}\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} + \boldsymbol{X}^T(\boldsymbol{X}\,\boldsymbol{\theta}_\star + \boldsymbol{\epsilon})) \\
&= (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{V}\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} + \\
&\qquad (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{V} - \boldsymbol{V})\,\boldsymbol{\theta}_\star + \\
&\qquad (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon} \\
&= \boldsymbol{\theta}_\star + (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{V}(\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star) + \\
&\qquad (\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon} .
\end{aligned}$$

Therefore, for any $\boldsymbol{c} \in \mathbb{R}^d$, we have:

$$\begin{aligned}
\boldsymbol{c}^T(\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star) &= \boldsymbol{c}^T(\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{V}(\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star) + \\
&\qquad \boldsymbol{c}^T(\boldsymbol{V} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon} \\
&= \langle \boldsymbol{c}, \boldsymbol{V}(\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\rangle_{\boldsymbol{V}_t^{-1}} + \langle \boldsymbol{c}, \boldsymbol{X}^T\boldsymbol{\epsilon}\rangle_{\boldsymbol{V}_t^{-1}} .
\end{aligned}$$

Since the matrix $\boldsymbol{V}_t$ is positive definite (hence $\boldsymbol{V}_t^{-1}$), the inner product is well-defined, so by the Cauchy-Schwarz inequality:

$$|\boldsymbol{c}^T(\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star)| \le \|\boldsymbol{c}\|_{\boldsymbol{V}_t^{-1}}(\|\boldsymbol{V}(\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_t^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}})$$

Now letting $\boldsymbol{c} = \boldsymbol{V}_t(\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star)$, so that $\|\boldsymbol{c}\|_{\boldsymbol{V}_t^{-1}} = \|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t}$ yields:

$$\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} \le \|\boldsymbol{V}(\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_t^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} . \tag{2}$$

At the start of the recalibration phase, we have $\boldsymbol{V} = \lambda\boldsymbol{I}$, thus Inequality (2) may be simplified into:

$$\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} \le \lambda\|(\hat{\kappa}_t \,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_t^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} .$$

For this analysis, we will assume that $\boldsymbol{X}^T\boldsymbol{X}$ is full rank and $2 \le t \le n_c + 1$, so the eigenvalues of $\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1}$ are:

$$\begin{aligned}
\lambda_i(\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1}) &= 1 - \lambda\lambda_i(\boldsymbol{V}_t^{-1}) \\
&= 1 - \frac{\lambda}{\lambda + \lambda_i(\boldsymbol{X}^T\boldsymbol{X}))} \\
&> 0 ,
\end{aligned}$$

which shows that $\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1}$ is positive definite, hence defining the inner product in the denominator of $\hat{\kappa}_t$.

Noting $\hat{\boldsymbol{\mu}}$ is deterministic, we can separate $\hat{\kappa}_t$ into the deterministic part and the random part by writing $\boldsymbol{r} = \boldsymbol{X}\boldsymbol{\theta}_\star + \boldsymbol{\epsilon}$:

$$\begin{aligned}
\hat{\kappa}_t &= \frac{\langle \hat{\boldsymbol{\mu}}, \boldsymbol{X}^T\boldsymbol{r}\rangle_{\boldsymbol{V}_t^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}} \\
&= \frac{1}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}\hat{\boldsymbol{\mu}}^T(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}_\star + \\
&\qquad \frac{1}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}\hat{\boldsymbol{\mu}}^T(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon} \\
&= \hat{\kappa}_{det} + \hat{\kappa}_{ran} .
\end{aligned}$$

Therefore, by the triangle inequality:

$$\begin{aligned}
\|\hat{\kappa}_t\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t^{-1}} &= \|\hat{\kappa}_{det}\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star + \hat{\kappa}_{ran}\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}} \\
&\le \|\hat{\kappa}_{det}\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t^{-1}} + \|\hat{\kappa}_{ran}\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}} .
\end{aligned}$$

1) The Deterministic Part of the Error: In this section, we will analyse $\hat{\kappa}_{det}$. Notice that the numerator can be written as:

$$\begin{aligned}
&\hat{\boldsymbol{\mu}}^T(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}_\star \\
&\quad = \hat{\boldsymbol{\mu}}^T(\boldsymbol{I} - \lambda(\lambda\boldsymbol{I} + \boldsymbol{X}^T\boldsymbol{X})^{-1})\boldsymbol{\theta}_\star \\
&\quad = \langle \hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star\rangle_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}} ,
\end{aligned}$$

which yields:

$$\hat{\kappa}_{det} = \frac{\langle \hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star\rangle_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}} .$$

This expression has some geometrical interpretation. Upon scaling $\hat{\boldsymbol{\mu}}$ with this expression, we have:

$$\hat{\kappa}_{det}\hat{\boldsymbol{\mu}} = \frac{\langle \hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star\rangle_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}\hat{\boldsymbol{\mu}} ,$$

which we realise as the weighted projection of $\boldsymbol{\theta}_\star$ onto $\hat{\boldsymbol{\mu}}$. What this means is that the correct magnitude for the initial guess is not required. As long as $\hat{\boldsymbol{\mu}}$ is parallel to $\boldsymbol{\theta}_\star$, our algorithm will give the correct κ under a noiseless regime.

We now analyse the weight for the inner product by varying the value for λ. Firstly, notice that $\lim_{\lambda\to 0}\hat{\kappa}_{det} = \frac{\langle \hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star\rangle}{\|\hat{\boldsymbol{\mu}}\|^2}$. Therefore, upon substituting this value, we have:

$$\lim_{\lambda\to 0}\hat{\kappa}_{det}\hat{\boldsymbol{\mu}} = \frac{\langle \hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star\rangle}{\|\hat{\boldsymbol{\mu}}\|^2}\hat{\boldsymbol{\mu}} ,$$

which corresponds to the standard Euclidean projection of $\boldsymbol{\theta}_\star$ onto $\hat{\boldsymbol{\mu}}$. On the other hand, λ can be tuned to be as large as possible. To analyse this behaviour, we rewrite $\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1}$ as follows:

$$\begin{aligned}
\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1} &= \boldsymbol{I} - (\boldsymbol{I} + \frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{X})^{-1} \\
&= \boldsymbol{I} - \boldsymbol{I} - \sum_{i=1}^{\infty}(-\frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{X})^i \\
&= \sum_{i=1}^{\infty}\frac{(-1)^{i+1}}{\lambda^i}(\boldsymbol{X}^T\boldsymbol{X})^i .
\end{aligned}$$

When λ is large, the first term dominates, hence:

$$\lim_{\lambda\to\infty} \hat{\kappa}_{det} = \frac{\langle \hat{\boldsymbol{\mu}}, \boldsymbol{\theta}_\star \rangle_{\frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{X}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{X}}} = \frac{\hat{\boldsymbol{\mu}}^T \frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}_\star}{\hat{\boldsymbol{\mu}}^T \frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\mu}}} = \frac{\langle \boldsymbol{X}\hat{\boldsymbol{\mu}}, \boldsymbol{X}\boldsymbol{\theta}_\star \rangle}{\|\boldsymbol{X}\hat{\boldsymbol{\mu}}\|^2} .$$

Therefore, we have:

$$\lim_{\lambda\to\infty} \hat{\kappa}_{det}\boldsymbol{X}\hat{\boldsymbol{\mu}} = \frac{\langle \boldsymbol{X}\hat{\boldsymbol{\mu}}, \boldsymbol{X}\boldsymbol{\theta}_\star \rangle}{\|\boldsymbol{X}\hat{\boldsymbol{\mu}}\|^2}\boldsymbol{X}\hat{\boldsymbol{\mu}} ,$$

which is the projection of the noiseless reward vectors onto the predicted reward vectors.

Therefore, the magnitude for the deterministic part of the error can be written as:

$$\|\hat{\kappa}_{det}\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t^{-1}} = \left\|\boldsymbol{\theta}_\star - \mathrm{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_t^{-1}} ,$$

which is the weighted norm of the orthogonal component of $\boldsymbol{\theta}_\star$ with respect to $\hat{\boldsymbol{\mu}}$ under the matrix transformation $(\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1})^{1/2}$.

When $\boldsymbol{X}^T\boldsymbol{X}$ is not a full-rank matrix, the matrix $\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}$ is no longer positive definite hence we lose our interpretation of the projection, since the inner product is no longer well-defined. However, our expression will remain correct as long as $\|\,\hat{\boldsymbol{\mu}}\,\|_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}} \neq 0$, which will be the restriction of our algorithm.

2) The Random Part of the Error: In this section, we will analyse the random part of the error and we will assume that our contexts are $\mathcal{F}^x_{t-1}$–measurable. Firstly, by the Cauchy-Schwarz inequality:

$$\begin{aligned}\hat{\kappa}_{ran} &= \frac{1}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}\langle \hat{\boldsymbol{\mu}}, \boldsymbol{X}^T\boldsymbol{\epsilon}\rangle_{\boldsymbol{V}_t^{-1}} \\ &\leq \frac{1}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}\|\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}}\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} .\end{aligned}$$

Therefore,

$$\|\hat{\kappa}_{ran}\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}} \leq \frac{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}}\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} = \psi_t\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} .$$

The term $\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}}$ is common with the second term of Inequality (2), so we begin by finding an upper bound to ψ_t. Expanding its denominator:

$$\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}} = \hat{\boldsymbol{\mu}}^T(\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1})\hat{\boldsymbol{\mu}} = \|\hat{\boldsymbol{\mu}}\|^2 - \lambda\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}} ,$$

so that ψ_t can be written as:

$$\psi_t = \frac{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2 - \lambda\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}}} = \frac{1}{\frac{\|\hat{\boldsymbol{\mu}}\|^2}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}}} - \lambda}$$

Therefore, ψ_t is maximised when $\|\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}}$ is maximised. Moreover, since $\frac{\|\hat{\boldsymbol{\mu}}\|^2}{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}}} > \lambda$, then $\psi_t > 0$ and we have avoided the singularity point. Maximising $\|\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}}$ yields:

$$\|\hat{\boldsymbol{\mu}}\|_{\boldsymbol{V}_t^{-1}} \leq \sqrt{\lambda_{max}(\boldsymbol{V}_t^{-1})}\|\hat{\boldsymbol{\mu}}\| .$$

Therefore, ψ_t can be bounded in terms of $\lambda_{max}(\boldsymbol{V}_t^{-1})$ as:

$$\psi_t \leq \frac{1}{\frac{1}{\lambda_{max}(\boldsymbol{V}_t^{-1})} - \lambda} .$$

Now $\lambda_i(\boldsymbol{V}_t^{-1}) = (\lambda + \lambda_i(\boldsymbol{X}^T\boldsymbol{X}))^{-1}$. Therefore, it must be true that $\lambda_{max}(\boldsymbol{V}_t^{-1}) = (\lambda + \lambda_{min}(\boldsymbol{X}^T\boldsymbol{X}))^{-1}$. Substituting this value to the upper bound of ψ_t above gives us:

$$\psi_t \leq \frac{1}{\lambda_{min}(\boldsymbol{X}^T_{\boldsymbol{t-1}}\boldsymbol{X}_{\boldsymbol{t-1}})} ,$$

where $\boldsymbol{X}$ is rewritten as its full form, $\boldsymbol{X}_{t-1}$.

Although the expression above is correct, it is not useful when $\boldsymbol{X}^T\boldsymbol{X}$ is not full-rank as it tells us that $\lambda_{\min}(\boldsymbol{X}^T\boldsymbol{X}) = 0$, thus ψ_t is unbounded. However, it is often the case that $\|\,\hat{\boldsymbol{\mu}}\,\|_{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}} \neq 0$ even though $\boldsymbol{X}^T\boldsymbol{X}$ is not full-rank. In other words, we allow $\boldsymbol{X}^T\boldsymbol{X}$ to be a non-full-rank matrix as long as $\hat{\boldsymbol{\mu}}$ does not lie in the null space of $\boldsymbol{I} - \lambda\boldsymbol{V}_t^{-1}$.

We have previously concluded that ψ_t will be larger as $\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_t^{-1}}$ is increased. However, since $\boldsymbol{x}_t\boldsymbol{x}_t^T$ is positive semi-definite for all t, we also have $\boldsymbol{V}_t \preceq \boldsymbol{V}_{t+1}$, which implies that $\boldsymbol{V}_t^{-1} \succeq \boldsymbol{V}_{t+1}^{-1}$. Since substituting $t = 1$ will give us 0 in the denominator, we step back and substitute $t = 2$ to give us a non-zero value in the denominator. This gives us another upper bound for ψ_t as:

$$\psi_t \leq \frac{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_2^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2 - \lambda\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_2^{-1}}} .$$

Since the two inequalities above hold simultaneously, it must be true that ψ_t is less than the minimum of the two upper bounds. Therefore, we conclude that $\psi_t \leq \phi_t$, where we have defined

$$\phi_t = \min\left\{\frac{1}{\lambda_{min}(\boldsymbol{X}^T_{t-1}\boldsymbol{X}_{t-1})}, \frac{\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_2^{-1}}}{\|\hat{\boldsymbol{\mu}}\|^2 - \lambda\|\hat{\boldsymbol{\mu}}\|^2_{\boldsymbol{V}_2^{-1}}}\right\} .$$

3) Putting it all together: Now we come back to Inequality (2). We can rewrite Inequality (2) as:

$$\begin{aligned}\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} &\leq \lambda\|(\hat{\kappa}_t\,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_t^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} \\ &\leq \lambda\left\|\boldsymbol{\theta}_\star - \mathrm{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_t^{-1}} + \\ &\qquad \lambda\phi_t\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} .\end{aligned}$$

To bound $\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}}$, we use the result presented by [33] after assuming that all the contexts are $\|\boldsymbol{x}_t\| \leq L$. Then, with probability at least $1 - \delta$,

$$\begin{aligned}\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} &\leq R\sqrt{2\log\left(\frac{\det(\boldsymbol{V}_t)^{1/2}\det(\lambda\boldsymbol{I})^{-1/2}}{\delta}\right)} \\ &\leq R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{(t-1)L^2}{\lambda d}\right)} .\end{aligned}$$

Notice that the two terms we are bounding are essentially the same variable, thus the probability that both inequalities hold is at least $1 - \delta$. We have finally arrived to the upper bound of our confidence ellipsoid for $2 \leq t \leq n_c + 1$, which is:

$$\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} \leq R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{(t-1)L^2}{\lambda d}\right)} + \lambda\phi_t R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{(t-1)L^2}{\lambda d}\right)} + \lambda\left\|\boldsymbol{\theta}_\star - \mathrm{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_t^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_t^{-1}} .$$

To analyse the upper bound for $t \geq n_c+1$, we may interpret this as a new bandit with initial guess $\hat{\kappa}_{n_c+1}\,\hat{\boldsymbol{\mu}} + \hat{\boldsymbol{\delta}}_{n_c+1}$ and initial scaled precision matrix $\boldsymbol{V} = \boldsymbol{V}_{n_c+1}$. Inequality (2) still applies under this condition, noting that $\hat{\kappa}_t = \hat{\kappa}_{n_c+1}$ for $t \geq n_c + 1$. Therefore, upon substituting $\boldsymbol{V} = \boldsymbol{V}_{n_c+1}$ and shifting the round number:

$$\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} \leq \|\boldsymbol{V}_{n_c+1}(\hat{\kappa}_{n_c+1}\,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_t^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} .$$

Now, again using $\boldsymbol{V}_t^{-1} \succeq \boldsymbol{V}_{t+1}^{-1}$, and since $t \geq n_c + 1$, we may bound the first term as:

$$\begin{aligned}
&\|\boldsymbol{V}_{n_c+1}(\hat{\kappa}_{n_c+1}\,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_t^{-1}} \\
&\quad\leq \|\boldsymbol{V}_{n_c+1}(\hat{\kappa}_{n_c+1}\,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star)\|_{\boldsymbol{V}_{n_c+1}^{-1}} \\
&\quad= \|\hat{\kappa}_{n_c+1}\,\hat{\boldsymbol{\mu}} - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_{n_c+1}} \\
&\quad\leq \lambda\left\|\boldsymbol{\theta}_\star - \mathrm{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_{n_c+1}^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_{n_c+1}^{-1}} + \\
&\qquad\lambda\phi_{n_c+1}\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_{n_c+1}^{-1}} .
\end{aligned}$$

This yields the following inequality:

$$\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} \leq \lambda\left\|\boldsymbol{\theta}_\star - \mathrm{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_{n_c+1}^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_{n_c+1}^{-1}} + \lambda\phi_{n_c+1}\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_{n_c+1}^{-1}} + \|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}} .$$

Notice that our upper bound for $\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_t^{-1}}$ includes the upper bound for $\|\boldsymbol{X}^T\boldsymbol{\epsilon}\|_{\boldsymbol{V}_{n_c+1}^{-1}}$ since our inequality holds simultaneously for all $t \geq 0$, thus both inequalities hold simultaneously with probability at least $1 - \delta$. Thus,

$$\|\hat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}_\star\|_{\boldsymbol{V}_t} \leq R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{(t-1)L^2}{\lambda d}\right)} + \lambda\phi_{n_c+1} R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{n_c L^2}{\lambda d}\right)} + \lambda\left\|\boldsymbol{\theta}_\star - \mathrm{proj}_{\hat{\boldsymbol{\mu}}}^{\boldsymbol{I}-\lambda\boldsymbol{V}_{n_c+1}^{-1}}\boldsymbol{\theta}_\star\right\|_{\boldsymbol{V}_{n_c+1}^{-1}}$$

holds simultaneously for $t \geq n_c + 1$.

Finally, combining two of the cases where $2 \leq t \leq n_c + 1$ and $t \geq n_c+1$ is done by simply substituting $k = \min(t, n_c + 1)$ to the relevant quantities in the second and third terms, which yields the desired result. □

An important consequence of this theorem is that, even though our algorithm is based on making a Gaussian noise modelling assumption, our confidence ellipsoid bound applies for any R–subgaussian noise.

Remark 1. *The key idea of our algorithm is to notice that, by warm-starting the bandit, we have a smaller value of the sum of the second and the third terms of Theorem 3's bound compared to the cold-started bandit. However, as time progresses, these terms start to grow and by halting updates to the estimate of κ, the last two terms will be left as constant, hence achieving the same or smaller error compared to a cold-started bandit. Thus, assuming that our initial guess is good enough, there exists an optimal number of recalibration phase rounds $n_c^\star$ such that the sum of the last two terms achieves its minimum. This depends on the contexts chosen during the round which depends on the bandit algorithm. Minimising this value requires future research on new bandit designs.*

IV. Experiments

We conducted experiments demonstrating the effectiveness of our algorithm on artificial and OpenML datasets used by [26] and [25] previously. For simplicity, with fewer hyperparmaters, we focus on Algorithm 2 using ϵ-greedy at deployment.

a) Static Configuration: We refer to *static*, Algorithm 2 and the Section III setup of a fixed $\hat{\kappa}_t$: a small number of rounds of the deployment phase are set aside to learn scalar κ. During this recalibration phase, we choose arms uniformly at random. This ensures consistency of Algorithm 1's MAP estimate, as samples of contexts and rewards are i.i.d. The κ estimate is only calculated once at completing of recalibration, while the value of $\hat{\boldsymbol{\delta}}_t$ in Algorithm 1 is calculated per deployment round.

b) Dynamic Variation: An alternative *dynamic* approach continues to update $\hat{\kappa}_t$ in every deployment round. In this regime, $\hat{\kappa}_t$ updates indefinitely without fixing it after a predetermined number of rounds.

A. Artificial Dataset

Our artificial dataset provides ground truth not typically available. We set an arbitrary true parameter $\boldsymbol{\theta}_\star = \begin{bmatrix}0.1 & 0.3 & 0.5 & 0.7\end{bmatrix}^T$, shared between arms. Then, 10^5 rounds of data is generated with 10 arms. For each round t and arm i, a random context vector $\boldsymbol{x}_t(i)$ generated with i.i.d. sampling from $\mathcal{U}(0, 1)$. The expected reward is then calculated via $\boldsymbol{\theta}^T\boldsymbol{x}_t(i)$ before noise $\epsilon_t(i) \sim \mathcal{N}(0, 0.25^2)$ is added. As the true expected reward is known in all arms, all arms' regrets can be calculated.

The pretraining dataset is produced in a similar manner with 10^4 rounds. The same method to generate the deployment dataset is repeated again using $\boldsymbol{\mu}_\star = \frac{3}{4}\boldsymbol{\theta}_\star$ as the true parameter, leading to true $\kappa_\star = \frac{4}{3}$.

In this experiment, the hyperparameters are chosen to be $\lambda = 10$ and $\epsilon = 0.001$. Another bandit is deployed as a pretraining method. Four regimes are assigned on top of that: one-cold start regime and three warm-start regimes with varying methods of choosing $\hat{\kappa}_t$. One is oblivious to unit mismatch ($\hat{\kappa}_t = 1$, equivalent to [26]), one has dedicated first few

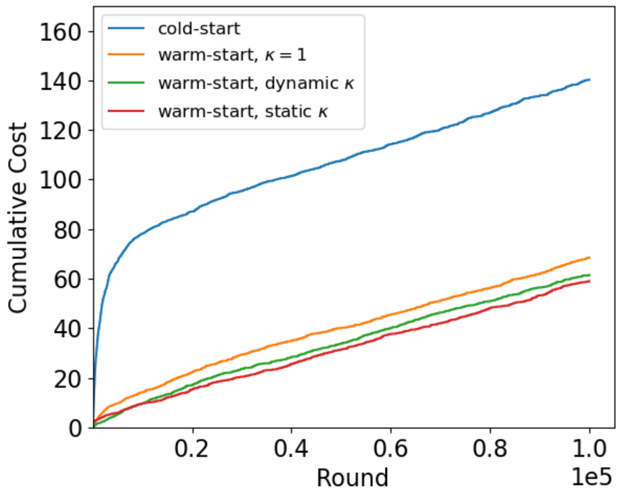

Fig. 1: Artificial data results comparing cold start, warm start with oblivious (no rescaling), dynamic, and static κ.

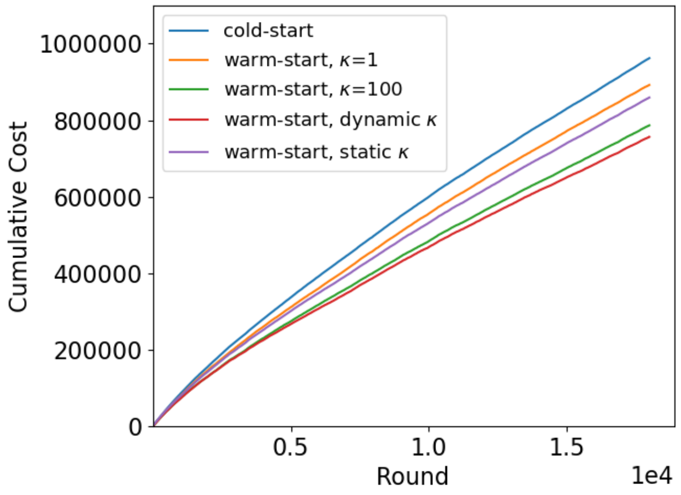

Fig. 2: Comparison between cold- and warm-start regimes on the OpenML "Letters" dataset.

rounds (static) and one is updated every round (dynamic). For the static method, we chose to calibrate the bandit at round 4. Each experiment is repeated 10 times and the average reward is recorded to account for randomness. Results are shown in Figure 1. It can be seen that the cold start regime performs the worst, followed by the warm start with oblivious unit mismatch, then the dynamic calibration and the static calibration performs the best. It is interesting to note that the dynamic calibration performs the best at the beginning before the static case overtakes it. This might be explained by the fact that in the dynamic case, the last two terms in Theorem 3 are still growing with t.

B. OpenML Dataset

The experiments of [25] and [26] use OpenML datasets. Here we use the same "Letters" dataset. Consisting of different properties or characteristics of letters along with the actual letter presented, this dataset was originally designed for classification. As is convention in bandit research, we adapt this data to a bandit setting. 26 arms are assigned to the letters 'A' to 'Z'. Contexts correspond to original features (properties of the letter presented), so that $\boldsymbol{x}_t(i) = \boldsymbol{x}_t(j)\ \forall i, j \in [26]$. Naturally, to create different expected reward predictions per arm, we use the disjoint model, necessitating the estimation of 26 latent variables $\boldsymbol{\theta}_\star(i)\ \forall i \in [26]$. The bandit will guess which letter is being presented, and at the end of the round it will be given a cost of 1 if it guesses incorrectly and 0 if it guesses correctly, that is, the cost is $c(a) = \mathbb{1}(a \neq y)$ for arm a and true label y. Since the bandit models costs instead of rewards, it is compelled to choose an action with minimum cost as opposed to maximum reward.

The first 10% of the data—2000 rounds—is used to pre-train each arm. The rewards of the remainder were scaled by 100 (simulating latent reward scaling) and used for the deployment phase.

We assessed five approaches capable of tackling this problem: cold-start, warm-start with oblivious unit mismatch (*i.e.*, $\hat{\kappa}_t = 1$), warm-start with the correct unit conversion (*i.e.*, $\hat{\kappa}_t = 100$), warm-start with $\hat{\kappa}_t$ updated in every round (dynamic) and with $\hat{\kappa}_t$ calibrated at round 100 (static). For warm start with calibrated $\hat{\kappa}_t$, whether once or every round, we learn different $\hat{\kappa}_t$ per arm. The hyperparameter used is $\epsilon = 0.0125$ following [25], while the choice of λ was observed to have little effect—matching findings of [26]—so we used $\lambda = 1$. For static, we calibrate $\hat{\kappa}_t$ at round 100 and ran a random policy prior to round 100 for reasons described above. To maintain the same effective number of rounds among all approaches, we cut the first 10% of the data for the cold-start regime, so that it starts at round 2001.

The experiment was repeated 10 times and the average cost is calculated and then plotted in Figure 2. As for the artificial data, cold start performs the worst, followed by warm starting with oblivious unit reward mismatch, and warm start with one recalibration. As we know the exact value $\kappa = 100$, we can plot the ideal graph should the bandit learner have known the true κ. This regime along with dynamic $\hat{\kappa}_t$ warm start, similar to this regime, yield the best performance.

a) Calibration Duration: The effect of calibration timing is captured in Figure 3, assessed with the Letter dataset over 10 repetitions. The more calibration, the more rounds Algorithm 2 spends following a purely random policy. On the other hand, rushed calibration observes insufficient rewards to estimate an accurate κ. In some cases, not all arms' rewards have been observed, rendering the bandit unable to estimate κ. In this case, recalibration at round 100 seems to be ideal. We have also observed that calibration at around round $4k$ yields a consistently performing disjoint model.

b) Linear Thompson Sampling: In Section III, we have stated that Algorithm 2 can be performed while deploying any one of several standard linear contextual bandits. As we presented a regret bound for Linear Thompson Sampling, it is natural to perform an experiment with LinThompson for empirical validation. In this experiment, we examine four regimes: cold start, warm start with oblivious unit reward mismatch, warm start with the correct κ *i.e.*, $\hat{\kappa}_t = 100$

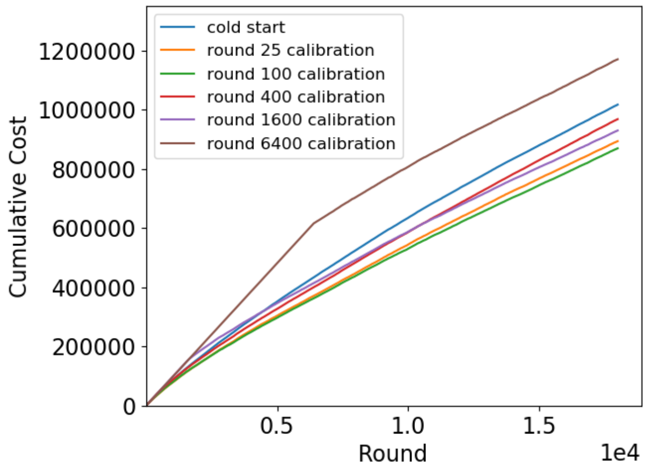

Fig. 3: Effect of calibration duration with OpenML "Letters" dataset.

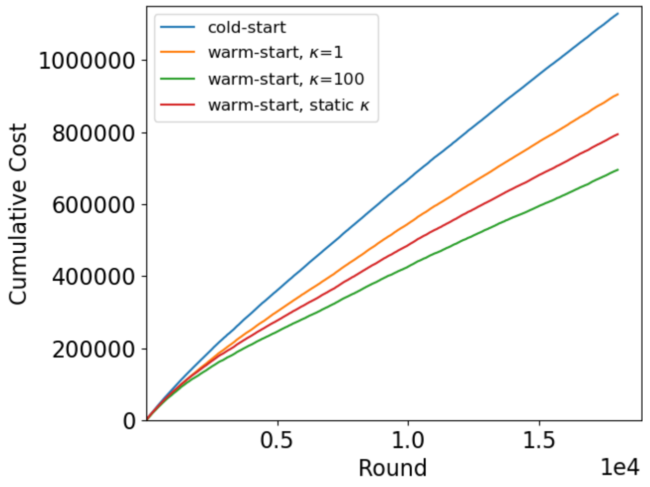

Fig. 4: Comparison between cold- and warm-start regimes with LinThompson deployed on OpenML "Letters" data.

and warm start with static recalibration at round 100. We choose hyperparameters $\lambda = 1$ as previously and $\beta = 0.2$ matching the selections by [26]. Figure 4 shows the cold-start regime performing worst, followed by warm start with $\hat{\kappa}_t = 1$. Warm starting with static recalibration performs better than these approaches. As expected, the regime with the correct κ performs best. This corresponds to an ablation study demonstrating possible gain in better estimation of κ.

C. Database Experiment

Index data structures help databases execute queries (*i.e.*, *workload*) faster by amortising query-time computation at the expense of (modest) space overhead. However, the sheer number of possible indices makes it difficult for human database administrators to choose a useful set of indices (*i.e.*, to *tune* the index). Multi-armed bandits have been employed to solve this problem for unseen workloads by identifying a performant set of indices [9], [10], [11]. However, cold-starting a database's bandit executes queries with an ineffective set of indices, for the sake of exploration, leading to poor early round performance and discouraging continued usage of the bandit. On the other hand, most database systems are embedded with a query optimiser capable of (inexpensively) estimating the cost of a query under a set of indices. This cost is however not measured in seconds—the unit used to minimise the execution time in the deployment phase. Rather, artificial units of a different scale are used to estimate work. Such a discrepancy may be remedied by the automatic scaling provided by our algorithm. In this experiment, our aim is to demonstrate the ability of our bandit to take the query optimiser's estimation as our initial guess while giving the bandit learner freedom to explore. It also demonstrates how our algorithm can adopt non-linear models via a neural network context feature embedding.

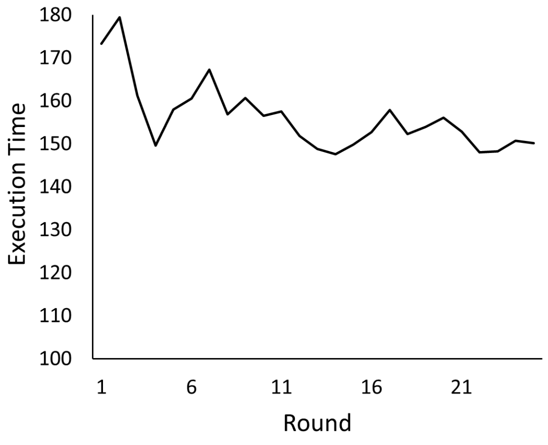

Fig. 5: Database experiment using LinUCB.

a) Method: Similar to [9], in each round, we generate a set of feasible arms (indices) based on individual queries. The raw context is fed into a neural network which outputs a cost estimate. We pre-train the network with the query optimiser's cost estimates as target outputs. Following [34], [35] and [36], we take the value of the last hidden layer as context for our bandit, and the negative of the last layer's weights as our initial guess $\hat{\boldsymbol{\mu}}$. The negative is taken since we use $-t_{exc}$ as our reward, while t_{exc} is workload execution time. The index is dropped at the end of each round. A total of 25 rounds are taken, the first of which involves no index, with ensuing execution time acting as a measuring stick on future rounds' index quality. In this experiment, we use LinUCB with exploration parameter $\alpha = 10$, $\lambda = 1$, and dynamic $\hat{\kappa}_t$.

b) Network Architecture: The neural network has three fully-connected hidden layers with 427, 161 and 61 neurons. ReLU activation functions are used across all layers but the last, which uses linear activations to mimic the linear model of our bandit setting.

c) Results: Figure 5 demonstrates our algorithm performing successfully with an accurate prior: the bandit recommends the same index deemed best by the query optimiser having different unit reward, while maintaining a moderate level of exploration. The former is important, *i.e.*, in the case where the query optimiser fails to produce accurate estimates (common for complex real-world workloads, [37], [38], [39]), the bandit will be able to mitigate misspecification via early round exploration, showing promise even in advanced database settings.

V. Conclusions

We have introduced a simple approach to accommodate reward drift in the form of rescaling rewards, when transitioning from pre-training to deployment in bandit learning. We introduce this setting, motivated by applications such as configuration management in database systems, where a bandit for recommending index selection [9] might be pre-trained on a vendor's servers, and then deployed to a server having different performance characteristics. The relationship between context and expected rewards may be maintained between pre-training and deployment, but only up to some latent constant.

We adopt a flexible framework for warm-starting common linear contextual bandits. We employ a probabilistic model of latent reward rescaling, compute exact MAP point estimates from deployment phase data, and then plug the corrected parameters in to the bandit initialisation. Two variations were considered: correcting parameters once (static) or per round (dynamic). Experiments with synthetic and OpenML datasets demonstrate the effectiveness of our approach. An experiment on database index selection demonstrates our algorithm's flexibility and ability to adopt non-linear models. Experimental results are complemented by regret analysis.

Acknowledgment

We gratefully acknowledge support from the Australian Research Council Discovery Early Career Researcher Award DE230100366, and L'Oreal-UNESCO For Women in Science '23 Fellowship.

References

[1] N. Cesa-Bianchi and G. Lugosi, *Prediction, learning, and games*. Cambridge University Press, 2006.

[2] A. Slivkins, "Introduction to multi-armed bandits," *Foundations and Trends in Machine Learning*, vol. 12, no. 1-2, pp. 1–286, 2019.

[3] T. Lattimore and C. Szepesvári, *Bandit algorithms*. Cambridge University Press, 2020.

[4] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *WWW*, pp. 661–670, 2010.

[5] L. Qin, S. Chen, and X. Zhu, "Contextual combinatorial bandit and its application on diversified online recommendation," in *SDM*, pp. 461–469, 2014.

[6] L. Tran-Thanh, S. Stein, A. Rogers, and N. R. Jennings, "Efficient crowdsourcing of unknown experts using bounded multi-armed bandits," *Artificial Intelligence*, vol. 214, pp. 89–111, 2014.

[7] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[8] D. Liu, G. Ernst, T. Murray, and B. I. P. Rubinstein, "Legion: Best-first concolic testing," in *ASE*, pp. 54–65, 2020.

[9] R. M. Perera, B. Oetomo, B. I. P. Rubinstein, and R. Borovica-Gajic, "DBA bandits: Self-driving index tuning under ad-hoc, analytical workloads with safety guarantees," in *ICDE*, pp. 600–611, 2021.

[10] R. M. Perera, B. Oetomo, B. I. P. Rubinstein, and R. Borovica-Gajic, "HMAB: self-driving hierarchy of bandits for integrated physical database design tuning," *VLDB*, vol. 16, no. 2, pp. 216–229, 2022.

[11] R. M. Perera, B. Oetomo, B. I. P. Rubinstein, and R. Borovica-Gajic, "No DBA? No regret! Multi-armed bandits for index tuning of analytical and HTAP workloads with provable guarantees," *TKDE*, vol. 35, no. 12, pp. 12855–12872, 2023.

[12] R. Marcus, P. Negi, H. Mao, N. Tatbul, M. Alizadeh, and T. Kraska, "Bao: Learning to steer query optimizers," in *SIGMOD*, 2021.

[13] T. Kaftan, M. Balazinska, A. Cheung, and J. Gehrke, "Cuttlefish: A lightweight primitive for adaptive query processing," 2018. arXiv:1802.09180.

[14] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," *Expert Systems with Applications*, vol. 41, no. 4, pp. 2065–2073, 2014.

[15] A. Kazerouni, M. Ghavamzadeh, Y. Abbasi Yadkori, and B. Van Roy, "Conservative contextual linear bandits," *NeurIPS*, vol. 30, 2017.

[16] P. Shivaswamy and T. Joachims, "Multi-armed bandit problems with history," in *AISTATS*, pp. 1046–1054, 2012.

[17] X. Wu, S. Cetintas, D. Kong, M. Lu, J. Yang, and N. Chawla, "Learning from cross-modal behavior dynamics with graph-regularized neural contextual bandit," WWW '20, p. 995–1005, 2020.

[18] L. Wang, C. Wang, K. Wang, and X. He, "Biucb: A contextual bandit algorithm for cold-start and diversified recommendation," in *2017 IEEE International Conference on Big Knowledge (ICBK)*, pp. 248–253, IEEE, 2017.

[19] C. Gentile, S. Li, and G. Zappella, "Online clustering of bandits," in *ICML*, pp. 757–765, 2014.

[20] S. Li, A. Karatzoglou, and C. Gentile, "Collaborative filtering bandits," in *SIGIR*, pp. 539–548, 2016.

[21] N. Korda, B. Szorenyi, and S. Li, "Distributed clustering of linear bandits in peer to peer networks," in *ICML*, pp. 1301–1309, 2016.

[22] K. Mahadik, Q. Wu, S. Li, and A. Sabne, "Fast distributed bandits for online recommendation systems," in *Proceedings of the 34th ACM International Conference on Supercomputing*, pp. 1–13, 2020.

[23] D. Bouneffouf, S. Parthasarathy, H. Samulowitz, and M. Wistub, "Optimal exploitation of clustering and history information in multi-armed bandit," in *IJCAI*, pp. 2016–2022, 2019.

[24] Y. Li, H. Xie, Y. Lin, and J. C. Lui, "Unifying offline causal inference and online bandit learning for data driven decision," in *TheWebConf*, p. 2291–2303, 2021.

[25] C. Zhang, A. Agarwal, H. D. Iii, J. Langford, and S. Negahban, "Warm-starting contextual bandits: Robustly combining supervised and bandit feedback," in *ICML*, pp. 7335–7344, 2019.

[26] B. Oetomo, R. M. Perera, R. Borovica-Gajic, and B. I. P. Rubinstein, "Cutting to the chase with warm-start contextual bandits," in *ICDM*, pp. 459–468, 2021.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[28] N. Jaitly, P. Nguyen, A. Senior, and V. Vanhoucke, "Application of pretrained deep neural networks to large vocabulary speech recognition," in *INTERSPEECH*, 2012.

[29] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training," in *AISTATS*, pp. 153–160, 2009.

[30] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?," in *AISTATS*, pp. 201–208, 2010.

[31] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *ICML*, pp. 127–135, 2013.

[32] M. Abeille, A. Lazaric, *et al.*, "Linear Thompson sampling revisited," *Electronic Journal of Statistics*, vol. 11, no. 2, pp. 5165–5197, 2017.

[33] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," *NeurIPS*, vol. 24, pp. 2312–2320, 2011.

[34] T. Zahavy and S. Mannor, "Deep neural linear bandits: Overcoming catastrophic forgetting through likelihood matching," *arXiv preprint arXiv:1901.08612*, 2019.

[35] C. Riquelme, G. Tucker, and J. Snoek, "Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling," in *ICLR*, 2018.

[36] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. Patwary, M. Prabhat, and R. Adams, "Scalable bayesian optimization using deep neural networks," in *ICML*, pp. 2171–2180, 2015.

[37] R. Borovica-Gajic, S. Idreos, A. Ailamaki, M. Zukowski, and C. Fraser, "Smooth scan: robust access path selection without cardinality estimation," *The VLDB Journal*, vol. 27, no. 4, pp. 521–545, 2018.

[38] R. Borovica-Gajic, S. Idreos, A. Ailamaki, M. Zukowski, and C. Fraser, "Smooth scan: Statistics-oblivious access paths," in *ICDE*, 2015.

[39] R. Borovica, I. Alagiannis, and A. Ailamaki, "Automated physical designers: what you see is (not) what you get," in *DBTest@SIGMOD*, 2012.