Knowledge and Information Systems (2020) 62:3697–3726
https://doi.org/10.1007/s10115-020-01456-1

REGULAR PAPER

# Tell me something my friends do not know: diversity maximization in social networks

Antonis Matakos[1] · Sijing Tu[1] · Aristides Gionis[1,2]

Received: 7 January 2019 / Revised: 22 February 2020 / Accepted: 22 February 2020 /
Published online: 23 April 2020

**Abstract**
Social media have a great potential to improve information dissemination in our society, yet they have been held accountable for a number of undesirable effects, such as polarization and filter bubbles. It is thus important to understand these negative phenomena and develop methods to combat them. In this paper, we propose a novel approach to address the problem of breaking filter bubbles in social media. We do so by aiming to maximize the diversity of the information exposed to connected social-media users. We formulate the problem of maximizing the diversity of exposure as a quadratic-knapsack problem. We show that the proposed diversity-maximization problem is inapproximable, and thus, we resort to polynomial nonapproximable algorithms, inspired by solutions developed for the quadratic-knapsack problem, as well as scalable greedy heuristics. We complement our algorithms with instance-specific upper bounds, which are used to provide empirical approximation guarantees for the given problem instances. Our experimental evaluation shows that a proposed greedy algorithm followed by randomized local search is the algorithm of choice given its quality-vs.-efficiency trade-off.

**Keywords** Diversity maximization · Filter bubble · Quadratic knapsack · Combinatorial optimization · Greedy algorithms

## 1 Introduction

Social media play a critical role in today's information society, not only by connecting people with their friends, but also as a means of news dissemination. A recent survey estimates that six out of ten adults in the USA get their news on social media [18]. Although initially it appeared that social media can contribute to the democratization of content generation and distribution, most recently, a series of negative effects and undesirable phenomena have emerged, such as *filter bubbles*, *polarization*, *fake news*, and more. An indication for the

✉ Antonis Matakos
antonis.matakos@aalto.fi

[1] Department of Computer Science, Aalto University, Espoo, Finland

[2] KTH Royal Institute of Technology, Stockholm, Sweden

extent of the problem can be seen by Facebook's own press release that *social media can have the unintended consequence of corroding democracy*.[1]

Given these negative effects, a recent body of research has focused on solving different aspects of the problem. Recent work in the area includes methods for detecting polarization [2,5,7,13,15,19], proposals of reducing polarization of opinions by network-engineering approaches [27,29], as well as recommending users to follow and content to bridge opposing views [14,23,28], and balancing information exposure in social networks [12].

One particular aspect of the problem is the emergence of *filter bubbles* in social media [31], where individuals are connected to like-minded net citizens or information sources of similar disposition, and as a result, not being exposed to information they disagree with, and effectively being isolated in their own cultural and ideological bubbles. By considering that a filter bubble may be attributed to the *lack of diversity* in the exposure of information, a desirable objective is to maximize the total diversity of the information that users receive in the social network. In this paper, we aim at achieving this objective by formulating a novel measure of diversity in a social network, which we call *diversity index*, and proposing efficient algorithms to maximize this measure.

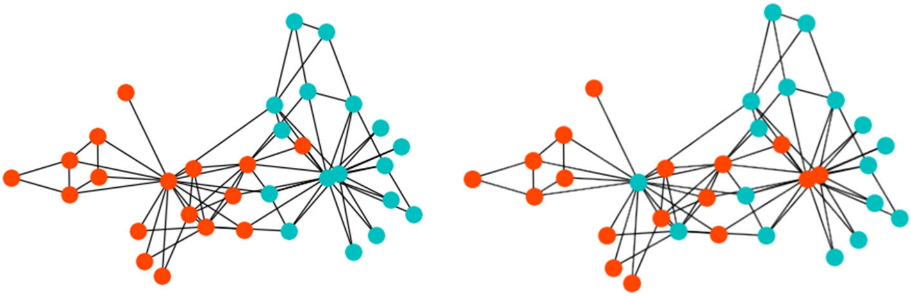
To justify our approach, we make a few assumptions. First, we assume that the operators of the social-network platform gather user data, which can be used to build user models and infer user leanings with respect to different topics. Second, we assume that it is possible to maximize the diversity of content consumed by a given user via injecting diverse content in the feed of the user. Furthermore, an important aspect of our approach is that diversity can also be increased if users have contacts with different leanings than theirs. Finally, we assume that it is best not to interfere with the organic operation of the network and make only a small number of recommendations that aim to maximize diversity.

Therefore, in addition to content that circulates in the network organically via shares and re-posts among users, the social-media platform may consider to strategically recommend suitably chosen content to selected individuals in order to increase diversity. Note that in order to increase transparency, the social-media platform may need to distinctly mark, or display in a specific format, such recommendations so that users are aware of its presence and objective. Additionally, users may need to opt in to receive such recommendations. We consider, however, that such design considerations are orthogonal to our study and beyond the scope of this paper.

A desirable property for such a mechanism of diversity-enabling recommendations is that the social-media platform should make a *small amount of recommendations*, as these can be perceived as interventions to the organic operation of the network. Thus, a natural question to ask is: *"given a fixed amount of content-recommendation activity, which users should we target and what recommendations to make, so as to maximally increase diversity?"* Intuitively, we would like to target users who are part of filter bubbles (their friends and themselves are exposed to similar content), are influential among their connections, and are likely to take into account the recommended content. The importance of leveraging the structure of social connections and their strength in order to break filter bubbles is highlighted by recent empirical studies,[2] which found that social trust has a higher impact on the likelihood of consuming a news story than the source of the story itself.

[1] http://money.cnn.com/2018/01/22/technology/facebook-democracy-social-media/index.html, January 22, 2018.

[2] https://www.americanpressinstitute.org/publications/reports/survey-research/trust-social-media/, March 20, 2017.



**(a)** Echo-chamber graph **(b)** Graph with diversified exposure

**Fig. 1** Toy graph with different exposure assignments

Our problem setting puts together all these components. To formulate a well-defined problem, we make additional modeling assumptions. We assume that we know the structure of the social graph and the influence that users exert on each other, expressed as edge weights on the social graph. We also assume that we can quantify the information exposure of each individual as a number in a prespecified range (we use $[-1,1]$), e.g., expressing the grade of the content consumed by an individual in a continuous spectrum, for example, *conservative-liberal*. Finally, we assume that we can estimate the degree to which an individual is likely to take into account a specific recommendation, e.g., the probability to re-post the recommended item. The problems of estimating these parameters, as well as the problem of making appropriate diversity-enabling recommendations for a given user, are beyond the scope of this paper and are left for future work. However, we believe that they are meaningful and realistic problems, which can be solved with good accuracy, and thus, supporting our overall approach.

***Example*** A toy example demonstrating our concept is shown in Fig. 1, using the karate club network, which is known to contain two communities. The colors on the nodes represent different exposure levels, say two different “news diets” that the network users consume. In Fig. 1a, each community has different exposure levels, leading to a network with echo chambers and no diversity. In Fig. 1b, we depict the optimal solution to our problem, where we ask for the best $k = 4$ users, to change their exposure and maximize the total network diversity—assuming that all users opt in to receive alternative news diets and the user cost is constant. In this simple example, the algorithm picks the two hubs of each community. □

From the technical point of view, we first define the *diversity index*, which is a global measure of diversity in a social network, reflecting the structure of the network and the exposure of users from different viewpoints. Then, we formulate the problem of maximizing diversity of exposure as the problem of maximizing the diversity index. We show that the diversity-maximization problem we define is a special case of the quadratic-knapsack problem (QKP) [10]. The proposed problem formulation combines our modeling assumptions in a concise way, while resulting in an interesting algorithmic problem through the novel connection to quadratic knapsack.

Our first result shows that the diversity-maximization problem is not only **NP**-hard, but also **NP**-hard to approximate within a multiplicative factor. Thus, we study a number of polynomial algorithms inspired by the quadratic-knapsack formulation, such as methods based on semidefinite-programming (SDP) relaxation and linearization techniques. We also propose two scalable greedy algorithms, which take advantage of the special structure of our problem.

Our results show that the SDP-based algorithm is the best performing on a diverse range of settings, followed very closely by one of the greedy methods. This is very useful because while the SDP algorithm is expensive, the greedy has linear complexity with respect to the number of nodes in the network, and thus, has excellent scalability properties.

Our relaxation provides upper bounds on the quality of solution. In addition, we propose alternative upper bounds with varying trade-offs of tightness-vs.-efficiency. All these bounds allow us to obtain empirical approximation guarantees for given problem instances. For instance, for the problem instances used in our experiments, we are able to assert that our algorithms give solutions with typical approximation factor between 1.5 and 2.5, despite the problem being **NP**-hard to approximate.

In summary, in this paper we make the following contributions:

- Inspired by the problem of breaking filter bubbles, we formulate the problem of maximizing the diversity of exposure, as a special case of the quadratic-knapsack problem.
- We prove that the diversity maximization problem is **NP**-hard to approximate within a multiplicative factor.
- We study several algorithms for the problem, including an SDP-based algorithm, an algorithm based on linearization, and two greedy methods.
- We develop upper bounds with different trade-offs of tightness-vs.-efficiency, which provide empirical approximation guarantees for given problem instances.
- We present an extensive experimental evaluation that provides evidence for the best performing methods and quality-vs.-efficiency trade-offs.

The rest of the paper is organized as follows. We start our presentation by reviewing the related work in Sect. 2. We then present our notation in Sect. 3, and we formally define the diversity-maximization problem in Sect. 4. The **NP**-hardness proof is also presented in Sect. 4. In Sect. 5, we discuss algorithms for the binary version of the problem, and we present upper bounds for the optimal solution. The extension of the diversity-maximization problem to the continuous case is discussed in Sect. 6. We present our experimental evaluation in Sect. 7, and we conclude in Sect. 8 by offering our final remarks and suggestions for future work.

An earlier version of this paper appeared in the ICDM 2018 conference [26]. In this submission, we have extended the conference version by studying continuous formulations of the diversity-maximization problem (Sect. 6) and conducting empirical evaluation of those continuous extensions (Sect. 7).

## 2 Related work

Our work relates to the emerging line of work on breaking filter bubbles and reducing polarization on social media. To the best of our knowledge, this is the first work to approach this problem from the point of view of increasing diversity of information exposure and formulating it as a quadratic-knapsack-style problem.

*Detecting polarization* Recently, a significant body of work has emerged that focuses on measures for characterizing polarization in online social media [2,7,13,19,27]. These works consider mainly the structure in social-media interactions and quantify polarization or compute node polarity scores using network-based techniques. Other papers study the emergence of polarization on various opinion-formation models: Dandekar et al. [8] generalize DeGroot's model to account for *biased assimilation*, while Vicario et al. [33] propose a variant of the *bounded-confidence model*, where discordant edges are rewired and two opposing opinion clusters emerge.

*Reducing polarization* Given the negative effects of fragmentation, there has been recent work that focuses on methods for reducing polarization [14,27,29]. Matakos et al. [27] study the problem of convincing a set of individuals to adopt a neutral opinion and act as mediators in the discussion. Musco et al. [29] study a similar problem, albeit with the dual objective of minimizing both polarization and disagreement among individuals. Garimella et al. [14] consider the problem of introducing new edges between the two sides of a controversy, so as to reduce polarization.

There are two key differences between these works and our approach. First, while these works focus on minimizing other measures of polarization, our aim is to maximize the diversity of content to which an individual is exposed. Second, while these works consider how to affect user opinions, we only consider the *exposure* of a user. We consider our setting more realistic since in practice it is difficult to know the user opinions.

A complementary line of work studies mechanisms that expose social-media users to content that is not aligned with their prior beliefs [23,28,34]. While these works focus on *how* to present information to users, addressing issues of interface and incentives, our work addresses the question of *who* to approach with the new information.

*Quadratic knapsack* Our formulation maps the diversity-maximization problem to a special case of the quadratic-knapsack problem (QKP). The general form of QKP was introduced by Gallo et al. [10]. A classical technique to solve 0–1 quadratic problems is to *linearize* them by introducing auxiliary variables and transforming the problem to an integer linear program (ILP) formulation. Glover et al. [16] presented a concise way to rewrite a 0–1 quadratic problem as an equivalent 0–1 linear program with only $n$ auxiliary variables and $4n$ constraints.

Of particular interest are *semidefinite-programming* (SDP) techniques, as they can yield tighter relaxations than using linearization, albeit at the expense of higher running time. Therefore, central to our work is the methodology of Helmberg et al. [20], who approach QKP by introducing a series of SDP relaxations of increasing tightness. Additionally, to strengthen the formulation, a number of inequalities defining the QKP polyhedron (called the Boolean quadric polytope) have been studied [30]. On a high level, our problem is also related to the MAX- CUT problem, which has been shown to admit an approximation ratio of 0.878 [17], using semidefinite programming. However, as we prove shortly, our problem is harder as it admits no polynomial approximation guarantee.

## 3 Diversity index

Consider a social network represented as a graph $G = (V, E, w)$, where the node set $V$ represents a set of individuals, the edge set $E$ represents social connections, and $w : E \rightarrow \mathbb{R}_+$ is a weight function that represents the strength of the social connections. The weight of an edge $(i, j) \in E$ is denoted by $w_{ij}$. The number of nodes and the number of edges are denoted by $n$ and $m$, respectively.

We write $\mathbf{A}$ to denote the adjacency matrix of the graph $G$, whose entry $A_{ij}$ is equal to $w_{ij}$ if $(i, j) \in E$, and 0 otherwise. We assume that the graph $G$ is undirected, and so $\mathbf{A}$ is symmetric. We also denote by $\mathbf{D}$ the diagonal matrix whose $i$th diagonal entry is equal to the weighted degree of individual $i$, i.e., $D_{ii} = \sum_j w_{ij}$. The *Laplacian matrix* of the graph $G$ is $\mathbf{L} = \mathbf{D} - \mathbf{A}$.

We assume that each individual $i \in V$ has an overall *exposure* to content represented by a value $s_i$. The value $s_i$ may represent the leaning of the content individual $i$ is exposed to on an issue, as measured by endorsed or shared news articles. Without loss of generality, we assume that $s_i \in [-1, 1]$. We also consider the discrete case where $s_i \in \{-1, 1\}$. The vector of exposure for all the individuals in the graph is denoted by $\mathbf{s} = [s_i]_{i \in V}$.

Given a vector of exposure $\mathbf{s}$ for the individuals of a network, we are interested in measuring the *network diversity* with respect to $\mathbf{s}$. Intuitively, high diversity should indicate that many individuals tend to have different exposures than those of their social connections. Furthermore, diversity should account for the strength of social connections. These considerations motivate our definition of the *diversity index* in a social network.

**Definition 3.1** (*Diversity index*) Given a graph $G = (V, E, w)$ and a vector of exposure $\mathbf{s} \in [-1, 1]^n$ for the individuals in $V$, the *diversity index* $\eta(G, \mathbf{s})$ of the graph $G$ with respect to the exposure vector $\mathbf{s}$ is defined as:

$$\eta(G, \mathbf{s}) = \sum_{(i,j) \in E} w_{ij} \left(s_i - s_j\right)^2. \tag{1}$$

An equivalent way of writing Eq. (1), using the laplacian $\mathbf{L}$ of the graph $G$ is:

$$\eta(G, \mathbf{s}) = \sum_{(i,j) \in E} w_{ij} \left(s_i - s_j\right)^2 = \mathbf{s}^\top \mathbf{D}\, \mathbf{s} - \mathbf{s}^\top \mathbf{A}\, \mathbf{s} = \mathbf{s}^\top \mathbf{L}\, \mathbf{s}. \tag{2}$$

Higher value for the diversity index indicates more diverse networks.

In the rest of the paper, given a vector $\mathbf{x}$ we write Diag($\mathbf{x}$) to denote the *matrix* with $\mathbf{x}$ as its diagonal and given a matrix $\mathbf{X}$, we write Diag($\mathbf{X}$) to denote the *vector* corresponding to the diagonal of $\mathbf{X}$. As it is common, we denote by $Tr(\mathbf{X})$ the *trace* of a matrix $\mathbf{X}$. Finally, we write $\mathbf{X} \succeq 0$ to denote that $\mathbf{X}$ is a positive semidefinite matrix.

## 4 Problem formulation

In this manuscript, we will first focus on the discrete case where exposure $s_i$ is either $-1$ or 1. This corresponds to the case where discussions are characterized by two dominant and opposing perspectives, which, exacerbated by filter bubbles, often leads to polarization and lack of diversity of exposure: Some examples are the fragmentation into liberals vs. conservatives, brexit vs bremain, right wing vs left wing. Additionally, this assumption leads to a more attractive formulation, while being at least as challenging computationally. We also investigate the continuous case but provide it more as an extension.

Our goal is to maximize the *diversity index* $\eta(G, \mathbf{s})$ of a graph $G$, assuming that we know the current exposure vector $\mathbf{s}$. We consider maximizing the diversity index by selecting individuals and "flipping" the leaning of their exposure (from $-1$ to 1, or vice versa), under a budget constraint.

As mentioned in Introduction, changing the exposure of a individual corresponds to recommendations that an individual possibly opts in. The issue of interface and communication with the individuals is of independent interest, and we consider it orthogonal to our work.

Given an exposure vector $\mathbf{s}$, after changing the exposure of $k$ individuals, the new exposure vector can be written as $\mathbf{y} = \mathbf{s} - \mathbf{s}'$, where $\mathbf{s}'$ is a sparse vector with at most $k$ nonzero elements. In particular, $s'_i = -2$ if $s_i = -1$ and $s'_i = 2$ if $s_i = 1$. Alternatively, we can write the new exposure vector as $\mathbf{y} = \mathbf{s} - 2\,\mathrm{Diag}(\mathbf{s})\,\mathbf{x}$, where $\mathbf{x} \in \{0, 1\}^n$ is an indicator vector with $x_i = 1$ if the exposure of the $i$th individual has changed and $x_i = 0$ otherwise. This formulation highlights the nature of the problem as a variable-selection problem.

We consider a knapsack-type constraint for $\mathbf{x}$ with a weight vector $\mathbf{b}$, where $b_i$ expresses the *cost* in altering the exposure of individual $i$ (for example, some individuals may have a strong predisposition toward certain issues). We define the following problem:

**Problem 1** *Given a graph $G = (V, E)$, an exposure vector $\mathbf{s}$, a node weight vector $\mathbf{b}$, a budget $k$, we ask to find a binary vector $\mathbf{x}$, such that the knapsack constraint $\mathbf{b}^\top \mathbf{x} \leq k$ is satisfied and the resulting diversity index $\eta(G, \mathbf{y})$ is maximized, where $\mathbf{y} = \mathbf{s} - 2\,\mathrm{Diag}(\mathbf{s})\,\mathbf{x}$.*

We now formulate the corresponding optimization problem. Using the definition of the diversity index from Eq. (2), we have

$$\begin{aligned}\mathbf{y}^\top \mathbf{L}\,\mathbf{y} &= (\mathbf{s} - 2\,\mathrm{Diag}(\mathbf{s})\,\mathbf{x})^\top\,\mathbf{L}\,(\mathbf{s} - 2\,\mathrm{Diag}(\mathbf{s})\,\mathbf{x})\\ &= \mathbf{s}^\top \mathbf{L}\,\mathbf{s} + 4\,\mathbf{x}^\top\,\mathrm{Diag}(\mathbf{s})\,\mathbf{L}\,\mathrm{Diag}(\mathbf{s})\,\mathbf{x}\\ &\quad -4\,\mathbf{s}^\top \mathbf{L}\,\mathrm{Diag}(\mathbf{s})\,\mathbf{x}.\end{aligned}$$

Recall that we want to maximize this quantity. The term $\mathbf{s}^\top \mathbf{L}\,\mathbf{s}$ is a constant and has no influence on the maximization and, therefore, can be removed from the objective. For convenience, we define

$$\begin{aligned}&\mathbf{Q} = \mathrm{Diag}(\mathbf{s})\,\mathbf{L}\,\mathrm{Diag}(\mathbf{s}),\ \ \mathbf{q} = \mathbf{s}^\top \mathbf{L}\,\mathrm{Diag}(\mathbf{s}),\ \text{ and}\\ &\mathbf{P} = \mathbf{Q} - \mathrm{Diag}(\mathbf{q}).\end{aligned} \tag{3}$$

Note that $\mathbf{P}$ is constant and does not depend on the optimization variable $\mathbf{x}$. We take advantage of the fact that $\mathbf{x} = \mathbf{x}^2$ to write our program in purely quadratic form. We define the following quadratic binary knapsack (QBK) problem:

$$\begin{array}{ll}\max & \mathbf{x}^\top \mathbf{P}\,\mathbf{x}\\ \text{subject to} & \mathbf{b}^\top \mathbf{x} \leq k, \quad \mathbf{b} \in \mathbb{R}^n\\ & \mathbf{x} \in \{0, 1\}^n\,.\end{array} \tag{QBK}$$

It is not difficult to see that QBK is **NP**-hard, by a simple transformation from MAX- CUT. It suffices to set $b_i = 0$, and the problem reduces to:

$$\begin{array}{ll}\max & \mathbf{y}^\top \mathbf{L}\,\mathbf{y}\\ \text{subject to} & \mathbf{y} \in \{-1, 1\}^n\,,\end{array}$$

which is equivalent to a general instance of MAX- CUT. Despite this similarity, our problem is harder than MAX- CUT. This is because MAX- CUT admits an approximation algorithm with ratio 0.878 [17], while it can be shown that QBK is *inapproximable*.

**Proposition 1** *The* QBK *problem is* **NP***-hard to approximate.*

The proposition asserts that there cannot be a polynomial-time approximation algorithm for the QBK problem with a multiplicative approximation guarantee. It does not preclude through the existence of an algorithm with additive approximation guarantee.

***Proof*** We prove the proposition by a reduction from SUBSET- SUM, a known **NP**-complete problem [11]. An instance of SUBSET- SUM consists of $n + 1$ positive integers $m_1, \ldots, m_n$ and $M$. The problem asks whether there is a subset $S \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in S} m_i = M$.

Define $A = \sum_{i=1}^{n} m_i$.

Given an instance of SUBSET- SUM, we construct an instance of QBK as follows: The underlying graph $G$ is a star graph with $n+1$ leaves. The central node is node 0. We assign weights to edges $(0, i)$ by $w_{0,i} = -m_i$, for $i = 1, \ldots, n$, and weight $w_{0,n+1} = M - A - 1$ to edge $(0, n+1)$. We set the budget $k = M$ and the node weights $b_0 = 0$, $b_i = m_i$, for $i = 1, \ldots, n$, and $b_{n+1} = M + 1$—for $b_{n+1}$, any other number bigger than $M$ also works, since the goal is to make selection of this coefficient infeasible. Finally, we set $s_i = 1$, for all $i = 0, 1, \ldots, n, n+1$, so that the matrix $\mathbf{P}$ is the Laplacian $\mathbf{L}$. Observe that the resulting laplacian has the form

$$\mathbf{L} = \begin{pmatrix} -M+1 & m_1 & \cdots & m_n & M-A-1 \\ m_1 & -m_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ m_n & 0 & \cdots & -m_n & 0 \\ M-A-1 & 0 & \cdots & 0 & -M+A+1 \end{pmatrix}.$$

Now consider a binary vector $\mathbf{x} = [x_i]_{i=0,\ldots,n+1}$. We interpret $\mathbf{x}$ as a solution to QBK, and the coordinates $[x_1, \ldots, x_n]$ as indicator variables for a solution to SUBSET- SUM.

First, note that due to the knapsack constraint $\mathbf{b}^\top \mathbf{x} \leq M$, since $b_{n+1} = M + 1$ and since $b_i \geq 0$ for all $i$, it is $x_{n+1} = 0$.

If $x_0 = 0$, then any feasible solution to QBK can be at most 0, and in fact, the value 0 can be obtained by the feasible vector $\mathbf{x} = \mathbf{0}$.

If $x_0 = 1$, let $S \subseteq \{1, \ldots, n\}$ be the set of all other nonzero coordinates in a feasible solution to QBK. The value of the solution is $f = (-M+1) + 2\sum_{i \in S} m_i - \sum_{i \in S} m_i = \sum_{i \in S} m_i - M + 1$. Due to the knapsack constraint $\sum_{i \in S} m_i \leq M$, it follows that if the answer to SUBSET- SUM is `no` we have $f \leq 0$, while if the answer to SUBSET- SUM is `yes` then $f$ can obtain the value 1.

We conclude that the optimal value to QBK is 1 if and only if the answer to SUBSET- SUM is `yes`, while the optimal value to QBK is 0 if and only if the answer to SUBSET- SUM is `no`.

Furthermore, any polynomial-time approximation algorithm with a finite (multiplicative) approximation guarantee that could be used to solve QBK will need to provide a nonzero value for QBK if and only if the answer to SUBSET- SUM is `yes`. Thus, no such algorithm can exist, unless $\mathbf{P} = \mathbf{NP}$. □

## 5 Algorithms

In this section, we discuss the proposed algorithms for the diversity-maximization problem and present upper bounds on the optimal solution.

We start by the observation that QBK is a nonconvex optimization problem (due to $\mathbf{P}$ being not positive semidefinite and the constraint $\mathbf{x} \in \{0, 1\}^n$), which is expected since convex problems can be solved in polynomial time. We can show, however, that we can still produce a convex semidefinite relaxation to this problem. Such a relaxation forms the basis for our first algorithm.

### 5.1 Semidefinite-programming relaxation

Semidefinite-programming (SDP) relaxations have long been studied in the optimization community. The idea was introduced by Lovasz [24], but it was arguably the seminal work of Goemans and Williamson for the MAX- CUT problem [17] that brought SDP relaxations into the spotlight.

We have noted that our problem (QBK) is a nonconvex 0–1 quadratic program with a linear constraint on $\mathbf{x}$. The general *quadratic knapsack* problem, first introduced by Gallo et al. [10], is defined for an arbitrary symmetric matrix. Since its introduction, a multitude of methods have been developed to solve the problem. Of particular interest to us is the work of Helmberg et al. [20], in which they introduce a series of SDP relaxations.

Now we present the SDP relaxation to the QBK problem.

**Lift to a matrix variable** As is the common practice with semidefinite relaxations, we "lift" the program to the space of square matrices. In particular, we lift vector $\mathbf{x}$ to a matrix $\mathbf{X}$ by introducing the constraint $\mathbf{X} = \mathbf{x}\,\mathbf{x}^\top$. This constraint is equivalent to $\mathbf{X}$ having rank 1 and being positive semidefinite. However, the rank 1 constraint is not convex; thus, we relax it to $\mathbf{X} \succeq \mathbf{x}\,\mathbf{x}^\top$. From the Schur complement, this is equivalent to

$$\begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \succeq 0.$$

Observe that the constraint $\mathbf{X} - \mathbf{x}\,\mathbf{x}^\top \succeq 0$ implies also that $\mathbf{X} \succeq 0$.

**Objective function** The objective of the SDP relaxation is written as a function of the new variable $\mathbf{X}$ in the trace form $Tr(\mathbf{P}\,\mathbf{X})$, since $\mathbf{x}^\top \mathbf{P}\,\mathbf{x} = Tr(\mathbf{x}^\top \mathbf{P}\,\mathbf{x}) = Tr(\mathbf{P}\,\mathbf{x}\,\mathbf{x}^\top)$.

**Integrality constraint** The integrality constraint $\mathbf{x} \in \{0, 1\}^n$ can be written as $\mathbf{x}^2 = \mathbf{x}$. In the SDP relaxation, we write this as $\mathrm{diag}(\mathbf{X}) = \mathbf{x}$. We note that the polytope corresponding to the 0–1 quadratic optimization problem is called the Boolean quadric polytope [30]. The Boolean quadric polytope has a number of facet-defining inequalities, which can used to tighten the relaxation by cutting off parts of the relaxation polytope.

**Knapsack constraint** We now proceed to describe how to express the linear knapsack-type constraint $\mathbf{b}^\top \mathbf{x} \le k$ with respect to the new variable $\mathbf{X}$. One straightforward way is to apply the constraint on the diagonal elements of $\mathbf{X}$, leading to

$$Tr(\mathrm{Diag}(\mathbf{b})\,\mathbf{X}) \le k. \tag{4}$$

In order to further tighten the relaxation, we replace constraint (4) by a tighter one, which is due to Helmberg et al. [20]. In their work, they show that the *square representation* constraint

$$Tr(\mathbf{b}\,\mathbf{b}^\top \mathbf{X}) \le k^2$$

is tighter than constraint (4).

**The resulting** SDP **relaxation** Putting everything together, our SDP relaxation becomes

$$\begin{aligned} \max \quad & Tr(\mathbf{P}\,\mathbf{X}) \\ \text{subject to} \quad & Tr(\mathbf{b}\,\mathbf{b}^\top \mathbf{X}) \le k^2 \\ & \mathbf{X} - \mathbf{x}\,\mathbf{x}^\top \succeq 0 \\ & \mathrm{diag}(\mathbf{X}) = \mathbf{x}. \end{aligned} \tag{SDP- QBK}$$

The SDP- QBK problem is convex and can be solved efficiently by readily available packages.

**Rounding** Let $Opt = (\mathbf{X}^*, \mathbf{x}^*)$ be an optimal solution of SDP- QBK, obtained by a convex-optimization solver.

The last step is to round the optimal solution $Opt$ to a binary vector $\bar{\mathbf{x}}$ that is a feasible solution for the QBK problem.

In order to derive such a binary solution for QBK, we follow the randomized-rounding approach proposed by Luo et al. [25]: Consider a semidefinite program (P) over a binary vector $\mathbf{x}$ and its relaxation (R) over a lifted variable matrix $\mathbf{X}$, where $\mathbf{X} = \mathbf{x}\,\mathbf{x}^\top$. Let $\mathbf{X}^*$ be the optimal solution to (P), and consider a random vector $\mathbf{z}$ drawn from a Gaussian distribution with zero mean covariance $\mathbf{X}^*$, or, $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{X}^*)$. It can be shown that $\mathbf{z}$ defines a distribution for which the quadratic objective of (R) is maximized and its quadratic constraints are satisfied *in expectation*. Then, a feasible binary solution $\bar{\mathbf{x}}$ to (P) can be constructed as follows:

1. solve (R) to find optimal solution $\mathbf{X}^*$;
2. draw $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{X}^*)$;
3. round $\mathbf{z}$ to a binary $\bar{\mathbf{x}}$;
4. repeat (3) until the constraints of (P) are satisfied.

As shown by Luo et al. [25], in certain cases, this randomized-rounding technique can give solutions with a provable quality guarantee. This is clearly not the case in our problem, as we have shown that it is inapproximable. However, the randomized-rounding technique can still be used as a powerful heuristic in the context of SDP relaxation.

In our case, after solving the relaxed problem SDP- QBK and obtaining an optimal solution $Opt = (\mathbf{X}^*, \mathbf{x}^*)$, we draw a random vector $\mathbf{z} \sim \mathcal{N}(\mathbf{x}^*, \mathbf{X}^* - \mathbf{x}^*\,\mathbf{x}^{*\top})$, The coordinates of $\mathbf{x}^*$ are between 0 and 1, and the coordinates of $\mathbf{z}$ are truncated to be between 0 and 1. The vector $\mathbf{z}$ is rounded to a binary vector $\bar{\mathbf{x}}$ using *randomized rounding*, i.e., $\bar{\mathbf{x}}_i$ is set to 1 with probability equal to $\mathbf{z}_i$.

The optimal solution $Opt = (\mathbf{X}^*, \mathbf{x}^*)$ of SDP- QBK maximizes the stochastic quadratic objective

$$\mathbb{E}_{\mathbf{z}\sim\mathcal{N}(\mathbf{x}^*, \mathbf{X}^* - \mathbf{x}^*\,\mathbf{x}^{*\top})}\left[\mathbf{z}^\top \mathbf{P}\,\mathbf{z}\right],$$

which is also the stochastic version of the objective for QBK. The optimal solution $Opt = (\mathbf{X}^*, \mathbf{x}^*)$ also satisfies the constraint

$$\mathbb{E}_{\mathbf{z}\sim\mathcal{N}(\mathbf{x}^*, \mathbf{X}^* - \mathbf{x}^*\,\mathbf{x}^{*\top})}\left[\mathbf{z}^\top \mathbf{b}\,\mathbf{b}^\top \mathbf{z}\right] \leq k^2.$$

Thus, the binary vector $\bar{\mathbf{x}}$ obtained by the randomized rounding satisfies the constraint $\bar{\mathbf{x}}^\top \mathbf{b}\,\mathbf{b}^\top \bar{\mathbf{x}} \leq k^2$ in expectation. In addition, until $\bar{\mathbf{x}}$ satisfies the knapsack constraint of the QBK problem $\mathbf{b}^\top \bar{\mathbf{x}} \leq k$, new randomized binary vectors $\bar{\mathbf{x}}$ are drawn. The resulting algorithm `SDP-Relax` is shown as Algorithm 1.

## 5.2 Glover's linearization

An alternative way to handle the quadratic programs is to perform *linearization*, i.e., reformulate the quadratic program as a linear program using auxiliary variables and constraints. A concise way to linearize a 0–1 quadratic programs is Glover's linearization [16]. According to this technique, we set $z_i = x_i \sum_{j=1}^{n} P_{ij} x_j$ and reformulate our program as:

$$\text{maximize} \qquad \sum_{i=1}^{n} z_i \qquad \text{(GLOVER)}$$

**Algorithm 1:** `SDP-Relax`

```
input  : matrix $\mathbf{P}$; node weights $\mathbf{b}$; budget $k$; number of iterations $I$
output: indicator vector $\bar{\mathbf{x}}$
solve SDP- QBK and obtain $Opt \leftarrow (\mathbf{X}^*, \mathbf{x}^*)$;
form covariance matrix $\Sigma \leftarrow \mathbf{X}^* - \mathbf{x}^* {\mathbf{x}^*}^\top$;
compute the Cholesky factorization $\Sigma = \mathbf{V}\,\mathbf{V}^\top$;
initialize $\bar{\mathbf{x}} \leftarrow \mathbf{0}$ and $f \leftarrow 0$;
for $i \leftarrow 1, \ldots, I$ do
    sample $\mathbf{z} \leftarrow \mathbf{x}^* + \mathbf{V}\,\mathbf{r}$, where $\mathbf{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
    // $\mathbf{z} \sim \mathcal{N}(\mathbf{x}^*, \mathbf{X}^* - \mathbf{x}^*\, {\mathbf{x}^*}^\top)$
    do
        $\bar{\mathbf{x}}' \leftarrow$ randomized_rounding($\mathbf{z}$);
    while $\mathbf{b}^\top \bar{\mathbf{x}}' > k$;
    if $f < \bar{\mathbf{x}}' \mathbf{P}\, \bar{\mathbf{x}}'^\top$ then
        $\bar{\mathbf{x}} \leftarrow \bar{\mathbf{x}}'$ and $f \leftarrow \bar{\mathbf{x}}' \mathbf{P}\, \bar{\mathbf{x}}'^\top$;
return $\bar{\mathbf{x}}$;
```

$$
\begin{array}{lll}
\text{subject to} & \mathbf{x} \in \{0, 1\}^n & \\
& \sum_{i=1}^{n} x_i \le k & \\
& x_i L_i \le z_i \le x_i U_i, \qquad i = 1, \ldots, n & (5) \\
& \sum_{j=1}^{n} P_{ij} x_j - U_i (1 - x_i) \le z_i, \qquad i = 1, \ldots, n, & (6) \\
& z_i \le \sum_{j=1}^{n} P_{ij} x_j - L_i (1 - x_i), \qquad i = 1, \ldots, n, & (7)
\end{array}
$$

where $L_i$ and $U_i$ are lower and upper bounds for $z_i$, respectively. Observe that we can easily obtain such bounds; for each $L_i$, it suffices to set $x_j = 0$ if $P_{ij} < 0$ and $x_j = 1$ if $P_{ij} \ge 0$, while to obtain $U_i$ it suffices to set $x_j = 0$ if $P_{ij} \ge 0$, and $x_j = 1$ if $P_{ij} < 0$.

Inequalities (5)–(7) enforce the following equivalence between problems QBK and GLOVER. If $x_i = 0$ for some $i$, then (5) ensures that $z_i = 0$ and (6)+(7) are redundant. If $x_i = 1$ for some $i$, then (6)+(7) ensure that $z_i = \sum_{j=1}^{n} P_{ij} x_j$ and (5) is redundant. In either case, $z_i = x_i \sum_{j=1}^{n} P_{ij} x_j$ for each $i$.

After formulating GLOVER, we solve the continuous relaxation of this integer program, and we obtain a bound on the initial program. The obtained fractional solution $\mathbf{x}$ is converted into a binary vector $\bar{\mathbf{x}}$ using a similar procedure as before: We repeatedly perform randomized rounding, where each $\bar{x}_i$ is set to 1 with probability equal to $x_i$, until we obtain a feasible solution.

## 5.3 Greedy algorithms

Solving an SDP problem, up to a desirable accuracy $\epsilon$, requires time polynomial in the problem size $n$ and $\log \frac{1}{\epsilon}$. However, SDP solvers are using expensive interior-point methods with running time $\tilde{\mathcal{O}}(n^3)$. Thus, the `SDP-Relax` algorithm discussed in the previous section is expected to produce solutions of high quality, but it is not scalable to problem instances of large size.

In this section, we present two greedy algorithms for the QBK problem, which scale linearly to the size of the input graph. Additionally, as we will see in our experimental evaluation, the greedy algorithms yield solutions of extremely high quality, in practice.

*Simple greedy* The first scalable algorithm is a simple greedy (S-Greedy), which takes advantage of the structural properties of $\mathbf{P}$. Recall from Eq. (3) that the entries of $\mathbf{P}$ incorporate information about the structure of the social network, as well as the exposure values of the neighbors of each node.

Notice that the diagonal entries $P_{ii}$ of matrix $\mathbf{P}$ take values in $[-D_{ii}, D_{ii}]$, where $D_{ii}$ is the weighted degree of node $i$, while the rest of the matrix is sparse. Also observe that in the 0–1 quadratic function $\mathbf{x}^\top \mathbf{P}\,\mathbf{x}$, setting an element of $\mathbf{x}$ to 1 means that the corresponding diagonal element of $\mathbf{P}$ is selected. Therefore, it is beneficial to select first those indices that correspond to the highest diagonal values of $\mathbf{P}$. In order to account for the node weight $\mathbf{b}$, we select the node with the highest ratio $P_{ii}/b_i$, that is, the most cost-effective node for its contribution to the objective.

Altogether, the S-Greedy algorithm selects nodes in descending order of value $P_{ii}/b_i$, while the total weight of selected nodes ($\sum_i b_i x_i$) does not exceed the budget $k$.

*Iterative greedy* An obvious drawback of S-Greedy is that nodes are selected independently, and thus, selected nodes may have the effect of canceling each other.

Our second greedy strategy (named iterative greedy, or simply I-Greedy) overcomes this drawback by selecting nodes iteratively and evaluating the gain in the objective function for each new node. The I-Greedy algorithm first generates a feasible solution $\mathbf{x}$ by initially setting all nodes to 0. Then, it iteratively sets the value of a variable from 0 to 1, so as to achieve the highest gain in the objective value, normalized for node cost, i.e., it selects the node $i$ that achieves the highest value of the ratio $\frac{\mathbf{x}'^\top \mathbf{P}\,\mathbf{x}'}{b_i}$, where $\mathbf{x}'$ differs from the current solution $\mathbf{x}$ as to having its $i$th coordinate equal to 1 instead of 0. The algorithm continues adding nodes, while the total weight of selected nodes ($\sum_i b_i x_i$) does not exceed the budget $k$.

To further explore the search space and allow the possibility of recovering from a bad choice during the greedy selection, we enhance the algorithm with an additional local search step. According to this, a node in the current solution is selected at random, removed from the solution, and other nodes are selected greedily to replace the removed node. The local search step is repeated for a given number of iterations $I$. The I-Greedy algorithm returns the best solution found during its execution.

The I-Greedy algorithm is described in Algorithm 2.

To analyze the running time of I-Greedy, consider the computation of the value $\mathbf{x}'^\top \mathbf{P}\,\mathbf{x}'$. A naïve implementation uses vector-matrix multiplication and results in complexity $\mathcal{O}(n^2)$. However, we can improve the running time considerably, by observing that multiplying a matrix with a binary vector is equivalent to selecting its rows or columns that correspond to indices with value 1 in the vector. Therefore, we can compute the updated value $\mathbf{x}'^\top \mathbf{P}\,\mathbf{x}'$ by selecting a single column and row for the new index and summing the nodes that are indexed by the current index set. Assuming that a solution has at most $\ell$ nodes, the cost is $\mathcal{O}(\ell)$. The total cost in selecting the best index is $\mathcal{O}(n\ell)$. Overall, the total running time of the I-Greedy is $\mathcal{O}(n\ell^2 I)$. In typical scenarios, we can assume $\ell, I << n$, making the algorithm very efficient.

**Algorithm 2:** `I-Greedy`

```
input  : matrix $\mathbf{P}$; node weights $\mathbf{b}$; budget $k$; iterations $I$
output: indicator vector $\mathbf{x}_b$
initialize $\mathbf{x}_b \leftarrow \mathbf{0}$, $\mathbf{x} \leftarrow \mathbf{0}$ and $f \leftarrow 0$;
for $i \leftarrow 1, \ldots, I$ do
    $\mathbf{x}' \leftarrow \mathbf{x}$;
    while $\mathbf{b}^\top \mathbf{x}' \leq k$ do
        $\mathbf{x} \leftarrow \mathbf{x}'$;
        $j^* \leftarrow \arg\max_j \left\{ \frac{\mathbf{x}'^\top \mathbf{P}\,\mathbf{x}'}{b_j} \mid \mathbf{x}' \leftarrow \mathbf{x} \text{ and } x'_j \leftarrow 1 \right\}$;
        $x'_{j*} \leftarrow 1$;
    if $f < \mathbf{x}^\top \mathbf{P}\,\mathbf{x}$ then
        $\mathbf{x}_b \leftarrow \mathbf{x}$ and $f \leftarrow \mathbf{x}^\top \mathbf{P}\,\mathbf{x}$;
    $r \leftarrow \mathit{random}\,\{j \mid x_j = 1\}$;
    $x_r \leftarrow 0$;
return $\mathbf{x}_b$;
```

### 5.4 Mixed-integer quadratic programming

For problem instances of small size, we can solve QBK optimally, using a mixed-integer quadratic programming package. Although the computational complexity of such a method is exponential in the worst case, powerful general-purpose solvers may work well for real-world (not worst-case) inputs.

By solving the problem optimally on small-size datasets, we can evaluate our more scalable techniques by checking how far off they are from the optimal solution. In our experiments, we use CPLEX, a standard mixed-integer quadratic programming solver.

### 5.5 Upper bounds

In this section, we derive three upper bounds for problem QBK. Our bounds are applicable to the special case of all nodes having the same cost ($\mathbf{b} = \mathbf{1}$). This is equivalent to setting a cardinality constraint on vector $\mathbf{x}$. The bounds also hold in the case that $b_i \geq 1$, but they may not be as tight in that case. The three bounds we present differ in computational complexity and tightness.

Computing a tight upper bound for our problem has several benefits. First, we can compare the upper bound with the value of the solution obtained by a method and having an estimation of the approximation for a particular problem instance. Second, a bound can be used to speed up some of our algorithms, e.g., in a branch-and-bound routine when computing the optimal solution, or as a cutting plane in the SDP relaxation algorithm.

The Rayleigh theorem for Hermitian matrices $\mathbf{M}$ provides an upper bound for the quantity $\mathbf{x}^\top \mathbf{M}\,\mathbf{x}$ based on the maximum eigenvalue $\lambda_{\max}(\mathbf{M})$ of $\mathbf{M}$:

$$\mathbf{x}^\top \mathbf{M}\,\mathbf{x} \leq \lambda_{\max}(\mathbf{M})\,\mathbf{x}^\top \mathbf{x}.$$

The constraint $\mathbf{1}^\top \mathbf{x} \leq k$ implies that $\mathbf{x}^\top \mathbf{x} \leq k$, and therefore:

$$\mathbf{x}^\top \mathbf{P}\,\mathbf{x} \leq \lambda_{\max}(\mathbf{P})\,\mathbf{x}^\top \mathbf{x} \leq \lambda_{\max}(\mathbf{P})\,k.$$

We can estimate the dominant eigenvalue $\lambda_{\max}(\mathbf{P})$ iteratively. Taking advantage of the sparsity of $\mathbf{P}$, the complexity of estimating this bound is $\mathcal{O}(rI)$, where $r$ is the number of nonzero elements in $\mathbf{P}$ and $I$ is the number of iterations required for convergence.

Our second bound can be computed in time $\mathcal{O}(n)$, based on the following lemma, which is a consequence of Gershgorin's theorem.

**Lemma 1** $\lambda_{\max}(\mathbf{M}) \leq R = \max_i \left\{ M_{ii} + \sum_{i \neq j} |M_{ij}| \right\}$.

Recall the definition of $\mathbf{P}$, and observe that $P_{ii} = -\mathbf{s}^\top \mathbf{L}_{\cdot i}\, s_i$ and $P_{ij} = s_i L_{ij}$, for $i \neq j$, where $\mathbf{L}_{\cdot i}$ is the $i$th column of the Laplacian matrix, and $L_{ij}$ the $(i, j)$ entry. It follows that $P_{ii} + \sum_{i \neq j} |P_{ij}| = P_{ii} + D_{ii}$, where $D_{ii}$ is the weighted degree of node $i$. From Lemma 1, it follows that $\lambda_{\max}(\mathbf{P}) \leq R = \max_i \{P_{ii} + D_{ii}\}$ and consequently

$$\mathbf{x}^\top \mathbf{P}\, \mathbf{x} \leq kR.$$

The third bound is based on the following observation: Given upper bounds on the rows of $\mathbf{P}$, due to the cardinality constraint on $\mathbf{x}$, the value of the objective function can be at most the sum of the $k$ highest row upper bounds. Accordingly, an upper bound on each row can be obtained by summing the top $k$ nonnegative nodes. This bound is more expensive than the other two, as it requires $\mathcal{O}(n^2)$ operations, but we expect it to be tighter.

## 6 Continuous extensions

Here, we extend our problem formulation and we provide two continuous variants. In the first variant, we want to select $k$ individuals and modify their exposure to some value in the interval $[-1, 1]$. The goal is again to maximize the diversity index. The choice of the algorithm is which individuals to select and a recommended exposure level for each one of them. We show that in the first variant the optimal solution is reached when the new exposure vector takes extreme values. We also propose a second variant of our continuous problem formulation, in which we introduce an $\ell_2$ constraint to bound the total change that the exposure vector $\mathbf{s}$ is subject to.

### 6.1 Bounded-box diversity maximization

We start our exposition with the first problem variant, where we assume that the entries of the input exposure vector $s_i$ as well as the entries of the output exposure vector $y_i$ can take any real value in the interval $[-1, 1]$.

**Problem 2** (Bounded-box diversity maximization) *Given a graph $G = (V, E, w)$ and an exposure index vector $\mathbf{s}$ with $s_i \in [-1, 1]$, $i = 1, \ldots, n$; change at most $k$ elements in $\mathbf{s}$ to get a new exposure index vector $\mathbf{y}$ with $y_i \in [-1, 1]$, $i = 1, \ldots, n$, such that the new diversity index $\eta(G, \mathbf{y})$ is maximized.*

Problem 2 has a direct mathematical programming form as in Eq. (8).

$$\begin{array}{ll} \underset{\mathbf{y}}{\text{maximize}} & \eta(G, \mathbf{y}) \\ \text{subject to} & ||\mathbf{y}||_\infty \leq 1, \\ \text{and} & \mathit{card}(\mathbf{y} - \mathbf{s}) \leq k. \end{array} \tag{8}$$

Recall that $\eta(G, \mathbf{y}) = \mathbf{s}^\top \mathbf{L}\,\mathbf{s}$, where $\mathbf{L}$ is the Laplacian matrix of $G$. Since $\mathbf{L}$ is a positive semidefinite matrix, without the cardinality constraint the maximization problem defined by Eq. (8) is convex. According to the representation theorem [4, Chapter 2, page 72], the optimal solution of convex maximization over a polyhedral set lies on its extreme points. Similarly, as we show in Lemma 2, under the cardinality constraint in Eq. (8), the solution is obtained for $y_i \in \{-1, 1, s_i\}$, for all $i = 1, \ldots, n$.

**Lemma 2** *In the optimal solution of Problem 2, it is $y_i \in \{-1, 1, s_i\}$, for all $i = 1, \ldots, n$.*

***Proof*** We write $\mathbf{y}^\top$ as $[\mathbf{y}_1^\top\ y\ \mathbf{y}_2^\top]$, where $y$ is any element in $\mathbf{y}$. Let $\ell$ be the coefficient of $y^2$ in the polynomial $\mathbf{y}^\top \mathbf{L}\,\mathbf{y}$. Then, $\ell$ is one of the elements at the diagonal of the Laplacian $\mathbf{L}$, and $\mathbf{y}^\top \mathbf{L} \mathbf{y}$ can be written as $C_1 + C_2 y + \ell y^2$. Here, $C_1$ and $C_2$ are independent of $y$. Since $\ell$ is a diagonal element of $\mathbf{L}$, it is nonnegative. It follows that $\arg\max_{y \in [-1,1]} \mathbf{y}^\top \mathbf{L}\,\mathbf{y}$ is either 1 or $-1$. □

According to Lemma 2, Problem 2 can be equivalently expressed by Eq. (9):

$$\begin{array}{lll} \underset{\mathbf{y}}{\text{maximize}} & \mathbf{y}^\top \mathbf{L}\,\mathbf{y} & \\ \text{subject to} & y_i \in \{-1, 1, s_i\}, \ \text{for all } i = 1, \ldots, n, & \\ & \operatorname{card}(\mathbf{x}) \le k, & \\ \text{and} & \mathbf{x} = \mathbf{y} - \mathbf{s}. & \end{array} \tag{9}$$

A drawback of the above formulation is that it does not explicitly show which entries of $\mathbf{s}$ are changed in the solution $\mathbf{y}$. To address this issue, we reformulate our maximization problem as follows: Let $\mathbf{a}$ be a binary vector, with $a_i = 1$ if $s_i$ is changed to 1, and $a_i = 0$ otherwise. Similarly, let $\mathbf{b}$ be a binary vector, with $b_i = 1$ if $s_i$ is changed to $-1$, and $b_i = 0$ otherwise. Let $\mathbf{s}_a = \mathbf{e} - \mathbf{s}$, and let $\mathbf{s}_b = -\mathbf{e} - \mathbf{s}$; We have

$$\mathbf{x} = \operatorname{Diag}(\mathbf{s}_a)\,\mathbf{a} + \operatorname{Diag}(\mathbf{s}_b)\,\mathbf{b}.$$

Then, Equation (9) can be written as follows:

$$\begin{array}{ll} \underset{\mathbf{a},\mathbf{b}}{\text{maximize}} & (\mathbf{s}+\mathbf{x})^\top \mathbf{L}\,(\mathbf{s}+\mathbf{x}) \\ \text{subject to} & \mathbf{a} \in \{0,1\}^n, \\ & \mathbf{b} \in \{0,1\}^n, \\ & a_i + b_i \neq 2, \ \text{for all } i = 1, \ldots, n, \text{ and} \\ & \mathbf{e}^\top(\mathbf{a}+\mathbf{b}) \le k. \end{array} \tag{10}$$

We first observe that the inequality constraint can be dropped without changing the optimal solution of the problem defined by Eq. (10).

**Lemma 3** *Dropping the inequality constraint ($a_i + b_i \neq 2$, for all $i = 1, \ldots, n$) does not influence the optimal solution of the problem defined by Eq. (10).*

***Proof*** Consider a solution with $a_i + b_i = 2$ for some $i = 1, \ldots, n$. Since $a_i$ and $b_i$ are binary, we have $a_i = b_i = 1$. Since $\mathbf{x} = \operatorname{Diag}(\mathbf{s}_a)\,\mathbf{a} + \operatorname{Diag}(\mathbf{s}_b)\,\mathbf{b}$, we have $x_i = s_{ai} + s_{bi} = -2s_i$, which means that $s_i$ is changed to $-s_i$. Furthermore, such a solution contributes exactly 2 to the cardinality of $\mathbf{a} + \mathbf{b}$. However, as proved in Lemma 2, if $s_i$ is changed in the optimal condition, then it should be changed to either 1 or $-1$. We conclude that $a_i + b_i \neq 2$ will not be violated in the optimal solution, and thus, it can be dropped. □

Let $\mathbf{v}$ be a binary vector of dimension $2n$ that concatenates vectors $\mathbf{a}$ and $\mathbf{b}$, i.e., $\mathbf{v}^\top = [\mathbf{a}^\top\ \mathbf{b}^\top]$. We also define $\mathbf{q}^\top = [\mathbf{s}^\top \mathbf{L}\,\mathrm{Diag}(\mathbf{s}_a)\quad \mathbf{s}^\top \mathbf{L}\,\mathrm{Diag}(\mathbf{s}_b)]$, and

$$\mathbf{P} = \begin{pmatrix} \mathrm{Diag}(\mathbf{s}_a)\,\mathbf{L}\,\mathrm{Diag}(\mathbf{s}_a) & \mathrm{Diag}(\mathbf{s}_a)\,\mathbf{L}\,\mathrm{Diag}(\mathbf{s}_b) \\ \mathrm{Diag}(\mathbf{s}_b)\,\mathbf{L}\,\mathrm{Diag}(\mathbf{s}_a) & \mathrm{Diag}(\mathbf{s}_b)\,\mathbf{L}\,\mathrm{Diag}(\mathbf{s}_b) \end{pmatrix}.$$

Then, Eq. (10) can be written as follows:

$$\begin{array}{ll} \underset{\mathbf{v}}{\text{maximize}} & \mathbf{v}^\top \mathbf{P}\,\mathbf{v} + 2\,\mathbf{q}^\top \mathbf{v} + c_1 \\ \text{subject to} & \mathbf{v} \in \{0,1\}^{2n} \\ \text{and} & \mathbf{e}^\top \mathbf{v} \le k \end{array} \tag{11}$$

Note that the inequality constraint in Eq. (10) has been dropped from Eq. (11), according to Lemma 3. The problem defined by Eq. (11) is an instance of the quadratic binary knapsack problem (QBK). As we saw in Sect. 4, QBK is an **NP**-hard and inapproximable problem [21,32]. To solve this problem by efficient heuristics, we use the same techniques as in the discrete version of our problem formulation, i.e., semidefinite programming relaxation and greedy algorithms.

**Semidefinite relaxation** We introduce a $2n \times 2n$ matrix $\mathbf{V}$ with $\mathbf{V} = \mathbf{v}\mathbf{v}^\top$. We also define

$$\tilde{\mathbf{P}} = \begin{pmatrix} \mathbf{P} & \mathbf{q} \\ \mathbf{q}^\top & c_1 \end{pmatrix}, \text{ and } \tilde{\mathbf{V}} = \begin{pmatrix} \mathbf{V} & \mathbf{v} \\ \mathbf{v}^\top & 1 \end{pmatrix}.$$

The objective function can be written as the Frobenius inner product of $\tilde{\mathbf{P}}$ and $\tilde{\mathbf{V}}$, which we denote by $\langle \tilde{\mathbf{P}}, \tilde{\mathbf{V}} \rangle$. We relax the definition $\mathbf{V} = \mathbf{v}\,\mathbf{v}^\top$ into $\mathbf{V} \succcurlyeq \mathbf{v}\,\mathbf{v}^\top$, and we obtain the following semidefinite-programming problem:

$$\begin{array}{ll} \underset{\tilde{\mathbf{V}}}{\text{maximize}} & \langle \tilde{\mathbf{P}}, \tilde{\mathbf{V}} \rangle \\ \text{subject to} & \tilde{\mathbf{V}} = \begin{pmatrix} \mathbf{V} & \mathbf{v} \\ \mathbf{v}^\top & 1 \end{pmatrix}, \\ & \tilde{\mathbf{V}} \succcurlyeq 0, \\ & \mathrm{diag}(\mathbf{V}) = \mathbf{v}, \\ \text{and} & \langle \mathbf{e}\,\mathbf{e}^\top - \mathbf{I}, \mathbf{V} \rangle \le k^2 - k. \end{array} \tag{12}$$

The last constraint is a relaxation of $\mathbf{e}^\top \mathbf{v} \le k$. Instead of relaxing it to $\langle \mathbf{e}\,\mathbf{e}^\top, \mathbf{V} \rangle \le k^2$, we use the tighter constraint $\langle \mathbf{e}\,\mathbf{e}^\top - \mathbf{I}, \mathbf{V} \rangle \le k^2 - k$, due to Helmberg et al. [20].

**Lemma 4** (Helmberg et al. [20]) *The constraint $\langle \mathbf{e}\,\mathbf{e}^\top - \mathbf{I}, \mathbf{V} \rangle \le k^2 - k$ is tighter than the constraint $\langle \mathbf{e}\,\mathbf{e}^\top, \mathbf{V} \rangle \le k^2$.*

**Rounding** By solving Eq. (12), we obtain an optimal matrix $\tilde{\mathbf{V}}^*$. The next step is to round $\tilde{\mathbf{V}}^*$ and obtain a binary vector $\mathbf{v}^*$. We obtain such a vector $\mathbf{v}^*$ by applying a randomized-rounding scheme, as before. Denote the optimal solution to the optimization problem by $\mathbf{v}^{\mathrm{opt}}$, and let $f(\mathbf{v}) = \mathbf{v}^\top \mathbf{P}\,\mathbf{v} + 2\,\mathbf{q}^\top \mathbf{v} + c_1$. The connection between $\mathbf{v}^*$, $\mathbf{v}^{\mathrm{opt}}$, and $\tilde{\mathbf{V}}^*$ is given by $f(\mathbf{v}^*) \le f(\mathbf{v}^{\mathrm{opt}}) \le \langle \tilde{\mathbf{P}}, \tilde{\mathbf{V}}^* \rangle$. Although we cannot have a guarantee on the quality of the solution $\mathbf{v}^*$, we can obtain an instance-specific bound by comparing whether $f(\mathbf{v}^*)$ is close to $\langle \tilde{\mathbf{P}}, \tilde{\mathbf{V}}^* \rangle$.

### 6.2 $\ell_2$-bounded diversity maximization

According to Lemma 2, the changed exposure index always takes extreme values, 1 or $-1$. However, changing the users' exposure index to extreme values may not lead to the desired effect in real-world situations. First, the users may not want their information exposure to be dramatically changed. Receiving too much information that radically challenges their existing ideas may harm the user experience and lead to abandonment of the platform. Second, if the users' exposure to information is different than the exposure of their social connections, it may harm communication in the platform and possibly lead to disputes. To tackle this situation, we add another constraint to our model, which prevents the modified exposure index from taking extreme values.

We use an $\ell_2$ constraint on the total change of the exposure index. In this new formulation, the feasible area without the cardinality constraint is no longer a polygon, and the problem can not be reduced to searching through the extreme points when the cardinality constraint is added. Formally, we define the second continuous variant of our diversity-maximization problem as follows.

**Problem 3** ($\ell_2$-bounded diversity maximization) *We are given a graph $G = (V, E, w)$, a nonnegative constant $\alpha$, and an exposure index vector $\mathbf{s}$ with $s_i \in [-1, 1]$, for $i = 1, \ldots, n$. We want to change at most $k$ elements in $\mathbf{s}$ and get a new exposure index vector $\mathbf{y}$ with $y_i \in [-1, 1]$, for $i = 1, \ldots, n$, such that $||\mathbf{y} - \mathbf{s}||_2^2 \leq \alpha$, and the new diversity index $\eta(G, \mathbf{y})$ is maximized.*

Problem 3 can be written as:

$$\begin{aligned} \underset{\mathbf{y}}{\text{maximize}} \quad & \mathbf{y}^\top \mathbf{L}\, \mathbf{y} && (13)\\ \text{subject to} \quad & ||\mathbf{y}||_\infty \leq 1, && (14)\\ & ||\mathbf{y} - \mathbf{s}||_2^2 \leq \alpha, && (15)\\ & \mathit{card}(\mathbf{y} - \mathbf{s}) \leq k. && (16) \end{aligned}$$

Problems related to the above have been studied in the literature. Boyd and Vandenberghe [6] show that in the case of a single constraint (15), the global maximum can be found by semidefinite relaxation. d'Aspremont et al. [9] give a semidefinite-programming relaxation method to solve the problem with constraints (15) and (16). On the other hand, the problem with constraints (14) and (15) was studied in the previous section. Finally, Ye [35] gives a general-form quadratic programming with a box constraints.

**Semidefinite Relaxation** We follow the approach of d'Aspremont et al. [9], and we solve Problem 3 by semidefinite-programming relaxation. Defining, as before, $\mathbf{x} = \mathbf{y} - \mathbf{s}$, our problem can be written as follows:

$$\begin{aligned} \underset{\mathbf{x}}{\text{maximize}} \quad & \left\langle \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix}, \begin{bmatrix} \mathbf{L} & \mathbf{L}\,\mathbf{s} \\ \mathbf{s}^\top \mathbf{L} & \mathbf{s}^\top \mathbf{L}\,\mathbf{s} \end{bmatrix} \right\rangle \\ \text{subject to} \quad & ||\mathbf{x} + \mathbf{s}||_\infty \leq 1, \\ & \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \succcurlyeq 0, \\ & \mathit{rank}\left( \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \right) = 1, \\ & \langle \mathbf{X}, \mathbf{I} \rangle \leq \alpha, \\ & \mathit{card}(\mathbf{x}) \leq k. \end{aligned} \qquad (17)$$

**Lemma 5** (d'Aspremont et al. [9]) *The constraint $\mathit{card}(\mathbf{x}) \leq k$ can be relaxed to $\mathbf{e}^\top |\mathbf{X}|\, \mathbf{e} \leq \alpha k$.*

***Proof*** According to the Cauchy–Schwarz inequality, $||\mathbf{x}||_1^2 \leq card(\mathbf{x})||\mathbf{x}||_2^2$. Thus, $||\mathbf{x}||_1^2 = \mathbf{e}^\top|\mathbf{x}||\mathbf{x}|^\top\mathbf{e} = \mathbf{e}^\top|\mathbf{X}|\,\mathbf{e}$, $||\mathbf{x}||_2^2 = \langle\mathbf{X}, \mathbf{I}\rangle$, and $card(\mathbf{x}) \leq k$ imply that $\mathbf{e}^\top|\mathbf{X}|\,\mathbf{e} \leq \alpha k$. □

The constraint $||\mathbf{x} + \mathbf{s}||_\infty \leq 1$ can be kept in its original form, as the infinity norm ball forms a convex set. After relaxing the constraint $card(\mathbf{x}) \leq k$ into $\mathbf{e}^\top|\mathbf{X}|\,\mathbf{e} \leq \alpha k$, according to Lemma 5, we can write the SDP relaxation as:

$$\begin{aligned}
\underset{\mathbf{x},\mathbf{X}}{\text{maximize}} \quad & \left\langle \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix}, \begin{bmatrix} \mathbf{L} & \mathbf{Ls} \\ \mathbf{s}^\top\mathbf{L} & \mathbf{s}^\top\mathbf{Ls} \end{bmatrix} \right\rangle \\
\text{subject to} \quad & \begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\top & 1 \end{bmatrix} \succcurlyeq 0, \\
& \mathbf{e}^\top|\mathbf{X}|\,\mathbf{e} \leq \alpha k, \\
& \langle\mathbf{X}, \mathbf{I}\rangle \leq \alpha, \\
& ||\mathbf{x} + \mathbf{s}||_\infty \leq 1, \\
& \mathbf{X}[i, i] \leq s_i^2 + 2|s_i| + 1, \ \text{for all } i = 1, \ldots, n.
\end{aligned} \tag{18}$$

**Rounding** We apply the Gaussian randomized-rounding method as before, to extract a solution $\mathbf{x}$ from the vector $\mathbf{x}^*$ and the matrix $\mathbf{X}^*$ obtained from the semidefinite relaxation. The difference now is that the discretization is no longer needed; we only change the top-$k$ entries with the largest value $|x_i|$, for $i = 1, \ldots, n$.

**Algorithm** After solving the semidefinite program, we can obtain the vector $\mathbf{y}$ through Algorithm 3. The procedure top$(x_0, k)$ returns a vector $\mathbf{x}'$ such that

$$x_i' = \begin{cases} x_{0i} & \text{if } |x_{0i}| \text{ is in the top-}k \text{ of } |\mathbf{x}_0|, \\ 0 & \text{otherwise.} \end{cases}$$

**Algorithm 3:** `SDP-Relax-top`

```
input  : The solution of the semidefinite relaxation $\mathbf{X}^*, \mathbf{x}^*, k, \alpha, \mathbf{L}$
output: The changed exposure vector $\mathbf{y}$
Initialize $\mathbf{y} \leftarrow \mathbf{s}$, $f = 0$, initialize $\mathbf{x}$.;
Form covariance matrix $\Sigma \leftarrow \mathbf{X}^* - \mathbf{x}^*\mathbf{x}^{*\top}$;
for $i \leftarrow 1, \ldots, 1000$ do
    sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{x}^*, \Sigma)$;
    do
        $\mathbf{x}' \leftarrow$ top$(\mathbf{x}_0, k)$;
    while $\mathbf{x}'^\top\mathbf{x}' > \alpha$;
    if $f < \mathbf{x}'^\top\mathbf{L}\,\mathbf{x}' + 2\,\mathbf{x}'^\top\mathbf{L}\,\mathbf{s}$ then
        $\mathbf{x} \leftarrow \mathbf{x}'$ and $f \leftarrow \mathbf{x}'^\top\mathbf{L}\,\mathbf{x}' + 2\,\mathbf{x}'^\top\mathbf{L}\,\mathbf{s}$;
$\mathbf{y} \leftarrow \mathbf{s} + \mathbf{x}$;
return $\mathbf{y}$;
```

## 7 Experiments

In this section, we present an experimental evaluation of the algorithms we presented. The goal of our experiments is threefold: First, we want to compare the performance of the algorithms in terms of the achieved value of the objective function. Second, we want to evaluate the scalability of the algorithms. Finally, we want to investigate the factors affecting the performance of the algorithms.

**Table 1** Dataset statistics

| Dataset | Nodes | Edges | Avg degree | Positive | Negative | $\eta$ |
|---|---|---|---|---|---|---|
| Karate | 34 | 78 | 4.58 | 17 | 17 | 40 |
| Karate-D | 34 | 78 | 4.58 | 18 | 16 | 172 |
| Books | 105 | 441 | 8.40 | 43 | 49 | 140 |
| Books-D | 105 | 441 | 8.40 | 54 | 51 | 896 |
| Twitter100 | 80 | 1 403 | 17.53 | 25 | 55 | 360 |
| Twitter100-D | 80 | 1 403 | 17.53 | 42 | 38 | 2 840 |
| Blogs | 1 222 | 16 717 | 27.36 | 636 | 586 | 5 676 |
| Elections | 18 893 | 269 696 | 14.27 | 6 612 | 12 281 | 112 656 |
| Twitter | 200 073 | 4 009 548 | 50 038.04 | 81 793 | 118 280 | 1 000 800 |

All experiments are conducted on an HPC machine with 8-cores and 32 GB of RAM.

**Datasets** We consider five datasets representing different types of social networks. We use networks where each node is associated with a value between −1 and 1, which we assume that reflects its exposure. We consider the following datasets:

Karate:[3] The well-known dataset representing a social network of a karate club at a US university in the 1970s. The social network is partitioned into two distinct equal-sized communities.

Books:[4] A network of books about US politics, sold by amazon.com. Edges represent frequently co-purchased books. Books are classified as *Liberal* (43), *Conservative* (49), and *Neutral* (13). Neutral books are randomly assigned to one of the two communities.

Blogs:[5] A directed network of hyperlinks between weblogs on US politics recorded in 2005 [1]. Blogs are classified as either *Liberal* or *Conservative*. We disregard edge directions and keep the largest connected component. The resulting dataset contains two communities with 636 and 586 nodes each.

Elections: A network of twitter followers of D. Trump and H. Clinton collected in the end of 2016. We consider two communities of users, partitioned by the usage of hashtags #maga and #imwithher. We keep the largest connected component and iteratively prune nodes to guarantee that every node has degree greater than 1.

Twitter: A large network of twitter users collected between 2011 and 2016, filtered for keywords related to three controversial topics: *gun control*, *abortion*, and *obamacare* [22]. For the exposure of the users, we use the ideology scores estimated by Barberá et al. [3]. We only expect the two greedy algorithms to scale on this dataset; therefore, in order to evaluate all our algorithms, we also generate a smaller dataset. We rank the nodes of the network according to their PageRank values and keep the largest connected component formed from the subgraph of the top-100 nodes. We refer to this smaller dataset as Twitter100.

To evaluate our algorithms in networks that are already diverse and where there is no latent community structure, we create a version of networks Karate, Books, and Twitter100, where nodes are assigned a random exposure value. The resulting networks are called Karate-D, Books-D, and Twitter100-D, respectively.

Table 1 shows the statistics of our datasets. All networks are treated as undirected. All edge weights and node costs are set to 1.

[3] https://networkdata.ics.uci.edu/data.php?id=105.

[4] https://networkdata.ics.uci.edu/data.php?id=8.

[5] https://networkdata.ics.uci.edu/data.php?id=102

*Performance evaluation* We evaluate the algorithms with respect to the diversity-index score they achieve. SDP-Relax is the SDP-based algorithms, Glover is the linearization algorithm, I-Greedy and S-Greedy are the two greedy algorithms, while IQP is the exact algorithm.

**Discrete problem setting** First, we evaluate for the discrete problem Problem 1. Table 2 shows the results obtained by the algorithms on all datasets. For the smaller datasets, Karate, Books, and Twitter100, we set $k = 0.1\,n, 0.2\,n, n$, while for the larger datasets, we set $k = 0.1\,n$ for Blogs, and $k = 0.01\,n$ for Elections. For the largest dataset Twitter, we set $k = 0.001\,n$.

We observe that SDP-Relax is the best performing algorithm: It finds solutions of quality very close to that of IQP, which is optimal. Particularly surprising is the performance of I-Greedy, which is almost equal to SDP-Relax. It even outperforms SDP-Relax slightly in some instances. On the other hand, I-Greedy performs less well for $k = n$, which is expected, given its greedy nature.

It is important to note that IQP terminates in reasonable time only for networks of up to 100 nodes. We also observe that the SDP relaxation is tight and achieves upper bounds very close to the optimal value (always less than 1.007 times the optimal). Glover, on the other hand, does not give tight relaxations: Its upper bounds can get quite off. Finally, S-Greedy manages to achieve good performance for small-size instances, due to picking first the high diagonal elements, but it fails to give good solutions for larger instances.

In addition, we evaluate the quality of the three upper bounds (Sect. 5.5). Table 3 shows the results. Bound3 is the most expensive to compute, but is also tightest. On the other hand, Bound1 is the cheapest to compute, but it can get quite bad. We also observe that it is more tight for diverse networks. This is due to the impact of the diagonal elements of **P** on the computation of the bound: The diagonal elements are smaller for diverse networks. Finally, Bound2 is fairly tight for $k = 0.1\,n$ but it gets worse for $k = n$. In general, we observe that for all bounds, the value is much closer to the optimal for small instances, which is somewhat expected. It is worth noting that for the case $k = n$, despite the fact that the optimal diversity-index value is the same no matter the initial assignment of exposures (since all exposures can be changed), the bounds obtained are different.

**Continuous problem settings** Next, we evaluate the continuous problem formulations, for bounded-box diversity maximization (Problem 2) and $\ell_2$-bounded diversity maximization (Problem 3). Table 4 contains the results for Problem 2, and Table 5 contains the results for Problem 3. The algorithms are evaluated on the same datasets as Problem 1. The exposure index of each node is randomly set to a value in the interval $[-1, 1]$. For Problem 3, we randomly assign each edge of the graph with weight in the interval $[0, 1]$.

Since Problem 2 can be formulated as quadratic knapsack, we apply similar methods as those in Problem 1, i.e., algorithms IQP, SDP-Relax, S-Greedy, and I-Greedy. As a comparison, we introduce the SDP-based SDP-Relax-Direct algorithm; the difference is that the SDP-Relax-Direct does not sample $\mathbf{z} \sim \mathcal{N}(\mathbf{x}^*, \mathbf{X}^* - \mathbf{x}^* {\mathbf{x}^*}^\top)$, but rather, it takes $\mathbf{z} \leftarrow \mathbf{x}^*$ in Algorithm 1. For the SDP-Relax algorithm, in Table 4 we also show the relaxation value of the solution before rounding. We can see that in most cases the relaxation value is very close to the rounded value, indicating that the quality of the solution obtained by the SDP-Relax algorithm is very high. We can also observe that SDP-Relax outperforms the I-Greedy for large values of $k$ and obtains solutions whose value is extremely close to the value of the exact algorithm IQP. However, for small values of $k$, I-Greedy is the best algorithm (apart from IQP). SDP-Relax-Direct is in comparison with the SDP-Relax when $k$ is small; however, when $k$ is large, the results are much worse. Finally, the performance of S-Greedy is the worst across all values of the parameter $k$.

**Table 2** Solution quality and bounds from the relaxations

| Dataset | $k$ | IQP | SDP-Relax | Glover | I-Greedy | S-Greedy |
|---|---|---|---|---|---|---|
| Karate | $0.1n$ | 184 | 184 (185.72) | 184 (209.12) | 184 | 184 |
| | $0.2n$ | 224 | 224 (236.52) | 216 (276.2) | 224 | 204 |
| | $n$ | 244 | 244 (253.92) | 208 (312.00) | 228 | 204 |
| Karate-D | $0.1n$ | 200 | 200 (215.84) | 196 (263.4) | 200 | 200 |
| | $0.2n$ | 220 | 220 (242.96) | 200 (319.36) | 212 | 208 |
| | $n$ | 244 | 244 (253.92) | 200 (372.00) | 220 | 192 |
| Books | $0.1n$ | 828 | 828 (831.24) | 828 (943.6) | 828 | 828 |
| | $0.2n$ | 1056 | 1048 (1089.04) | 996 (1320.04) | 1056 | 992 |
| | $n$ | 1236 | 1224 (1273.72) | 1068 (1788.00) | 1192 | 1012 |
| Books-D | $0.1n$ | 1060 | 1048 (1091.8) | 996 (1313.28) | 1052 | 1008 |
| | $0.2n$ | 1140 | 1124 (1193.92) | 1120 (1555.28) | 1136 | 1016 |
| | $n$ | 1236 | 1228 (1273.76) | 1128 (1990.0) | 1144 | 972 |
| Twitter100 | $0.1n$ | 1700 | 1700 (1700.76) | 1696 (1918.08) | 1700 | 1696 |
| | $0.2n$ | 2396 | 2396 (2405.52) | 2356 (3166.2) | 2396 | 2368 |
| | $n$ | – | 3172 (3216.84) | 2932 (5 624.00) | 3160 | 2588 |
| Twitter100-D | $0.1n$ | 2972 | 2968 (3011.76) | 2888 (3671.96) | 2968 | 2860 |
| | $0.2n$ | – | 3.028 (3102.12) | 2916 (4 285.28) | 3044 | 2860 |
| | $n$ | – | 3172 (3216.84) | 3100 (5 984.00) | 3064 | 2948 |
| Blogs | $0.1n$ | – | – | 39 512 (50 636.24) | 39 516 | 35 556 |
| Elections | $0.01n$ | – | – | – | 471 800 | 469 928 |
| Twitter | $0.001n$ | – | – | – | – | 6 715 012 |

**Table 3** Upper bounds

| Dataset | $k$ | Optimal | Bound1 | Bound2 | Bound3 |
|---|---|---|---|---|---|
| Karate | $0.1\,n$ | 46 | 122 | 63.62 | 57 |
| | $0.2\,n$ | 56 | 262 | 130.63 | 83 |
| | $n$ | 61 | 934 | 452.32 | 145 |
| Karate-D | $0.1\,n$ | 50 | 99 | 76.43 | 59 |
| | $0.2\,n$ | 55 | 169 | 118.23 | 90 |
| | $n$ | 61 | 570 | 302.43 | 145 |
| Books | $0.1\,n$ | 207 | 535 | 297.6 | 245 |
| | $0.2\,n$ | 264 | 1 035 | 560.20 | 387 |
| | $n$ | 309 | 5 235 | 2 766.04 | 857 |
| Books-D | $0.1\,n$ | 265 | 552 | 363.26 | 338 |
| | $0.2\,n$ | 285 | 872 | 494.53 | 439 |
| | $n$ | 309 | 3 528 | 1 249.18 | 778 |
| Twitter100 | $0.1\,n$ | 425 | 890 | 498.27 | 481 |
| | $0.2\,n$ | 599 | 1 590 | 855.50 | 807 |
| Twitter100-D | $0.1\,n$ | 743 | 1 176 | 1 035.32 | 809 |

Problem 3 can no longer be transformed into a quadratic-knapsack problem; however, we can still use the SDP-Relax-top algorithm outlined in Algorithm 3. Results for Problem 3 on our datasets are shown in Table 5. The vector $\mathbf{x}$ at the left side of each algorithm indicates the changes on the exposure vector $\mathbf{s}$ of that algorithm, and the parameter $\alpha$ is the upper bound on $||\mathbf{x}||_2^2$, i.e., $||\mathbf{x}||_2^2 \leq \alpha$. For our experiments, we set $\alpha = \frac{1}{20}\alpha_{\max}$, where $\alpha_{\max} = \sum_i^n \max\{(1-\mathbf{s}_i)^2, (-1-\mathbf{s}_i)^2\}$ is the maximum value that $||\mathbf{x}||_2^2$ can take, taking into account the bounded-box constraints. As we can see again by comparing the relaxation value with the rounded value, the SDP-Relax-top algorithm gives solutions of high quality. As a comparison, we adapted the I-Greedy and the S-Greedy in Problem 2 such that for each element in $\mathbf{s}$ that changed to 1 or $-1$, we check the total changes on $\mathbf{s}$ does not violate the $\ell_2$ constraint. It turns out the with the adapted greedy algorithm, the process stops after a small number of elements in $\mathbf{x}$ are changed. Due to the $\ell_2$ constraint, no more elements in $\mathbf{x}$ can be changed, and as a result, the value of the solution does not increase with $k$. We conclude that for Problem 3 the I-Greedy algorithm is not as effective as the SDP-Relax-top algorithm.

**Scalability** We also perform a scalability analysis of the algorithms. The results of problem 1 are shown in Table 6. We are able to run all algorithms for the smaller datasets, Karate, Books, and Twitter100, although IQP did not terminate after two hours on Twitter100 for $k = n$ and Twitter100-D for $k = 0.2\,n$. For the Blogs dataset, Glover, I-Greedy, and S-Greedy are scalable, while IQP and SDP-Relax run out of memory. I-Greedy and S-Greedy are very scalable, and run fast even on big datasets. S-Greedy scales well even on the very large network, Twitter.

All in all, for the polynomial algorithms, the running time is in line with their theoretical complexity, while IQP is very fast for some instances but does not terminate within two hours for some other instances.

Next, Table 7 shows scalability results for Problem 2 and Table 8 for Problem 3. It is worth mentioning that for the semidefinite-programming relaxation method, the total time taken on SDP-Relax is much larger on solving SDP (in the parentheses), while it takes

**Table 4** Results on bounded-box diversity maximization

| Dataset (diversity index) | $k$ | `IQP` | `SDP-Relax` | `SDP-Relax-Direct` | `S-Greedy` | `I-Greedy` |
|---|---|---|---|---|---|---|
| Karate (12.64) | $0.1n$ | 91.83 | 91.83 (97.88) | 91.83 (97.88) | 91.83 | 91.83 |
| | $0.2n$ | 123.84 | 120.19 (142.12) | 122.83 (142.12) | 116.79 | 123.54 |
| | $n$ | 244.00 | 244.00 (253.96) | 192.00 (253.96) | 127.65 | 224.00 |
| Karate-D (91.23) | $0.1n$ | 109.40 | 108.22 (116.65) | 108.22 (116.65) | 105.40 | 109.40 |
| | $0.2n$ | 122.30 | 121.93 (130.20) | 117.08 (130.20) | 112.22 | 122.30 |
| | $n$ | 138.55 | 138.24 (139.97) | 138.24 (139.97) | 115.62 | 138.19 |
| Books (98.30) | $0.1n$ | 469.53 | 454.03 (505.76) | 461.38 (505.76) | 467.08 | 469.53 |
| | $0.2n$ | 649.51 | 599.47 (740.03) | 565.54 (740.03) | 518.30 | 637.00 |
| | $n$ | 1 236.00 | 1 208.00 (1 273.75) | 976.00 (1 273.75) | 976.00 | 1 188.00 |
| Books-D (460.10) | $0.1n$ | 543.42 | 526.12 (568.82) | 523.04 (568.82) | 519.57 | 535.42 |
| | $0.2n$ | 599.98 | 585.01 (629.95) | 548.67 (629.95) | 557.65 | 590.92 |
| | $n$ | 647.98 | 627.92 (670.51) | 578.83 (670.51) | 574.37 | 625.86 |
| Twitter100 (247.72) | $0.1n$ | 1 003.70 | 983.04 (1 082.72) | 974.29 (1082.72) | 992.89 | 1 000.61 |
| | $0.2n$ | – | 1 393.67 (1 674.54) | 1 390.13 (1 674.54) | 1 399.95 | 1 472.40 |
| | $n$ | – | 3 168.00 (3 216.87) | 2 976.00 (3 216.87) | 1 533.05 | 3 120.00 |
| Twitter100-D (842.03) | $0.1n$ | – | 1 260.69 (1 456.98) | 1 274.59 (1 456.98) | 1 292.25 | 1 302.78 |
| | $0.2n$ | – | 1 566.86 (1 897.54) | 1 543.89 (1 897.54) | 1 445.11 | 1 629.49 |
| | $n$ | – | 3 168.00 (3 216.87) | 2 928.00 (3 168.00) | 1 445.11 | 3 084.00 |
| Blogs (4 621.66) | $0.1n$ | – | – | – | 19 390.79 | 26 827.98 |
| Elections (166 311.56) | $0.1n$ | – | – | – | 178 518.59 | – |

**Table 5** Results on $\ell_2$-bounded diversity maximization

| Dataset (diversity index) | $k$ | $\alpha$ | $\|\|\mathbf{x}\|\|_2^2$ | `SDP-Relax-top` | $\|\|\mathbf{x}\|\|_2^2$ | `S-Greedy` | $\|\|\mathbf{x}\|\|_2^2$ | `I-Greedy` |
|---|---|---|---|---|---|---|---|---|
| Karate (12.64) | $0.1\,n$ | 3.82 | 3.33 | 58.21 (87.47) | 2.88 | 36.69 | 2.88 | 36.69 |
| | $0.2\,n$ | 3.82 | 3.77 | 66.62 (88.74) | 2.88 | 36.69 | 2.88 | 36.69 |
| | $0.3\,n$ | 3.82 | 3.34 | 56.87 (88.87) | 2.88 | 36.69 | 2.88 | 36.69 |
| | $n$ | 3.82 | 3.18 | 57.97 (88.87) | 2.88 | 36.69 | 2.88 | 36.69 |
| Karate-D (44.23) | $0.1\,n$ | 3.82 | 3.28 | 83.28 (139.59) | 1.89 | 45.72 | 2.76 | 45.72 |
| | $0.2\,n$ | 3.82 | 1.83 | 104.74 (146.65) | 2.96 | 60.86 | 2.76 | 45.72 |
| | $0.3\,n$ | 3.82 | 3.12 | 114.54 (149.61) | 2.96 | 60.86 | 2.76 | 45.72 |
| | $n$ | 3.82 | 3.70 | 128.32 (149.97) | 2.96 | 60.86 | 2.76 | 45.72 |
| Books (98.30) | $0.1\,n$ | 12.04 | 9.09 | 326.63 (477.71) | 9.70 | 189.60 | 11.41 | 190.73 |
| | $0.2\,n$ | 12.04 | 8.88 | 337.57 (486.15) | 9.70 | 189.60 | 11.41 | 190.73 |
| | $0.3\,n$ | 12.04 | 9.83 | 360.31 (486.27) | 9.70 | 189.60 | 11.41 | 190.73 |
| | $n$ | 12.04 | 10.42 | 385.46 (486.27) | 9.70 | 189.60 | 11.41 | 190.73 |
| Books-D (254.57) | $0.1\,n$ | 12.04 | 5.45 | 422.00 (715.52) | 11.77 | 217.62 | 4.58 | 235.63 |
| | $0.2\,n$ | 12.04 | 6.39 | 517.22 (746.86) | 11.77 | 217.62 | 9.63 | 328.27 |
| | $0.3\,n$ | 12.04 | 6.49 | 586.56 (757.94) | 11.77 | 217.62 | 11.92 | 367.64 |
| | $n$ | 12.04 | 8.25 | 661.19 (761.31) | 11.77 | 217.62 | 11.92 | 367.64 |
| Twitter100 (247.72) | $0.1\,n$ | 9.25 | 5.28 | 615.36 (969.13) | 7.33 | 400.66 | 8.37 | 444.42 |
| | $0.2\,n$ | 9.25 | 7.25 | 731.64 (1 013.15) | 7.33 | 400.66 | 8.37 | 444.42 |
| | $0.3\,n$ | 9.25 | 8.24 | 813.64 (1 036.86) | 7.33 | 400.66 | 8.37 | 444.42 |
| | $n$ | 9.25 | 9.25 | 1 055.41 (1 055.41) | 7.33 | 400.66 | 8.37 | 444.42 |
| Twitter100-D (842.03) | $0.1\,n$ | 9.25 | 4.28 | 1 212.13 (1 789.66) | 8.84 | 608.24 | 8.03 | 582.62 |
| | $0.2\,n$ | 9.25 | 4.89 | 1 370.44 (1 903.76) | 8.84 | 608.24 | 8.03 | 582.62 |
| | $0.3\,n$ | 9.25 | 5.28 | 1 546.35 (1 974.15) | 8.84 | 608.24 | 8.03 | 582.62 |
| | $n$ | 9.25 | 9.25 | 2 084.27 (2 084.27) | 8.84 | 608.24 | 8.03 | 582.62 |

**Table 6** Running times (in seconds)

| Dataset | $k$ | IQP | SDP-Relax | Glover | I-Greedy | S-Greedy |
|---|---|---|---|---|---|---|
| Karate | $0.1n$ | 0.093 | 1.355 | 0.814 | 0.009 | 0.002 |
| | $0.2n$ | 0.274 | 1.326 | 0.575 | 0.018 | 0.001 |
| | $n$ | 1.620 | 1.820 | 0.587 | 0.035 | 0.001 |
| Karate-D | $0.1n$ | 0.172 | 1.436 | 0.692 | 0.010 | 0.002 |
| | $0.2n$ | 0.275 | 1.271 | 0.613 | 0.019 | 0.002 |
| | $n$ | 0.329 | 1.460 | 0.558 | 0.036 | 0.001 |
| Books | $0.1n$ | 0.098 | 158.297 | 6.259 | 0.078 | 0.002 |
| | $0.2n$ | 0.334 | 165.299 | 5.157 | 0.154 | 0.002 |
| | $n$ | 2.543 | 213.720 | 4.744 | 0.266 | 0.006 |
| Books-D | $0.1n$ | 0.503 | 146.344 | 6.138 | 0.123 | 0.002 |
| | $0.2n$ | 1.493 | 138.263 | 6.225 | 0.150 | 0.002 |
| | $n$ | 3.726 | 188.170 | 4.744 | 0.253 | 0.006 |
| Twitter100 | $0.1n$ | 0.855 | 44.813 | 3.745 | 0.042 | 0.001 |
| | $0.2n$ | 71.362 | 50.523 | 3.139 | 0.083 | 0.002 |
| | $n$ | $> 7200$ | 56.670 | 2.870 | 0.086 | 0.002 |
| Twitter100-D | $0.1n$ | 40.284 | 40.972 | 3.687 | 0.041 | 0.002 |
| | $0.2n$ | $> 7200$ | 39.720 | 2.980 | 0.077 | 0.001 |
| | $n$ | $> 7200$ | 42.811 | 3.007 | 0.072 | 0.002 |
| Blogs | $0.1n$ | – | – | 947.980 | 10.070 | 0.103 |
| Elections | $0.01n$ | – | – | – | 333.727 | 12.961 |
| Twitter | $0.001n$ | – | – | – | – | 3 000.676 |

little time for SDP-Relax-Direct at the rounding stage; the time taken for solving SDP is comparable with solving the greedy algorithms on small datasets. For larger datasets, the greedy algorithm is more scalable.

From Table 8, we can see with the $\ell_2$ constraints, it takes significantly more time than the previous problems; however, rounding step alone takes less time. While both algorithms draw random vectors from a Gaussian distribution, there are differences in three aspects. Firstly, the size of covariance matrix of SDP-Relax in Problem 2 is $(2n + 1) \times (2n + 1)$, while the size of covariance matrix of Problem 3 is $(n + 1) \times (n + 1)$; secondly, in Problem 2, we round every entry of the vector into an integer; thirdly, for some subroutines in Problem 3, it happens that the rounding results violate the cardinality constraint, so the subroutines are repeated again until the cardinality constraint is satisfied. The greedy algorithms are much faster.

**Case study** We conclude the experiments by taking a closer look at the first five nodes selected by IQP in the Twitter100 dataset. We characterize the nodes by ranking them according to three measures: the size of their "echo chamber," defined as the number of their neighbors with the same exposure, their centrality, measured by PageRank score, and their degree. The results are shown in Table 9. We observe that the selected nodes are among the highest ranked nodes in all three categories. It appears that the most important feature when changing the exposure of an individual is the size of their echo chamber. This is in line with the observed performance of S-Greedy that implements this logic and performs well for small $k$.

**Table 7** Running times (in seconds)

| Dataset | $k$ | `IQP` | `SDP-Relax` | `SDP-Relax-Direct` | `S-Greedy` | `I-Greedy` |
|---|---|---|---|---|---|---|
| Karate | $0.1\,n$ | 0.91 | 1.79 (0.68) | 1.27 (0.68) | 0.00 | 0.00 |
| | $0.2\,n$ | 78.41 | 1.78 (0.69) | 1.38 (0.69) | 0.00 | 0.00 |
| | $n$ | 93.32 | 1.61 (0.70) | 1.42 (0.70) | 0.00 | 0.01 |
| Karate-D | $0.1\,n$ | 6.25 | 19.92 (0.68) | 1.22 (0.68) | 0.00 | 0.00 |
| | $0.2\,n$ | 12.32 | 21.22 (0.68) | 1.29 (0.68) | 0.00 | 0.00 |
| | $n$ | 13.01 | 9.31 (0.70) | 1.31 (0.70) | 0.00 | 0.00 |
| Books | $0.1\,n$ | 81.61 | 356.56 (6.65) | 9.59 (6.65) | 0.17 | 0.03 |
| | $0.2\,n$ | 109.64 | 283.33 (6.61) | 9.75 (6.61) | 0.25 | 0.05 |
| | $n$ | 960.20 | 141.06 (6.70) | 9.76 (6.70) | 0.25 | 0.20 |
| Books-D | $0.1\,n$ | 19.36 | 227.52 (6.66) | 9.65 (6.66) | 0.10 | 0.08 |
| | $0.2\,n$ | 17.13 | 214.98 (6.59) | 9.55 (6.59) | 0.15 | 0.24 |
| | $n$ | 63.64 | 46.69 (6.76) | 9.86 (6.76) | 0.09 | 0.28 |
| Twitter100 | $0.1\,n$ | 233.44 | 91.40 (3.87) | 5.47 (3.87) | 0.00 | 0.01 |
| | $0.2\,n$ | – | 93.38 (3.82) | 5.79 (3.82) | 0.00 | 0.01 |
| | $n$ | – | 45.28 (3.98) | 5.67 (3.98) | 0.00 | 0.09 |
| Twitter100-D | $0.1\,n$ | 41.27 | 81.10 (3.83) | 5.32 (3.83) | 0.09 | 0.05 |
| | $0.2\,n$ | 1 681.95 | 105.76 (3.88) | 5.65 (3.88) | 0.12 | 0.13 |
| | $n$ | – | 31.66 (3.96) | 5.66 (3.96) | 0.06 | 0.24 |
| Blogs | $0.1\,n$ | – | – | – | 1.58 | 13.07 |
| Elections | $0.1\,n$ | – | – | – | 21.95 | – |

**Table 8** Running times (in seconds)

| Dataset | $k$ | SDP-Relax-top | S-Greedy | I-Greedy |
|---|---|---|---|---|
| Karate | $0.1\,n$ | 5.08 (5.02) | 0.00 | 0.00 |
| | $0.2\,n$ | 5.52 (5.45) | 0.00 | 0.00 |
| | $0.3\,n$ | 4.78 (4.72) | 0.00 | 0.00 |
| | $n$ | 3.89 (3.80) | 0.00 | 0.00 |
| Karate-D | $0.1\,n$ | 4.80 (4.73) | 0.00 | 0.00 |
| | $0.2\,n$ | 6.73 (6.67) | 0.00 | 0.00 |
| | $0.3\,n$ | 5.80 (5.73) | 0.00 | 0.00 |
| | $n$ | 3.86 (3.78) | 0.00 | 0.01 |
| Books | $0.1\,n$ | 2573.13 (2569.48) | 0.00 | 0.01 |
| | $0.2\,n$ | 3227.13 (3223.47) | 0.00 | 0.02 |
| | $0.3\,n$ | 3138.83 (3135.14) | 0.00 | 0.03 |
| | $n$ | 2485.89 (2482.08) | 0.00 | 0.11 |
| Books-D | $0.1\,n$ | 3080.75 (3077.09) | 0.00 | 0.01 |
| | $0.2\,n$ | 3111.62 (3107.96) | 0.00 | 0.02 |
| | $0.3\,n$ | 2884.86 (2881.20) | 0.00 | 0.03 |
| | $n$ | 2208.06 (2204.32) | 0.00 | 0.11 |
| Twitter100 | $0.1\,n$ | 595.71 (593.75) | 0.00 | 0.00 |
| | $0.2\,n$ | 646.11 (644.19) | 0.00 | 0.01 |
| | $0.3\,n$ | 749.53 (747.61) | 0.00 | 0.02 |
| | $n$ | 432.58 (428.61) | 0.00 | 0.06 |
| Twitter100-D | $0.1\,n$ | 644.68 (642.81) | 0.00 | 0.00 |
| | $0.2\,n$ | 490.46 (488.56) | 0.00 | 0.01 |
| | $0.3\,n$ | 564.60 (562.71) | 0.00 | 0.01 |
| | $n$ | 573.19 (569.15) | 0.00 | 0.05 |

**Table 9** Characteristics of the first five nodes selected by IQP on Twitter100

| # | Echo chamber | Degree | PageRank |
|---|---|---|---|
| 1 | 3 | 6 | 7 |
| 2 | 7 | 11 | 11 |
| 3 | 8 | 12 | 9 |
| 4 | 15 | 13 | 13 |
| 5 | 1 | 3 | 3 |

## 8 Conclusion

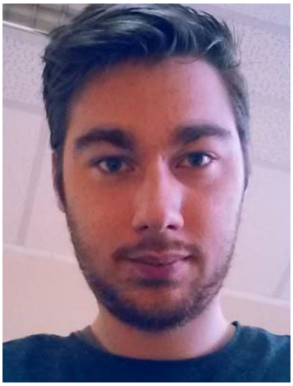
In this paper, we considered the problem of diversifying user exposure to content in social networks. We formally defined the *diversity index* of a social network and formulated the problem of maximizing diversity. We showed that the diversity-maximization problem is **NP**-hard to approximate. Despite this result, we studied algorithms that in practice offer solutions of high quality, including an SDP-based algorithm, an algorithm based on linearization, and two scalable greedy methods. Furthermore, we provided several upper bounds with varying tightness-vs.-efficiency trade-off. Finally, we extended our problem formulation with two continuous variants of our problem, with corresponding SDP relaxations. Our experiments with real data demonstrate the effectiveness of our algorithms in the diversity-maximization problem.

**Acknowledgements** Open access funding provided by Aalto University. This work has been supported by three Academy of Finland projects (286211, 313927, and 317085) and the EC H2020 RIA project "SoBigData" (654024)."

## References

1. Adamic LA, Glance N (2005) The political blogosphere and the 2004 U.S. election: divided they blog. In: International Workshop on Link Discovery, LinkKDD
2. Akoglu L (2014) Quantifying political polarity based on bipartite opinion networks. In: ICWSM
3. Barberá P (2014) Birds of the same feather tweet together: Bayesian ideal point estimation using twitter data. Polit Anal 23(1):76–91
4. Bazaraa MS, Sherali HD, Shetty CM (2006) Nonlinear programming: theory and algorithms. Wiley, Hoboken OCLC: ocm61478842
5. Bessi A, Zollo F, Vicario MD, Puliga M, Scala A, Caldarelli G, Uzzi B, Quattrociocchi W (2016) Users Polarization on Facebook and Youtube. PLoS ONE 11(8):e0159641
6. Boyd SP, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
7. Conover M, Ratkiewicz J, Francisco MR, Gonçalves B, Menczer F, Flammini A (2011) Political polarization on Twitter. In: ICWSM
8. Dandekar P, Goel A, Lee DT (2013) Biased assimilation, homophily, and the dynamics of polarization. PNAS 110(15):5791–5796
9. d'Aspremont A, Ghaoui LE, Jordan MI, Lanckriet GR (2007) A direct formulation for sparse PCA using semidefinite programming. SIAM Rev 49:434–448
10. Gallo G, Hammer PL, Simeone B (1980) Quadratic knapsack problems. In: Padberg MW (ed) Combinatorial optimization. Springer, pp 132–149
11. Garey MR, Johnson DS (1979) Computers and intractability. A guide to the theory of NP-completeness. WH Freeman and Co, New York
12. Garimella K, Gionis A, Parotsidis N, Tatti N (2017) Balancing information exposure in social networks. In: NIPS
13. Garimella K, Morales GDF, Gionis A, Mathioudakis M (2016) Quantifying controversy in social media. In: WSDM
14. Garimella VRK, Morales GDF, Gionis A, Mathioudakis M (2017) Reducing controversy by connecting opposing views. In: WSDM
15. Garrett RK (2009) Echo chambers online? Politically motivated selective exposure among internet news users1. J Comput-Mediat Commun 14(2):265–285
16. Glover F (1975) Improved linear integer programming formulations of nonlinear integer problems. Manag Sci 22(4):455–460
17. Goemans MX, Williamson DP (1995) Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J ACM 42(6):1115–1145
18. Gottfried J, Shearer E (2016) News use across social media platforms 2016. Pew Research Center, Washington
19. Guerra PHC, W M Jr, Cardie C, Kleinberg R (2013) A measure of polarization on social media networks based on community boundaries. In: WSDM
20. Helmberg C, Rendl F, Weismantel R (1996) A semidefinite programming approach to the quadratic knapsack problem. J Comb Optim 4:197–215

21. Kellerer H, Pferschy U, Pisinger D (2004) Introduction to np-completeness of knapsack problems. In: Knapsack problems. Springer, pp 483–493
22. Lahoti P, Garimella K, Gionis A (2018) Joint non-negative matrix factorization for learning ideological leaning on twitter. In: WSDM
23. Liao QV, Fu W-T (2014) Can you hear me now? Mitigating the echo chamber effect by source position indicators. In: Proceedings of the 17th ACM conference on computer supported cooperative work & social computing, CSCW '14. ACM, New York, NY, USA, pp 184–196
24. Lovasz L (2006) On the Shannon capacity of a graph. IEEE Trans Inf Theory 25(1):1–7
25. Luo Z-Q, Ma W-K, So AM-C, Ye Y, Zhang S (2010) Semidefinite relaxation of quadratic optimization problems. IEEE Signal Process Mag 27(3):20–34
26. Matakos A, Gionis A (2018) Tell me something my friends do not know: Diversity maximization in social networks. In: ICDM
27. Matakos A, Terzi E, Tsaparas P (2017) Measuring and moderating opinion polarization in social networks. DMKD 31(5):1480–1505
28. Munson SA, Resnick P (2010) Presenting diverse political opinions: how and how much. In: CHI
29. Musco C, Musco C, Tsourakakis CE (2017) Minimizing polarization and disagreement in social networks. *ArXiv e-prints*
30. Padberg M (1989) The boolean quadric polytope: some characteristics, facets and relatives. Math Program 45(1):139–172
31. Pariser E (2011) The filter bubble: what the internet is hiding from you. Penguin Press, New York
32. Pisinger D (2007) The quadratic knapsack problem—a survey. Discrete Appl Math 155(5):623–648
33. Vicario MD, Scala A, Caldarelli G, Stanley HE, Quattrociocchi W (2016) Modeling confirmation bias and polarization. *CoRR*. arXiv:1607.00022
34. Vydiswaran V, Zhai C, Roth D, Pirolli P (2015) Overcoming bias to learn about controversial topics. JASIST 66:1655–1672
35. Ye Y (1999) Approximating quadratic programming with bound and quadratic constraints. Math Program 84(2):219–226



**Antonis Matakos** received his BSc degree in computer science from the University of Ioannina, Greece, in 2014, and his MSc degree in computer science from the same university in 2017. His research interests are in the area of algorithmic data mining and its applications on social networks and computational sociology. He is particularly interested in studying graph algorithms for social network analysis and algorithms for Web data. Since August 2017, he is PhD student at the Department of Computer Science at Aalto University, under the supervision of prof. Aristides Gionis.

**Sijing Tu** is currently a master's student majoring in machine learning, data science, and artificial intelligence in Aalto University. She received her B.E. degree on information security from Huazhong University of Science and Technology, Hubei, China. Her research interests include graph mining, combinatorial optimization, and computational complexity theory.

**Aristides Gionis** is a WASP professor in the Department of Computer Science in KTH Royal Institute of Technology, Sweden. He is currently an adjunct professor in Aalto University, Finland, and a fellow in the ISI Foundation of Turin, Italy. He was a visiting professor in the University of Rome in 2016. In his previous appointment, he was a senior research scientist and group leader in Yahoo! Research, Barcelona. He obtained his PhD in 2003 from Stanford University, USA. He is currently serving as an action editor in the Data Management and Knowledge Discovery journal (DMKD), an associate editor in the ACM Transactions on Knowledge Discovery from Data (TKDD), and an associate editor in the ACM Transactions on the Web (TWEB). He has contributed in several areas of data science, such as algorithmic data analysis, Web mining, social-media analysis, data clustering, and privacy-preserving data mining. His current research is funded by the Wallenberg AI, Autonomous Systems and Software Program (WASP) in Sweden, the Academy of Finland (projects MLDB and AIDA), and the European Commission with an ERC AdG project (REBOUND) and via a RIA project (SoBigData++).