Detecting Network Effects: Randomizing Over Randomized Experiments*

Martin Saveski
MIT
msaveski@mit.edu

Jean Pouget-Abadie
Harvard University
jeanpougetabadie@g.harvard.edu

Guillaume Saint-Jacques
MIT
gsaintja@mit.edu

Weitao Duan
LinkedIn
wduan@linkedin.com

Souvik Ghosh
LinkedIn
sghosh@linkedin.com

Ya Xu
LinkedIn
yaxu@linkedin.com

Edoardo M. Airoldi
Harvard University
airoldi@fas.harvard.edu

ABSTRACT

Randomized experiments, or A/B tests, are the standard approach for evaluating the causal effects of new product features, i.e., treatments. The validity of these tests rests on the "stable unit treatment value assumption" (SUTVA), which implies that the treatment only affects the behavior of treated users, and does not affect the behavior of their connections. Violations of SUTVA, common in features that exhibit network effects, result in inaccurate estimates of the causal effect of treatment. In this paper, we leverage a new experimental design for testing whether SUTVA holds, without making any assumptions on how treatment effects may spill over between the treatment and the control group. To achieve this, we simultaneously run both a completely randomized and a cluster-based randomized experiment, and then we compare the difference of the resulting estimates. We present a statistical test for measuring the significance of this difference and offer theoretical bounds on the Type I error rate. We provide practical guidelines for implementing our methodology on large-scale experimentation platforms. Importantly, the proposed methodology can be applied to settings in which a network is not necessarily observed but, if available, can be used in the analysis. Finally, we deploy this design to LinkedIn's experimentation platform and apply it to two online experiments, highlighting the presence of network effects and bias in standard A/B testing approaches in a real-world setting.

KEYWORDS

A/B testing, Network effects, Network interference, SUTVA.

*Martin Saveski and Jean Pouget-Abadie contributed equally to this work.

KDD'17, August 13-17, 2017, Halifax, NS, Canada

978-1-4503-4887-4/17/08…$15.00
DOI: 10.1145/3097983.3098192

1 INTRODUCTION

Before deploying new features or products, it is common practice to run randomized experiments—or A/B tests—to estimate the effects of the suggested change. A part of the population is randomly selected to receive the new feature/product (treatment), while another randomly selected part of the population is given the product status quo (control). The goal of comparing the treatment population to the control population is to impute the difference between two universes that we cannot observe simultaneously: one where everyone receives treatment and another where no one does. This imputation, formalized in the theory of causal inference [16], relies on a fundamental independence assumption, known as the "stable unit treatment value assumption" (SUTVA) [9, 26]. It states that every user's behavior is affected only by their own treatment and not by the treatment of any other user.

However, this independence assumption may not always hold, particularly when testing products with social components that, by design, connect users and allow them to interact with one another. Consider the example of testing a feed ranking algorithm: If user A and user B are connected, by changing the ranking of items on user A's feed, we impact their engagement with their feed and indirectly change the items that appear on user B's feed. User B may well have been placed in the control group, but their behavior was impacted by the assignment of user A to the treatment group. This spillover of treatment effects between users violates SUTVA and leads to inaccurate treatment effect estimates [7].

There are three common approaches to minimizing the effects of spillovers: (*i*) Cluster-based randomization [2, 10, 34], where users are clustered based on their connections and treatment is assigned at a cluster level; (*ii*) Multi-level designs [15, 30], where treatment is applied with different proportions; (*iii*) Designs and analysis that assume specific models of interference [3, 8, 12, 28, 31], e.g. that the interference effect is proportional to the number of neighbors treated.

While mitigating interference is the end goal of the field of causal inference with network interference, a precursor to that question is to detect whether SUTVA holds in the experiments we run. When we plan an experiment in which we suspect that SUTVA might be violated, a common solution is to use cluster-based randomized

assignments, which minimize the number of edges cut between treated and control units. Under certain assumptions [10], this design can partially mitigate the problem of interference. However, in practice, cluster-based randomized assignments have higher variance than the completely randomized assignment, making it more difficult to accurately estimate the treatment effect. Furthermore, cluster-based assignments require an appropriate choice of clustering of the population, which can be a challenging even if the graph of interactions between units is known. If we have a robust way of testing whether SUTVA holds, we can decide whether the standard completely randomized assignment is valid and whether running a cluster-based randomized assignment is necessary.

Traditionally, testing whether SUTVA holds has been done through post-experiment analysis: Rosenbaum [25] was the first to state two sharp null hypotheses (i.e., there is no treatment effect on anyone) which imply that SUTVA does not hold. Under these restricted null hypotheses, we can obtain exact distribution of network parameters. More recent work [1, 4] explicitly tackles testing for the non-sharp null that SUTVA holds by considering the distribution of chosen network effect parameters for a subpopulation of the graph under SUTVA. While the final test is dependent on the choice of network parameter and subpopulation, the main appeal of this analysis-based approach is that it does not require running a modified experiment.

Rather than focusing on a post-experiment analysis approach, we explore new designs tailored to test for network effects. The main idea behind our approach is to simultaneously run a completely randomized and a cluster-based randomized experiment on different, randomly selected, parts of the population and to compare the resulting treatment effect estimates. If the two estimates are very different, that is an indication of network effects. However, if the two estimates are very similar, we expect SUTVA to hold. This allows us to test whether SUTVA holds without making any modeling assumptions of how units interact with each other. Moreover, if SUTVA holds, we are still able to estimate the treatment effect using the part of the population that received a completely randomized assignment. If SUTVA does not hold, we can still rely on the estimate obtained from the cluster-based randomized experiment.

This work, together with [24], is part of a two-paper series. In [24], we introduce the methodology and main theoretical results, while in this paper we present the practical implementation details, and provide an in-depth analysis of the experimental results. In doing so, we make the following main contributions:

- We present the randomized experimental design for testing whether SUTVA holds, detailed more thoroughly in [24], and present the essential results on variance estimation and bounding the Type I error rate (Section 2).
- We provide detailed implementation guidelines to help practitioners deploy the proposed design on large-scale experimentation platforms. We discuss how to compute balanced clustering at scale (Section 3.2), as well as how to stratify to reduce variance (Section 3.3).
- We deploy our framework on LinkedIn's experimental platform and we report the results of two large-scale experiments (Section 3), achieving significance in one of them, highlighting the presence of network effects.

2 THEORETICAL FRAMEWORK

In this section, we present our two-stage experimental design for testing whether SUTVA holds and the main theoretical results. We provide more complete exposition in [24].

2.1 Two assignment strategies

First, we briefly review the notation and main results for the completely randomized (CR) design and the cluster-based randomized (CBR) design. Let $G = (V, E)$ be a graph of N units ($|V| = N$), for which we measure outcomes $(Y_i)_{i \in V}$, and on which we can assign an intervention with two conditions: treatment and control. Each unit's potential outcome is defined as a function of the entire assignment vector $\mathbf{Z} \in \{0,1\}^N$ of units to treatment buckets: $Y_i(\mathbf{Z})$. If SUTVA holds, $Y_i(\mathbf{Z}) = Y_i(Z_i)$. The causal estimand of interest is the *Total Treatment Effect* (TTE) given by:

$$\mu := \frac{1}{N} \sum_{i \in V} [Y_i(\mathbf{Z} = \mathbf{1}) - Y_i(\mathbf{Z} = \mathbf{0})]. \tag{1}$$

Finally, for any vector $\mathbf{u} \in \mathbb{R}^n$, let $\overline{u} = \frac{1}{n} \sum_{i=1}^n u_i$ and $\sigma^2(\mathbf{u}) = \frac{1}{n-1} \sum_{i=1}^n (u_i - \bar{u})^2$.

Completely Randomized Assignment. In a completely randomized experiment, we sample the assignment vector $\mathbf{Z}$ uniformly at random from the set $\{z \in \{0,1\}^N : \sum z_i = n_t\}$, where n_t (resp. n_c) is the number of units assigned to treatment (resp. control). Let $Y_t := \{Y_i : Z_i = 1\}$ and $Y_c := \{Y_i : Z_i = 0\}$. Recall the definition of the difference-in-means estimator:

$$\hat{\mu}_{cr} := \overline{Y_t} - \overline{Y_c}.$$

Cluster-based Randomized Assignment. In a cluster-based randomized assignment, we randomize over clusters of units in the graph, rather than individual units. We suppose that each unit in V is assigned to one of m clusters. We sample the cluster assignment vector $\mathbf{z}$ uniformly at random from $\{v \in \{0,1\}^m : \sum v_i = m_t\}$. Units in cluster j are assigned to the same treatment bucket as cluster j: $Z_i = 1 \Leftrightarrow z_j = 1$ if $i \in j$, where m_t (resp. m_c) is the number of clusters assigned to treatment (resp. control). We let Y' be the vector of *aggregated* potential outcomes, defined as $Y'_j := \sum_{i \in j} Y_i$, the sum of all outcomes within that cluster. Let $Y'_t := \{Y'_j : z_j = 1\}$, $Y'_c := \{Y'_j : z_j = 0\}$. The aggregated difference-in-means estimator is given by:

$$\hat{\mu}_{cbr} := \frac{m}{N} \left(\overline{Y'_t} - \overline{Y'_c} \right).$$

2.2 A two-stage randomized design

When SUTVA holds, $\hat{\mu}_{cr}$ and $\hat{\mu}_{cbr}$ are unbiased estimators of the total treatment effect under a completely randomized assignment and a cluster-based randomized assignment respectively [22]. When SUTVA does not hold, their unbiasedness is no longer guaranteed. In fact, when interference is present, we expect the estimate of the total treatment effect to be different under a completely randomized design than under a cluster-based randomized design.

Consider for example the case where only the unit's immediate neighborhood affects their outcome:

$$\forall i, \mathbf{Z}, \mathbf{Z}', [\ \forall j \in N(i),\ Z_i = Z'_j\ ] \implies Y_i(\mathbf{Z}) = Y_i(\mathbf{Z}'), \tag{2}$$

where $N(i) := \{j \in V : (i,j) \in E\}$ be the neighborhood of i.

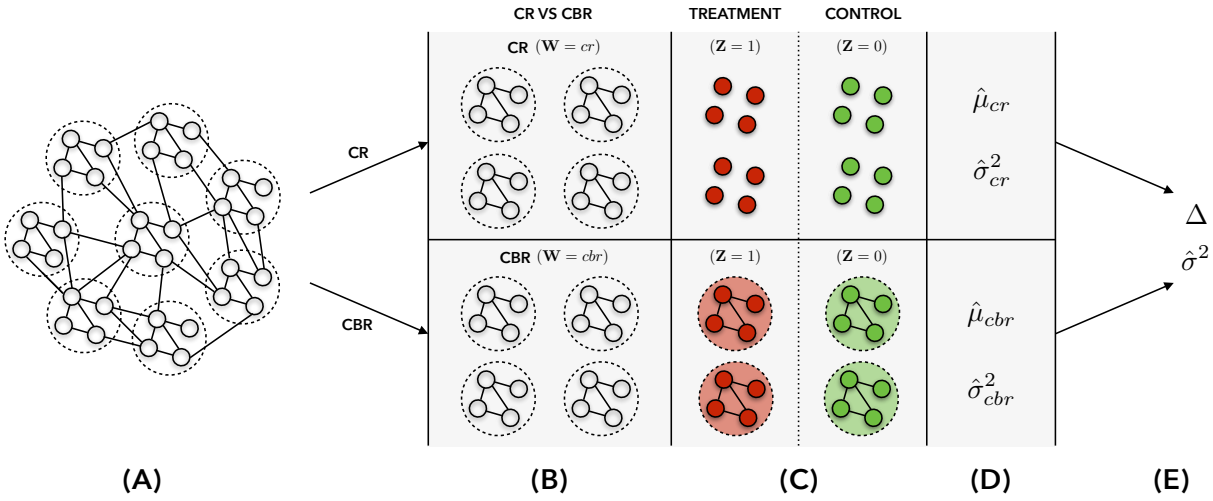

Figure 1: Illustration of the proposed experimental design for detecting network effects. (A) Graph of all units and the connections between them; the dashed circles represent (equally-sized) clusters. (B) Assigning clusters to treatment arms: completely randomized (CR) and cluster-based randomized assignment (CBR). (C) Assigning units to treatment buckets—treatment and control—using the corresponding strategy. (D) Computing the treatment effect within each treatment arm: $\hat{\mu}_{cr}$ and $\hat{\mu}_{cbr}$, and variance: $\hat{\sigma}^2_{cr}$ and $\hat{\sigma}^2_{cbr}$. (E) Computing the difference of the estimates from each treatment arm: $\Delta = \hat{\mu}_{cr} - \hat{\mu}_{cbr}$, and the total variance: $\hat{\sigma}^2 = \hat{\sigma}^2_{cr} + \hat{\sigma}^2_{cbr}$.

In other words, if the assignment $\mathbf{Z}$ is such that a unit i's neighborhood is assigned entirely to treatment (resp. control), then we observe $Y_i(\mathbf{Z}) = Y_i(\mathbf{1})$ (resp. $Y_i(\mathbf{Z}) = Y_i(\mathbf{0})$), in which case we can estimate the treatment effect for unit i. The probability of assigning the whole neighborhood of a unit to treatment or to control is very small under a completely randomized assignment. One way to increase this probability is to assign entire *clusters* of units to treatment or to control (Section 2.1). Cluster-based randomization designs are used to reduce the bias under a completely randomized assignment when interference is believed to happen primarily through each unit's first-degree neighborhood.

Thus, the main idea behind this work is to set up a two-level experimental design to test for interference, such that different parts of the graph G will receive treatment through different randomized strategies. By comparing the estimates from each randomized strategy, we test the *effect of the randomized strategy*, which is null under SUTVA. This is done by first assigning units to treatment *arms* and then, within each treatment arm, applying a specific assignment strategy to assign units to treatment *buckets* (Figure 1). The two-stage design works as follows:

(*i*) We initially cluster the graph G into m clusters C. Note that we do not necessarily need to fully observe the graph, we just need to have a meaningful clustering of the users.

(*ii*) We sample the units to treatment arms assignment vector $\mathbf{W} \in \{cr, cbr\}^N$ using a cluster-based randomized assignment. We denote by $\omega \in \{cr, cbr\}^m$ the corresponding cluster assignment vector to treatment arms *CR* ($\omega_j = cr$) and *CBR* ($\omega_j = cbr$). Let m_{cr} and m_{cbr} be the number of clusters assigned to treatment arms *CR* (completely randomized assignment) and *CBR* (cluster-based randomization assignment) respectively, and let n_{cr} and $n_{cbr} = N - n_{cr}$ be the resulting number of units assigned to each arm.

(*iii*) Conditioned on $\mathbf{W}$, we sample $\mathbf{Z}_{cr} \in \{0,1\}^{n_{cr}}$ using a completely randomized assignment to assign units in treatment arm *CR* to treatment and control. Let $n_{cr,t}$ and $n_{cr,c}$ be the number of units that we wish to assign to treatment and control respectively.

(*iv*) Still conditioned on $\mathbf{W}$, we sample $\mathbf{Z}_{cbr}$ using a cluster-based randomized assignment to assign units in treatment arm *CBR* to treatment and control. Let $m_{cbr,t}$ and $m_{cbr,c}$ be the number of clusters assigned to treatment and control respectively.

The resulting assignment vector $\mathbf{Z}$ of units to treatment and control is such that $\mathbf{Z}_{cr} \perp\!\!\!\perp \mathbf{Z}_{cbr} | \mathbf{W}$.

2.3 Testing for the SUTVA null

Next, we present a statistical test for accepting or rejecting the SUTVA null. We provide a more detailed, step by step, derivation in [24]. We define the two estimates of the causal effect for this experiment as follows:

$$\hat{\mu}_{cr}(\mathbf{W}, \mathbf{Z}) := \overline{Y}_{cr,t} - \overline{Y}_{cr,c}, \tag{3}$$

$$\hat{\mu}_{cbr}(\mathbf{W}, \mathbf{Z}) := \frac{m_{cbr}}{n_{cbr}} \left(\overline{Y'}_{cbr,t} - \overline{Y'}_{cbr,c} \right), \tag{4}$$

where we have introduced the following notation:

$$Y_{cr,t} := \{Y_i : W_i = cr \wedge Z_i = 1\},$$
$$Y_{cr,c} := \{Y_i : W_i = cr \wedge Z_i = 0\},$$
$$Y'_{cbr,t} := \{Y'_j : \omega_j = cbr \wedge z_j = 1\},$$
$$Y'_{cbr,c} := \{Y'_j : \omega_j = cbr \wedge z_j = 0\}.$$

In order to test whether the estimates of each arm are significantly different, we must divide the difference of the estimates by its variance under the null. It is uncommon in randomized experiments to know the variance of the chosen estimators exactly, but we can usually settle for an empirical upper-bound.

We consider the following variance estimators, computable from the observed data:

$$\hat{\sigma}^2_{cr} \coloneqq \frac{\hat{S}_{cr,t}}{n_{cr,t}} + \frac{\hat{S}_{cr,c}}{n_{cr,c}}, \tag{5}$$

$$\hat{\sigma}^2_{cbr} \coloneqq \frac{m^2_{cbr}}{n^2_{cbr}} \left(\frac{\hat{S}'_{cbr,t}}{m_{cbr,t}} + \frac{\hat{S}'_{cbr,c}}{m_{cbr,c}} \right), \tag{6}$$

where we introduced the following empirical variance quantities in each treatment arm and treatment bucket:

$$\hat{S}_{cr,t} \coloneqq \sigma^2(Y_i : W_i = cr \wedge Z_i = 1),$$
$$\hat{S}_{cr,c} \coloneqq \sigma^2(Y_i : W_i = cr \wedge Z_i = 0),$$
$$\hat{S}'_{cbr,t} \coloneqq \sigma^2(Y'_j : \omega_j = cbr \wedge z_j = 1),$$
$$\hat{S}'_{cbr,c} \coloneqq \sigma^2(Y'_j : \omega_j = cbr \wedge z_j = 0).$$

Finally, we consider the sum of these two empirical variance quantities:

$$\hat{\sigma}^2 \coloneqq \hat{\sigma}^2_{cr} + \hat{\sigma}^2_{cbr}. \tag{7}$$

As shown in [24], the following results holds when the *clustering is balanced* (i.e., all clusters have the same number of nodes):

$$E_{\mathbf{W},\mathbf{Z}}\left[\hat{\mu}_{cbr} - \hat{\mu}_{cr}\right] = 0, \tag{8}$$

$$\mathrm{var}_{\mathbf{W},\mathbf{Z}}\left[\hat{\mu}_{cr} - \hat{\mu}_{cbr}\right] \leq E_{\mathbf{W},\mathbf{Z}}[\hat{\sigma}^2]. \tag{9}$$

Note that Eq. 9 is the only statement requiring the clustering to be balanced. When SUTVA holds, both estimators have the same expectation under their respective assignment strategy regardless of whether the clustering is balanced. In order to control the variance however, as is done in Eq. 9, we restrict ourselves to balanced clusterings. There are now multiple ways in which we can reject the null. If we reject the null that SUTVA holds when:

$$\frac{|\hat{\mu}_{cr} - \hat{\mu}_{cbr}|}{\sqrt{\hat{\sigma}^2}} \geq \frac{1}{\sqrt{\alpha}}, \tag{10}$$

then the type I error of our test is no greater than α if $\hat{\sigma}^2 \geq \mathrm{var}[\hat{\mu}_{cr} - \hat{\mu}_{cbr}]$, which we can only guarantee in expectation (Eq. 9). A less conservative approach is to suppose that the quotient follows a normal distribution $\mathcal{N}(0,1)$, for which we obtain $(1-\alpha) \times 100\%$ confidence intervals:

$$CI^{1-\alpha}(T) = \left(T - z_{\frac{\alpha}{2}}, T + z_{1-\frac{\alpha}{2}}\right), \tag{11}$$

where $z_{\frac{\alpha}{2}}$ and $z_{1-\frac{\alpha}{2}}$ are the $\frac{\alpha}{2}$ quantile of the standard normal distribution.

While bounding the Type I error of our test allows us to assume SUTVA under which the moments of our test statistic become tractable, the same cannot be said of the Type II error, where we must assume a specific interference model. We refer the reader to [24], where we show that under the following popular model of interference, the Type II error rate of our suggested design and test decreases as the number of edges cut by the initial clustering also decreases, with all else being equal:

$$Y_i = \beta_0 + \beta_1 Z_i + \beta_2 \rho_i + \epsilon_i,$$

where $\rho_i = \frac{1}{N} \sum_{i \in V} |N(i) \cap C(i)| \;/\; |N(i)|$, with $N(i)$ being the neighborhood of unit i and $C(i)$ being the units in i's cluster.

Finally, note that the results presented in this section hold regardless of whether the true network of interactions between users is observed. Knowing this network, at least partially, allows us to find more meaningful clusters of users and increases our ability to detect network effects when they are present i.e. reduce the Type II error. One can achieve similar results by using any network or domain knowledge deemed relevant for the experiment at hand, not necessarily the true network.

2.4 Incorporating Stratification

So far, we have used completely randomized assignment to assign clusters to treatment arms and, in the CBR treatment arm, to assign clusters to treatment buckets. Under this randomization strategy, it is possible that—by chance—we may end up with very different populations in the two treatment arms, or in the CBR arm, different treatment and control groups. For example, we may assign all clusters with highly active users in the same treatment arm. Stratification prevents such scenarios by design. Instead of randomizing all clusters at once, we first divide them into more homogeneous groups (strata) and we randomize within each stratum. Stratification has two key advantages: (*i*) it ensures that all covariates used to create the strata will be balanced, (*ii*) it improves the precision of the treatment effect estimator. In this section, we extend our test for detecting network effects to incorporate stratification.

Suppose that each graph cluster $c \in C$ is assigned to one of S strata. In this section, we assume that the strata are given, but in Section 3.3 we show how to construct them. We denote by $V(s)$ the nodes in the graph which belong to strata s. Within each strata $s \in [1,S]$, we assign $m_{cr}(s)$ clusters completely at random to the *CR* treatment arm and $m_{cbr}(s)$ clusters to the *CBR* treatment arm, denoted by vector $\mathbf{W}(s)$, sampled uniformly at random from vectors $\{\mathbf{v} \in \{cr, cbr\}^{|V(s)|} : \sum_j \mathrm{I}[v_j = cr] = m_{cr}(s)\}$.

Let $\mathbf{Z}_{cr}(s)$ be the assignment of units the treatment arm *CR* to treatment buckets within strata s. If $V_{cr}(s)$ is the subset of V in strata s assigned to treatment arm *CR*, $\mathbf{Z}_{cr}(s)$ is chosen uniformly at random from the vectors $\{\mathbf{u} \in \{0,1\}^{|V_{cr}(s)|} : \sum_i u_i = n_{cr,t}(s)\}$. Similarly, let $z_{cbr}(s)$ be the assignment of clusters in treatment arm *CBR* to the treatment buckets within strata s. If $C_{cbr}(s)$ is the subset of clusters in strata s assigned to treatment arm *CBR*, $z_{cbr}(s)$ is chosen uniformly at random from the vectors $\{\mathbf{v} \in \{0,1\}^{|C_{cbr}(s)|} : \sum_j v_j = m_{cbr,t}(s)\}$.

Let $n_{cbr}(s)$ be the total number of units assigned to treatment arm *CBR* and $m_{cbr}(s)$ be the total number of clusters assigned to treatment arm *CBR* within strata s. We can extend the previous estimators of the average treatment effect under stratification. Let

$$Y_{cr,t}(s) \coloneqq \{Y_i : i \in V_{cr}(s) \wedge Z_i = 1\},$$
$$Y_{cr,c}(s) \coloneqq \{Y_i : i \in V_{cr}(s) \wedge Z_i = 0\},$$
$$Y'_{cbr,t}(s) \coloneqq \{Y'_j : j \in C_{cbr}(s) \wedge z_j = 1\},$$
$$Y'_{cbr,c}(s) \coloneqq \{Y'_j : j \in C_{cbr}(s) \wedge z_j = 0\}.$$

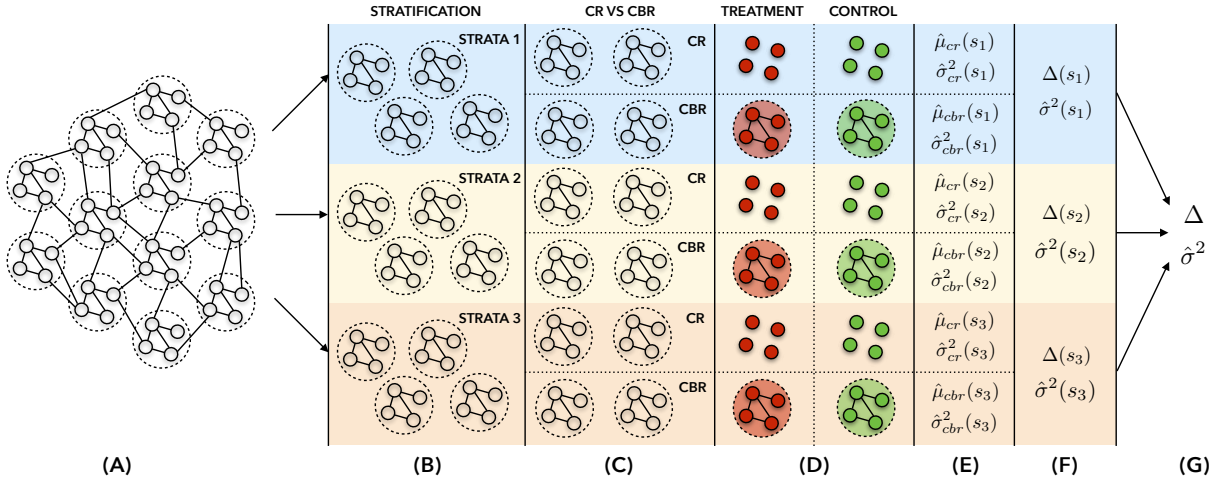

Figure 2: Illustration of the proposed experimental design for detecting network effects, using stratification to reduce variance and improve covariate balance. (A) Graph of all units and their connections; the dashed circles represent clusters. (B) Assigning clusters to strata (Section 2.4), (C) Assigning clusters within each strata to treatment arms: completely randomized (CR) and cluster-based randomized assignment (CBR). (D) Assigning units within strata to treatment buckets: treatment and control, using corresponding assignment strategy. (E) Computing the treatment effects of each treatment arm within each strata: $\hat{\mu}_{cr}(s)$ and $\hat{\mu}_{cbr}(s)$, and variance within each strata: $\hat{\sigma}^2_{cr}(s)$ and $\hat{\sigma}^2_{cbr}(s)$. (F) Computing the difference between the estimated effects using CR and CBR within each strata: $\Delta(s)$, and sum variances in each strata: $\hat{\sigma}^2(s)$. (G) Aggregating the differences across strata to compute the overall difference in differences (Δ) and the total variance ($\hat{\sigma}^2$).

The stratified estimators are given by:

$$\hat{\mu}_{cr}(s) \coloneqq \overline{Y_{cr,t}(s)} - \overline{Y_{cr,c}(s)}$$
$$\hat{\mu}_{cbr}(s) \coloneqq \frac{m_{cbr}(s)}{n_{cbr}(s)} \left(\overline{Y'_{cbr,t}(s)} - \overline{Y'_{cbr,c}(s)} \right)$$
$$\Delta(s) \coloneqq \hat{\mu}_{cr}(s) - \hat{\mu}_{cbr}(s).$$

Note that for every strata s, the following quantity $\hat{\sigma}^2(s)$ upper-bounds $\mathrm{var}_{\mathbf{W}(s),\mathbf{Z}(s)}[\Delta(s)]$ in expectation:

$$\hat{\sigma}^2(s) \coloneqq \frac{\hat{S}_{cr,t}(s)}{n_{cr,t}(s)} + \frac{\hat{S}_{cr,c}(s)}{n_{cr,c}(s)} + \frac{m^2_{cbr}(s)}{n^2_{cbr}(s)} \left(\frac{\hat{S}'_{cbr,t}(s)}{m_{cbr,t}(s)} + \frac{\hat{S}'_{cbr,c}(s)}{m_{cbr,c}(s)} \right).$$

Since the assignment of units to treatment is independent across strata, we can extend the previous test across strata by considering the following numerator and denominator:

$$\Delta = \sum_{s \in [1,S]} \frac{m(s)}{M} \Delta(s)$$
$$\hat{\sigma}^2 = \sum_{s \in [1,S]} \left(\frac{m(s)}{M} \right)^2 \hat{\sigma}^2(s)$$

We refer the reader to Figure 2 for an illustration.

3 EXPERIMENTS ON LINKEDIN'S PLATFORM

3.1 Experimental Scenario

Major Internet companies like Google [29], Microsoft [18], Facebook [5], or LinkedIn [35] rely heavily on experimentation to understand the effects of each product decision, before deploying any changes to the majority of their user base. As a result, these companies have each built mature experimentation platforms. However, how many of the experiments run on these platforms violate SUTVA is an open question. Together with the team running LinkedIn's experimentation platform, we applied the proposed theoretical framework to test for interference in two randomized experiments on LinkedIn.

LinkedIn users can interact with content posted by their connections through an algorithmically sorted feed. To improve the user experience, teams at LinkedIn continually modify the feed ranking algorithm and seek to determine the impact of these changes on key user metrics by running randomized experiments. Experiments of this kind are a typical case where the treatment effects may spill over between the treatment and control units: if a user is assigned to an effective treatment then they are more likely to interact with the feed, which in turn will impact the feeds of their connections. The goal of our experiments is to determine whether these spillover effects significantly bias the treatment effect estimators or SUTVA can be safely assumed.

In remainder of this section, we provide a detailed, step by step, description of the deployment of our design on LinkedIn's experimentation platform. We show how to find balanced clusters at scale, evaluating four different algorithms (Section 3.2); how to construct strata based on cluster covariates in order to ensure covariate balance and reduce the variance of the treatment effect estimates (Section 3.3); how to use lagged outcomes to reduce variance even further (Section 3.4); and finally, we report the results of two experiments testing different feed ranking algorithms (Section 3.5). The guidelines provided in this section are not specific to LinkedIn and can be applied to any large-scale experimentation platform.

3.2 Clustering the graph

The main challenge of implementing the proposed test for interference is finding a clustering algorithm that (*i*) finds clusters of highly inter-connected nodes, (*ii*) generates balanced clusters, and (*iii*) scales to the size of the LinkedIn graph. The LinkedIn graph contains hundreds of millions of nodes and billions of edges. As a result, we restrict our search to *parallelizable streaming* algorithms. Furthermore, because of the way social networks tend to be structured, most clustering (community detection) techniques find clusters with highly skewed size distribution [13, 19]. We therefore further restrict our search to clustering algorithms which explicitly enforce balance. We report the results of our evaluation of the relevant state-of-the-art clustering algorithms. Before presenting the empirical results, we provide a brief justification of why we need balanced clustering.

Why balanced clustering is necessary. As stated in Section 2.3, having clusters of equal size simplifies the theoretical analysis of the variance of our estimators under the null hypothesis. There are also two main practical reasons for partitioning the graph into clusters of equal size: (*i*) variance reduction, and (*ii*) balance on pre-treatment covariates.

First, clusters of equal size will tend to have more similar aggregated outcomes per cluster (Y'), which leads to smaller variance of the estimator, $\hat{\sigma}^2_{cbr}$. In particular, the values of $\hat{S}'_{cbr,t}$ and $\hat{S}'_{cbr,c}$ in Equation 6 will tend to be smaller.

Second, due to homophily [21], users who are connected will tend to be more similar to each other, leading to homogeneous clusters. Thus, if the clusters are not balanced, then large clusters of similar users will tend to dominate the treatment/control population (depending on where the clusters were randomly assigned), making it harder to achieve balance on pre-treatment covariates.

Thus, even if unbalanced clustering results in clusters of higher quality, in terms of number of edges cut, that will not necessarily help us to achieve our ultimate goal of detecting network effects. Finally, note that these observations are not specific to our design, but hold for any cluster-based randomized design.

Evaluation of balanced clustering algorithms. To test different clustering algorithms, we extracted a subgraph of the full LinkedIn graph containing only active members from the Netherlands and the connections between them. The subgraph contains >2.5M nodes and >300M edges. We picked the Netherlands because (*i*) it is a tightly-connected graph, and (*ii*) it fits in memory on a single machine, which allowed us to compare the streaming algorithms to state-of-the-art batch algorithms. We tested four algorithms.

METIS [17] is a widely-used *batch* graph clustering algorithm, and thus serves as our baseline to compare the quality of the clustering achieved by the streaming algorithms. It consists of three phases: (*i*) coarsening of the original graph, (*ii*) partitioning of the coarsened graph, (*iii*) uncoarsening of the partitioned graph.

Balanced Label Propagation (BLP) [33] is an iterative algorithm that greedily maximizes edge locality by (*i*) given the current cluster assignment, determining the reduction in edges cut from moving a node to another cluster, (*ii*) solving a Linear Program to find an optimal relocation of nodes while maintaining balance, and (*iii*) moving the nodes to the desired clusters according to the relocation found in step *ii*. Note that step *i* and *iii* can be easily parallelized and ran in streaming fashion. Step *ii* requires solving an LP with linear number of variables and quadratic number of constraints w.r.t. the number of clusters.

Table 1: Evaluation of the different balanced clustering algorithms. We report the percentage of within-cluster edges per clustering of the Netherlands LinkedIn graph. The values in bold represent the best performance. For BLP, we report results only for $k = 100$ and $k = 300$, since the running times of one iteration for larger values of k were too long.

Number of clusters (k)	BLP	reFENNEL	reLDG	METIS
100	26.7%	31.7%	**35.6%**	35.0%
300	22.7%	27.7%	**29.9%**	29.4%
500	-	26.1%	**27.7%**	27.0%
1000	-	23.9%	**24.7%**	23.8%

Restreaming Linear Deterministic Greedy (reLDG) [23] is a restreaming version of the Linear Deterministic Greedy algorithm [27]. Nishimura and Ugander [23] show that restreaming significantly increases the quality of the clusters compared to a single pass. LDG assigns each node u to a cluster i according to:

$$\arg\max_{i \in 1 \dots k} |C_i \cap N(u)| \left(1 - \frac{|C_i|}{C}\right), \tag{12}$$

where $C(i)$ is the set of all nodes in cluster i in the most recent assignment, $N(u)$ is the set of neighbors of u, and C is the capacity (i.e., maximum number of nodes) allocated for each cluster (usually set to $\frac{n}{k}$ to achieve perfect balance). The first term maximizes the number of edges within clusters, while the second term enforces balance on the cluster sizes.

Restreaming FUNNEL (reFUNNEL) [23] is a restreaming version of the FUNNEL algorithm [32], which is itself a streaming generalization of the modularity maximization. It assigns nodes to clusters as:

$$\arg\max_{i \in 1 \dots k} |C_i \cap N(u)| - \alpha |C_i|,$$

where α is a hyper-parameter. Note that, unlike LDG, FUNNEL ensures only approximate balance, unless $\alpha \geq \lceil \frac{n}{k} \rceil$. Nishimura and Ugander [23] suggest increasing α in each restreaming pass to achieve best results. We run with linearly and logarithmically increasing schedules.

We set the number of clusters to $k = \{100, 300, 500, 1000\}$. For each algorithm and value of k, we measured the percentage of edges found within the clusters (Table 1). We ran BLP for 10 iterations, and reLDG and reFUNNEL for 20 iterations. In both cases this was enough for the algorithms to converge. We found that for larger numbers of clusters ($k \geq 300$) running one iteration of the BLP algorithm using the *GLPK* solver (GNU Linear Programming Kit) takes more than a day. It is worth noting that the bottleneck of the

Table 2: Results of clustering the full LinkedIn graph. We ran the parallel version of *reLDG* on 350 Hadoop nodes for 35 iterations.

Number of clusters (k)	Percantage of edges within clusters	Cluster sizes (mean ± std)
1000	35.59%	43893.2 ± 634.5
3000	28.54%	14631.1 ± 109.3
5000	26.16%	8778.6 ± 199.3
7000	22.77%	6270.5 ± 40.5
10000	21.09%	4389.3 ± 67.2

algorithm is solving the LP, which depends on the number of clusters, rather than the size of the graph. reLDG and reFUNNEL do not have this limitation as their running time, for a single pass, is $O(nk)$ and, in practice, larger values of k do not significantly increase the running time. In terms of clustering quality, reLDG consistently outperforms all other algorithms (Table 1), including METIS which requires the full graph to be loaded in memory. reFUNNEL performs worse than reLDG and METIS, except for $k = 1000$ when it achieves similar clustering quality as METIS. Finally, BLP lags behind, performing significantly worse than all other methods.

Clustering the full graph. Next, we apply the reLDG—the best performing balanced clustering algorithm—on the full LinkedIn graph containing hundreds of millions of nodes and billions of edges. As mentioned above, reLDG's running time is $O(nk)$ and can be easily parallelized [23]. We ran the parallel version of reLDG on 350 Hadoop nodes for 35 iterations. We set $k = \{1000, 3000, 5000, 7000, 10000\}$ and a leniency of 1% for the balance constraint, to slightly sacrifice balance for better clustering quality. We find that even when clustering the graph in 1000 clusters more than one third (35.59%) of all edges are between nodes within the same clusters (Table 2). Expectedly, as we increase k, the number of edges within clusters decreases, with $k = 10000$ having 21.09% of the edges within clusters. We observe that most clusters are of similar size, except for very few clusters that are smaller due to the algorithm's leniency. We also looked at the distributions of the number of edges within each cluster (proxy for possible network effects) and the number of edges to other clusters (proxy for possible spillovers). We find that although there is some heterogeneity between the clusters, there are very few outliers (Table 2). Finally, we analyzed the sensitivity of clustering quality to different initializations. For $k = \{3000, 5000\}$, we run reLDG with four random initializations and, in both cases, we found very small differences—with standard deviation of 0.03%—between different runs.

3.3 Stratifying the clusters

To ensure balance on cluster-level covariates and to reduce the variance of the treatment effect estimates—as discussed in Section 2.4—we use stratification to assign clusters to treatment arms, and in the CBR treatment arm, clusters to treatment buckets. Stratification produces the greatest gains in variance reduction when the covariates used to stratify are predictive of the outcomes. While we cannot observe the outcomes before we run the experiment, we do have historical data about the past behavior of the users, including the pre-treatment outcomes (the key metric of interest just before launching the experiment). Although, we hope that the treatment will significantly increase the outcome, we do expect that the pre-treatment outcomes will be highly correlated with the post-treatment outcomes. Note that even in the worst-case scenario when the selected covariates fail to predict the post-treatment outcomes, the treatment effect estimates still remain unbiased and have no more sampling variance than the estimates we would have obtained using a completely randomized assignment [14].

Table 3: The effect of stratification on pre-treatment variance. We report the empirical variance of the difference-in-difference-in-means estimator (Δ) using the pre-treatment outcomes. To avoid disclosing raw numbers all values are multiplied by a constant.

Number of clusters (k)	No stratification	Balanced k-means stratification
1000	0.890	1.000
3000	0.592	0.650
5000	0.590	0.545
7000	0.445	0.451
10000	0.400	0.372

We describe each cluster using four covariates: number of edges within the cluster, number of edges to other clusters, and two metrics that characterize users' engagement with the LinkedIn feed averaged over all users in the cluster (one of which is the pre-treatment outcome). To group clusters into strata, we used balanced k-means clustering. We experimented with two algorithms: [20] led to more balanced groups of clusters (strata) but does not scale to more than 5000 data points, whereas [6] is faster, more scalable but outputs clusters (strata) that are slightly less balanced. We report results only for the latter.

We cannot measure the effects of stratification on the post-treatment variance since we can run the experiment only once, either with or without stratification. However, we can measure the effects on the pre-treatment variance. Table 3 shows the pre-treatment variance of the difference-in-difference-in-means estimator under our design with and without stratification. For small values of $k = \{1000, 3000\}$ stratifying increases the variance. However, for larger values of $k = \{5000, 7000, 10000\}$ stratification leads to smaller or similar variance.

3.4 Variance reduction using lagged outcomes

In order to further reduce the variance of the estimator of the network effect, we define a new variable as the difference between post-treatment and pre-treatment outcomes, as suggested in [11]:

$$Y_i^* = Y_{i,t} - Y_{i,t-1},$$

where $Y_{i,t}$ is the outcome of unit i at period t and $Y_{i,t-1}$ is the outcome of unit i one period unit prior to t. Since this is only a question of choosing the appropriate outcome variable, this does not change the validity of our procedure. In practice, we chose 2 and 4 weeks as the difference between t and $t-1$ in the first and the second experiment, respectively.

Table 4: Results of two online experiments testing for network effects ran on 20% and 36% of all LinkedIn users. The outcomes are related to the level of users' engagement with the feed. We report results pre-treatment, post-treatment, and post-treatment using the variance reduction technique explained in Section 3.4. We refer to the first treatment arm as BR, instead of CR, since we used a Bernoulli randomized assignment instead of a completely randomized assignment (Section 3.5). To avoid disclosing raw numbers all values are multiplied by a constant, except for the final row which displays the two-tailed p-value of the test statistic T under assumption of normality $T \sim \mathcal{N}(0,1)$.

	Experiment 1			Experiment 2		
Statistic	pre-treatment	post-treatment	post-treatment ($\mathbf{Y} = \mathbf{Y_t} - \mathbf{Y_{t-1}}$)	pre-treatment	post-treatment	post-treatment ($\mathbf{Y} = \mathbf{Y_t} - \mathbf{Y_{t-1}}$)
BR Treatment Effect ($\hat{\mu}_{br}$)	-0.0261	0.0432	0.0559	0.0230	0.2338	0.2108
CBR Treatment Effect ($\hat{\mu}_{cbr}$)	0.0638	0.1653	0.0771	0.2733	0.8123	0.5390
Delta ($\Delta = \hat{\mu}_{br} - \hat{\mu}_{cbr}$)	-0.0899	-0.1221	-0.0211	-0.2504	-0.5785	-0.3281
BR standard deviation ($\hat{\sigma}_{br}$)	0.0096	0.0098	0.0050	0.3269	0.3414	0.2911
CBR standard deviation ($\hat{\sigma}_{cbr}$)	0.0805	0.0848	0.0260	0.9332	0.9966	0.5613
Delta standard deviation ($\hat{\sigma}$)	0.0811	0.0856	0.0265	0.9367	1.0000	0.5712
p-value (2-tailed)	**0.2670**	**0.1530**	**0.4246**	**0.5753**	**0.2560**	**0.0483**

3.5 Experimental results

We ran two separate experiments on the LinkedIn graph, in August and November 2016, respectively. The experiments tested two separate changes in the feed ranking algorithm. Note that the treatment did not have any trivial network effect, e.g, it did not specifically encourage users to interact with each other. The outcome of interest is the number of sessions in which the user actively engages with their feed.

Practical Considerations. In order to address some of the challenges of running experiments in real-time on a non-fixed population, the LinkedIn experimentation platform [35] is implemented to run Bernoulli randomized assignments (BR) and not completely randomized assignments. A Bernoulli randomized assignment assigns each unit to treatment or control with probability p independently at random, whereas a completely randomized assignment (CR) assigns exactly $n_t = \lfloor p \cdot N \rfloor$ units to treatment, chosen uniformly at random. For large sample sizes, as it is the case in our experiments, the difference between a CR and BR assignment is negligible for the purpose of our test. The edge cases where no units are assigned to treatment or control are very unlikely when N is large. In [24], we provide a formal explanation for why running a Bernoulli randomized assignment does not affect the validity of our test in practice.

With these constrains in mind, we run our experiments as follows:

(*i*) We cluster the LinkedIn graph into balanced clusters (Section 3.2). We set the number of clusters to 3000 in the first experiment and to 10000 in the second experiment.
(*ii*) We stratify the clusters based on their covariates using balanced k-means stratification (Section 3.3).
(*iii*) In each stratum, we randomly assign clusters to the CBR treatment arm, and subsequently to the treatment or control bucket.
(*iv*) Units that were not assigned to the CBR treatment arm are passed to the main experimentation pipeline, where a subpopulation is first sampled at random, and then assigned to treatment or control using Bernoulli randomization.

Prior to launching the experiments, we ran a series of balance checks on the background characteristics of the users to ensure that there are no systematic differences between the populations in the two treatment arms and between the treatment and control groups within each arm. We compared the distributions of number of connections per user, levels of user activity, and the number of premium member accounts.

We report the results of the two experiments in Table 4. To avoid disclosing raw numbers, we multiply all values by a constant, expect for the last row, which displays the two-tailed p-value.

Experiment 1. We ran the experiment for two weeks on 20% of the LinkedIn users: 10% assigned to Bernoulli randomization (BR) and 10% assigned to cluster-based randomization (CBR). We first test whether there is any systematic bias of the outcomes in the assignment prior to experiment. We apply the test for network effects on pre-treatment outcomes (A/A test) and, as expected, we do not find any significant effects. Next, we test for network effects post-treatment: we fail to reject the null hypothesis that SUTVA holds. The treatment effects within each arm were not as significant as expected, which potentially led to smaller networks effects and not enough evidence for our test to reject the null. We observe that most of the variance comes from the CBR treatment arm. Using the lagged outcomes ($\mathbf{Y} = \mathbf{Y_t} - \mathbf{Y_{t-1}}$) reduces the variance ($\hat{\sigma}$) 3.2 times, but also reduces the difference in treatment effects ($\hat{\mu}_{br} - \hat{\mu}_{cbr}$) 5.8 times. We still fail to reject the null.

Experiment 2. In this experiment, we used a larger test population, 36% of all LinkedIn users: 20% assigned to Bernoulli randomization (BR) and 16% assigned to cluster-based randomization (CBR). We also ran the experiment for a longer period of time: 4 weeks. As in experiment 1, we first test for network effects using pre-treatment outcomes (A/A test) and we do not find significant effects. Testing post-treatment, we also fail to reject the null. However, by applying the variance reduction technique described in Section 3.4, we reduce the standard deviation ($\hat{\sigma}$) 1.8 times, while also reducing the difference in treatment effects ($\hat{\mu}_{br} - \hat{\mu}_{cbr}$) 1.8 times. We find significant network effects: we *reject* the null that SUTVA holds ($p = 0.048$).

Although we did not reject the null hypothesis twice, in both experiments we found positive network effects: the difference-in-means estimate was higher in the cluster-based randomization treatment arm ($\hat{\mu}_{cbr}$) than the Bernoulli randomization one ($\hat{\mu}_{br}$). Given the nature of the outcome variable, this behavior is expected. In fact, the outcome of interest, which measures the number of sessions in which a user engages with an item on their feed, yields itself to positive interference effects: the more of user's connections engage with items on their feed, the more content to engage on the user's feed there will be. The fact that we rejected the null in the second experiment, but not in the first, suggests that small treatment effects in the first experiment were likely the reason why no strong interference effects were observed.

4 FUTURE WORK

This work opens many avenues for future research. In this section, we highlight a few.

First, the proposed design compares a cluster-based randomized assignments with a completely randomized assignment. A very similar test could be investigated where instead of comparing these two designs, each treatment arm assigns different proportions of treated units. An interesting result would be to understand how the Type I and Type II error of these two hierarchical designs compare.

Second, there is a growing literature on identifying heterogeneous treatment effect, which we believe can be adapted to this framework. Cursory analysis of the experiments run showed that interference effects seemed strongest for moderate users of the LinkedIn platforms, but were weaker for very recurrent users of LinkedIn and less-recurrent users.

Finally, as mentioned at the end of Section 2, the type II error of our test is strongly dependent on how "good" our initial clustering of the graph is. With all else being equal, this means cutting fewer (possibly weighted) edges of the graph. A follow-up to our work would be to explore clustering algorithms which manage both objectives of minimizing edges cut but also managing the final empirical variance.

ACKNOWLEDGMENTS

We would like to thank Guillaume Basse, Igor Perisic, Edmond Awad, and Dean Eckles for useful comments and discussions. This work was partially done while Martin Saveski and Guillaume Saint-Jacques were interns at LinkedIn.

REFERENCES

[1] Peter M Aronow. 2012. A general method for detecting interference between units in randomized experiments. *Sociological Methods & Research* (2012).
[2] Peter M Aronow, Joel A Middleton, and others. 2013. A class of unbiased estimators of the average treatment effect in randomized experiments. *Journal of Causal Inference* (2013).
[3] Peter M Aronow and Cyrus Samii. 2013. Estimating average causal effects under interference between units. *preprint arXiv:1305.6156* (2013).
[4] Susan Athey, Dean Eckles, and Guido W Imbens. 2016. Exact P-values for Network Interference. *J. Amer. Statist. Assoc.* (2016).
[5] Eytan Bakshy, Dean Eckles, and Michael S Bernstein. 2014. Designing and deploying online field experiments. In *Proceedings of the international conference on World wide web*.
[6] Arindam Banerjee and Joydeep Ghosh. 2006. Scalable clustering algorithms with balancing constraints. *Data Mining and Knowledge Discovery* (2006).
[7] Guillaume Basse and Edoardo Airoldi. 2017. Limitations of design-based causal inference and A/B testing under arbitrary and network interference. *preprint arXiv:1705.05752* (2017).
[8] Guillaume W Basse and Edoardo M Airoldi. 2015. Optimal design of experiments in the presence of network-correlated outcomes. *preprint arXiv:1507.00803* (2015).
[9] David Roxbee Cox. 1958. Planning of experiments. (1958).
[10] Eckles Dean, Karrer Brian, and Ugander Johan. 2017. Design and Analysis of Experiments in Networks: Reducing Bias from Interference. *Journal of Causal Inference* (2017).
[11] Alex Deng, Ya Xu, Ron Kohavi, and Toby Walker. 2013. Improving the sensitivity of online controlled experiments by utilizing pre-experiment data. In *Proceedings of the ACM international conference on Web search and data mining*.
[12] Laura Forastiere, Edoardo M Airoldi, and Fabrizia Mealli. 2016. Identification and estimation of treatment and interference effects in observational studies on networks. *preprint arXiv:1609.06245* (2016).
[13] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* (2010).
[14] Alan S Gerber and Donald P Green. 2012. *Field experiments: Design, analysis, and interpretation*. WW Norton.
[15] Michael G Hudgens and M Elizabeth Halloran. 2008. Toward Causal Inference with Interference. *J. Amer. Statist. Assoc.* (2008).
[16] Guido W Imbens and Donald B Rubin. 2015. *Causal Inference in Statistics, Social, and Biomedical Sciences*. Cambridge University Press.
[17] George Karypis and Vipin Kumar. 1998. Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed computing* (1998).
[18] Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, and Nils Pohlmann. 2013. Online controlled experiments at large scale. In *Proceedings of ACM SIGKDD international conference on Knowledge discovery and data mining*.
[19] Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. 2009. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics* (2009).
[20] Mikko I Malinen and Pasi Fränti. 2014. Balanced k-means for clustering. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*.
[21] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* (2001).
[22] Joel A Middleton and Peter M Aronow. 2011. Unbiased estimation of the average treatment effect in cluster-randomized experiments. (2011).
[23] Joel Nishimura and Johan Ugander. 2013. Restreaming graph partitioning: simple versatile algorithms for advanced balancing. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*.
[24] Jean Pouget-Abadie, Martin Saveski, Guillaume Saint-Jacques, Weitao Duan, Ya Xu, Souvik Ghosh, and Edoardo Airoldi. 2017. Testing for arbitrary interference on experimentation platforms. *preprint arXiv:1704.01190* (2017).
[25] Paul R Rosenbaum. 2007. Interference between units in randomized experiments. *J. Amer. Statist. Assoc.* (2007).
[26] Donald B Rubin. 1990. Formal mode of statistical inference for causal effects. *Journal of statistical planning and inference* (1990).
[27] Isabelle Stanton and Gabriel Kliot. 2012. Streaming graph partitioning for large distributed graphs. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*.
[28] Daniel L Sussman and Edoardo M Airoldi. 2017. Elements of estimation theory for causal effects in the presence of network interference. *preprint arXiv:1702.03578* (2017).
[29] Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. 2010. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*.
[30] Eric J Tchetgen Tchetgen and Tyler J VanderWeele. 2012. On causal inference in the presence of interference. *Statistical Methods in Medical Research* (2012).
[31] Panos Toulis and Edward Kao. 2013. Estimation of Causal Peer Influence Effects. In *International Conference on Machine Learning*.
[32] Charalampos Tsourakakis, Christos Gkantsidis, Bozidar Radunovic, and Milan Vojnovic. 2014. Fennel: Streaming graph partitioning for massive scale graphs. In *Proceedings of the ACM international conference on Web search and data mining*.
[33] Johan Ugander and Lars Backstrom. 2013. Balanced label propagation for partitioning massive graphs. In *Proceedings of the ACM international conference on Web search and data mining*.
[34] Johan Ugander, Brian Karrer, Lars Backstrom, and Jon Kleinberg. 2013. Graph cluster randomization: Network exposure to multiple universes. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*.
[35] Ya Xu, Nanyu Chen, Addrian Fernandez, Omar Sinno, and Anmol Bhasin. 2015. From infrastructure to culture: A/B testing challenges in large scale social networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.