CHANGE DETECTION METHODS FOR NON-STATIONARY STOCHASTIC LINEAR BANDITS

Linghang Meng[⋆] *and Jie Wu*[⋆†]

[⋆]Cloud Computing Research Institute, China Telecom
[†]Department of Computer and Information Sciences, Temple University

ABSTRACT

This paper investigates the linear bandit problem in non-stationary environments, focusing on the piecewise stationary setting where the environment undergoes abrupt changes at unknown time points. In such settings, the lower bound for the dynamic regret is $\Omega(d\sqrt{MT})$, where d is the dimension of the features, M is the number of stationary segments, and T is the time horizon. We employ the change detection-based framework to address this problem and design two algorithms, LBCD-AW and LBCD-CUSUM, by incorporating different change detection techniques. We provide theoretical analysis for both algorithms and show that the proposed LBCD-AW achieves nearly optimal regret in T. Simulation experiments demonstrate that our algorithms outperform existing methods in environments with abrupt changes and exhibit comparable performance to state-of-the-art algorithms in environments with gradual changes.

***Index Terms*—** Change Detection, Linear Bandits, Nonstationary Bandits, Performance Analysis

1. INTRODUCTION

The multi-armed bandit (MAB) problem is a fundamental framework for sequential decision-making under uncertainty. In this problem, an agent is tasked with selecting an arm from a set of available options at each time step to maximize the cumulative reward over a time horizon [1]. A key challenge in MAB lies in balancing the trade-off between exploration and exploitation. This balance makes MAB models particularly well-suited for sequential decision-making tasks where decisions must be made in real time with incomplete information. The versatility of MAB has led to its successful application in a wide range of domains [2, 3, 4, 5, 6].

The classical bandit problem assumes a stationary environment in which the reward distributions of arms remain constant over time. However, real-world environments are inherently non-stationary [7, 8]. Existing studies on non-stationary bandits typically distinguish between two types of environmental changes: gradual shifts and abrupt shifts. The former refers to scenarios where the underlying reward distributions evolve slowly over time, requiring learning algorithms to continuously adapt to these incremental changes. In contrast, abrupt shifts involve sudden, discrete changes in the environment, necessitating algorithms that can rapidly update their estimates of the environment once a change occurs.

For gradual changes, the extent of variation in the environment is often characterized by the total variation of environment parameter over the time horizon, defined as $B_T = \sum_{t=1}^{T-1} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}\|_2$. Forgetting-based methods are particularly well suited to such settings. These methods rely on discarding outdated observations to better adapt to recent changes. Some approaches implement this by discounting past rewards [9], while others employ sliding-window techniques [10]. These mechanisms can also be combined with other bandit algorithms, such as Thompson Sampling [11] and Exp3 [12]. By smoothly balancing past and recent information, these methods can achieve near-optimal regret with respect to both T and B_T. On the other hand, change-detection-based methods are suited for environments with abrupt changes [13, 14, 15, 16, 17, 18, 19]. Such approaches typically combine a bandit algorithm with a change detection module. When a change is detected, the algorithm is restarted and the estimate of the environment is updated accordingly. However, a key limitation of these approaches lies in the need for forced exploration across all actions. Since a bandit algorithm naturally tends to exploit a small subset of actions, it may fail to detect changes in other actions. To ensure that all changes can be detected with small delay, these methods impose forced exploration over all actions. This strategy, however, may result in additional linear regret.

This paper focuses on the stochastic linear bandit problem. In the linear bandit setting, most existing works employ forgetting-based methods to address gradual changes [20, 21, 22]. Under abrupt changes, these methods yield a regret bound of $\widetilde{\mathcal{O}}(B_T^{1/4}T^{3/4})$, which exceeds the theoretical lower bound of $\Omega(d^{2/3}B_T^{1/3}T^{2/3})$. This gap arises because forgetting strategies cannot eliminate the influence of outdated information quickly enough after a sudden shift. The work of [23] also adopts a change-detection perspective by maintaining multiple slave bandit models and tracking their reward estimation accuracy. When environmental changes degrade estimation quality, new models are generated while outdated ones are discarded. However, subsequent studies [21, 22] have shown that this approach performs well only under low-noise settings, while its effectiveness deteriorates in noisier environments. In addition, [24] proposed a black-box algorithm that employs a prior-free change detection mechanism. Their theoretical results can specialize to the linear bandit setting and yield near-optimal regret. Nonetheless, as highlighted by the recent work [25], the detection mechanism in [24] requires an extremely long time horizon to be triggered. In practice, its performance is comparable to random-restart strategies.

This paper reveals two key insights beyond existing studies. First, the linear bandit setting induces a correlation among the arms through the shared environment parameter. When the environment changes, the rewards of all actions are affected. As a result, it is unnecessary to enforce forced exploration of all actions to detect environmental changes, making it possible to achieve sub-linear regret. Second, our experiment results show that even in a gradual change environment, a change detection algorithm can still identify environmental shifts as long as the drift of the environment parameter vector exceeds a threshold. In this paper, we employ the linear bandits with change detection (LBCD) framework. Within this framework, we design a simple yet effective detection method, the Averaged Window (AW) method, Other detection methods, such

Algorithm 1 Linear bandits with change detection (LBCD)

Require: Change detection method $\mathcal{F}$, window length W
Initialize linearUCB method.
Initialize reward estimation of all arms using INITEST
for $t = 1, 2, \cdots, T$ **do**
  Choose arm $\boldsymbol{A}_t = \boldsymbol{a}_k$ using linearUCB
  Play arm k and recieve reward X_t
  Update the linearUCB sub-routine using $\boldsymbol{a}_k$ and X_t
  if arm k's estimation not done, i.e. $w_k < W$ **then**
    Update reward estimate of arm k:
      $r_k \leftarrow (r_k * w_k + X_t)/(w_k + 1)$, $w_k \leftarrow w_k + 1$
    if arm k's estimation is done, i.e. $w_k = W$ **then**
      Initialize change detection $\mathcal{F}_k$ for arm k
  else
    Detect change on arm k: IsChange $= \mathcal{F}_k(X_t)$
    if IsChange **then**
      Re-initialize linearUCB
      Re-initialize estimation using INITEST
Procedure INITEST
for each arm index $k = 1, \ldots, K$ **do**
  Initialize counter and estimate: $w_k \leftarrow 0, r_k \leftarrow 0$

as CUSUM, can also be incorporated. We establish regret bounds for the proposed LBCD-AW and LBCD-CUSUM algorithms, the regret of LBCD-AW is nearly optimal in T.

2. PROBLEM FORMULATION AND LOWER BOUND

The environment provides the learner with a finite set of available actions $\mathcal{A} \subset \mathbb{R}^d$. At each round $t \geq 1$, the learner chooses an action $\boldsymbol{A}_t \in \mathcal{A}$ and receives a reward X_t such that $X_t = \langle \boldsymbol{A}_t, \boldsymbol{\theta}_t \rangle + \eta_t$, where $\boldsymbol{\theta}_t \in \mathbb{R}^d$ is the unknown environment parameter and η_t is a Gaussian noise with variance σ^2. The environment parameter $\boldsymbol{\theta}_t$ may change, requiring the learner to track the environmental shifts and adapt its action selection accordingly. The goal of the learner is to minimize the dynamic regret $R(T) = \sum_{t=1}^{T} \max_{\boldsymbol{a} \in \mathcal{A}} \langle \boldsymbol{a} - \boldsymbol{A}_t, \boldsymbol{\theta}_t \rangle$. We focus on the piecewise stationary setting, where the environment undergoes abrupt changes at certain unknown points in time. Formally, the time horizon T is divided into M stationary segments. We use t_i to denote the i-th changepoint. For all time steps in the i-th segment, $\boldsymbol{\theta}_t = \boldsymbol{\theta}_i$. The changepoints $\{t_2, \ldots, t_M\}$ are unknown to the learner. In this setting, the learner must simultaneously identify the optimal actions and adapt to the changes at the boundaries of these segments.

We can establish the following regret lower bound for the piecewise stationary linear bandit problem, which provides insights into the best possible regret achievable. Proofs for all the theorems and lemmas are available online[1].

Theorem 1. *For any piecewise stationary linear bandit problem with $M - 1$ change points, the dynamic regret of any linear bandit algorithm satisfies $R(T) = \Omega(d\sqrt{MT})$, where d is the dimension and T is the time horizon.*

3. PROPOSED ALGORITHMS

We combine change detection techniques with the linearUCB algorithm to address the challenges of non-stationary environments. The framework is summarized in Algorithm 1. It consists of two kinds of sub-routines. The linearUCB sub-routine estimate the environment parameters and select actions based on the principle of optimism in the face of uncertainty. Alongside this, we estimate the reward mean for each action using the first W observations for that action. After the mean reward estimate for the action is calculated with enough observations, a change detection sub-routine is launched and monitors new reward feedbacks. When a change detection sub-routine alerts a change, the framework resets all the sub-routines.

Compared to algorithms in [17, 18, 19], which introduce change detection techniques into the piecewise stationary MAB problem, Algorithm 1 has a significant distinction: it does not require the additional steps of forced exploration of all actions to ensure timely detection of environmental changes. This advantage arises from the inherent properties of the linear bandit setting. Specifically, when the environment's parameter shifts from $\boldsymbol{\theta}_i$ to $\boldsymbol{\theta}_{i+1}$, the mean rewards of all actions change simultaneously. Consequently, even if the linearUCB sub-routine restricts its action selection to a few high-reward actions, the mean rewards of these actions will still reflect the environment change. As a result, the change detection sub-routines can raise alerts without the need for forced exploration of all actions. The elimination of forced exploration reduces the selection of sub-optimal actions and thus reduces the algorithm's regret.

We introduce two change detection methods, both of which can be integrated into Algorithm 1 as subroutines for detecting changes. The Averaged Window (AW) Method monitors potential changes in the environment by comparing the running average of rewards within a window of length W to the reward mean estimated at the start of the stationary segment. When the deviation exceeds a pre-defined threshold b, the method signals a potential change in the environment. The Cumulative Sum (CUSUM) Method [26] detects changes by maintaining cumulative statistics that track the deviation of observed rewards from the reference mean. Since the change direction is unknown, the reward may increase or decrease after a change. So we need to detect changes in both directions. At each round we first calculate the deviations $s^+ = X_t - \hat{\mu} - \epsilon, s^- = \hat{\mu} - \epsilon - X_t$, where ϵ is a noise tolerance parameter that helps the algorithm ignore minor fluctuations. Then we calculate the cumulative statistics $g^+ \leftarrow \max\left(g^+ + s^+, 0\right), g^- \leftarrow \max\left(g^- + s^-, 0\right)$. If either statistic exceeds threshold h, the algorithm signals a change.

Algorithm 1 can becomes computationally inefficient when the action space is large. To address this issue, we can cluster the actions based on their contextual features under the assumption that actions with similar contexts yield similar rewards. Within each cluster, we aggregate samples from all actions to estimate a shared reward mean. When the environment changes and the optimal action shifts from one cluster to another, the mean reward of the affected cluster also changes. Then we can detect change on the estimated cluster-level reward mean. This clustering-based approach can significantly improve the efficiency of the algorithm.

4. REGRET BOUNDS OF THE ALGORITHMS

The regret of the framework presented in Algorithm 1 is primarily determined by the false alarms and detection delays in change detection. Figure 1 illustrates the dynamics of environment changes and the corresponding change detection results during the execution of the algorithm. The horizontal axis represents the timeline. Black ticks on the timeline indicate the changepoints t_i. The environment experiences $M - 1$ changepoints, resulting in M stationary segments. We use $\mathcal{I}_i$ to denote these segments. Blue ticks represent the detection points where the algorithm issues change detection signals, denoted as τ_j. Each detection point triggers a restart of the algorithm, with N denoting the total number of restarts. We use $\mathcal{J}_j$ to

[1] https://zenodo.org/records/17149159

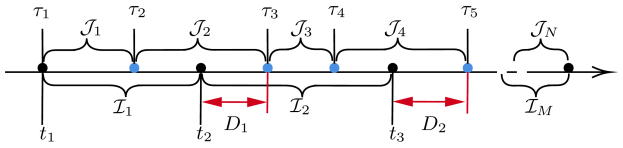

Fig. 1: Illustration of environmental changes and change detections.

denote the j-th restart period. It is evident that the number of total false alarms is $F = N - M$. After each changepoint, the algorithm experiences a delay represented as D_i. We have $D_i, D = \sum_{i=1}^{M} D_i$. Using the definitions of F and D provided here, we can derive the following regret upper bound for the framework.

Theorem 2. *By setting the confidence parameter of the linearUCB subroutine as δ, with probability at least $1-2\delta$, the dynamic regret of Algorithm 1 is bounded by $C_1 d\sqrt{(M+F)T}\log((1+L^2)T/(M+F)) + C_2\sqrt{d(M+D+F)^3}$, where d is the dimension of the contextual feature, $C_1 = 2\sqrt{2}(\frac{\sqrt{\lambda}S}{\sqrt{d\log 2}} + \sqrt{1+\frac{2\log(1/\delta)}{d\log 2}})$, $C_2 = \frac{8L^2S}{3\sqrt{\lambda}}$ are constants, and λ is the regularization coefficient of the LinearUCB algorithm.*

This theorem establishes the relationship between the dynamic regret of Algorithm 1 and key variables such as T, M, D, and F. The constants C_1 and C_2 depend on L, S, d, λ and δ. Typically, L, S, d and λ are constants independent of T, and δ is set as $1/T$. In this case, we have $C_1 \sim \sqrt{\log T}$ and $R(T) = \widetilde{\mathcal{O}}(\sqrt{d}(M+D+F)^{\frac{3}{2}} + d\sqrt{(M+F)T})$. Comparing this upper bound to Theorem 1, it is evident that both D and F must be controlled to achieve optimal regret. Typically, there is a trade-off between detection delay and false alarms in the design of change detection algorithms.

4.1. Regret Bound of LBCD-AW

For the proposed LBCD-AW algorithm, consider the setting where the Gaussian noise has a variance of σ^2, and the parameters are b, W, the following lemma provides an upper bound on the false alarms.

Lemma 1. *For the LBCD-AW algorithm, the expected total number of false alarms F satisfies $\mathbb{E}(F) \leq T\exp(-(b/\sqrt{2\sigma^2/W})^2/2)$. Moreover, if $b \geq 2\sigma\sqrt{\log(T^2/\alpha M)/W}$, we have $\Pr(F \geq \alpha M) \leq 1/T$, where $\alpha > 0$ is a constant to be determined later.*

As shown in this lemma, the expected number of false alarms decreases exponentially with increasing detection threshold b. The constant α can take any positive value. On average, α represents the expected number of false alarms within each stationary segment. If we expect only one false alarm for each segment, the threshold can be set as $b = 2\sigma\sqrt{\log(T^2/M)/W}$.

To ensure low detection delay, the change in reward means must exceed a certain threshold. Specifically, we assume for any arm $k \in [K]$ and any stationary segment $i \in [M-1]$, the change is sufficiently large, such that $\left|a_k^T\theta_i - a_k^T\theta_{i+1}\right| > \Delta$, where Δ represents the minimum required change in the expected reward for all arms. Under this assumption, the following lemma provides an upper bound on the total detection delay.

Lemma 2. *When $\Delta \geq b + \frac{2\sigma}{\sqrt{W}}\sqrt{\log((M-1)\beta T/2)}$, the total detection delay satisfies $\Pr(D \geq (M-1)(KW+\beta)) \leq 1/T$, where $\beta \geq 1$ is a constant to be determined later.*

Since the bandit algorithm selects arms in a random way, in the worst-case scenario, each arm needs to be selected sufficiently many times to detect the change. Consequently, the delay bound provided in this lemma is proportional to KW. The parameter β in the theorem accounts for additional interactions beyond KW. In practice,

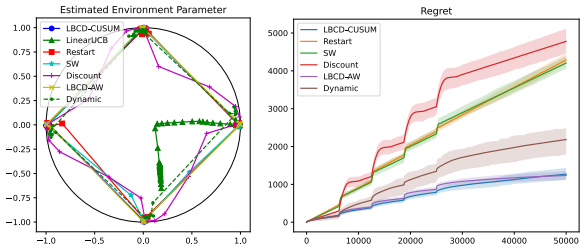

Fig. 2: Experiment results in the abrupt change environment.

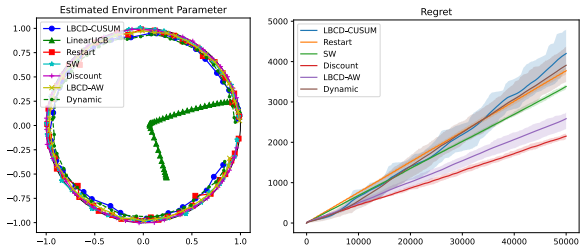

Fig. 3: Experiment results in the gradual change environment.

after running for enough steps, the LinearUCB module tends to focus on selecting a small subset of actions rather than sampling all actions evenly. As a result, the actual delay is likely to be much smaller than the bound in the theorem. By substituting the conclusions of Lemma 1 and Lemma 2 into Theorem 2, we can derive the following upper bound.

Theorem 3. *When $M \leq \sqrt{T}$, we can set the sliding window as $W = 16\sigma^2/\Delta^2 \cdot \log(T^2/M)$ and the threshold as $b = \Delta/2$, then the regret of LBCD-AW is bounded by $C_1 d\sqrt{2MT}\log((1+L^2)T/M) + C_2\sqrt{dM^3(3+32K(\sigma/\Delta)^2\log T)^3}$, where C_1, C_2 are constants defined in Theorem 2.*

Lemma 1 requires b to be sufficiently large to control the number of false alarms, while Lemma 2 demands that Δ exceeds b by a certain margin to control the detection delay. To satisfy both requirements simultaneously, in this theorem, we set W to be proportional to σ^2/Δ^2, and b is chosen as $\Delta/2$. If both K and σ/Δ are relatively small compared to T, the derived bound can be written as $R(T) = \mathcal{O}(\sqrt{dM^3(\log T)^3} + d\sqrt{MT}\log(T/M))$, which is nearly optimal in d and T.

4.2. Regret Bound of LBCD-CUSUM

The CUSUM algorithm accumulates the deviations between the current observations and a reference value. In this work, we use the estimated mean as the reference value. The estimation error directly affects the expected deviation step in each round. By choosing W large enough, we can ensure that the estimation error is, with high probability, no greater than $\Delta/4$. Then, by setting $\epsilon \sim \Delta/2$, we can guarantee that the expected deviation step is at most $-\Delta/4$ under the no-change regime, and at least $\Delta/4$ after a change occurs. Under this condition, by setting h properly, we can control both the detection delay and the false alarm, which leads to the result below.

Theorem 4. *For the LBCD-CUSUM algorithm, suppose the reference window is sufficiently large. Set the algorithm parameters as $h = 2\sigma^2/\Delta \cdot \log T$ and $\epsilon = \Delta/4$. Then, the false alarm satisfies $\Pr(F=0) \geq 1 - 1/T$, and the detection delay satisfies $\Pr\left(D \leq \frac{2MK\sigma}{\Delta}\left(\sqrt{T\log T} + \frac{2\sigma}{\Delta}\log T\right)\right) \geq 1 - 1/T$. Consequently, with high probability the cumulative regret is upper bounded by $\widetilde{\mathcal{O}}\left(d\sqrt{MT}\right) + \widetilde{\mathcal{O}}\left(d^{\frac{1}{2}}M^{\frac{3}{2}}K^{\frac{3}{2}}\left(\frac{\sigma}{\Delta}\right)^{\frac{3}{2}}T^{\frac{3}{4}}\right)$.*

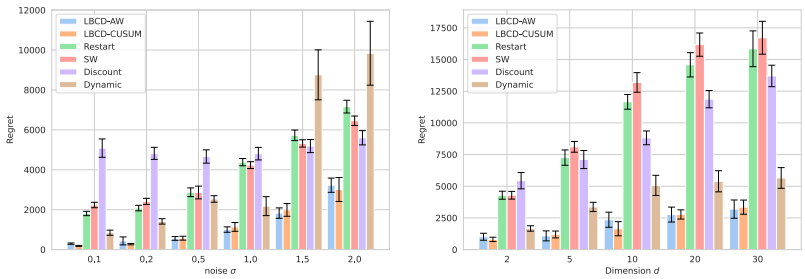

Fig. 4: Impact of noise and feature dimension.

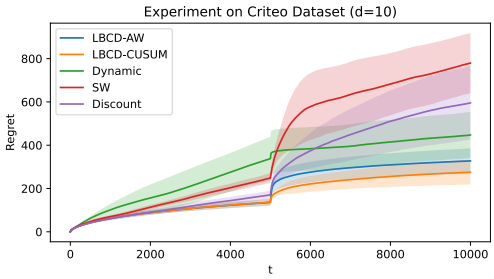

Fig. 5: Experiment on Criteo Live dataset.

5. EXPERIMENTS

We compared the proposed algorithm against several baseline methods, including Sliding Window linearUCB (SW, [20]), Discounted linearUCB (Discount, [21]), and Restarted linearUCB (Restart, [22]). These methods are based on forgetting mechanisms. They use the total variation B_T of the environment to quantify the magnitude of changes and tune parameters accordingly. Therefore, in our experiments, once the total variation of the environment is simulated, we calculate B_T and set the parameters of the baseline algorithms to their optimal values. Besides, we also compared the proposed algorithm with the dynamic linearUCB (Dynamic) algorithm [23]. We used the publicly available implementation provided by the authors. We also conducted experiments using the traditional linearUCB [27, 28]. The two algorithms proposed in this paper, based on averaged window and CUSUM, are labeled as LBCD-AW and LBCD-CUSUM, respectively.

5.1. Abruptly changing environment

In this experiment, during $t \in [1, 25000]$ the environment parameters changed abruptly from $(1, 0)$ to $(0, 1)$, then to $(-1, 0)$, and finally to $(0, -1)$. During $t \in [25001, 50000]$, the environment parameters returned to $(1, 0)$ and remained static. We randomly generated 50 candidate actions on the unit circle. The noise level was set to $\sigma = 1$, which is relatively high compared to the mean rewards. For LBCD-AW, we set $W = 50$ and $b = 0.8$. For LBCD-CUSUM, we set $W = 50$, $\epsilon = 1$, and $h = 10$. The average cumulative regret over 100 trials is shown in the right subplot of Figure 2. The left subplot of Figure 2 shows the changes in the estimated environment parameters for each algorithm. Specifically, the estimated parameters were sampled every 1000 rounds, averaged over 100 repeted trials, and plotted sequentially over time. In the left subplot of Figure 2, after each abrupt change, the lines corresponding to LBCD-AW and LBCD-CUSUM quickly jumps to the new positions. In contrast, SW, Restart, and Discount algorithms lag behind in tracking the abrupt changes. Consequently, as shown in the right subplot, the cumulative regrets of LBCD-AW and LBCD-CUSUM are significantly lower than that of other algorithms. The Dynamic algorithm can also detect changes in the environment parameters relatively quickly. However, compared to the method proposed in this paper, the convergence of its estimate is slower, resulting in a higher regret than the proposed algorithm.

5.2. Gradually changing environment

The environment parameters start at $(0, 0)$ and gradually drift counterclockwise along the unit circle, completing one full rotation. All other settings remain the same as in the first experiment. For LBCD-AW, we set $W = 50$ and $b = 0.5$. For LBCD-CUSUM, we set $W = 50$, $\epsilon = 1$, and $h = 10$. The experiment results are shown in Figure 3. From the left subplot, we observe that the yellow and blue lines, corresponding to LBCD-AW and LBCD-CUSUM, align well with the unit circle, indicating that the proposed algorithms can effectively capture the overall trend of the parameter changes.

5.3. Impact of noise and feature dimension

For the abrupt change setup, we analyzed the impact of noise level σ and feature dimension d using simulated data. Specifically, we set $\sigma \in [0.1, 0.2, 0.5, 1.0, 1.5, 2.0]$ and $d \in [2, 5, 10, 20, 30]$, and recorded the average regret of each algorithm over 50 independent runs. The results are presented in Figure 4. As shown in the figure, the proposed LBCD-AW and LBCD-CUSUM consistently outperform all baselines across different settings.

5.4. Experiment on Criteo Live Dataset

The Criteo live traffic dataset records 30 days of Criteo live traffic data. Each record corresponds to a banner displayed to a user. Detailed information about each banner is provided, based on which the bandit model needs to predict whether the banner will be clicked. Following similar setting in [21], we employ PCA to reduce the feature dimension to 10. From all the data, 10,000 positive samples and 10,000 negative samples are randomly selected to form a sample pool. In each round, 100 candidates are selected from the sample pool, and the model is tasked with choosing the optimal action from these candidates. To enhance the efficiency, we use the k-means algorithm to cluster all context features from the sample pool into 100 clusters. Actions in the same cluster exhibit similar rewards, and we detect changes in the rewards of these clusters. At $t = 5000$, a shift in user preferences occurs, requiring the bandit model to adapt to the change. The results are presented in Figure 5, from which it can be observed that the LBCD-CUSUM and LBCD-AW algorithm proposed in this paper demonstrates the best performance.

6. CONCLUSIONS

This paper addresses the problem of non-stationary linear bandits, focusing on piecewise stationary settings with limited abrupt changes. We employ the change detection-based algorithmic framework and introduce two specific algorithms: LBCD-AW and LBCD-CUSUM. We provide detailed theoretical analysis for the proposed framework. Regret upper bounds are derived for LBCD-AW and LBCD-CUSUM. It is proved that under appropriate parameter settings, the LBCD-AW algorithm achieves the optimal regret rate. We conduct experiments on both simulated data and real-world data. Experimental results show that the proposed methods significantly outperform other algorithms in abrupt change environments and perform well in gradual change environments.

7. REFERENCES

[1] Tor Lattimore and Csaba Szepesvári, *Bandit algorithms*, Cambridge University Press, 2020.

[2] Eric M Schwartz, Eric T Bradlow, and Peter S Fader, "Customer acquisition via display advertising using multi-armed bandit experiments," *Marketing Science*, vol. 36, no. 4, pp. 500–522, 2017.

[3] Sofía S Villar, Jack Bowden, and James Wason, "Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges," *Statistical science: a review journal of the Institute of Mathematical Statistics*, vol. 30, no. 2, pp. 199, 2015.

[4] Tor Lattimore, Koby Crammer, and Csaba Szepesvári, "Linear multi-resource allocation with semi-bandit feedback," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[5] I-Hong Hou, "Distributed no-regret learning for multi-stage systems with end-to-end bandit feedback," in *Proceedings of the Twenty-fifth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2024, pp. 41–50.

[6] Nida Zamir and I-Hong Hou, "Deep index policy for multi-resource restless matching bandit and its application in multi-channel scheduling," in *Proceedings of the Twenty-fifth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2024, pp. 71–80.

[7] Quang Minh Nguyen and Eytan Modiano, "Learning to schedule in non-stationary wireless networks with unknown statistics," in *Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2023, pp. 181–190.

[8] Lihong Li, Wei Chu, John Langford, and Robert E Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.

[9] Levente Kocsis and Csaba Szepesvári, "Discounted ucb," in *2nd PASCAL Challenges Workshop*, 2006, vol. 2, pp. 51–134.

[10] Aurélien Garivier and Eric Moulines, "On upper-confidence bound policies for switching bandit problems," in *International conference on algorithmic learning theory*. Springer, 2011, pp. 174–188.

[11] Francesco Trovo, Stefano Paladino, Marcello Restelli, and Nicola Gatti, "Sliding-window thompson sampling for non-stationary settings," *Journal of Artificial Intelligence Research*, vol. 68, pp. 311–364, 2020.

[12] Omar Besbes, Yonatan Gur, and Assaf Zeevi, "Stochastic multi-armed-bandit problem with non-stationary rewards," *Advances in neural information processing systems*, vol. 27, 2014.

[13] Robin Allesiardo, Raphaël Féraud, and Odalric-Ambrym Maillard, "The non-stationary stochastic multi-armed bandit problem," *International Journal of Data Science and Analytics*, vol. 3, pp. 267–283, 2017.

[14] Réda Alami, Odalric Maillard, and Raphael Féraud, "Memory bandits: a bayesian approach for the switching bandit problem," in *NIPS 2017-31st conference on neural information processing systems*, 2017.

[15] Fang Liu, Joohyun Lee, and Ness Shroff, "A change-detection based framework for piecewise-stationary multi-armed bandit problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018, vol. 32.

[16] Lilian Besson and Emilie Kaufmann, "The generalized likelihood ratio test meets klucb: an improved algorithm for piecewise non-stationary bandits," in *Proceedings of Machine Learning Research vol XX*, 2019, vol. 1, p. 35.

[17] Lilian Besson, Emilie Kaufmann, Odalric-Ambrym Maillard, and Julien Seznec, "Efficient change-point detection for tackling piecewise-stationary bandits," *Journal of Machine Learning Research*, vol. 23, no. 77, pp. 1–40, 2022.

[18] Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie, "Nearly optimal adaptive procedure with change detection for piecewise-stationary bandit," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 418–427.

[19] Subhojyoti Mukherjee and Odalric-Ambrym Maillard, "Distribution-dependent and time-uniform bounds for piecewise iid bandits," *arXiv preprint arXiv:1905.13159*, 2019.

[20] Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu, "Learning to optimize under non-stationarity," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1079–1087.

[21] Yoan Russac, Claire Vernade, and Olivier Cappé, "Weighted linear bandits for non-stationary environments," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[22] Peng Zhao, Lijun Zhang, Yuan Jiang, and Zhi-Hua Zhou, "A simple approach for non-stationary linear bandits," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 746–755.

[23] Qingyun Wu, Naveen Iyer, and Hongning Wang, "Learning contextual bandits in a non-stationary environment," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 495–504.

[24] Chen-Yu Wei and Haipeng Luo, "Non-stationary reinforcement learning without prior knowledge: An optimal black-box approach," in *Conference on learning theory*. PMLR, 2021, pp. 4300–4354.

[25] Argyrios Gerogiannis, Yu-Han Huang, and Venugopal Veeravalli, "Is prior-free black-box non-stationary reinforcement learning feasible?," in *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Emtiyaz Khan, Eds. 03–05 May 2025, vol. 258 of *Proceedings of Machine Learning Research*, pp. 2692–2700, PMLR.

[26] Ewan S Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[27] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári, "Improved algorithms for linear stochastic bandits," *Advances in neural information processing systems*, vol. 24, 2011.

[28] Peter Auer, "Using confidence bounds for exploitation-exploration trade-offs," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 397–422, 2002.

[29] Ahmed Touati and Pascal Vincent, "Efficient learning in non-stationary linear markov decision processes," *arXiv preprint arXiv:2010.12870*, 2020.