Minimizing Polarization and Disagreement in Social Networks via Link Recommendation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Individual's opinions are fundamentally shaped and evolved by their interactions with other people, and social phenomena such as disagreement and polarization are now tightly woven into daily life. The quantification and optimization of these concepts have been the subject of much recent research behind a wealth of high-impact data mining applications. In particular, researchers have addressed the question of how such concepts can be optimized by influencing the opinion of a small number of individuals or by designing the network from scratch.

Here, rather than a "design-from-scratch" approach or altering the initial opinion, we study the optimization problem of recommending k new links to minimize the sum of polarization and disagreement in a social network with n nodes and m edges. We show that our objective function of this combinatorial optimization problem is not submodular, although it is monotone. We propose a simple greedy algorithm with a constant-factor approximation that solves the problem in cubic running time, and we provide theoretical analysis of the approximation guarantee for the algorithm. To overcome the computation challenge for large networks, we also provide a fast algorithm with computation complexity $\widetilde{O}(mk\epsilon^{-2})$ for any $\epsilon > 0$, where the $\widetilde{O}(\cdot)$ notation suppresses the $\mathrm{poly}(\log n)$ factors. Extensive experiments on real datasets demonstrate both the efficiency and effectiveness of our algorithms.

1 Introduction

Social networks and social media play a prominent part in the propagation, evolution, and formulation of opinions [30], leading to fundamental changes in how humans share and shape opinions. Particularly, in recent years, the tremendous prevalence of online social media platforms produces various social phenomena, such as polarization and disagreement. The identification [45] and optimization of these quantities are fundamental tasks behind a myriad of high-impact data mining applications, and thus have received considerable attention.

We focus on the phenomena of disagreement and polarization. Disagreement [36, 15] characterizes how much acquaintances disagree in their opinions, globally across the network. Polarization [36, 33, 15] measures how equilibrium expressed opinions deviate from the average. However, existing recommender system, trained on real-data, with the goal to increase user engagement may stop the user from being exposed to diverse opinions and naturally end up creating "echo-chambers". In other words, the recommended links minimizing disagreement may lead to greater polarization [36] since connections between users with similar mindsets are preferred for such system. Yet, exposure to diverse content is necessary to obtain a complete picture about a topic [26]. Thus, there is a need for a radically different approach to suggest links that decrease both disagreement and polarization, which motivates our work.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

In this paper, we address the following optimization problem: given a social network with n nodes and m edges, and an integer k, how to strategically recommend k links to the individuals so that the polarization and disagreement is minimized. We show that our objective function of this combinatorial optimization problem is not submodular, although it is monotone. To tackle the exponential computation complexity, we resort to greedy algorithm extended for non-submodular function [5] by iteratively adding the most promising edges. We propose a simple greedy algorithm with a constant-factor approximation that solves the problem in cubic running time. To confront the computation challenge for large networks, we also provide a fast algorithm with computation complexity $\widetilde{O}(mk\epsilon^{-2})$ for any $\epsilon > 0$, where $\epsilon > 0$ is the error parameter and the $\widetilde{O}(\cdot)$ notation suppresses the $\mathrm{poly}(\log n)$ factors. We confirm our theoretical and algorithmic performance by executing extensive experiments on various real networks, which show that our algorithms are efficient and effective, outperforming several other strategies of creating edges.

Omitted proofs and implementation are provided as supplementary material.

2 Related work

Modeling opinion dynamics. Opinion dynamics has been the subject of intense recent research to model social learning processes in various disciplines [28, 18, 4]. These models capture the mechanisms and factors influencing opinion formulation, shedding light on understanding the whole process of opinion shaping and diverse phenomena taking place in social media. In the past decades, numerous relevant models have been proposed [37], among which the Friedkin-Johnsen (FJ) model [19] is one of the most popular models. After its establishment, the FJ model has been extended in a variety of ways [28, 27, 44, 13], by incorporating different factors affecting opinion dynamics, such as peer pressure [39], susceptibility to persuasion [1], and opinion leader [24]. Under the formalism of the FJ model or its variants, some relevant quantities, properties and explanations have been broadly studied, including the equilibrium expressed opinions [17, 6], sufficient condition for the stability [38], the average initial opinion [17], interpretations [23, 6], and so on.

Optimization problems in opinion dynamics. Recently, several optimization problems related to opinion dynamics have been formulated and studied for different objectives. For example, a long line of work has been devoted to maximizing the overall opinion by using different strategies, such as identifying a fixed number of individuals and setting their expressed opinions to 1 [24], changing agent's internal opinions [44], as well as modifying individuals' susceptibility to persuasion [1, 8]. [43] studies the related problem of allocating seed users to opposing campaigns with the goal to maximize the expected number of users who are co-exposed to both campaigns. In additon, [22] studies the related problem of balancing the information exposure. These studies have far-reaching implications in product marketing, public health campaigns, and political candidates. Another major and increasingly important focus of research is optimizing some social phenomena, such as maximizing the diversity [32, 34], minimizing conflict [12], disagreement [20], and polarization [33]. A recent work considers the problem of minimizing the sum of polarization and disagreement by changing the underlying network topology [36]. Different from the "design-from-scratch" approach, we instead concentrate on edge recommendation, a practical incremental approach that suggests modifications to an existing network to achiever our goal. Moreover, our proposed algorithms differ significantly from existing ones, in spite of the fact our objective functions are identical to that in [36].

Link recommendation strategy. Our problem aims to optimize an objective via link recommendation. Here, we review some instances of related edge recommendation problems. [14, 40, 16] improves the centrality of a node and [3] strategically fights opinion control in social networks. As for opinion dynamics, the operation of creating edges is actually convenient and practical, which is equivalent to making friendship between individuals. In [21], the edge addition strategy is explored on the endorsement graph, with an aim to reduce controversy, and it also takes account of the acceptable probability of the recommended edge. In [6], addition of edges is discussed in order to reduce the social cost, namely the lack of agreement. Moreover, [12] exploits link recommendation strategy to reduce the risk of conflict, [26] considers link insertions to reduce structural bias which may trap a reader in a "polarized" bubble with no access to other opinions. Our work is motivated in part by these studies, while we address a new optimization problem, and present new algorithms, with the faster one being nearly linear.

3 Preliminary

In this section, we give a brief introduction to some essential concepts and tools, in order to facilitate the description of the problem and its related greedy algorithms.

Graph and Related Matrix. Let $\mathcal{G} = (V, E)$ be a connected undirected simple graph (network) with $|V| = n$ nodes and $|E| = m$ edges. The adjacency relation of all nodes in $\mathcal{G}$ is encoded in its adjacency matrix $\boldsymbol{A} = (a_{ij})_{n\times n}$. If nodes i and j are adjacent by an edge e, then $a_{ij} = a_{ji} = 1$; $a_{ij} = a_{ji} = 0$ otherwise. Let N_i be the set of neighbours of node i satisfying $N_i = \{j|\{i, j\} \in E\}$. Then, the degree d_i of a node i is $d_i = \sum_{j=1}^{n} a_{ij} = \sum_{j\in N_i} a_{ij}$, and the diagonal degree matrix of $\mathcal{G}$ is defined as $\boldsymbol{D} = \mathrm{diag}(d_1, d_2, \ldots, d_n)$.

The Laplacian matrix of $\mathcal{G}$ is defined to be $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$. There is also an alternative construction of $\boldsymbol{L}$ by using the incidence matrix $\boldsymbol{B} \in \mathbb{R}^{|E|\times|V|}$, an $m \times n$ signed edge-node incidence matrix. The element b_{ev}, $e \in E$ and $v \in V$, of $\boldsymbol{B}$ is defined as follows: $b_{ev} = 1$ if node v is the head of edge e, $b_{ev} = -1$ if node v is the tail of edge e, and $b_{ev} = 0$ otherwise. For an edge $e \in E$ with two end nodes i and j, the row vector of $\boldsymbol{B}$ corresponding to e can be written as $\boldsymbol{b}_{ij} \triangleq \boldsymbol{b}_e = \boldsymbol{e}_i - \boldsymbol{e}_j$ where $\boldsymbol{e}_i$ denote the i-th standard basis vector of appropriate dimension. Then the Laplacian matrix $\boldsymbol{L}$ of $\mathcal{G}$ can also be represented as $\boldsymbol{L} = \boldsymbol{B}^\top \boldsymbol{B}$, which indicates that $\boldsymbol{L}$ is symmetric and positive semidefinite. Moreover, the Laplacian matrix $\boldsymbol{L}$ of a connected graph $\mathcal{G}$ has a unique zero eigenvalue.

Let $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{n-1}$ be the nonzero eigenvalues of $\boldsymbol{L}$ of a connected graph $\mathcal{G}$, and $\boldsymbol{u}_i$ be the corresponding orthogonal eigenvectors. Then, $\boldsymbol{L}$ has an eigendecomposition of form $\boldsymbol{L} = \boldsymbol{U}\Lambda\boldsymbol{U}^\top = \sum_{i=1}^{n-1} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^\top$ where $\Lambda = \mathrm{diag}\,(0, \lambda_1, \lambda_2, .., \lambda_{n-1})$ and $\boldsymbol{u}_i$ is the i-th column of matrix $\boldsymbol{U}$. Let $\lambda_{\max}$ and $\lambda_{\min}$ be, respectively, the maximum and nonzero minimum eigenvalue of $\boldsymbol{L}$. Then, $\lambda_{\max} = \lambda_{n-1} \leq n$ [41], and $\lambda_{\min} = \lambda_1 \geq 1/n^2$ [31].

The forest matrix of graph $\mathcal{G}$ is defined as $\boldsymbol{\Omega} = (\boldsymbol{I} + \boldsymbol{L})^{-1} = (\omega_{ij})_{n\times n}$ [25, 11]. Matrix $\boldsymbol{\Omega}$ is a doubly stochastic matrix [9, 10], satisfying $\boldsymbol{\Omega}\mathbf{1} = \mathbf{1}$ and $\mathbf{1}^\top\boldsymbol{\Omega} = \mathbf{1}^\top$ where $\mathbf{1}$ denotes the all-ones vector. For a pair of nodes v_i and v_j, their forest distance r_{ij} is defined as $r_{ij} \triangleq \boldsymbol{b}_{ij}^\top \boldsymbol{\Omega} \boldsymbol{b}_{ij}$, obeying relation $0 \leq r_{ij} \leq 2$.

Greedy Algorithms for Non-Submodular Functions. For submodular maximization problems, the greedy algorithm has become a prevalent choice for solving them with a guaranteed $(1 - 1/e)$ approximation ratio. However, there are a large class of important set functions optimization in network topology design fail to be submodular. To alleviate this issue, a generalized approximation ratio has been proved and further provides a performance guarantee for non-submodular functions based on two quantities: submodularity ratio γ and curvature α, which characterize how close these functions are from being sub- or supermodular [5].

Notice that for a non-decreasing function f, it follows that its submodular ratio $\gamma \in [0, 1]$ and $\gamma = 1$ if and only if f is a submodular function. The curvature $\alpha \in [0, 1]$ and $\alpha = 0$ if and only if f is a supermodular function. Given those two quantities, the greedy algorithm enjoys a tight approximation guarantee of $\frac{1}{\alpha}\left(1 - e^{-\alpha\gamma}\right)$ for a larger class of optimization problems [5]. Details about these two quantities are provided in the supplementary material.

4 Problem Formulation

In this section, we briefly discuss the FJ model of opinion formation, as well as the definitions and measures for disagreement and polarization. Then we formally introduce the problem for optimizing the polarization-disagreement index in a social network.

Friedkin-Johnsen Model. The FJ model is one of the most popular models for opinion dynamics [19]. In the FJ model, each node $i \in V$ has two opinions: internal (or innate) opinion and the expressed opinion, both in the interval $[0, 1]$. The internal opinion for each node, denoted by $\boldsymbol{s}_i$, remains unchanged. Let $\boldsymbol{z}_i(t)$ be the expressed opinion of node i at time t, which evolves as

$$\boldsymbol{z}_i(t+1) = \frac{\boldsymbol{s}_i + \sum_{j\in N_i} a_{ij}\boldsymbol{z}_j(t)}{1 + \sum_{j\in N_i} a_{ij}}. \tag{1}$$

Let $\boldsymbol{s} = (\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n)^\top$ be the initial opinion vector. Let $\boldsymbol{z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n)^\top$ be the equilibrium expressed opinion vector, which be represented as $\boldsymbol{z} = (\boldsymbol{I} + \boldsymbol{L})^{-1}\boldsymbol{s}$ [6].

In the FJ model, individuals interact with their acquaintances and exchange opinions whereas the opinions of nodes often do not reach consensus, leading to disagreement, polarization and other important phenomena, which have been the subject of many recent works. In this paper, we study the disagreement and polarization using definitions given in [36].

For a graph $\mathcal{G} = (V, E)$ with expressed opinion vector $\boldsymbol{z}$, its disagreement $D(\mathcal{G}, \boldsymbol{z})$ is defined as $D(\mathcal{G}, \boldsymbol{z}) = \sum_{(i,j)\in E} (\boldsymbol{z}_i - \boldsymbol{z}_j)^2$. Let $\bar{\boldsymbol{z}}$ be the mean-centered equilibrium vector given by $\overline{\boldsymbol{z}} = \boldsymbol{z} - \frac{\boldsymbol{z}^\top \mathbf{1}}{n}\mathbf{1}$. Then the polarization $P(\mathcal{G}, \boldsymbol{z})$ is defined as: $P(\mathcal{G}, \boldsymbol{z}) = \sum_{i\in V} \overline{\boldsymbol{z}}_i^2 = \overline{\boldsymbol{z}}^\top \overline{\boldsymbol{z}}$. We further introduce the polarization-disagreement (P-D) index, i.e., the objective we are concerned with. For a graph $\mathcal{G} = (V, E)$ with expressed opinion vector $\boldsymbol{z}$, the polarization-disagreement index $\mathcal{I}(\mathcal{G}, \boldsymbol{z})$ is the sum of the polarization $P(\mathcal{G}, \boldsymbol{z})$ and disagreement $D(\mathcal{G}, \boldsymbol{z})$: $\mathcal{I}(\mathcal{G}, \boldsymbol{z}) = P(\mathcal{G}, \boldsymbol{z}) + D(\mathcal{G}, \boldsymbol{z})$. Convenient matrix-vector expressions for the above quantities are provided by the following proposition [12, 36, 45], we will use $D(\mathcal{G})$, $P(\mathcal{G})$ and $\mathcal{I}(\mathcal{G})$ to denote $D(\mathcal{G}, \boldsymbol{z})$, $P(\mathcal{G}, \boldsymbol{z})$ and $\mathcal{I}(\mathcal{G}, \boldsymbol{z})$ for simplicity.

Proposition 1 *$D(\mathcal{G})$, $P(\mathcal{G})$ and $\mathcal{I}(\mathcal{G})$ can be conveniently expressed in terms of quadratic forms as*

$$\begin{aligned} D(\mathcal{G}) &= \boldsymbol{z}^\top \boldsymbol{L}\boldsymbol{z} = \overline{\boldsymbol{z}}^\top \boldsymbol{L}\overline{\boldsymbol{z}} = \boldsymbol{s}^\top (\boldsymbol{I}+\boldsymbol{L})^{-1}\boldsymbol{L}(\boldsymbol{I}+\boldsymbol{L})^{-1}\boldsymbol{s}, \\ P(\mathcal{G}) &= \overline{\boldsymbol{z}}^\top \overline{\boldsymbol{z}} = \overline{\boldsymbol{s}}^\top (\boldsymbol{I}+\boldsymbol{L})^{-1}(\boldsymbol{I}+\boldsymbol{L})^{-1}\overline{\boldsymbol{s}}, \\ \mathcal{I}(\mathcal{G}) &= \overline{\boldsymbol{z}}^\top \boldsymbol{L}\overline{\boldsymbol{z}} + \overline{\boldsymbol{z}}^\top \overline{\boldsymbol{z}} = \overline{\boldsymbol{s}}^\top (\boldsymbol{I}+\boldsymbol{L})^{-1}\overline{\boldsymbol{s}}. \end{aligned}$$

Remark 1 *In this paper, we will generally assume that the internal opinions $\boldsymbol{s}$ are mean-centered, that is, $\bar{\boldsymbol{s}} = \boldsymbol{s}$. Note that in such case, $\boldsymbol{z}$ will also be mean-centered. In fact, as will be shown in supplementary material, whether the opinions $\boldsymbol{s}$ are mean-centered or not, the edges which we select to augment the network are the same since the variation of our objective remains unchanged.*

Formulation of Problem. Our problem is based on the following fact. For a connected undirected network $\mathcal{G}(V, E)$, given a set T consisting of edges not in E, if we augment the network by adding the edges in T, the P-D index of the new graph will not increase. In our subsequent analysis, we will use the following notation to improve readability of our lemmas. We use $\mathcal{G} + T$ to denote the graph augmented by adding the edges in T, that is $\mathcal{G} + T = (V, E \cup T)$. To evaluate the variation of the P-D index when adding edges, we define an objective function on the edge set $f : 2^{E_C} \to \mathbb{R}_+$ as

$$f(T) = \mathcal{I}(\mathcal{G}) - \mathcal{I}(\mathcal{G} + T), \tag{2}$$

in the sequel, we will use $f(e)$ and $\mathcal{G} + e$, respectively, to denote $f(\{e\})$ and $\mathcal{G} + \{e\}$ for simplicity. The variation of the P-D index under the perturbation of a single edge can be expressed by the following lemma.

Lemma 4.1 *For a candidate edge $e \in E_C$ connecting node u and v with row vector $\boldsymbol{b}_e = \boldsymbol{e}_u - \boldsymbol{e}_v$, one obtains*

$$f(e) = \frac{\boldsymbol{s}^\top \boldsymbol{\Omega}\boldsymbol{b}_e\boldsymbol{b}_e^\top \boldsymbol{\Omega}\boldsymbol{s}}{1 + \boldsymbol{b}_e^\top \boldsymbol{\Omega}\boldsymbol{b}_e} = \frac{(\boldsymbol{z}_u - \boldsymbol{z}_v)^2}{1 + r_{uv}} \geq 0. \tag{3}$$

Lemma 4.1 indicates that the addition of any nonexisting edge can reduce the P-D index. Then the following problem arises naturally: given a candidate edge set E_C, how to optimally select a subset T of E_C subject to a cardinality constraint, so that the polarization-disagreement index of the new graph is minimized. Such an optimization problem has been the subject of many recent papers [20, 36]. Yet we focus on edge recommendation rather than global structural results due to the following reason: in practice, graph edits can typically be made only in small amounts, either because of budget constraints, or because of practical considerations. Similar idea was previously used in [12]. We now give a mathematical formulation of Problem 1 in a formal way as follows.

Problem 1 *Given a connected graph $\mathcal{G} = (V, E)$, an integer k, a candidate edge set E_C consisting of nonexistent edges, find a subset $T \subset E_C$ with $|T| = k$, and add them to graph $\mathcal{G}$ forming a new graph $\mathcal{G} + T = (V, E \cup T)$, so that the polarization-disagreement index $\mathcal{I}(\mathcal{G} + T)$ is minimized. That is:*

$$\underset{T\subset E_C,\, |T|=k}{\text{minimize}} \quad \mathcal{I}(\mathcal{G} + T). \tag{4}$$

5 SPGREEDY: Simple Greedy Algorithm

In this section, we first study the characterization of our problem, and then propose a simple greedy algorithm, followed by some analysis in terms of the approximation guarantee.

Problem Characterization. We start with detailing the theoretic challenges of Problem 1. The complexity of Problem 1 comes along two dimensions—searching for the best edge subset delivering the maximum of the objective, and assessing the impact of a given subset of edges upon the objective. The former is inherently a combinatorial problem, which is computationally infeasible to solve in a naïve brute-force manner while the latter involves cubic-time matrix inversion. To be specific, for each augmented edge set T coming from $\binom{|E_C|}{k}$ possible subsets, we need to compute the P-D index of the resultant graph, yielding an exponential complexity $O\big(\binom{|E_C|}{k} \cdot n^3\big)$.

To tackle the exponential complexity, we resort to greedy heuristics. Firstly, the objective function is monotone respect to the edge set T, as we mentioned in Lemma 4.1. But, we show that, unlike in the case in other standard optimization settings, neither submodularity nor supermodularity holds for our objective function. We present above results by the following theorem.

Theorem 5.1 *The function $f(T)$ defined above is a non-submodular monotonically increasing function of the edge set T.*

Thanks to the empirical success of applying greedy strategy on a significantly larger class of non-submodular functions [5], in spite of the non-submodularity of our objective $f(T)$, the greedy algorithm can still become a proper choice for solving Problem 1, the details of which will be discussed in the next subsection.

Simple Greedy Algorithm. Our simple greedy algorithm, denoted as SPGREEDY, exploits the performance guarantee for non-submodular functions. The general approach is as follows. We will first assess candidate edges with respect to how much their addition to the network decreases the term $\mathcal{I}(\mathcal{G})$, and then iteratively add the most promising edges to the network until budget is reached.

Initially, the augmented edge set T is initialized with an empty set. Then k edges are iteratively selected to the augmented edge set from set $E_C \setminus T$. At each iteration of the greedy algorithm, the edge e in candidate set is chosen that gives the largest marginal gain $\mathcal{I}(T) - \mathcal{I}(T \cup \{e\})$, i.e. $f(e)$. The algorithm stops until k edges are selected to be added to T. A naïve greedy algorithm takes time $O(k|E_C|n^3)$, which is computationally intractable even for small-size networks. Actually, as shown in the proof of Lemma 4.1, with $(\boldsymbol{I} + \boldsymbol{L})^{-1}$ already computed, we can view the addition of a single edge e as a rank-1 update to the original matrix $(\boldsymbol{I} + \boldsymbol{L})^{-1}$, which can be calculated with a run-time of $O(n^2)$ by using Sherman-Morrison formula [35], instead of inverting the matrix again in $O(n^3)$ in each loop.

The above analysis leads to a simple greedy algorithm SPGREEDY$(\mathcal{G}, E_C, k, \boldsymbol{s})$, which is outlined in Algorithm 1. To begin with, this algorithm requires $O(n^3)$ time to compute the inverse of $\boldsymbol{I} + \boldsymbol{L}$, and then it performs in k rounds, with each round mainly including two steps: computing $f(e)$ (Line 4) in $O(|E_C|n^2)$ time, and updating $(\boldsymbol{I} + \boldsymbol{L})^{-1}$ (Line 8) in $O(n^2)$ time. Thus, the total running time of Algorithm 1 is $O(n^3 + k|E_C|n^2)$, which is much faster than the naïve algorithm.

Approximation Guarantee. As we mentioned before, the performance of the greedy algorithm for non-submodular function can be evaluated by its submodularity ratio γ and curvature α. Hence, we derive the bounds for these two quantities of our Problem 1.

Lemma 5.1 *The submodularity ratio γ of set function $f(T) = \mathcal{I}(G) - \mathcal{I}(G+T)$ is bounded by $1 > \gamma \geq \left(\frac{1+\lambda_1(\boldsymbol{L})}{1+\lambda_{n-1}(\boldsymbol{L}_{E_C})}\right)^2 > 0$, and its curvature α is bounded by $0 < \alpha \leq 1 - \left(\frac{1+\lambda_1(\boldsymbol{L})}{1+\lambda_{n-1}(\boldsymbol{L}_{E_C})}\right)^2 < 1$.*

Lemma 5.1, together with the approximation guarantee stated before, yields a performance analysis for the greedy algorithm. It should be noticed that although the bounds given in Lemma 5.1 are too weak to derive a satisfactory approximation, but the greedy algorithm has been shown to perform very close to the optimal solutions for our problem on the experimental aspect.

Algorithm 1: SPGREEDY$(\mathcal{G}, E_C, k, \boldsymbol{s})$

Input : A connected graph $\mathcal{G}$; a candidate edge set E_C; an integer $k \leq |E_C|$; an initial opinion vector $\boldsymbol{s}$

Output : A subset of $T \subset E_C$ and $|T| = k$

Initialize solution $T = \emptyset$
Compute $(\boldsymbol{I} + \boldsymbol{L})^{-1}$
for $i = 1$ *to* k **do**
  Compute $f(e)$ for each $e \in E_C \setminus T$
  Select e_i s.t. $e_i \leftarrow \arg\max_{e \in E_C \setminus T} f(e)$
  Update solution $T \leftarrow T \cup \{e_i\}$
  Update the graph $\mathcal{G} \leftarrow \mathcal{G}(V, E \cup \{e_i\})$
  Update $(\boldsymbol{I} + \boldsymbol{L})^{-1} \leftarrow (\boldsymbol{I} + \boldsymbol{L})^{-1} - \frac{\boldsymbol{\Omega} \boldsymbol{b}_e \boldsymbol{b}_e^\top \boldsymbol{\Omega}}{1 + \boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{b}_e}$
return T

6 FASTGREEDY: Fast Greedy Algorithm

Compared with the naïve algorithm, computation time of Algorithm 1 is significantly reduced. However, it is still computationally unacceptable when employed on large networks with millions of vertices. In this section, we address this challenge by presenting an efficient approximation algorithm, solving the problem in time $\widetilde{O}(mk\epsilon^{-2})$.

The core step for solving Problem 1 is to calculate the impact of each edge upon the objective i.e., $f(e)$. According to (3), to evaluate $f(e)$, we need to estimate two terms $\boldsymbol{s}^\top \boldsymbol{\Omega} \boldsymbol{b}_e \boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{s}$ and $\boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{b}_e$. We will show how to approximate these two quantities in nearly linear time using Johnson-Lindenstrauss Lemma [29, 2] and Fast SDDM Solvers[42]. Moreover, the results returned by our algorithm are demonstrated to provide proper approximations to $f(e)$.

Lemma 6.1 (JL Lemma [29]) *Given fixed vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n \in \mathbb{R}^d$ and a real number $\epsilon > 0$. Let p be a positive integer such that $p \geq 24 \log n / \epsilon^2$ and $\boldsymbol{R}_{p \times d}$ a random matrix with each entry being $1/\sqrt{p}$ or $-1/\sqrt{p}$ with identical probability. Then, with probability at least $1 - 1/n$, the following statement holds for any pair of i and j, $1 \leq i, j \leq n$:*

$$(1 - \epsilon)\|\boldsymbol{v}_i - \boldsymbol{v}_j\|^2 \leq \|\boldsymbol{R}\boldsymbol{v}_i - \boldsymbol{R}\boldsymbol{v}_j\|^2 \leq (1 + \epsilon)\|\boldsymbol{v}_i - \boldsymbol{v}_j\|^2.$$

Lemma 6.2 (Fast SDDM Solver [42]) *There is a nearly linear time solver $\boldsymbol{y} = \text{SOLVE}(\boldsymbol{S}, \boldsymbol{b}, \epsilon)$ which takes an SDDM matrix $\boldsymbol{S}_{n \times n}$ with m nonzero entries, a vector $\boldsymbol{b} \in \mathbb{R}^n$, and an error parameter $\delta > 0$, and returns a vector $\boldsymbol{x} \in \mathbb{R}^n$ satisfying $\|\boldsymbol{y} - \boldsymbol{S}^{-1}\boldsymbol{b}\|_{\boldsymbol{S}} \leq \delta \|\boldsymbol{S}^{-1}\boldsymbol{b}\|_{\boldsymbol{S}}$ with high probability, where $\|\boldsymbol{y}\|_{\boldsymbol{S}} \stackrel{\text{def}}{=} \sqrt{\boldsymbol{y}^\top \boldsymbol{S} \boldsymbol{y}}$. The solver runs in expected time $\widetilde{O}(m)$.*

We first approximate the denominator, which can be written as $\boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{b}_e = \boldsymbol{b}_e^\top \boldsymbol{\Omega}(\boldsymbol{I} + \boldsymbol{L})\boldsymbol{\Omega} \boldsymbol{b}_e = \boldsymbol{b}_e^\top \boldsymbol{\Omega} \left(\boldsymbol{I} + \boldsymbol{B}^\top \boldsymbol{B}\right) \boldsymbol{\Omega} \boldsymbol{b}_e = \|\boldsymbol{\Omega} \boldsymbol{b}_e\|^2 + \|\boldsymbol{B}\boldsymbol{\Omega} \boldsymbol{b}_e\|^2$. In this way, we have reduced the estimation of the denominator of (3) to the calculation of the quadratic form of $\|\boldsymbol{B}\boldsymbol{\Omega} \boldsymbol{b}_e\|^2$ and $\|\boldsymbol{\Omega} \boldsymbol{b}_e\|^2$ in $\mathbb{R}^m$ and $\mathbb{R}^n$. To reduce the computation cost, we will apply the JL lemma to reduce the dimensions. Let $\boldsymbol{Q}_{p \times m}$ and $\boldsymbol{P}_{p \times n}$ be two random $\pm 1/\sqrt{p}$ matrices where p will be decided later, then we can simply project matrices $\boldsymbol{B}\boldsymbol{\Omega}$ and $\boldsymbol{\Omega}$ onto low-dimensional subspace $\boldsymbol{Q}\boldsymbol{B}\boldsymbol{\Omega}$ and $\boldsymbol{P}\boldsymbol{\Omega}$. However, this still does not help to reduce the computation time, since direct computation of the above ℓ_2 norm involves matrix inversion, leading a run-time of $O(n^3)$. In order to avoid inverting matrix $\boldsymbol{I} + \boldsymbol{L}$, we will utilize the fast SDDM linear system solvers [42] to approximate the above two terms.

Let $\boldsymbol{X} = \boldsymbol{Q}\boldsymbol{B}, \overline{\boldsymbol{X}} = \boldsymbol{B}\boldsymbol{\Omega}, \boldsymbol{X}' = \boldsymbol{Q}\overline{\boldsymbol{X}}$, let $\tilde{\boldsymbol{X}}_i = \text{SOLVE}(\boldsymbol{I} + \boldsymbol{L}, \boldsymbol{X}_i, \delta_1)$, satisfying $\|\tilde{\boldsymbol{X}}_i - \boldsymbol{X}'_i\|_{\boldsymbol{I}+\boldsymbol{L}} \leq \delta_1 \|\boldsymbol{X}'_i\|_{\boldsymbol{I}+\boldsymbol{L}}$ with $\delta_1 \leq \frac{\epsilon\sqrt{2(1-\epsilon/12)}}{32n(n+1)\sqrt{(1+\epsilon/12)(n+1)}}$. Making use of the combination of Lemmas 6.1 and 6.2, the term $\|\boldsymbol{Q}\boldsymbol{B}\boldsymbol{\Omega} \boldsymbol{b}_e\|^2$ can be efficiently approximated as stated in the following lemma.

Lemma 6.3 *Given an undirected graph $\mathcal{G} = (V, E)$ with Laplacian matrix $\boldsymbol{L}$, a parameter $\epsilon \in (0, \frac{1}{2})$, then, the following relation holds:*

$$(1 - \frac{\epsilon}{12})^2 \|\overline{\boldsymbol{X}}\, \boldsymbol{b}_e\|^2 \leq \|\tilde{\boldsymbol{X}}\, \boldsymbol{b}_e\|^2 \leq (1 + \frac{\epsilon}{12})^2 \|\overline{\boldsymbol{X}}\, \boldsymbol{b}_e\|^2. \tag{5}$$

In a similar way, let $\boldsymbol{Y} = \boldsymbol{P}, \overline{\boldsymbol{Y}} = \boldsymbol{\Omega}, \boldsymbol{Y}' = \boldsymbol{P}\overline{\boldsymbol{Y}}$, let $\tilde{\boldsymbol{Y}}_i = \textsc{Solve}(\boldsymbol{I} + \boldsymbol{L}, \boldsymbol{Y}_i, \delta_2)$ with $\delta_2 \leq \frac{\epsilon\sqrt{2(1-\epsilon/12)}}{32(n+1)\sqrt{(1+\epsilon/12)(n+1)}}$. Then, the following lemma holds, giving an efficient approximation to $\|\boldsymbol{Q}\boldsymbol{\Omega}\boldsymbol{b}_e\|^2$.

Lemma 6.4 *Given an undirected graph $\mathcal{G} = (V, E)$ with Laplacian matrix $\boldsymbol{L}$, a parameter $\epsilon \in (0, \frac{1}{2})$, then, the following relation holds:*

$$(1 - \frac{\epsilon}{12})^2 \|\overline{\boldsymbol{Y}}\, \boldsymbol{b}_e\|^2 \leq \|\tilde{\boldsymbol{Y}}\, \boldsymbol{b}_e\| \leq (1 + \frac{\epsilon}{12})^2 \|\overline{\boldsymbol{Y}}\, \boldsymbol{b}_e\|^2. \tag{6}$$

Having Lemma 6.3, 6.4, the term $\boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{b}_e$ can be efficiently approximated by $\|\tilde{\boldsymbol{X}}\, \boldsymbol{b}_e\|^2 + \|\tilde{\boldsymbol{Y}}\, \boldsymbol{b}_e\|^2$ satisfying

$$(1 - \frac{\epsilon}{3}) \boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{b}_e \leq \|\tilde{\boldsymbol{X}}\, \boldsymbol{b}_e\|^2 + \|\tilde{\boldsymbol{Y}}\, \boldsymbol{b}_e\|^2 \leq (1 + \frac{\epsilon}{3}) \boldsymbol{b}_e^\top \boldsymbol{\Omega} \boldsymbol{b}_e.$$

We continue to approximate the numerator of $f(e)$, which can be written as $\|\boldsymbol{\Omega}\boldsymbol{s}\|^2_{\boldsymbol{b}_e \boldsymbol{b}_e^\top}$. We will use the fast SDDM matrix solver again to approximate this term.

Lemma 6.5 *Given an undirected graph $\mathcal{G} = (V, E)$ with Laplacian matrix $\boldsymbol{L}$ and forest matrix $\boldsymbol{\Omega}$, a parameter $\epsilon \in (0, \frac{1}{2})$, and the internal opinion vector $\boldsymbol{s}$, let $\boldsymbol{q} = \textsc{Solve}\,(\boldsymbol{I} + \boldsymbol{L}, \boldsymbol{s}, \delta_3)$, where $\delta_3 \leq \frac{\epsilon}{9n^2}$. Then, the following relation holds: $\left| \|\boldsymbol{q}\|^2_{\boldsymbol{b}_e \boldsymbol{b}_e^\top} - \|\boldsymbol{\Omega}\boldsymbol{s}\|^2_{\boldsymbol{b}_e \boldsymbol{b}_e^\top} \right| \leq \frac{\epsilon}{3}$.*

In the previous subsection, we have considerably approximated computation of each part of $f(e)$, and now we are able to propose an algorithm COMP approximating $f(e)$ for every edge e in the candidate set E_C based on Lemmas 6.3, 6.4 and 6.5. The outline of algorithm COMP is shown in Algorithm 2, whose performance is given in Theorem 6.1.

Theorem 6.1 *For $0 \leq \epsilon \leq 1/2$, the value $\hat{f}(e)$ returned by* COMP *satisfies $|f(e) - \hat{f}(e)| \leq \frac{4}{5}\epsilon$ with high probability.*

Algorithm 2: $\textsc{Comp}(\mathcal{G}, E_C, \boldsymbol{s}, \epsilon)$

Input : A graph $\mathcal{G}$; a candidate edge set E_C; an initial opinion vector $\boldsymbol{s}$; a real number $0 \leq \epsilon \leq 1/2$
Output : $\{(e, \hat{f}(e) | e \in E_C\}$

Set δ_1, δ_2, δ_3 according to Lemmas 6.3, 6.4 and 6.5,
$p \leftarrow \lceil 24 \log n / (\frac{\epsilon}{12})^2 \rceil$
Generate random Gaussian matrices $\boldsymbol{P}_{p \times n}$, $\boldsymbol{Q}_{p \times m}$
Compute approximations $\boldsymbol{q}$ to $\boldsymbol{\Omega}\boldsymbol{s}$, $\tilde{\boldsymbol{X}}$ to $\overline{\boldsymbol{X}} \stackrel{\text{def}}{=} \boldsymbol{B}\boldsymbol{\Omega}$, and $\tilde{\boldsymbol{Y}}$ to $\overline{\boldsymbol{Y}} \stackrel{\text{def}}{=} \boldsymbol{\Omega}$
for $i = 1$ *to* p **do**
  $\boldsymbol{q} \leftarrow \textsc{Solve}(\boldsymbol{I} + \boldsymbol{L}, \boldsymbol{s}, \delta_3)$
  $\tilde{\boldsymbol{X}}_i \leftarrow \textsc{Solve}(\boldsymbol{I} + \boldsymbol{L}, \boldsymbol{X}_i, \delta_1)$
  $\tilde{\boldsymbol{Y}}_i \leftarrow \textsc{Solve}(\boldsymbol{I} + \boldsymbol{L}, \boldsymbol{Y}_i, \delta_2)$
for *each* $e \in E_C$ **do**
  compute $\hat{f}(e) = \frac{\boldsymbol{b}_e^\top \boldsymbol{q}\boldsymbol{q}^\top \boldsymbol{b}_e}{1 + \|\tilde{\boldsymbol{X}}\boldsymbol{b}_e\|^2 + \|\tilde{\boldsymbol{Y}}\boldsymbol{b}_e\|^2}$
return $\{(e, \hat{f}(e) | e \in E_C\}$

Exploiting Algorithm 2 to approximate $f(e)$, we develop a fast greedy algorithm $\textsc{FastGreedy}(\mathcal{G}, E_C, \boldsymbol{s}, k, \epsilon)$ in Algorithm 3 to solve Problem 1. Overall, it follows the greedy strategy (Algorithm 1). In detail, Algorithm 3 performs k rounds (Lines 2-6). In each round, it takes time $\widetilde{O}(m\epsilon^{-2})$ to call COMP to compute the approximation of the marginal gain $\hat{f}(e)$, then iteratively select the element with the highest impact score and update. Therefore, the time complexity of Algorithm 3 is $\widetilde{O}(mk\epsilon^{-2})$.

Algorithm 3: FASTGREEDY$(\mathcal{G}, E_C, \boldsymbol{s}, k, \epsilon)$

Input : A graph $\mathcal{G}$; a candidate edge set E_C; an initial opinion vector $\boldsymbol{s}$; an integer $k \leq |E_C|$; a real number $0 \leq \epsilon \leq 1/2$

Output : T: a subset of E_C and $|T| = k$

Initialize solution $T = \emptyset$

for $i = 1$ *to* k **do**

  $\{e, \hat{f}(e) | e \in E_C \setminus T\} \leftarrow$ COMP$(\mathcal{G}, E_C \setminus T, \boldsymbol{s}, \epsilon)$

  Select e_i s.t. $e_i \leftarrow \arg\max_{e \in E_C \setminus T} \hat{f}(e)$

  Update solution $T \leftarrow T \cup \{e_i\}$

  Update the graph $\mathcal{G} \leftarrow \mathcal{G}(V, E \cup \{e_i\})$

return T

Table 1: Running time and results.

Network	n	m	Running Time (s)		$\Delta\mathcal{I}(\mathcal{G})$		Ratio
			SPGREEDY	FASTGREEDY	SPGREEDY	FASTGREEDY	
GrQc	4,158	13,422	7.83	11.13	-4.9966	-4.9774	0.9962
USgrid	4,941	6,594	10.31	10.70	-7.3406	-7.2720	0.9906
Erdos992	5,094	7,515	10.91	8.98	-6.1616	-6.0984	0.9897
Bcspwr10	5,300	8,271	12.03	11.57	-6.0290	-5.9820	0.9922
Reality	6,809	7,680	19.81	7.83	-5.7861	-5.7183	0.9883
PagesGovernment	7,057	89,429	21.10	27.27	-2.7580	-2.7484	0.9965
WikiElec	7,115	100,753	21.13	27.74	-4.7478	-4.7105	0.9921
Dmela	7,393	25,569	23.59	15.70	-4.9245	-4.8988	0.9948
HepPh	11,204	117,619	59.37	79.10	-4.2728	-4.2641	0.9980
Anybeat	12,645	49,132	81.86	44.41	-5.4685	-5.4180	0.9908
PagesCompany	14,113	52,126	106.40	62.88	-5.2711	-5.2554	0.9970
CondMat	21,363	91,286	319.86	101.95	-4.5719	-4.5659	0.9987
Gplus	23,628	39,194	416.24	42.99	-5.3146	-5.2672	0.9911
Brightkite$*$	56,739	212,945	—	274.97	—	—	—
WikiTalk$*$	92,117	360,767	—	397.34	—	—	—
Douban$*$	154,908	327,162	—	494.19	—	—	—
Citeseer$*$	227,320	814,134	—	1460.20	—	—	—
TwitterFollows$*$	404,719	713,319	—	1009.49	—	—	—
FourSquare$*$	639,014	3,214,986	—	3748.63	—	—	—
IMDB$*$	896,305	3,782,447	—	9076.97	—	—	—
YoutubeSnap$*$	1,134,890	2,987,624	—	6918.97	—	—	—

7 Experiments

In this section, we present numerical results to evaluate the performance of our two greedy algorithms SPGREEDY and FASTGREEDY. To this end, extensive experiments are designed and executed on real networks of various types and scales to validate the effectiveness and efficiency of our algorithms.

Datasets. The studied realistic networks are representative selected from various domains, details on the statistics of the dataset can be found at the supplementary material.

Machine and reproducibility. All experiments were conducted on a machine equipped with 32G RAM and 4.2 GHz Intel i7-7700 CPU. Our code can be found at the supplementary material.

Methods. The sets of k edges are found with the following seven strategies. (1) *Optimum*: selecting edge set $|T| = k$ with minimum $\mathcal{I}(\mathcal{G} + T)$ by exhaustive search. (2) *Random*: selecting k edges at random. (3) *Betweenness*: selecting top-k edges with the highest betweenness [7]. (4) *DegProduct*: selecting top-k edges with the largest product of two end-points degrees. (5) *DegSum*: selecting top-k edges with the largest sum of two end-points degrees. (6)*FastGreedy*: selecting top-k edges with maximum $f(e)$ returned by algorithm FASTGREEDY. (7) *SpGreedy*: selecting top-k edges with maximum $f(e)$ returned by algorithm SPGREEDY.

Opinions and evaluation metrics. For the initial opinions vector $\boldsymbol{s}$, each element $\boldsymbol{s}_i$ is uniformly distributed in the range of $[0, 1]$. The performance of all above-mentioned methods is evaluated by the impact of their selected edges on the drop of the P-D index $\Delta\mathcal{I}(\mathcal{G}) = \mathcal{I}(\mathcal{G} + T) - \mathcal{I}(\mathcal{G})$, with a smaller $\Delta\mathcal{I}(\mathcal{G})$ corresponding to an more effective method.

Results. We first evaluate the effectiveness of our algorithms, we execute experiments on four small realistic networks: Karate, Dolphins, Netscience and Diseasom. These networks are small, allowing us to compute the optimal set of edges. For each case, we add $k = 1, 2, \ldots, 8$ edges to augment the

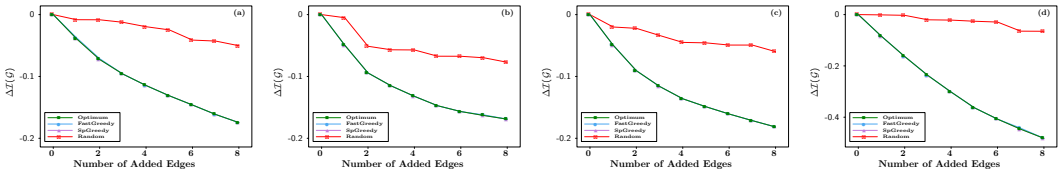

Figure 1: Optimization results of four methods for edge addition on datasets (a) Karate, (b) Dolphins, (c) Netscience and (d) Diseasom for varying k.

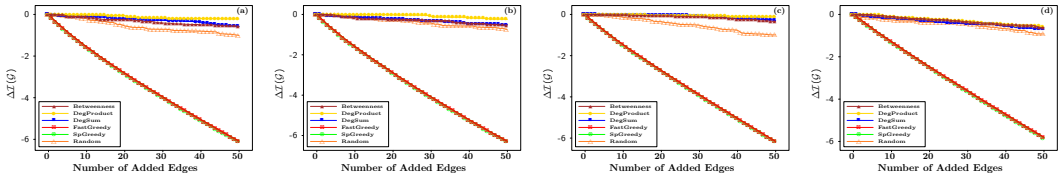

Figure 2: Optimization results for six methods of edge addition on datasets: (a) Yeast, (b) GridWorm, (c) Erdos992 and (d) Reality for varying k.

network from $|E_C| = 30$ candidate edges. For the approximation algorithm FASTGREEDY, we set $\epsilon = 0.3$. Note that a smaller ϵ corresponds to a stronger approximation ratio but less efficiency. As shown in Figures 1, the solutions returned by our two greedy algorithms and the optimum solution are almost the same so that the three curves are overlapped, demonstrating that our greedy algorithms perform much better than the theoretical guarantee. In addition, both of our algorithms are much better than those returned by the random scheme.

To further show the effectiveness of our algorithms, we proceed to compare the results of our methods with the baseline schemes on four larger real-world networks: Yeast, GridWorm, Erdos992 and Reality, which are too large to get the optimal solutions by brute-force search. For each network, the performance of different methods for varying k and $|E_C| = 10000$ candidate edges are displayed in Figures 2. We can see that SPGREEDY achieves the best performance as expected, and the proposed FASTGREEDY (a): is very close to the SPGREEDY method, and (b): consistently outperforms all four alternative methods.

In Table 1 we provide the results of running time and the drop of the P-D index $\Delta\mathcal{I}(\mathcal{G})$ returned by our two greedy algorithms. We observe that FASTGREEDY is significantly faster than SPGREEDY, especially for large networks, while both algorithms almost yield the same value on $\Delta\mathcal{I}(\mathcal{G})$. It is worth noting that SPGREEDY is not applicable to the last eight networks marked with "$*$" due to the limitations of time and memory. In comparison with SPGREEDY, FASTGREEDY approximately computes $\Delta\mathcal{I}(\mathcal{G})$ within several hours. Therefore, our algorithm FASTGREEDY achieves remarkable improvement in efficiency and is scalable to large networks with more than 10^6 nodes.

8 Conclusions

In this paper, we considered the problem of minimizing the disagreement polarization index $\mathcal{I}(\mathcal{G})$ by strategically recommending k links. This problem belongs to the class of discrete optimization that has found vast applications in various domains. We showed the objective function is monotone but non-submodular. We then presented two greedy algorithms to solve the optimization problem: the former returns a constant-factor approximation of the optimum solutions in time $O(n^3)$, while the latter runs in time $\widetilde{O}(mk\epsilon^{-2})$. On the theoretic side, we provided analysis of the approximation guarantee for these two algorithms. On the experimental aspect, we performed extensive experiments on real-life networks, which demonstrate that both of our algorithms lead to almost optimal solutions, and consistently outperform several alternative baseline heuristics. Particularly, our second algorithm could yield a good approximation solution quickly on networks with more than one million nodes within 2 hours, demonstrating excellent scalablity to massive networks.

References

[1] Rediet Abebe, Jon Kleinberg, David Parkes, and Charalampos E Tsourakakis. Opinion dynamics with varying susceptibility to persuasion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1089–1098. ACM, 2018.

[2] Dimitris Achlioptas. Database-friendly random projections. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 274–281. ACM, 2001.

[3] Victor Amelkin and Ambuj K Singh. Fighting opinion control in social networks via link recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 677–685, 2019.

[4] Brian DO Anderson and Mengbin Ye. Recent advances in the modelling and analysis of opinion dynamics on influence networks. *International Journal of Automation and Computing*, 16(2):129–149, 2019.

[5] Andrew An Bian, Joachim M Buhmann, Andreas Krause, and Sebastian Tschiatschek. Guarantees for greedy maximization of non-submodular functions with applications. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 498–507. JMLR. org, 2017.

[6] David Bindel, Jon Kleinberg, and Sigal Oren. How bad is forming your own opinion? *Games and Economic Behavior*, 92:248–265, 2015.

[7] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[8] TH Chan, Zhibin Liang, and Mauro Sozio. Revisiting opinion dynamics with varying susceptibility to persuasion via non-convex local search. In *The World Wide Web Conference*, pages 173–183. ACM, 2019.

[9] P. Yu Chebotarev and E. V. Shamis. The matrix-forest theorem and measuring relations in small social groups. *Automation and Remote Control*, 58(9):1505–1514, 1997.

[10] P. Yu Chebotarev and E. V. Shamis. On proximity measures for graph vertices. *Automation and Remote Control*, 59(10):1443–1459, 1998.

[11] Pavel Chebotarev. Spanning forests and the golden ratio. *Discrete Appl. Math.*, 156(5):813–821, 2008.

[12] Xi Chen, Jefrey Lijffijt, and Tijl De Bie. Quantifying and minimizing risk of conflict in social networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1197–1205. ACM, 2018.

[13] Uthsav Chitra and Christopher Musco. Analyzing the impact of filter bubbles on social network polarization. In *Proceedings of the Thirteenth ACM International Conference on Web Search and Data Mining*, pages 115–123. ACM, 2020.

[14] Pierluigi Crescenzi, Gianlorenzo D'angelo, Lorenzo Severini, and Yllka Velaj. Greedily improving our own closeness centrality in a network. *ACM Transactions on Knowledge Discovery from Data*, 11(1):9, 2016.

[15] Pranav Dandekar, Ashish Goel, and David T Lee. Biased assimilation, homophily, and the dynamics of polarization. *Proceedings of the National Academy of Sciences*, 110(15):5791–5796, 2013.

[16] Gianlorenzo D'Angelo, Martin Olsen, and Lorenzo Severini. Coverage centrality maximization in undirected networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, AAAI '19, pages 501–508, 2019.

[17] Abhimanyu Das, Sreenivas Gollapudi, Rina Panigrahy, and Mahyar Salek. Debiasing social wisdom. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 500–508. ACM, 2013.

[18] Yucheng Dong, Min Zhan, Gang Kou, Zhaogang Ding, and Haiming Liang. A survey on the fusion process in opinion dynamics. *Information Fusion*, 43:57–65, 2018.

[19] Noah E Friedkin and Eugene C Johnsen. Social influence and opinions. *Journal of Mathematical Sociology*, 15(3-4):193–206, 1990.

[20] J. Gaitonde, J. Kleinberg, and Éva Tardos. Adversarial perturbations of opinion dynamics in networks. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 471–472, 2020.

[21] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. Reducing controversy by connecting opposing views. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 81–90. ACM, 2017.

[22] Kiran Garimella, Aristides Gionis, Nikos Parotsidis, and Nikolaj Tatti. Balancing information exposure in social networks. In *Advances in Neural Information Processing Systems*.

[23] Javad Ghaderi and Rayadurgam Srikant. Opinion dynamics in social networks with stubborn agents: Equilibrium and convergence rate. *Automatica*, 50(12):3209–3215, 2014.

[24] Aristides Gionis, Evimaria Terzi, and Panayiotis Tsaparas. Opinion maximization in social networks. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 387–395. SIAM, 2013.

[25] VE Golender, VV Drboglav, and AB Rosenblit. Graph potentials method and its application for chemical information processing. *Journal of Chemical Information and Computer Sciences*, 21(4):196–204, 1981.

[26] Shahrzad Haddadan, Cristina Menghini, Matteo Riondato, and Eli Upfal. Repbublik: Reducing polarized bubble radius with link insertions. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, Mar 2021.

[27] Guang He, Wenbing Zhang, Jing Liu, and Haoyue Ruan. Opinion dynamics with the increasing peer pressure and prejudice on the signed graph. *Nonlinear Dynamics*, 99:1–13, 2020.

[28] Peng Jia, Anahita MirTabatabaei, Noah E Friedkin, and Francesco Bullo. Opinion dynamics and the evolution of social power in influence networks. *SIAM Review*, 57(3):367–397, 2015.

[29] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[30] Heidi Ledford. How facebook, twitter and other data troves are revolutionizing social science. *Nature*, 582(7812):328–330, 2020.

[31] Huan Li and Aaron Schild. Spectral subspace sparsification. In *Proceedings of 2018 IEEE 59th Annual Symposium on Foundations of Computer Science*, pages 385–396, 2018.

[32] Erika Mackin and Stacy Patterson. Maximizing diversity of opinion in social networks. In *2019 American Control Conference*, pages 2728–2734. IEEE, 2019.

[33] Antonis Matakos, Evimaria Terzi, and Panayiotis Tsaparas. Measuring and moderating opinion polarization in social networks. *Data Mining and Knowledge Discovery*, 31(5):1480–1505, 2017.

[34] Antonis Matakos, Sijing Tu, and Aristides Gionis. Tell me something my friends do not know: Diversity maximization in social networks. *Knowledge and Information Systems*, 62(9):3697–3726, 2020.

[35] Carl D Meyer, Jr. Generalized inversion of modified matrices. *SIAM Journal on Applied Mathematics*, 24(3):315–323, 1973.

[36] Cameron Musco, Christopher Musco, and Charalampos E Tsourakakis. Minimizing polarization and disagreement in social networks. In *Proceedings of the 2018 World Wide Web Conference*, pages 369–378. International World Wide Web Conferences Steering Committee, 2018.

[37] Anton V Proskurnikov and Roberto Tempo. A tutorial on modeling and analysis of dynamic social networks. part i. *Annual Reviews in Control*, 43:65–79, 2017.

[38] Chiara Ravazzi, Paolo Frasca, Roberto Tempo, and Hideaki Ishii. Ergodic randomized algorithms and dynamics over networks. *IEEE Trans. Control Netw. Syst.*, 1(2):78–87, 2015.

[39] Justin Semonsen, Christopher Griffin, Anna Squicciarini, and Sarah Rajtmajer. Opinion dynamics in the presence of increasing agreement pressure. *IEEE Trans. Cybern.*, 49(4):1270–1278, 2019.

[40] Liren Shan, Yuhao Yi, and Zhongzhi Zhang. Improving information centrality of a node in complex networks by adding edges. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI '18, pages 3535–3541. AAAI Press, 2018.

[41] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal of Computing*, 40(6):1913–1926, 2011.

[42] Daniel A Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM J. Matrix Anal. Appl.*, 35(3):835–885, 2014.

[43] Sijing Tu, Cigdem Aslay, and Aristides Gionis. Co-exposure maximization in online social networks. In *Advances in Neural Information Processing Systems*.

[44] Pinghua Xu, Wenbin Hu, Jia Wu, and Weiwei Liu. Opinion maximization in social trust networks. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 1251–1257, 2020.

[45] Wanyue Xu, Qi Bao, and Zhongzhi Zhang. Fast evaluation for relevant quantities of opinion dynamics. *arXiv preprint arXiv:2101.08143*, 2021.

Checklist

- Did you include the license to the code and datasets? [Yes] See Section 7.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [No]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [No]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [No]