Adaptive Experimentation with Delayed Binary Feedback

Zenan Wang
JD.com
Mountain View, CA, USA
wangzenan5@jd.com

Carlos Carrion
JD.com
Mountain View, CA, USA
carlos.carrion@jd.com

Xiliang Lin
JD.com
Mountain View, CA, USA
xiliang.lin@jd.com

Fuhua Ji
JD.com
Beijing, China
jifuhua@jd.com

Yongjun Bao
JD.com
Beijing, China
baoyongjun@jd.com

Weipeng Yan
JD.com
Beijing, China
paul.yan@jd.com

ABSTRACT

Conducting experiments with objectives that take significant delays to materialize (e.g., conversions, add-to-cart events, etc.) is challenging. Although the classical “split sample testing” is still valid for the delayed feedback, the experiment will take longer to complete, which also means spending more resources on worse-performing strategies due to their fixed allocation schedules. Alternatively, adaptive approaches such as “multi-armed bandits” are able to effectively reduce the cost of experimentation. But these methods generally cannot handle delayed objectives directly out of the box. This paper presents an adaptive experimentation solution tailored for delayed binary feedback objectives by estimating the real underlying objectives before they materialize and dynamically allocating variants based on the estimates. Experiments show that the proposed method is more efficient for delayed feedback compared to various other approaches and is robust in different settings. In addition, we describe an experimentation product powered by this algorithm. This product is currently deployed in the online experimentation platform of JD.com, a large e-commerce company and a publisher of digital ads.

1 INTRODUCTION

Experimentation has been widely used in the tech industry and especially for content optimization in websites and online advertising. A typical experiment system will assign users or requests to different groups and display one variant of contents among several possibilities accordingly. Then users’ interactions with the content such as clicks and purchases etc. are collected to construct metrics like click-through rate (CTR), conversion rate (CVR), user return rate, dwell time, etc. for analyzing user engagement [14]. One key aspect of the system that does not receive a lot of attention is that there could be significant delays between a user’s visit to the page and their actions. A click may be instantaneous, but a purchase could take hours or even days for a user to complete. Using objectives with delays could introduce problems to the experiment.

The traditional process of assigning a fixed portion of users to competing alternatives is also known as online A/B/n testing and is readily available in major online experimentation platforms [12]. The biggest advantage of A/B/n testing is that it’s easy to implement and can easily support a variety of metrics of interest, including objectives with delays. However, using a delayed objective in an A/B/n test means it takes longer to finish the experiment than it otherwise would, which in turn exacerbates two common problems that A/B/n testing is criticized for. First, A/B/n testing is prone to peeking, which inflates the Type-I error. Because the A/B/n tests are designed to be analyzed only when the experiments end, peeking results and making decisions before the end of an experiment could lead to erroneous conclusions. Having to run a longer experiment for delayed objectives creates more opportunities for mistakes. Although there are advanced methods to address the peeking issue, such as sequential tests [9], as far as we know, the prominent methods today may not work easily with delayed feedback.

In recent years, adaptive tests have gained traction. Especially, “multi-armed bandits” test designs [7, 19, 20] increasingly becomes an alternative to the A/B/n testing when experimenters are only concerned with one primary metric. These bandit tests have the key...
advantage of reducing the opportunity costs from the experimenta-
tion, allocating traffic that would be diverted to inferior variants
to variants with more potential gains, as [20] points out. However,
widely-used “multi-armed bandits” test designs require the metric
or objective of interest to provide feedback in a reasonable time
frame in order to update the assignment rule to the variants. Con-
sequently, adaptive tests have found the most success with metrics
with near-instantaneous feedback, especially CTR.

Developing bandit algorithms for delayed feedback has become
a hot topic recently [13], for which we give an overview in Sec-
tion 2. But there are very few practical solutions that are directly
applicable to our use case, optimizing for CVR. Aside from the
fact that conversions are often delayed, another complexity for the
CVR objective is that we would never observe negative feedback. If
we have not observed a conversion from one particular user, it’s
because either she will convert in the future or she would never
convert to begin with. And it is impossible to distinguish between
these two possibilities. Metrics with such a property are common.
For example, computing user return rates also face a similar issue: a
user may return at some time in the future or she may never return,
but both cases are observably the same [5].

In this paper, we present a Thompson Sampling (TS) algorithm
to extend the “multi-armed bandits” test designs to binary metrics
with significantly delayed feedback. In our practical application
discussed in detail subsequently, we focus on conversion rate (CVR)
as the key metric. Nevertheless, the underlying ideas of the pro-
posed TS algorithm are readily applicable to other binary metrics
with delayed feedback and can be extended for delayed continuous
metrics.

We contribute to the literature by formulating a common real-
world problem and proposing a coherent and cogent solution that
is practically appealing and easy to implement. Our proposed al-
gorithm leverages a hybrid model within a Thompson Sampling
Bandit framework [13, 19]. The key features of our proposed algo-

rithm are

1. Modeling the objective using two latent variables, the event-
tual conversion, and the delay;
2. Computing the delay-corrected feedback during the exper-
iment using Expectation-Maximization method;
3. Selecting the optimal creatives based on the delay-corrected
feedback.

We use simulations to benchmark our proposed algorithm against
other approaches, and also present a real-world example of advertis-
ers using our proposed algorithm after it is deployed. Our solution
is deployed in the online experimentation platform of JD.com, a
large e-commerce company and a publisher of digital ads, and it
allows advertisers to optimize their ads creatives exposed to on
average tens of millions of requests per day.

2 RELATED WORKS

This paper belongs to the fast-growing literature of using bandit al-
gorithms for web-content optimization (e.g. personalized news rec-
ommendation, personalized creatives for ads, etc.) [1, 6, 15, 18, 20].
However, unlike this paper, almost all the applied bandit papers
are focusing on optimizing instantaneous metrics, and more specif-
ically CTR (see [1, 4, 6, 7] for example), because a key assumption

behind their algorithm and analyses is that the reward needs to be
immediately available after an action is taken. For advertisers and
decision-makers, CVR, sales, and other delayed metrics are often
more important than CTR because those are more directly related
to the business Key Performance Indicators (KPIs). CTR is used as
a proxy for its convenience, but may not lead to the optimum in
the desired KPIs. In Schwartz et al. [18]’s analysis, they found that
customer acquisition would drop 10% if the firm were to optimize
CTR instead of conversion directly.

In the broader online learning algorithms literature, there are a
handful of research projects extending bandit algorithms to delayed
feedback [10, 16, 17, 21, 22]. These research projects seek to address
delayed feedback issues under different settings and mostly focus
on theoretical analysis of the algorithms through the derivation of
complexity bounds for regret. In the influential empirical study
of Thompson Sampling [4], the authors discuss the impact of the
delays on the TS algorithm, but they only consider the fixed and
non-random delays. A more general problem of learning with delays
is discussed in [10], but the proposed modifications only apply to
the delays with known parameters. In the more recent work [23],
the authors consider stochastic delays which are more reasonable
for practical applications. But unlike in our setting, all the delays
are assumed to be observed eventually, which is not applicable for
CVR because non-converted clicks are never observed.

The closest works to ours in terms of the problem settings are
[3], [21] and [22], where the feedback are not only stochastically
delayed, but also can be missing entirely. Chapelle [3] proposes
to treat the conversion rate estimation as an offline supervised
learning problem, and set it up as a version of the statistical
censoring problem. Similar to our paper, Vernade et al. [21] tries an
online approach, but they focus more on the theoretical properties
and assume the delay distribution is known, which is not very
applicable in practice. The authors’ recent follow-up paper [22]
relaxes the assumption to allow for unknown delay distribution,
but introduces a hyperparameter m, which is essentially a timeout
limit. If feedback has not been received within m rounds, their
algorithm will label it as a non-conversion. It is an interesting
approach but has some limitations for practitioners to use. First,
it’s not clear how to choose a proper m. Second, the estimated CVR
is biased, and more likely to underestimate the conversion.

3 PROBLEM SETUP AND CHALLENGES

In the example used throughout this paper, our goal is to find the
creative with the best CVR among a set of competing alternatives.
Following the standards of the online advertising industry, we use
the post-click attribution model, which means that CVR is defined
as the percentage of the converted clicks among all the clicks. And
a click is considered converted if a purchase occurred at some time
after the click.

One unique aspect of the CVR (and other delayed binary feed-
back) problem is that the positive feedbacks take time to be observed
while the negative feedbacks are never observed. Therefore, we use
a hybrid model with two latent random variables to capture this
dynamic. Formally, for each click i in the experimental group k, the
outcome of the click is characterized by the following two latent
variables:
We will bring back the group subscript in Section 4.2.

only when there is no conversion delay; whereas when there is $n$ will use lowercase letters such as $N$ where $A$ common practice to measure CVR at any given time $t$ to represent contemporaneous counts at time $t$, and uppercase letters such as $N_t$ to represent the cumulative counts up to time $t$.

Using the latent variables defined above, we can rewrite the naive CVR as

$$\frac{1}{N_t} \sum_{i=1}^{t} \sum_{i' = 1}^{n_i} C_{i' s} \cdot \{ D_{i' s} \leq t - s \},$$

and thus it is trivial to show that $\hat{\theta}_t$ is an unbiased estimator of $\theta$ only when there is no conversion delay; whereas when there is any delay, it systematically underestimates the $\theta$. Therefore, the naive CVR is not suitable to be used with the bandit algorithm as an outcome metric if the real conversion is delayed. As shown by the red line in the Figure 1, using the naive CVR as the reward may not help identify the best alternative, when the delay distributions vary across competing treatment groups.

This problem can be addressed if the delay distribution is known. We can use the delay distribution to calculate an unbiased delay-corrected CVR estimator. For example, following Vernade et al. [21],

$$\hat{\theta}_t = \frac{N_{t \text{convert}}}{\sum_{s=1}^{t} n_s P(D \leq t - s)}$$

The proof of unbiasedness of this estimator is presented in Appendix A.1. The green line in Figure 1 shows that delay-corrected CVR indeed performs much better in recovering the ground truth, and thus identifying the best treatment group.

However, in practice the delay distribution is unknown. Moreover, the delay distributions could reasonably be very different across treatment groups and experiments because the treatment could leave impacts on the delays too. As a result, we could not simply use a delay distribution estimated from historical data, but have to estimate a delay distribution for each treatment group during each experiment instead.

During an active experiment, the delay time is right-censored at any given time, i.e. we cannot observe conversion delays longer than a threshold at any time of estimation. In the canonical survival analysis problems, all the events will eventually happen, so the right-censored portion implies the tail probability of the delay distribution [11]. In contrast, in our problem, the clicks that are not yet converted (i.e. right-censored portion) may either convert in the future or not convert at all. And the composition of those two types depends on the unknown true conversion rate. Therefore, in order to accurately estimate the delay distribution, we need to know the conversion rate first. We have come full circle.

3.1 Challenges

A common practice to measure CVR at any given time $t$ in the online advertising industry is to compute the naive CVR, i.e.

$$\hat{\theta}_t = \frac{N_{t \text{convert}}}{N_t},$$

where $N_t$ and $N_{t \text{convert}}$ respectively represent the total number of clicks and conversions up until time $t$.

Throughout this paper, we will use lowercase letters such as $n_t$ to represent contemporaneous counts at time $t$, and uppercase letters such as $N_t$ to represent the cumulative counts up to time $t$.

4 METHOD

In this section, we describe the system we proposed and implemented to conduct adaptive experiments with the CVR as the target metric. As shown in Figure 2, our system has two major components on top of the standard ad-serving infrastructure. The first component takes the click and purchase logs as inputs and estimates CVRs for each treatment group in an experiment. The second component computes the assignment probability based on all the estimated CVRs from the first component. If a stopping rule is not met, new ads will be displayed to users according to the assignment probability. Then the process repeats. Each such cycle represents a time step in our notations. It should be noted that the specific stopping criterion is independent of our proposed algorithm and should be set in accordance with the specific application. For example, an experiment can be set to stop whenever the top-performing treatment receives more than 95% assignment probability for 24 hours.

We will describe each component in detail in the following subsections.

4.1 CVR estimation

In this subsection, we describe the approach to estimating CVR for each group. Because the same procedure is applied to all the treatment groups in an experiment, we will focus on one treatment group and continue omitting the group subscript for the simplicity of illustration.
As we have mentioned in Section 3, there are two latent variables for click \( i \) in each treatment group, eventual conversion indicator \( C_i \) and conversion delay \( D_i \). We assume the data generating process is that, whenever a click occurs, noted as \( i \), a Bernoulli variable \( C_i \) will be drawn, indicating whether this click will eventually become a conversion. Then if the click will convert, a continuous variable \( D_i \) will be drawn and dictate how long it takes for the conversion to be observed.

Formally, we assume both variables are independent and identically distributed across \( i \) and follow:

\[
\begin{align*}
C_i & \sim \text{Bernoulli}(\theta) \\
D_i | C_i = 1 & \sim \text{distribution with CDF } F(\cdot; \lambda)
\end{align*}
\]

The \( \theta \) is the unknown true CVR that we want to estimate, and \( \lambda \) is a parameter that characterizes the delay distribution. We do not require the delay distribution to be any specific form except that it can be parameterized. Note that, because of the Bernoulli assumption, the above-described data generating process is only suitable for binary feedback. It’s possible to extend our framework to delayed continuous feedback by choosing a different distribution to be observed.

Both \( C \) and \( D \) are not always observable at an observation time \( t \). Instead, we observe the following variables:

- \( Y_{it} \in \{0, 1\} \) indicating whether click \( i \)’s conversion has already occurred at \( t \);
- \( E_{it} \in \mathbb{R}^{+} \) is the elapsed time since the click \( i \) till \( t \) if \( Y_{it} = 0 \), and the elapsed time since the click till conversion if \( Y_{it} = 1 \), i.e.:

\[
E_{it} = \begin{cases} 
\text{click} - t, & \text{if } Y_{it} = 0 \\
D_i, & \text{if } Y_{it} = 1
\end{cases}
\]

We apply the Expectation-Maximization (EM) method to find the maximum likelihood estimates for \( \theta \) and \( \lambda \) [8]. At any given observation time \( t \), EM solves a maximum likelihood problem of the form:

\[
\max_{\theta, \lambda} \sum_{i=1}^{N_t} \log \left\{ P(C_i = 0, y_{it}, e_{it}; \theta, \lambda) + P(C_i = 1, y_{it}, e_{it}; \theta, \lambda) \right\}
\]

After some reformulation and applying Jensen’s inequality, the above objective function is equivalent to:

\[
\max_{\theta, \lambda} \sum_{i=1}^{N_t} \left\{ q(C_i = 0) \log P(C_i = 0, y_{it}, e_{it}; \theta, \lambda) + q(C_i = 1) \log P(C_i = 1, y_{it}, e_{it}; \theta, \lambda) \right\}
\]

where \( q(c) = P(c|y, e; \theta, \lambda) \propto P(c|y, e; \theta, \lambda) \).

The EM method will iterate through the Expectation Step and the Maximization Step to find the solution to the above maximization problem. We detail those two steps below.

**4.1.1 Expectation Step.** For a given click and its corresponding data point \((y_{it}, e_{it})\), we need to compute the posterior probability of the eventual conversion conditioned on the observed data:

\[
w_{it} \equiv P(C_i = 1|y_{it}, e_{it}; \theta, \lambda)
\]

When \( y_{it} = 1 \), \( w_{it} \) simply equals 1, because it is trivial that \( C_i = 1 \) for certain. When \( y_{it} = 0 \),

\[
w_{it} = \frac{Pr(C_i = 1|y_{it} = 0, e_{it}; \theta, \lambda)}{Pr(y_{it} = 0, e_{it}; \theta, \lambda)}
\]

\[
= \frac{Pr(Y_{it} = 0, e_{it}; C_i = 1; \theta, \lambda)}{Pr(Y_{it} = 0, e_{it}; \theta, \lambda)} = \frac{Pr(Y_{it} = 0, e_{it}; C_i = 1; \theta, \lambda)Pr(C_i = 1)}{Pr(Y_{it} = 0, e_{it}; \theta, \lambda)}
\]

\[
= \frac{(1 - F(e_{it}))\theta}{1 - \theta + (1 - F(e_{it}))\theta}
\]

\( F(\cdot) \) is the CDF of \( e_{it} \). The expectation step of the EM algorithm then becomes:

\[
\hat{\theta} = \frac{\sum_{i=1}^{N_t} \sum_{t=1}^{T} w_{it} f(y_{it}, e_{it}; \theta)}{\sum_{i=1}^{N_t} \sum_{t=1}^{T} w_{it}}
\]

\[
\hat{\lambda} = \frac{\sum_{i=1}^{N_t} \sum_{t=1}^{T} w_{it} \left( y_{it} - f(y_{it}, e_{it}; \hat{\theta}) \right)^2}{\sum_{i=1}^{N_t} \sum_{t=1}^{T} w_{it}}
\]

\( f(\cdot) \) is the CDF of \( e_{it} \).
4.1.2 Maximization Step. In this step, we take the $w_{it}$ as given and maximize Equation 2

$$\max_{\theta, \lambda} \sum_{i} N_{i} (1 - w_{it}) \log P(C_{i} = 0, y_{it}, e_{it}; \theta, \lambda)$$

$$+ w_{it} \log P(C_{i} = 1, y_{it}, e_{it}; \theta, \lambda)$$

Because

$$P(C_{i} = 0, y_{it}, e_{it}; \theta, \lambda) = \begin{cases} 0, & \text{if } y_{it} = 1 \\ 1 - \theta, & \text{if } y_{it} = 0 \end{cases}$$

$$P(C_{i} = 1, y_{it}, e_{it}; \theta, \lambda) = \begin{cases} f(e_{it}), & \text{if } y_{it} = 1 \\ (1 - F(e_{it}))\theta, & \text{if } y_{it} = 0 \end{cases}$$

the objective function becomes

$$\max_{\theta, \lambda} \sum_{i} N_{i} w_{it} \log \theta + (1 - w_{it}) \log(1 - \theta)$$

$$+ \sum_{i} N_{i} w_{it} y_{it} f(e_{it}) + w_{it}(1 - y_{it}) \log(1 - F(e_{it}))$$

A nice result from the derivation above is that, regardless of the delay distribution $F(\cdot)$, there is always a separation between $\theta$ and $\lambda$. In other words, they can be optimized independently. This separation result comes from the fact the delay distribution is independent of the true conversion rate.

4.1.3 Exponential Delay Distribution. Up to this point, we have been agnostic about the distribution of the delay. Depending on the use cases and settings, one may choose different delay distributions to fit the data and our approach should work for all the parameterized delay distributions. But to give readers a more in-depth illustration of our approach work in practice, we are going to assume the delay follows a exponential distribution for the following sections.

For our use cases, we find that exponential distribution can best fit the conversion delay. Chapelle [3] also reaches the same conclusion after analyzing the conversion data at Criteo.

Plugging the probability density function and cumulative distribution function of exponential distribution into Equation 4, we can solve for optimal $\lambda^{*}$ analytically:

$$\hat{\lambda}_{t}^{*} = \frac{N_{convert}^{(l)}}{\sum_{i} w_{it}y_{it}}$$

Because of the separation, we could use the estimator described in Equation 1 for $\theta$. With the exponential distribution, the estimator is:

$$\hat{\theta}_{t}^{*} = \frac{N_{convert}^{(l)}}{\sum_{s=1}^{L} n_{s}(1-e^{-\lambda^{*}(t-s)})}$$

In practice, we find that this estimator for $\theta$ is more stable than the $\theta$ estimator solved from Equation 4.

4.1.4 E-M iterations. At each time step $t$, we iterate the E-M steps for a few cycles to make sure the resulted estimates are stable. Then the final estimates are saved and used as the priors for the next time step. Let $L$ represent the total number of the E-M cycles. At time $t$ and cycle $l$ ($0 < l \leq L$), we compute the following:

$$w_{it}^{(l)} = \begin{cases} 1, & \text{if } y_{it} = 1 \\ \frac{\beta_{it}^{(l-1)} e^{-\lambda_{it}^{(l-1)} e_{it}}}{1 - \beta_{it}^{(l-1)} + e^{-\lambda_{it}^{(l-1)} e_{it}}}, & \text{if } y_{it} = 0 \end{cases}$$

$$\hat{\lambda}_{t}^{(l)} = \frac{N_{convert}^{(l)}}{\sum_{i} w_{it}^{(l)} y_{it}}$$

$$\hat{\theta}_{t}^{(l)} = \frac{N_{convert}^{(l)}}{\sum_{s=1}^{L} n_{s}^{(l)}(1-e^{-\hat{\lambda}_{t}^{(l-1)}(t-s)})}$$

where $\hat{\lambda}_{t}^{(0)} = \lambda_{t-1}^{(L)}(L)$, $\hat{\theta}_{t}^{(0)} = \theta_{t-1}^{(L)}$.

4.2 Bandit Integration

After the unbiased CVRs are estimated in each treatment group for an experiment, we use a multi-armed bandit algorithm to compute the assignment probability for each group. The assignment probabilities will be used to assign requests to groups, and are updated at each time step.

We propose to use the Thompson Sampling method with a delay-corrected sample size and a Beta-Bernoulli prior. Specifically, we assume the eventual conversion in each treatment group follows a Bernoulli distribution with a group-specific probability $\theta_{k}$, consistent with what we have been assuming. And in a Bayesian framework, $\theta_{k}$ has a Beta$(\alpha_{k}, \beta_{k})$ prior at time $t$.

Before the experiment starts, at $t = 0$ we set diffuse priors and let $\alpha_{k0} = 1, \beta_{k0} = 1, \forall k \in K$. In the subsequent time-step $t$, we update $\alpha_{kt}$ and $\beta_{kt}$ following:

$$\alpha_{kt} = \alpha_{kt-1} + N_{convert_{kt}}$$

$$\beta_{kt} = \max(1 - N_{convert_{kt}} + \frac{N_{convert_{kt}}}{\beta_{kt-1}}, 1)$$

Then the assignment probability of a group is the posterior probability that the group offers the highest expected CVR. We compute these values using Monte Carlo simulations following the procedure outlined in Scott [19].

Algorithm 1 presents the entire procedure of our method for exponentially distributed delays.

5 EXTENSION TO DELAYED CONTINUOUS FEEDBACK

The proposed algorithm described previously focuses on the case of binary delayed feedback metrics, e.g. conversion rate (CVR). There are many important metrics such as Gross Merchandise Value that are not binary but face the same issues of delay and censoring. This algorithm can be extended to those cases of continuous metrics and even count metrics by redefining the eventual conversion $C$ variable. The random variable $C$ could be defined as a mixed random variable with a discrete component still corresponding to the case without a response (e.g. no purchase is made), and a continuous
Algorithm 1 TS to identify the group with best CVR

Input: K groups

Parameter: Number of E-M cycles each step, L

1: Let t = 0.
2: ∀k ∈ K, 𝜃_k^0 ← 0.1, 𝜆_k^0 ← 1/105
3: ∀k ∈ K, 𝛼_k ← 1
4: ∀k ∈ K, 𝑝_k ← 1/K
5: while NOT exit condition do
6:   A batch of requests arrives
7:   for request i do
8:     Sample from a multinomial distribution with K groups and 𝑝_k for each k ∈ K
9:     Assign i to the sampled group
10:   end for
11: Collect click and conversion data
12: for group k do
13:   Collect click and conversion data
14:   Load previously computed 𝜃_k^{t-1} and 𝜆_k^{t-1}
15: for l in {1, 2, ..., L} do
16:   Update 𝜃_k^{(l)} for each click i in k as in Equation (7)
17:   Update 𝜆_k^{(l)} as in Equation (9) and (8)
18: end for
19: 𝜃_k^t ← 𝜃_k^{(L)}, 𝜆_k^t ← 𝜆_k^{(L)}
20: Update α_k = 1 + N_k convergent
21: Update 𝛼_k = max(1 − N_k convergent + N_k convergent 𝜆_k^t, 1)
22: Repeatedly sample from Beta(𝛼_k, 𝛼_k) for all k ∈ K and 𝑝_k equals the empirical proportion of Monte Carlo samples in which the draw from k is maximal.
23: end for
24: end while

6 SIMULATIONS

In this section, we present the simulation results that establish the validity of our approach and compare it against other approaches.

For all the simulations, we consider a setup with three treatment groups in a simulated experiment. All groups have different event conversion rates and a delay distribution with different means.

We compare our algorithm Delay-corrected Thompson Sampler (D-TS) against four other algorithms.

(1) Random. As the name suggests, this algorithm randomly chooses a treatment group to display with equal probability. This can be interpreted as the classic “split-testing”.

(2) Naive Thompson Sampler. This algorithm only uses the observed conversions at the assignment time and ignores the possible delays. It behaves in the same way as the standard Thompson Sampler for CTR [6].

(3) Delay-corrected UCB. This is a variant of the Upper Confidence Bound (UCB) algorithm proposed by Vernade et al. [21], where the sample size is replaced with the delay-corrected sample size plus some additional adjustments 3. The original paper assumes a known delay distribution, but we use estimated distribution here. The estimation follows the same EM procedure as that of our D-TS algorithm.

(4) Full Bayesian. This algorithm assumes that the delay distribution follows the exponential distribution and uses the Beta priors for 𝜽 and 𝜆. Moreover, the numerical posterior is computed and consumed by a Bayesian UCB bandit. The biggest drawback with this approach is that it is extremely time-consuming to compute, taking as much as 100 times longer than the time used by the delay-corrected methods.

In Table 1, we present the benchmark results for getting one batch of assignments from different algorithms starting from the raw log data. The benchmark test was run on a 2019 model 16-inch MacBook Pro with 16 GB Ram and 2.3 GHz 8-Core 19 Intel CPU. Each algorithm is repeated 50 times. Although these results should not be taken for their face value because the algorithms are not fully optimized for production, they show that the delay-corrected algorithm with EM procedure is reasonably fast whereas the Full Bayesian approach is too slow for any practical use.

| Algorithm | min  | mean | median | max  |
|-----------|------|------|--------|------|
| Random    | 0.014| 0.018| 0.017  | 0.038|
| Naive TS  | 0.018| 0.025| 0.023  | 0.039|
| D-UCB     | 0.345| 0.423| 0.402  | 0.714|
| Full Bayesian | 38.803| 46.299| 44.027 | 56.261|
| D-TS      | 0.332| 0.433| 0.434  | 0.642|

The main metric we use to compare algorithms is cumulative regret. For each treatment group k at the time t, we consider the rewards 𝑟_t(k) as the total number of eventual conversions. Regrets at each time t are defined as the difference between the best possible rewards at time t and the rewards from the algorithm assignment plan. Mathematically, the cumulative regret is:

\[ R_t = \sum_{s=1}^{t} \max_k r_s(k) - r_s(k^*) \]

If a bandit algorithm is able to find the best group, it means that the cumulative regret should level off after some time.

The simulation results for 4 different environments are presented in Figure 3. In Figure 3a, we compare the cumulative regret of the five bandit policies in a setting with relative high CVRs, \( \theta = (0.5, 0.4, 0.3) \), and exponentially distributed delays with \( \lambda = \]

3Their paper also proposed a D-KLUCB algorithm which is claimed to be better than the D-UCB, but it is complicated to implement.
Delay could be a reasonable assumption to use in practice. Even though our approach takes much longer to converge. In the low CVR setting, our method continues to deliver the best performance. As more data are collected, the creative with the highest conversion rate will gradually have more chance to be displayed. Throughout the experiment, all the reports and relevant statistical results are displayed in a dashboard in real-time and readily available to the advertisers.

We discuss a case study based on the results from the first CVR experiment run by a large cellphone manufacturer after we launched the product. The advertiser sets up 2 creatives for the same item: one dark version and one light version. We keep track of the orders of each click for 15 days. The experiment lasted about 3 weeks, with 130 orders recorded for the dark version and 237 orders for the light version.

In the left panel of Figure 4, we present the estimated delay-corrected CVRs of both versions of the creatives. The solid lines indicate the point estimate of the CVRs, whereas the dashed lines indicate the CVRs fifteen days after the experiment ends. The green and red lines represent the 10th and 90th percentiles of the posterior of estimated CVRs. This left figure shows that after a period of learning, the estimated CVRs from our algorithm are able to “predict” the eventual CVRs of each creative after the experiment. The right panel presents the impression count for each version of the creative through the experiment. It shows the exploration and exploitation of the bandit algorithm and the fact that the algorithm eventually allocates more traffic to the higher CVR creative.

Although the online case study was not designed for comparing our algorithm against the other approaches, we can still use its data to check whether our estimated CVR is a better signal for the eventual CVR compared to the naive CVR. Similar to Figure 1, we compare the delay corrected CVR estimate against the uncorrected (or naive) CVR estimate during the experiment in Figure 5. The green and red lines represent the delay-corrected CVR and naive CVR estimations, respectively, and the dashed line indicates the eventual CVR fifteen days after the experiment ends. This figure shows that our delayed-corrected estimates are much closer to the eventual CVR compared to the naive estimator and its performance improves as time progresses. In the right panel, the naive estimate greatly underestimates the eventual CVR even at the end of the experiment.
This result is consistent with our argument made in Section 3.

experiment. The main reason is the naive estimator considers clicks that have not converted as a negative immediately, while that the delay-corrected takes into account the potential delay to conversion. This result is consistent with our argument made in Section 3.

8 CONCLUSION

An adaptive experimentation algorithm to identify the best treatment group from a set of competing treatment groups with respect to a delayed binary feedback objective was presented. This algorithm is applicable to a variety of situations common in digital advertising and has the potential to be extended to support more metrics. For our application, the algorithm powers a product that allows advertisers to identify the best creative for an ad from a set of advertising creatives for a delayed feedback outcome, i.e. conversion rate (CVR). Moreover, simulations were presented to demonstrate that the algorithm outperforms benchmarks. In addition, we discussed the deployment and presented a case study where the algorithm was used by an advertiser (a large cellphone manufacturer) to identify the optimal advertising creative for their advertising campaign. This algorithm is currently deployed in the online experimentation platform of JD.com, a large e-commerce company and a publisher of digital ads.

REFERENCES

[1] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. 2009. Explore/exploit schemes for web content optimization. In 2009 Ninth IEEE International Conference on Data Mining. IEEE, 1–10.

[2] Takeshi Amemiya. 1985. Advanced econometrics. Harvard university press.

[3] Olivier Chapelle. 2014. Modeling delayed feedback in display advertising. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 1097–1105.

[4] Olivier Chapelle and Lihong Li. 2011. An empirical evaluation of thompson sampling. Advances in neural information processing systems 24 (2011), 2249–2257.

[5] Georges Dupret and Mounia Lalmas. 2013. Absence time and user engagement: evaluating ranking functions. In Proceedings of the sixth ACM international conference on Web search and data mining. 173–182.

[6] Tong Geng, Xiliang Lin, and Harikesh S Nair. 2020. Online Evaluation of Audiences for Targeted Advertising via Bandit Experiments. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34. 13273–13279.

[7] Tong Geng, Xiliang Lin, Harikesh S Nair, Jun Hao, Bin Xiang, and Shurui Fan. 2021. Comparison Lift: Bandit-based Experimentation System for Online Advertising. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35. 15117–15126.

[8] David R Hunter and Kenneth Lange. 2004. A tutorial on MM algorithms. The American Statistician 58, 1 (2004), 30–37.

[9] Ramesh Johari, Leo Pekelis, and David J Walsh. 2015. Always valid inference: Bringing sequential analysis to A/B testing. arXiv preprint arXiv:1512.04922 (2015).

[10] Pooria Joulani, András Gyorgy, and Csaba Szepesvári. 2013. Online learning under delayed feedback. In International Conference on Machine Learning. PMLR, 1453–1461.

[11] John D Kalbfleisch and Ross L Prentice. 2011. The statistical analysis of failure time data. John Wiley & Sons.

[12] Ron Kohavi, Diane Tang, and Ya Xu. 2020. Trustworthy online controlled experiments: A practical guide to a/b testing. Cambridge University Press.

[13] Tor Lattimore and Csaba Szepesvári. 2020. Bandit algorithms. Cambridge University Press.

[14] Janette Lehmann, Mounia Lalmas, Elad Yom-Tov, and Georges Dupret. 2012. Models of user engagement. In International conference on user modeling, adaptation, and personalization. Springer, 164–175.

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

[16] Larkin Liu, Richard Downe, and Joshua Reid. 2019. Multi-armed bandit strategies for non-stationary reward distributions and delayed feedback processes. arXiv preprint arXiv:1902.08593 (2019).

[17] Ciara Pike-Burke, Shipra Agrawal, Csaba Szepesvári, and Steffen Grunewalder. 2018. Bandits with delayed, aggregated anonymous feedback. In International Conference on Machine Learning. PMLR, 4105–4113.

[18] Eric M Schwartz, Eric T Bradlow, and Peter S Fader. 2017. Customer acquisition for non-stationary reward distributions and delayed feedback processes. arXiv preprint arXiv:1706.09186 (2017).

[19] Claire Vernade, Olivier Cappé, and Vianney Perchet. 2017. Stochastic bandit models for delayed conversions. arXiv preprint arXiv:1706.09186 (2017).

[20] Claire Vernade, Alexandra Carpentier, Tor Lattimore, Giovanni Zappella, Beyza Ermiş, and Michael Brueckner. 2020. Linear bandits with stochastic delayed feedback. In International Conference on Machine Learning. PMLR, 9712–9721.

[21] Yingzhen Zhou, Rennan Xu, and Jose Blanchet. 2019. Learning in generalized linear contextual bandits with stochastic delays. Advances in Neural Information Processing Systems 32 (2019), 5197–5208.
A APPENDIX

A.1 Unbiasedness of delay-corrected estimator

Proposition 1. If the delay variable $D_i$ is independent and identically distributed across $i$, then

$$\hat{\theta}_t = \frac{\sum_{s=1}^{t} \sum_{s'=1}^{n} \mathbb{1} \{ D_{i} \leq t - s \}}{\sum_{s=1}^{t} n_s P(D \leq t - s)}$$

is an unbiased estimator for $\theta = E[C_i]$.

Proof. The $\hat{\theta}_t$ can be re-written as

$$\hat{\theta}_t = \frac{\sum_{s=1}^{t} \sum_{s'=1}^{n} \mathbb{1} \{ D_{i} \leq t - s \}}{\sum_{s=1}^{t} n_s P(D \leq t - s)}.$$

So

$$E[\hat{\theta}_t] = E\left[\frac{\sum_{s=1}^{t} \sum_{s'=1}^{n} \mathbb{1} \{ D_{i} \leq t - s \}}{\sum_{s=1}^{t} n_s P(D \leq t - s)}\right]$$

$$= \frac{\sum_{s=1}^{t} \sum_{s'=1}^{n} E[C_i] \mathbb{1} \{ D_{i} \leq t - s \}}{\sum_{s=1}^{t} n_s P(D \leq t - s)}$$

$$= \frac{\sum_{s=1}^{t} \sum_{s'=1}^{n} \theta P(D \leq t - s)}{\sum_{s=1}^{t} n_s P(D \leq t - s)}$$

$$= \frac{\theta \sum_{s=1}^{t} n_s P(D \leq t - s)}{\sum_{s=1}^{t} n_s P(D \leq t - s)}$$

$$= \theta$$

□

B ONLINE RESOURCES

The R source code for the simulation exercises will be available upon publication.