Separation strategies for three pitfalls in A/B testing

Luo Lu
Twitter Inc.
1355 Market Street, Suite 900
San Francisco, California, USA
llu@twitter.com

Chuang Liu
Twitter Inc.
1355 Market Street, Suite 900
San Francisco, California, USA
chuang@twitter.com

ABSTRACT

A/B testing has become an invaluable tool for guiding complex business decisions. It has a long history in the natural sciences and hence is known under many other names such as controlled experiments, randomized experiments, treatment/control studies as well as others. Twitter provides a global communication network with a vast number of use cases such as expressing opinions, communication with friends, following news and more. Due to the heterogeneity of use cases and the intrinsic complexity of user behavior, A/B testing is used extensively by Twitter to guide product launches and iterating on existing products. In this paper, we discuss our best practices for three common pitfalls observed while reviewing experiment designs and metrics at Twitter. These pitfalls are generic enough to be of benefit to the wider community. They are "dilution", "carry over effect" and "novelty impact". All of the three corresponding strategies, coincidentally, involve separation.

1. INTRODUCTION

1.1 Background

As online products and business become increasingly sophisticated, A/B testing is now an important tool for gaining business insights and driving decisions making. Some of the most well-known users of A/B testing include social networks such as Facebook [2, 1] and LinkedIn, search providers such Bing and Google[10], e-commerce sites such as Amazon and eBay as well as others [9]. A/B testing is deceptively difficult to master, as famously said by Ron Kohavi, "Getting numbers is easy, getting numbers you can trust is quite difficult." [7]. We also rephrase this as "Getting numbers is easy, using the numbers correctly is quite difficult." In order to obtain correct results and correctly draw conclusions from them, it is crucial for a company to understand how to accurately and efficiently set up, implement and interpret the results from an experiment. Due to the complexity of the products and user behaviors, we consider each experiment on a case by case basis and the list of "to-do" and "not-to-do" is extensive. Marvelous insights and experiences have been shared in [7, 3, 5, 6]. We would like to add some of our own experiences from the thousands of experiments at Twitter that we think are generic enough for the wider community.

1.2 A/B testing at Twitter

A brief understanding of Twitter's experimental framework is required to grasp some of the concepts covered in the following sections. Any experimental framework involves at least three components, a "randomization algorithm", an "assignment method" and a "data path" [8]. Twitter's framework is no different, however the details within each component may be unfamiliar. The randomization algorithm divides users into two or more groups (called buckets). Users in the control bucket are presented the original application or website. Users in the other buckets, called case buckets, are presented a new variant of the application or website. We call this assignment procedure a "choose call". It involves filtering the user based on parameters such as OS type or country as well as a hash based randomization. In order to scale up to a large fleet of servers, we compute the hash value as a hash of the user id and the experiment id [4, 8]. It is up to the experimenter to decide how to implement the filters in the "choose call" and we will discuss in the following sections some pitfalls associated with it. Secondly, the assignment method relates to the implementation of how variants are delivered to the user. This does not relate to any pitfalls so we don't discuss it here. Finally, the data path is the process of collecting relevant metrics from each bucketed user. We call the time at which Twitter's framework starts recording metrics from the user the "scribe impression" or the user being "scribed". This point may be the same as the "choose call" or it could be at a later time. Similarly, it is also up to the experimenter to choose the location of the "scribe impression". These concepts will be used in the pitfalls discussed below.

Furthermore, at Twitter, we primarily use "users" as experimental units. Some experiments, like advertisement targeting algorithm changes could use "user request" as experimental units by randomly serving different variants per request, and measuring how users respond. However, most of Twitter product operates around either getting users into a healthy state by improved presentation of content or improved content recommendations. For these experiments, it is important to provide a consistent experience to each user during the experimental period [4]. For metrics, we need to measure the long term impact on important user met-

rics, and answer questions like "Does getting users to follow more accounts help get these users back on Twitter more often down the road?". To address these needs, most of the experiments in Twitter use "user" as the experimental unit and keep each user in one bucket during the experiment.

At last, compared to choosing among statistical methods for A/B testing, it is much trickier to decide among types of data to test. Given two buckets of users in a time period (lasts for m days), there are different levels of granularity to compare the user engagement. Let's use "Tweet" as an example, we list them below from the "coarsest" to the "finest":

1. Unique user: the percentage of users in each bucket who tweet at least once in the time period;
2. Daily unique user: the average number of days a user tweets at least once;
3. Average action taken: the average number of Tweets sent by each user.

The first measurement is the "coarsest" since it only depends on whether each user tweet or not, without differentiate from the users who tweet one hundred times and the users who tweet only once during the experiment. While for the third data type, each user's weight is decided by the number of tweets he/she sent. The second data type is in between the first and the third ones: the weight of the user who tweets the most is at most m times of the user who tweets only once. By comparing the three data types between buckets we are able to answer three different questions and each one cannot be replaced by another. In other words, if the users in the case bucket tweet more than the users in the control bucket users in regards of the first data type, it does not necessarily leads to the same order for the second or the third data type. Same thing happens for the other two data types. All of the three data types can be used to do ab test and might give different and even opposite results. As that being said, which data type to choose is critical to the ab test results and is usually determined by the goal we set for each experiment as well as the "target" users. For the features aiming at boost the number of tweet for the "mature" users, we expect the unique user tweet percentage is close to 100% before and during the experiment for both buckets and only the second or the third data type will be sensitive enough to detect the improvement. Inherently, from the coarsest to the finest data type, their sensitivity to the (over) active users increases and accordingly less and less conservative. For the flexibility for the experimenters to choose from, we have all three data type metrics available from the pipeline. To achieve that, we store the numbers for each bucket as: S_i: the number of unique users who are scribed on day i; D_i: the number of unique users who have been scribed since day 1 till day i; $S_i(\phi)$: the number of unique users who are scribed and take the action ϕ on day i (ϕ could be "Tweet", "Retweet", "Follow", "Be followed", etc.); $D_i(\phi)$: the number of unique users who have been scribed and taken the action ϕ since day 1 to day i and $T_i(\phi)$: total number of ϕ taken by the users on day i.

1.3 Contribution

In this paper, we focus on three common pitfalls for experimentation: "dilution", "carryover effect" and "novelty impact". For the sake of convenience, we define all case buckets as "adding a new feature" to Twitter. The users in the case bucket are the ones who see the "new feature" while the users in the control bucket do not see it.

1. *Dilution.* Dilution happens when many of the users in the case bucket do not experience the new feature. Twitter products and features vary greatly in their user coverage, and many of them only affect a fraction of users. Dilution deteriorates the A/B testing power [8], and hence may not fully capture the effect of the new feature. In order to address this problem, we separate "choose call" – the logic to check if the new feature should be visible to a user, and "scribe impression"– the logic to decide if a user should be counted for the experiment. This separation gives us increased flexibility to address dilution problems.

2. *Carry over effect.* Carry over effect happens when past experiments, which a user was part of, affect their behaviors in current experiments. It could also happen within a single experiment if being part of current experiment affects their chances to get into the experiment again in following days. One solution is to run an A/A test prior to running the experiment and "re-bucket" the users if there are significant changes in the key metrics [5]. This paper focuses on the cases when the past experiment is one iteration of the current experiment. We claim that the previous bucketing is reusable as long as no bias appears to the number of users who are scribed, S_i or D_i. They are also the denominator of all the key metrics based on the three "data types" in section 1.2. Separating the scribed users into "first time visitors" and "return visitors" and recording their daily (or hourly) counts in a "bucket size status table", gives us an efficient method for checking bucket sanity without requiring extra tests, as well as to decide whether a re-bucketing is necessary to address carry over effects.

3. *Novelty impact.* Novelty impacts happen when users behave differently when they are first exposed to an experiment [8, 5]. It is a common and well known issue among user interface experiments because it takes time for users to get used to the new changes. Yet there is little literature discussing possible solutions. [5] observes that in general, novelty impact rarely reverses the initial effect, for instance where the feature is initially negative until users learn it and get used to it, then it starts to be positive. We do see a couple of experiments as exceptions of that, mostly due to external effect (such as an OS update which invalided the experiment code) instead of user behavior change. Based on different sources of novelty impact such as user curiosity, learning curve, user structure lagging and external effect, there are different ways to get the novelty impacted days. In this paper, we aim to provide the practitioner with a generic method based on the daily metric, to find a separation date after which the metric has stabilized.

2. PITFALL I – DILUTION

Twitter has a vast spectrum of features. Apart from the "me" profile, there are three timelines: a home timeline serves Tweets from followed accounts; a discovery timeline serves Tweets and account recommendations; an activity timeline provides information about activities in a user's social circle. Twitter enables users to express themselves via Tweets, Retweets, Favorites etc. and provides a search box for users to search for Tweets, friends and events.

Most users use only a subset of these features. Dilution occurs if a significant proportion of users in the case bucket do not see or use the feature experimented on. In this case, even if the new feature effectively changes the "target" users' behavior, we may not be able to measure it due to dilution.

In most experiments, it is trivial to focus only on the "target" users, both conceptually and practically. For example, if the new feature is to send all new users an email right after they sign up, introducing how Twitter works, the target users are the users who receive the email. In this case, all the users to whom we show the new feature are also the ones we want to scribe. In other words, the "choose call" and "scribe impression" location should overlap, when the email is sent. Unfortunately, this is not always the case.

2.1 Example

The "nearby timeline" experiment. Right now Twitter has three timelines besides the "me" profile page: home timeline, discovery timeline and activity timeline. On mobile, users swipe to switch between timelines. This experiment is to insert a new timeline "nearby timeline" as the second timeline on mobile and shift discovery and activity timelines to the third and forth timeline respectively. The "nearby timeline", as the name suggests, is to provide local news or Tweets for Twitter users. Hence it only targets to the users who have geotag switched on. "Geotag" is the feature that informs Twitter of the users' physical locations every time they open their Twitter apps. The left plot in Figure 1 is the flow chart of the experiment design. Aware of the potential dilution problem, the experimenters narrowed down the users to all mobile Twitter users who have geotag switched on (we call them "qualified users" now). In their experiment design, the experimenters "scribe impression" for the qualified users when they land on the first (home) timeline and show them the "nearby" timeline for those in the case bucket.

2.2 Diagnosis

Even though this experiment focuses on the users with geotag on, dilution still exists. The majority of Twitter users start their Twitter experience from their home timeline. When this happens, experiment logic enables the "nearby timeline" for users in the case bucket and they can potentially swipe. However, a large part of users only use their home timeline, and never swipe into the new "nearby" timeline. As a result, the metrics for the case bucket will be diluted significantly due to the users who do not swipe to the new timeline during the experiment.

2.3 Separation Strategy

The fundamental problem is those users who are qualified do not see or use the new feature. In order to address this, the experiment framework provides an API for experimenters to pick users based on users properties and the product feature they are using. It also provides an option to separate "choose call" and "scribe impression".

2.4 Best practice

Where to make the "choose call" should be defined by the experimenters, according to the position and characteristics of the new feature. The principle is to (1) include only users who can potentially see the new feature, given (2) inserting the new feature is technically feasible. In the "nearby" timeline case, we are targeting the mobile Twitter users who have geotagged switched on and swipe to the second timeline during the experiment. However, it is too late to insert the new timeline after they swipe, we make the "choose call" for all the mobile Twitter users who have geotagged on.

Once we enable the new feature for the users assigned to the case bucket in the "choose call", not all of them will see the new feature, depending on their habit or pattern of using Twitter. We only want to "scribe" the users who see the new feature in the case bucket to maximize power and "scribe impression" at the equivalent location for the control bucket. Hence, in this example, we only "scribe" the users who swipe to the second timeline during the experiment, i.e."discover" timeline in the control bucket and "nearby" timeline in the case bucket (Figure 1 right plot).

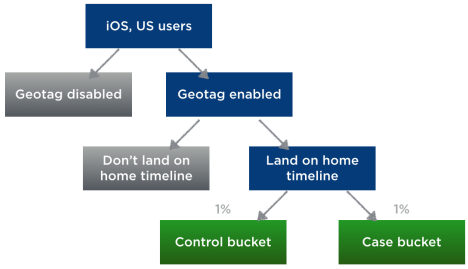
This design separates the location of "choose call" and "scribe impression". It guarantees flexibility in engineering when to insert the new feature, meanwhile enables us to narrow down to the "target" users for metrics. This best practice is for experiments which have inherently unavoidable gaps between "showing the feature to the users" and "the user experiencing the feature". It also requires the bucketing pipeline to be flexible enough to allow for this separation.

3. PITFALL II – CARRY OVER EFFECT

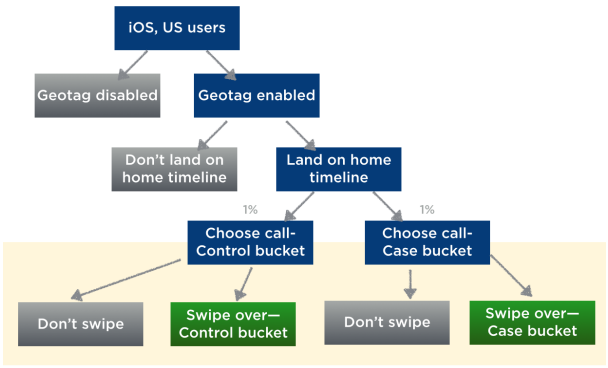
It is common practice at Twitter that product teams iterate on experiments until it reaches a "ready" state, where each iteration corresponds to a version. Before an experiment starts, product teams have certain expectation on how users will respond to it, which may or may not be true. After an experiment is live, metrics tell us the truth. Based on these metrics, product teams continue to make small adjustments and fix bugs until the experiment is "ready". The final results are based only on metrics from the last version when the experiment is in a "ready" state.

Twitter continuously runs a large number of experiments, and most of the experiments go through multiple versions. It is well known that experiments running in the past may affect users' behaviors in the new experiments that affect validity of experiment results. [5] describes ways to address carryover effect caused by past experiments via randomized bucketing logic. Here, we only focus on the type of carryover effect caused between versions of the same experiment.

As mentioned in previous sections, in order to give users a consistent experience, the mapping from a user to an experiment bucket is deterministic. If a user's experience in a old version causes him or her to either stop or reduce use of the feature, it will cause a sample bias in the subsequent ver-



(a) Pitfall – dilution due to the users in case bucket who do not swipe



(b) Best practice – only scribe the users who swipe to the second timeline in both the control and the case bucket.

Figure 1: Avoid dilution in bucketing

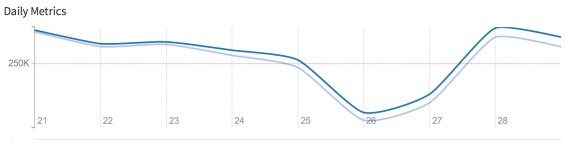


Figure 2: Experiment version 1: daily count of users in the case bucket (light blue curve) shrinks during the experiment compared to the control bucket (dark blue curve).

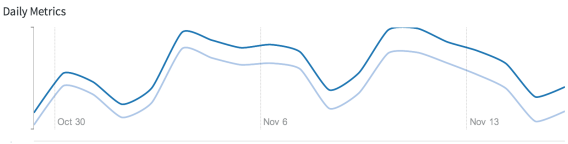


Figure 3: Experiment version 2: bucket size imbalance is observed at the beginning of this version

sions. This is because experiments only count users who see or use the new feature (see Section 2). We will illustrate how the carry over effect happens using the following example.

3.1 Example

This experiment has two versions with a negligible change between them. Figure 2 and 3 are the daily bucket size (S_i in Section 1.2) charts of the two versions, light blue for case bucket and dark blue for control bucket. It shows the number of users being scribed in each bucket every day. The bucket setting is 1 : 1 for the control and the case buckets. The first version starts at a 383.4K users for the control bucket and 382.7K users for the case bucket, which is consistent with the bucket setting. However, along the eight days of version 1, a gap gradually emerges between the two curves, showing much less users in the case bucket compared to control at the end of the first version. As described in Section 1.2, when the experiment id and bucket setting are unchanged, the assignment of users to buckets will be unchanged between versions since the input of the hash function is unchanged. Therefore, the same gap shows up at the beginning of the second version. The experimenters find out that the case bucket performs overwhelmingly better than the control bucket in the second version but not for the first. Given that usually we use the last version's results as the final results, a clear victory is claimed for the case bucket. Is that true?

3.2 Diagnosis

Bucket size abnormally is usually a red flag in A/B testing. It could indicate an error in one or some of the bucketing logic: randomization algorithm, assignment method or data path. Code bugs could be another possibility. Sometime, the new feature changes the user behavior for the users in the case bucket, reflected by a gap of the daily bucket size between the buckets. Some of the sources may affect the correctness of the experiment. Before we trust the experiment results, it is critical to understand the reason behind the bucket size abnormality if any. In the example, the gap between the buckets gradually enlarges over the time and we suspect it is caused by "organic" lose of users in the case bucket – one situation of carry over effect. The strategy in the next section helps us verify the hypothesis.

3.3 Separation Strategy

Using the same notation as in Section 1.2, we show how we can make use of $D_i(\phi)$ and $S_i(\phi)$ to check bucket size status. At first, we separate each "scribe impression" to be either "first time visitor" or "return visitor". "first time visitor" refers to the first time a user is being scribed, the rest of the cases are called "return visitor". In spite of what the names suggest, "first time visitor" and "return visitor" do not refer to actual people, but different types of scribes. The main difference between these two types and the reason for separating them lie here: from each user's perspective, the first time being scribed happens before exposed to the new feature but not for the subsequent ones. For example, if the new feature is to give a more detailed description when you click "follow" for that person, the scribe impression location is when you make the click, at which you are unaware of the new feature. On the other hand, you might have already noticed the longer description for the second or third time you make the "click" (i.e. be scribed "return visitor") and your altitude towards the new feature will potentially influence counts of "return visitor". Therefore, whether the

gap in the bucket size between the buckets comes from "first time visitor" or "return visitor" becomes a critical clue of the reason behind the scene.

Denote m as the experiment length, which could represent either number of days or hours, etc., B(control/case) as the bucket size setting for each bucket. The formula for the daily count of "first time visitor" (a_i) ,"return visitor" (r_i) and the "return percentage" in the control bucket are

$$\begin{aligned} a_1 &= S_1 \\ r_1 &= 0 \\ a_i &= S_i - S_{i-1} \\ r_i &= D_i - a_i w_i = r_i / S_{i-1} \end{aligned}$$

for $i = 2, \ldots, m$. And $a'_i, r'_i, w'_i (i = 1, \ldots, m)$ are the analogs for the case bucket.

For each day $i = 1, \ldots, m$, following are the two bucket size status testing:

1. whether the daily "first time visitor" counts are within expectation $H_0 : a_i/a'_i = B(\text{control})/B(\text{case})$;
2. whether the daily "return percentage" agrees for the two buckets $H_0 : w_i = w'_i$.

For the statistical testing, we assume $a_i|a_i + a'_i$, $a'_i|a_i + a'_i$, $r_i|S_{i-1}$ and $r'_i|S'_{i-1}$ all follow a binomial distribution. For testing "first visitor", we use confidence interval generated from Binomial$(a_i+a'_i, B(\text{control})/(B(\text{control})+B(\text{case}))$ and Binomial$(a_i+a'_i, B(\text{case})/(B(\text{control})+B(\text{case})))$. For testing "return percentage", we use t test.

Therefore, we have two test results each day, for "first time visitor" and for "return percentage". Typically, consecutive two or three day abnormalities in either test is a red flag. Abnormality in "first time visitor" is usually caused by experiment logic error or code error. While for "return percentage" abnormalities, if there is a significantly higher percentage of return visitors in case bucket, it is likely that the new feature successfully encourages the users to return more often and vice versa. For most of the features, it is a win if the former happens since we always want to increase "scribe impression" at Twitter. An exception is when the new feature is functional, that is to help user accomplish a task, being scribed more than once indicates an inefficiency of the new feature.

3.4 Best practice

Table 1 is the bucket size status table of the first version and it tells us where the gap between buckets comes from. The "first time visitor" counts are normal for both the buckets while case bucket is losing "return visitor" at the second half of the experiment. One of the possible explanation is that part of the users in case bucket dislike the new feature and thus return less than if they were in the control bucket. Therefore, at the end of the first version, there are less unhappy users in the case bucket relative to the control bucket. The second version starts with the same group of users which also has this bias in the user structure. It is not hard to imagine that the results of the second version will be biased toward the case bucket since the proportion of the users who love the new feature is higher than in the control. For both versions, we need to be very careful when interpreting the daily metrics because they lost a different portion of (unhappy) users over days. For version 1, we could get around the sample bias issue as long as we look into the aggregated metrics ("Unique user" data type in Section 1.2) across the whole period, and count everyone who has ever been scribed from day one. However, we cannot do the same for the second version because sample bias exists from day one due to the carry over effect. If the experiment results from the second version are used, there is a high risk of making a wrong decision.

Checking the bucket size status table frequently during the experiment is one of the most important sanity checks. The table also serves as signal of how the users are enjoying the new feature. Opening a new experiment is an alternative of starting a new version using the same experiment id. It shuffles the users again and avoid the carry over effect from the previous version, at the cost of user experience inconsistency. Always open a new experiment if the user structure has changed at the end of the current version, indicated by the "return percentage" abnormality.

4. PITFALL III – NOVELTY IMPACT

The majority of experiments aim to improve long term user experience. For experiments involving a user interface change, it may take several attempts for the users to become familiar. For experiments involving social components, such as encouraging more social interactions, it may take time for the impact to mature.

Novelty impact is when the short term user behavior is not a good indicator of the longer term user behavior. It is a well known phenomenon in A/B testing. However, the associated methods and discussion is still scarce in literature.

4.1 Example

The left plot in Figure 4 is the daily chart for "mention sent" from an experiment during September 25 and October 9, 2013. The case bucket (red curve) has higher percentage of users who sent mentions than the control bucket (blue curve) for the first couple of days. Then the gap between the curves shrinks gradually and at last flips to the other side. The average daily "mention sent" percentage for the case bucket is 5.64% larger than the control bucket, which is significant based on our statistical tests.

4.2 Diagnosis

In most cases, we can only afford to run the experiment for a much shorter time compared to the desired long term effect. This short time frame is sensitive to transient user behavior near the beginning, i.e. novelty impact. It is possible that ignoring them will create misleading or even opposite conclusions.

1. *Curiosity.* Even if we put a new button on the Twitter user interface which leads to nowhere, there will be many users who click on the button due to curiosity for the first couple of days. This causes an initial bump in the engagement of the new feature. After the users

Bucket		Oct20	Oct21	Oct22	Oct23	Oct24	Oct25	Oct26	Oct27	Oct28	Oct29
Control	$a_i(k)$	383.40	142.63	111.51	98.13	86.76	78.88	65.68	64.38	69.38	63.26
	$w_i(\%)$	0	31.40	27.80	25.15	22.90	20.99	18.35	17.98	18.85	17.90
Case	$a_i(k)$	382.65	142.50	111.56	98.26	87.02	77.54	64.68	64.47	68.77	63.59
	$w_i(\%)$	0	31.32	27.63	25.01	**22.62**	**20.80**	**18.17**	**17.67**	**18.62**	**17.59**

Table 1: . Bucket size status table: numbers in bold font are numbers outside of confidence interval. At the last 6 days, the case bucket has significantly less "return visitors" than the control bucket

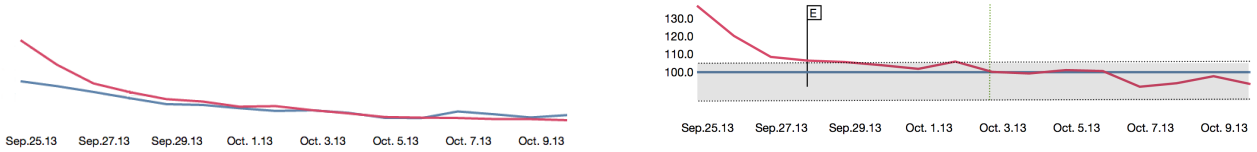


Figure 4: Novelty impact detection

The blue and red curves on the left plot are the daily percentage of users who sent mentions during the experiment for control and case bucket respectively$(a_i, b_i,\ i = 1, \ldots, 15)$. The curves on the right plot are the daily percentages relative to control (therefore, the blue curve is always at 100%). The shadow to the right of the vertical dashed line marks the confidence interval generated from the second half of the red curve z_i from step 2 of Algorithm1. The flag on September 28 represents the last day that novelty impact is detected.

become used to the new feature, their behavior will regress to the long term effect of the new feature.

2. *Learning curve.* Sometimes it takes time for the users to get used to the new feature, during which time we may see a decrease in the user's engagement with the new feature. However, after the users become familiar with the new feature, they may increase their engagement again. In those cases, the novelty impact is more likely to express as a dent at the beginning of the experiment.

3. *User type structure.* At the first couple of days of the experiment, most of the users being scribed are frequent users for Twitter. Lets use a mock example to illustrate this. Assume Group A are the users who use Twitter once everyday while Group B are those who use Twitter once every week. The number of users in Group A and Group B is of the ratio 65 : 35. Experiment E is general enough to cover all the users from both groups once they sign in Twitter. For the first day of the experiment, there will be about 65 : 5 number of users being scribed from Group A and B. The aggregated ratio for the first two days would be around 65 : 10. At the seventh day of the experiment we start getting the same ratio as the overall Twitter user population. For this example, novelty impact exists in "Unique user" data type but not for the other two. What is a good representative group of users for the experiment depends on the different user composition of the "target" users of the new feature.

4. *External Effect.* As mentioned earlier, some changes in the environment of the experiment, such as OS or browser updates, could disrupt the implementation of the experiment. Strictly speaking, this is more of a bug than an inherent novelty impact. Nevertheless, a change may be observed on the metrics and the separation strategy in the next subsection will help us capture it. In fact, there have been a couple of experiments in Twitter where novelty impact is identified at the same day for almost all the metrics, followed by abrupt and dramatic changes in the time series. This suspicious phenomenon successfully caught people's attentions and external changes were found to be the culprits.

4.3 Separation strategy

Different types of novelty impact (caused by different reasons) can be corrected using various models and algorithms. Given that a large part of the novelty impact involves multiple reasons which are hard to isolate from each other, we introduce a generic method that requires no knowledge of the source of the novelty impact. Given two time series $a_0, a_1, \ldots, a_m$ for control bucket and $b_0, b_1, \ldots, b_m$ for case bucket. a_i and b_i could be the percentage of users who take certain action in that bucket ("Daily unique user") or the average number of actions taken by the users in that bucket at daily or hourly basis ("Average action taken"). Algorithm 1 describes how we detect novelty impact automatically. We use the ratio of a_i and b_i to cancel the daily (hourly) effect and assume at first the ratio for the second part has stabilized. The algorithm generates an upper and lower bound for the ratio from the stabilized part. It then proceeds to trace back from the middle to record the first value which exceeds the confidence interval for two consecutive points. If the value we get is the close to the middle point, we recommend that experimenters should continue the experiment for more days (hours).

4.4 Best practice

After removing the novelty impacted days, the mean of daily "mention sent" percentage is 3.14% lower in case than control and turns out to be insignificant. The length that novelty impact lasts usually varies among the metrics and the new features. Customizing the days we exclude due to novelty

Algorithm 1 Algorithm for detecting novelty impact

Step 1. For each day/hour, calculate the ratio of case and control metrics: $z_i = b_i/a_i$ $(i = 0, 1, \ldots, m)$.
Step 2. Estimate the confidence interval (lb,ub) based on $z_{[m/2]}, z_{[m/2+1]}, \ldots, z_m$.
▷ Normal distribution is the rule of thumb distribution for z_i
Step 3. For $k \in \{[m/2]-1, [m/2]-3\}$
if $\min_{j\in\{k,k-1,k-2\}} z_j > ub$ or $\max_{j\in\{k,k-1,k-2\}} z_j < lb$ **then**
z_i has not been stabilized, return
end if
Step 4. $k = [m/2] - 4$
while $k \geq 3$ **do**
▷ The number "3" is a parameter
if $\min_{j\in\{k,k-1,k-2\}} z_j > ub$ or $\max_{j\in\{k,k-1,k-2\}} z_j < lb$ **then**
break
end if
$k = k - 1$
end while
if $k \geq 3$ **then**
Novelty impact is detected and lasts from the first day until day k.
elseif
No novelty impact is detected.
end if
return

impact guarantee us the largest power in the subsequent statistical testing.

5. CONCLUSION

We have discussed three pitfalls in user based A/B testing together with three corresponding separation strategies for addressing them. The first pitfall, dilution, usually related to the bucketing logic, is commonly observed when there is a gap between when the new feature shown to the user and when the user engages with it. The solution we recommended is to separate "choose call" and "scribe impression", which allows us to scribe only part of the users for metric calculation. The second pitfall describes when a new feature influences a user's behavior and the effect is carried over to next time the same user engages with the feature. This could be within the same experiment version or happen at the next version. Using the bucket status table and separating "first time visitor" and "return visitor" helps detect this problem and distinguish it from other reasons for bucket size anomaly. This pitfall is usually organic and unavoidable. Nevertheless, ignoring it can potentially lead to biased results especially when starting a new version. Finally, we proposed a generic algorithm for detecting and correcting for novelty impact on time series data. Furthermore, it sends an alarm when the time series has not yet stabilized, which indicates we should run the experiment for a longer time. We hope that these separation strategies will benefit the wider community and also raise awareness in the field to these pitfalls.

6. ACKNOWLEDGMENTS

We wish to thank Jimmy Chen, Linus Lee, Robert Chang, Matt Knox and Madhu Muthukumar for their valuable discussions.

7. REFERENCES

[1] E. Bakshy and D. Eckles. Uncertainty in online experiments with dependent data: an evaluation of bootstrap methods. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1303–1311. ACM, 2013.

[2] E. Bakshy, D. Eckles, and M. S. Bernstein. Designing and deploying online field experiments. In *Proceedings of the 23rd international conference on World wide web*, pages 283–292. International World Wide Web Conferences Steering Committee, 2014.

[3] T. Crook, B. Frasca, R. Kohavi, and R. Longbotham. Seven pitfalls to avoid when running controlled experiments on the web. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1105–1114. ACM, 2009.

[4] R. K. et al. Practical guide to controlled experiments on the web: Listen to your customers not to the hippo. *KDD*, 2007.

[5] R. Kohavi, A. Deng, B. Frasca, R. Longbotham, T. Walker, and Y. Xu. Trustworthy online controlled experiments: Five puzzling outcomes explained. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 786–794. ACM, 2012.

[6] R. Kohavi, A. Deng, R. Longbotham, and Y. Xu. Seven rules of thumb for web site experimenters. *KDD*, 2014.

[7] R. Kohavi and R. Longbotham. Unexpected results in online controlled experiments. *ACM SIGKDD Explorations Newsletter*, 12(2):31–35, 2011.

[8] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18(1):140–181, 2009.

[9] J. Overgoor. Experiments at airbnb. *http://nerds.airbnb.com/experiments-at-airbnb/*, May 2014.

[10] D. Tang, A. Agarwal, D. O'Brien, and M. Meyer. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 17–26. ACM, 2010.