Modeling the Impact of Timeline Algorithms on Opinion Dynamics Using Low-rank Updates

Tianyi Zhou
KTH Royal Institute of Technology
Stockholm, Sweden
tzho@kth.se

Stefan Neumann*
TU Wien
Vienna, Austria
stefan.neumann@tuwien.ac.at

Kiran Garimella
Rutgers University
New Brunswick, USA
kg766@comminfo.rutgers.edu

Aristides Gionis
KTH Royal Institute of Technology
Stockholm, Sweden
argioni@kth.se

ABSTRACT

Timeline algorithms are key parts of online social networks, but during recent years they have been blamed for increasing polarization and disagreement in our society. Opinion-dynamics models have been used to study a variety of phenomena in online social networks, but an open question remains on how these models can be augmented to take into account the fine-grained impact of user-level timeline algorithms. We make progress on this question by providing a way to model the impact of timeline algorithms on opinion dynamics. Specifically, we show how the popular Friedkin–Johnsen opinion-formation model can be augmented based on *aggregate information*, extracted from timeline data. We use our model to study the problem of minimizing the polarization and disagreement; we assume that we are allowed to make small changes to the users' timeline compositions by strengthening some topics of discussion and penalizing some others. We present a gradient descent-based algorithm for this problem, and show that under realistic parameter settings, our algorithm computes a $(1 + \epsilon)$-approximate solution in time $\widetilde{O}(m\sqrt{n}\log(1/\epsilon))$, where m is the number of edges in the graph and n is the number of vertices. We also present an algorithm that provably computes an ϵ-approximation of our model in near-linear time. We evaluate our method on real-world data and show that it effectively reduces the polarization and disagreement in the network. Finally, we release an anonymized graph dataset with ground-truth opinions and more than 27 000 nodes (the previously largest publicly available dataset contains less than 550 nodes).

CCS CONCEPTS

• **Information systems → Social networks**; • **Theory of computation → Graph algorithms analysis**.

KEYWORDS

Opinion dynamics, Friedkin–Johnsen model, social-network analysis, polarization, disagreement

ACM Reference Format:
Tianyi Zhou, Stefan Neumann, Kiran Garimella, and Aristides Gionis. 2024. Modeling the Impact of Timeline Algorithms on Opinion Dynamics Using Low-rank Updates. In *Proceedings of the ACM Web Conference 2024 (WWW '24), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3589334.3645714

1 INTRODUCTION

Online social networks are used by millions of people on a daily basis and they are integral parts of modern societies. However, during the last decade there has been growing criticism that timeline algorithms, employed in online social networks, create filter bubbles and increase the polarization and disagreement in societies.

Despite significant research effort, our understanding of these phenomena is still limited. One of the main challenges is that polarization and disagreement appear at a *global network-level*, whereas timeline algorithms operate on a *local user-level*. So, on the one hand, opinion dynamics are commonly studied in the context of the graph structure of the social network. On the other hand, timeline algorithms provide a personalized ranking of content (such as posts on Facebook or 𝕏) and only consider users' local neighborhoods in the graph (e.g., k-hop neighborhoods), without considering the global polarization and disagreement. Providing models that bridge the gap between these two levels of abstraction is a major challenge to facilitate our understanding of the underlying phenomena.

A popular way to study the network-level polarization and disagreement is using opinion-formation models, and one of the most popular abstractions is the Friedkin–Johnsen (FJ) model [10]. The vanilla version of the FJ model, however, is not sufficient to model real-world online social networks, since it assumes that the underlying graph is static, based only on friendship relations, and not taking into account *additional* relations and interactions based on recommendations from timeline algorithms.

To address these limitations a lot of attention has been devoted to augmenting the FJ model to understand phenomena that are more closely aligned with the real world [6, 7, 22, 26, 27, 32]. However, existing augmentations are rather simplistic: they either study a small number of edge additions or deletions [26, 32] or they directly

*This work was done while the author was at KTH Royal Institute of Technology.

WWW '24, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0171-9/24/05
https://doi.org/10.1145/3589334.3645714

perform global changes to the graph structure to minimize the polarization and disagreement [6, 7, 22] (see Section 2 for a more detailed description of existing approaches). Most importantly, these papers assume that the graph structure is manipulated directly, which does not align with how timeline algorithms interact with the underlying graph structure. Hence, the augmentations studied in existing papers provide no way of incorporating the properties of timeline algorithms into opinion-formation models. They also provide no means of updating a timeline algorithm's recommendations to reduce polarization and disagreement.

Our contributions. In this paper, we make progress on these issues by introducing an augmentation of the FJ model that *combines a fixed underlying graph and a network that is based on aggregate information of a timeline algorithm.*

In particular, we obtain our *aggregate information* by aggregating along the topics that are discussed in the social network. First, for each user we consider how many posts of their timeline are from each topic. This provides us with the topic distribution on the timeline of each user. Second, for each topic we consider how frequently posts by the users are displayed by the timeline algorithm. This provides us with a distribution for each topic, indicating how influential each user is for this topic. We argue that this is a realistic way to obtain aggregate information for a large range of timeline algorithms in real-world platforms, e.g., on 𝕏 or Reddit.

Based on the aggregate information, we introduce a low-rank graph update, which encodes the social-network connections created by the timeline algorithm's recommendations. In other words, we use the aggregate information and the low-rank graph to bridge between the network-level opinion dynamics and the user-level recommendations of a timeline algorithm. Our model is the first that allows to quantify how timeline algorithms impact polarization and disagreement; we also show that our model can be computed in nearly-linear time. Details are presented in Section 4.

Next, we use our model to study how a timeline algorithm's recommendations need to be adapted to reduce polarization and disagreement, by allowing small changes to the aggregate information. More concretely, we allow small changes to the timelines of the users, such as reducing a user's interest in a highly polarizing topic and slightly strengthening a less controversial topic in the user's timeline. Incorporating these types of the changes into real-world timeline algorithms is quite practical.

For the problem of reducing polarization and disagreement, we provide a gradient descent-based algorithm, called GDPM, and show that under realistic parameter settings it computes a $(1+\epsilon)$-approximate solution in time $\widetilde{O}(m\sqrt{n}\log(1/\epsilon))$, where n is the number of vertices and m is the number of edges in the original graph. The details are presented in Section 5.2.

To obtain our efficient optimization algorithm, we have to overcome significant computational challenges. In particular, since it is possible that the number of edges introduced by the low-rank graph is much larger than in the original graph, even writing down the edges introduced by the recommender system may be infeasible in practice. Therefore, in Section 5.1 we show that we can efficiently approximate the opinions, the polarization, and the disagreement in time that is *near-linear* in the size of the *original* graph.

Furthermore, we experimentally evaluate our algorithm on 27 real-world datasets. Our results show that GDPM can efficiently reduce the *disagreement–polarization index* proposed by Musco et al. [22]. We also qualitatively evaluate which topics are favored and which topics are penalized when reducing the polarization and the disagreement. Additionally, our experiments show that our algorithms are orders of magnitude faster than baseline algorithms and that they scale to graphs with millions of nodes and edges.

Finally, we make our code and two anonymized 𝕏 datasets available for research purposes [30]. Our anonymized graph datasets contain ground-truth opinions and the graph structure for more than 27 000 nodes. The previoulsy largest publicly available dataset contains less than 550 nodes [9].

We include all omitted proofs in the full version of this paper [31]. Our code and our data is available online [30].

2 RELATED WORK

Over the past few years, researchers have studied the phenomena of political polarization on social media [15, 25]. The work includes understanding the impact of polarized discussions [2, 18] as well as developing mitigation strategies [1, 20].

From a practical point of view, there have been various attempts to develop algorithmic solutions to reduce polarization. Several works propose approaches that expose users to opposing viewpoints on online social networks [12–14]. Munson and Resnick [21] design a browser extension that visualizes the bias of a user's content consumption.

To study polarization in online social networks theoretically, researchers resorted to opinion formation models and in recent years the most popular model in this context is the the Friedkin–Johnsen (FJ) model [10]. It has been popular to augment the FJ model with abstractions of algorithmic interventions [3, 6, 29, 32]. Other works in this research area study the impact of adversaries [5, 11, 28] and viral content [27], as well basic properties of the FJ model [4].

Several works in this area dealt with the question of minimizing the polarization and disagreement using small updates to the underlying graph [19, 22, 32]. Zhu et al. [32] and Rácz and Rigobon [26] allow k edge updates to the underlying graph. Musco et al. [22] allow to redistribute all edge weights arbitrarily, whereas Cinus et al. [7] allow edge updates under the constraints that the vertex degrees must stay the same and that no new edges are added to the graph. The main limitation of these works is that the graph updates performed in their algorithms have no clear correspondence with operations of timeline algorithms; for instance, it is unclear how the graph updates proposed in [7, 22] should be incorporated into a timeline algorithm. In contrast, incorporating the changes to the aggregate information that we study in this paper is feasible in practice. We believe that this is a significant contribution to this line of work.

3 PRELIMINARIES

Linear algebra. Let $G = (V, E, w)$ be an undirected, connected, weighted graph with $n = |V|$ vertices and $m = |E|$ edges. We set $\mathbf{L} = \mathbf{D} - \mathbf{A}$ to the Laplacian of G, where $\mathbf{D}$ is the diagonal matrix with $\mathbf{D}_{ii} = \sum_{j\colon (i,j)\in E} w_{ij}$ and $\mathbf{A}$ is the weighted adjacency matrix with $\mathbf{A}_{ij} = w_{ij}$.

For $\mathbf{X} \in \mathbb{R}^{n \times k}$, we denote the Frobenius norm by $\|\mathbf{X}\|_F = (\sum_{i,j} \mathbf{X}_{ij}^2)^{1/2}$. The spectral norm of $\mathbf{X}$ is $\|\mathbf{X}\|_2 = \sigma_{\max}(A)$, where $\sigma_{\max}(\mathbf{A})$ is the largest singular value of $\mathbf{A}$. We also use the 1-norm of a matrix, which is given by $\|\mathbf{X}\|_{1,1} = \sum_{ij} |\mathbf{X}_{ij}|$. We write $\mathbf{X}_i$ to denote the i-th row of $\mathbf{X}$.

We write $\mathbf{I}$ to denote the identity matrix and $\mathbf{1}$ to denote the vector with all entries equal to 1; the dimension will typically be clear from the context. Given a vector $\mathbf{v}$, we write $\operatorname{diag}(\mathbf{v})$ to denote the diagonal matrix with $\operatorname{diag}(\mathbf{v})_{ii} = \mathbf{v}_i$. For vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, we write $\mathbf{u} \odot \mathbf{v} \in \mathbb{R}^n$ to denote their Hadamard product, i.e., $(\mathbf{u} \odot \mathbf{v})_i = \mathbf{u}_i \mathbf{v}_i$. We define $\|\mathbf{v}\|_2 = (\sum_i \mathbf{v}_i^2)^{1/2}$ to be the Euclidean norm of $\mathbf{v}$. For a vector $\mathbf{v}$ and a convex set Q, $\operatorname{Proj}_Q(\mathbf{v})$ denotes the orthogonal projection of $\mathbf{v}$ onto Q.

We write $\operatorname{sgn}(x) \colon \mathbb{R} \to \{-, 0, +\}$ to denote the sign of x. We use the notation $\widetilde{O}(T)$ to denote running times of the form $T \log^{O(1)}(n)$. We write $\operatorname{poly}(n)$ to denote numbers bounded by $n^{O(1)}$.

Friedkin–Johnsen (FJ) model. Let G and $\mathbf{L}$ be as defined above. In the FJ model [10], each node i has a fixed *innate opinion* $\mathbf{s}_i$ and an *expressed opinion* $\mathbf{z}_i^{(t)}$ at time t. Initially, $\mathbf{z}_i^{(0)} = \mathbf{s}_i$, and at time $t+1$ every node i updates their expressed opinion as the weighted average of its own innate opinion and the expressed opinions of its neighbors:

$$\mathbf{z}_i^{(t+1)} = \frac{\mathbf{s}_i + \sum_{j \colon (i,j) \in E} w_{ij} \mathbf{z}_i^{(t)}}{1 + \sum_{j \colon (i,j) \in E} w_{ij}}. \quad (1)$$

We write $\mathbf{s} \in \mathbb{R}^n$ and $\mathbf{z}^{(t)} \in \mathbb{R}^n$ to denote the vectors of innate and expressed opinions, respectively. It is known that in the limit, the expressed opinions converge to $\mathbf{z} = \lim_{t \to \infty} \mathbf{z}^{(t)} = (\mathbf{I} + \mathbf{L})^{-1} \mathbf{s}$.

We assume that the innate opinions are mean-centered and in the interval $[-1, 1]$, i.e., $\sum_{i \in V} \mathbf{s}_i = 0$ and $\mathbf{s}_i \in [-1, 1]$ for all $i \in V$. The latter implies that $\mathbf{z}_i^{(t)} \in [-1, 1]$. We note that these assumptions are made without loss of generality as they can always be achieved by rescaling the opinions $\mathbf{s}$.

For mean-centered opinions, the *polarization index* $P(G)$ measures the variance of the opinions and is given by $P(G) = \sum_{i \in V} \mathbf{z}_i^2$. The *disagreement index* $D(G)$ describes the tension along edges in the network and is given by $D(G) = \sum_{(i,j) \in E} w_{ij} (\mathbf{z}_i - \mathbf{z}_j)^2$. Finally, the *disagreement–polarization index* $I(G)$, on which we will focus for the rest of the paper, is given by

$$I(G) = P(G) + D(G) = \mathbf{s}^\top (\mathbf{I} + \mathbf{L})^{-1} \mathbf{s}, \quad (2)$$

where the last equality was shown by Musco et al. [22]. They also observe that the function $f(\mathbf{L}) = \mathbf{s}^\top (\mathbf{I} + \mathbf{L})^{-1} \mathbf{s}$ is convex if $\mathbf{L} \in \mathcal{L}$ is from a convex set of Laplacians $\mathcal{L}$ [24].

4 PROBLEM FORMULATION

In this section, we formally introduce our augmented version of the FJ model and we state the optimization problem we study for minimizing the disagreement–polarization index. In our model, we show how aggregate information from a timeline algorithm can be used to obtain a low-rank graph update for the FJ model. At a high level, we start with the initial adjacency matrix $\mathbf{A}$, which only contains interaction-information (such as who follows whom), and add an adjacency matrix $\mathbf{A}_\mathbf{X}$ based on the aggregate information.

Table 1: Notation

Variable	Meaning
$G = (V, E)$	Original graph, vertex set, edge set
n	Number of vertices in the original graph
m	Number of edges in the original graph
k	Number of topics
$\mathbf{X}$	User–topic matrix (variable of our algorithm)
$\mathbf{Y}$	Influence–topic matrix (fixed)
$\mathbf{A}$	Adjacency matrix of the original graph
$\mathbf{A}_\mathbf{X}$	Low-rank adjacency matrix based on aggregate information
$\mathbf{L}$	Laplacian of the original graph
$\mathbf{L}_\mathbf{X}$	Laplacian of the low-rank graph
$\mathbf{s}$	Innate opinions
$\mathbf{z}$	Expressed opinions for the original graph
$\mathbf{z}_\mathbf{X}$	Expressed opinions after adding the low-rank update to the original graph
$\widetilde{\mathbf{z}}_\mathbf{X}$	Approximation of $\mathbf{z}_\mathbf{X}$
$f(\mathbf{X})$	Objective function value for $\mathbf{X}$
$\mathbf{X}^{(L)}$	Entry-wise lower bound for $\mathbf{X}'$ in Problem 2
$\mathbf{X}^{(U)}$	Entry-wise upper bound for $\mathbf{X}'$ in Problem 2
θ	Parameter used to define $\mathbf{X}^{(L)}$ and $\mathbf{X}^{(U)}$
Q	Feasible set of matrices $\mathbf{X}$ with $\mathbf{X}^{(L)} \leq \mathbf{X} \leq \mathbf{X}^{(U)}$
C	Percentage of extra edge weight added by low-rank update

The *aggregate information* that we consider is as follows. We consider k different topics and two row-stochastic matrices $\mathbf{X} \in [0,1]^{n \times k}$ and $\mathbf{Y} \in [0,1]^{k \times n}$, i.e., $\sum_{j=1}^k \mathbf{X}_{ij} = 1$, for all $i = 1, \dots, n$, and $\sum_{r=1}^n \mathbf{Y}_{jr} = 1$, for all $j = 1, \dots, k$, and we assume $k \leq n$. Here, $\mathbf{X}$ models how user timelines are formed based on various topics; more concretely, we assume that $\mathbf{X}_{ij}$ is the fraction of posts in user i's timeline from topic j. The matrix $\mathbf{Y}$ models which users are recommended by the timeline algorithm for each topic; that is, when the algorithm recommends contents for topic j, then a fraction of $\mathbf{Y}_{jr}$ of the contents was composed by user r. We believe that it is possible to obtain this type of aggregate data for providers of online social networks in the real-world, for instance, by monitoring users' timelines (to get $\mathbf{X}$) and the recommendations of timeline algorithms (to get $\mathbf{Y}$).

Observe that if we consider the product $\mathbf{XY}$, a $(\mathbf{XY})_{ij}$-fraction of the recommended contents in the timeline of user i is composed by user j. This can also be viewed as the impact that a user j has on another user i. Since in general $\mathbf{XY}$ is a non-symmetric matrix, we also add the transposed term $\mathbf{Y}^\top \mathbf{X}^\top$, which ensures symmetry of the adjacency matrix. This can be interpreted as the impact of users' audience to them, for instance, users want to create content that is liked by their audience.

Thus, we consider a scaled version of $\mathbf{XY} + \mathbf{Y}^T \mathbf{X}^T$. In the following lemma, we show that this matrix adds (weighted) edges of total weight $2n$.

Lemma 1. *It holds that* $\left\|\mathbf{XY} + \mathbf{Y}^T \mathbf{X}^T\right\|_{1,1} = 2n$.

To obtain a more fine-grained control over how many edges we add to the original graph, we consider a scaled version of $\mathbf{XY} + \mathbf{Y}^T \mathbf{X}^T$. More concretely, based on the result from Lemma 1, we add the low-rank adjacency matrix given by

$$\mathbf{A}_\mathbf{X} = \frac{CW}{2n} \left(\mathbf{XY} + \mathbf{Y}^T \mathbf{X}^T \right),$$

where $C > 0$ is a parameter that is fixed throughout the paper and $W = \sum_{(i,j)\in E} w_{ij}$ is the total weight of edges in the original graph G. Observe that Lemma 1 implies that $\|\mathbf{A_X}\|_{1,1} = CW$ and thus if we add the edges in $\mathbf{A_X}$ to the graph, the total weight of edges increases by a C-fraction.[1] In practice, it may be realistic to think of $C = 10\%$ or $C = 50\%$.

After adding the edges $\mathbf{A_X}$, which are based on the aggregate information, the new adjacency matrix becomes

$$\mathbf{A} + \mathbf{A_X} = \mathbf{A} + \frac{CW}{2n}\left(\mathbf{XY} + \mathbf{Y}^T\mathbf{X}^T\right),$$

where $\mathbf{A}$ is the adjacency matrix of the original graph and $\mathbf{A_X}$ is the adjacency matrix of the edges that are introduced by the low-rank update. Next, we write

$$\mathbf{L_X} = \operatorname{diag}(\mathbf{A_X}\mathbf{1}) - \mathbf{A_X}$$

to denote the Laplacian associated with the adjacency matrix $\mathbf{A_X}$. Note that the Laplacian of the combined graph is given by $\mathbf{L} + \mathbf{L_X}$, where $\mathbf{L}$ is the Laplacian of the original graph that only contains the follow-information.

Now, after adding the edges from the low-rank update, the expressed equilibrium opinions that are produced by the FJ opinion dynamics are given by $\mathbf{z_X} = (\mathbf{I} + \mathbf{L} + \mathbf{L_X})^{-1}\mathbf{s}$.

Next, we formally introduce the optimization problem that we study. Intuitively, the problem states that we wish to minimize the disagreement–polarization index (Eq. (2)), while allowing small changes to the aggregate information. In particular, we allow to make changes to how the users' timelines are composed of different topics. The formal definition is as follows.

Problem 2. *Given a graph $G = (V, E)$ with adjacency matrix $\mathbf{A}$ and Laplacian $\mathbf{L}$, user–topic matrix $\mathbf{X} \in [0,1]^{k\times n}$, influence–topic matrix $\mathbf{Y} \in [0,1]^{k\times n}$, and lower and upper bound matrices $\mathbf{X}^{(L)}$ and $\mathbf{X}^{(U)}$, respectively, find a matrix $\mathbf{X}' \in [0,1]^{n\times k}$ to satisfy*

$$\begin{aligned} &\min_{\mathbf{X}'} && f(\mathbf{X}') = \mathbf{s}^T\,(\mathbf{I} + \mathbf{L} + \mathbf{L}_{\mathbf{X}'})^{-1}\,\mathbf{s}, \\ &\textit{such that} && \left\|\mathbf{X}'_i\right\|_1 = 1, \quad \textit{for all } i = 1, \ldots, n, \textit{ and} \\ & && \mathbf{X}^{(L)} \le \mathbf{X}' \le \mathbf{X}^{(U)}. \end{aligned} \tag{3}$$

In Problem 2, we write $\mathbf{X}'_i$ to denote the i-th row of the matrix-valued variable $\mathbf{X}'$. The first constraint ensures that $\mathbf{X}'$ is a row-stochastic matrix. Furthermore, the matrices $\mathbf{X}^{(L)} \in [0,1]^{n\times k}$ and $\mathbf{X}^{(U)} \in [0,1]^{n\times k}$ are part of the input and they give entry-wise lower and upper bounds for the entries in $\mathbf{X}'$, i.e., we require $0 \le \mathbf{X}^{(L)}_{ij} \le \mathbf{X}'_{ij} \le \mathbf{X}^{(U)}_{ij} \le 1$ for all i, j. This constraint can be interpreted as a quantification of how much we can increase/decrease the attention of user i to topic j without the risk of making non-relevant recommendations and without violating ethical considerations. We further assume that $\mathbf{X}^{(L)} \le \mathbf{X} \le \mathbf{X}^{(U)}$, which corresponds to the assumption that the initial matrix $\mathbf{X}$ is a feasible solution to our optimization problem.

In the following, we let Q denote the set of all matrices $\mathbf{X}'$ that satisfy the constraints of Problem 2. Observe that Q is a convex set, since it is the intersection of a box and a hyperplane (the first constraint is equivalent to the hyperplane constraint $\langle \mathbf{X}'_i, \mathbf{1}\rangle = 1$, since all entries in $\mathbf{X}'$ are in the interval $[0,1]$; the second constraint is a box constraint). Furthermore, observe that the constraints are independent across different rows $\mathbf{X}'_i$, which we will exploit later.

Since the objective function and Q are convex, Problem 2 can be solved optimally in polynomial time. However, if we use a off-the-shelf solver for this purpose, its running time will be prohibitively high in practice (see Section 6). Even more, already *a single* computation of the gradient is impractical when done naïvely (see Section 6). We address these challenges in the following section.

[1] We note that while here we only guarantee that the *global* increase of edges is a C-fraction, in the full version [31] we show that also on a *local* user-level, most individual node-degrees are increased by approximately a C-fraction.

5 OPTIMIZATION ALGORITHM

In this section, we present a gradient-descent algorithm, which converges to an optimal solution for Problem 2. We present bounds for its running time and its approximation error after a given number of iterations. We also show that we can approximate the expressed opinions $\mathbf{z_X}$ highly efficiently. We conclude the section by presenting two greedy baseline algorithms.

5.1 Efficient estimation of expressed opinions

To understand the impact of the low-rank update on the user opinions, it is highly interesting to inspect the expressed opinions $\mathbf{z_X}$: comparing them with the original expressed opinions $\mathbf{z}$ will offer us insights into the impact of the timeline algorithm. However, even though in Lemma 1 we bound the total *weight* of edges that are added, their *number* could still be $\Omega(n^2)$, since the matrix $\mathbf{A_X}$ might be dense. Thus, even writing down $\mathbf{A_X}$ would result in running times of $\Omega(n^2)$ and would be prohibitively expensive. Therefore, one challenge is to show how to compute $\mathbf{z_X}$ efficiently.

In the following proposition, we show that since $\mathbf{A_X}$ has small rank, we can exploit the Woodbury identity to obtain an approximation $\widetilde{\mathbf{z}}_\mathbf{X}$ (see the full version [31] for the pseudocode). By using such an approximation we can achieve much faster running times, while still obtaining provably small errors. In the following proposition we use $\mathbf{U} = \begin{pmatrix}\mathbf{X} & \mathbf{Y}^\top\end{pmatrix}$ and $\mathbf{V} = \begin{pmatrix}\mathbf{Y} \\ \mathbf{X}^\top\end{pmatrix}$.

Proposition 3. *Let $\epsilon > 0$. Suppose $\left(-\frac{2n}{CW}\mathbf{I} + \mathbf{V}\mathbf{M}^{-1}\mathbf{U}\right)^{-1}$ exists and $\left\|\mathbf{V}\mathbf{M}^{-1}\mathbf{U}\right\|_2 \le 0.99\frac{2n}{CW}$. There exists an algorithm which computes $\widetilde{\mathbf{z}}_\mathbf{X}$ with $\|\widetilde{\mathbf{z}}_\mathbf{X} - \mathbf{z_X}\|_2 \le \epsilon$ in expected time $\widetilde{O}((mk + nk^2 + k^3)\log(W/\epsilon))$.*

Proof sketch. The algorithm is based on the observation that using the Woodbury matrix identity with $\mathbf{M} = \mathbf{I} + \mathbf{L} + \operatorname{diag}(\mathbf{A_X}\mathbf{1})$, and $\mathbf{U}$ and $\mathbf{V}$ as before, we get that

$$\mathbf{z_X} = \mathbf{M}^{-1}\mathbf{s} + \frac{CW}{2n}\mathbf{M}^{-1}\mathbf{U}\left(\mathbf{I} - \frac{CW}{2n}\mathbf{V}\mathbf{M}^{-1}\mathbf{U}\right)^{-1}\mathbf{V}\mathbf{M}^{-1}\mathbf{s}.$$

Now the algorithm (pseudocode in the full version [31]) basically computes this quantity from right to left. Our main insight here is that we can compute the quantities $\mathbf{M}^{-1}\mathbf{s}$ and $\mathbf{M}^{-1}\mathbf{U}$ using the Laplacian solver from [17]. Here, we approximate $\mathbf{M}^{-1}\mathbf{U}$ column-by-column using the call $\operatorname{Solve}(\mathbf{M}, \mathbf{w}_j, \epsilon_\mathbf{R})$, where $\mathbf{w}_j$ is the j'th column of $\mathbf{U}$ and $\epsilon_\mathbf{R}$ is a suitable error parameter. The remaining matrix multiplications are efficient since $\mathbf{U}$ has only $2k$ columns and since matrix $\mathbf{V}$ has only $2k$ rows.

To obtain our guarantees for the approximation error, we have to perform an intricate error analysis to ensure that errors do not compound too much. This is a challenge since we solve $\mathbf{I} - \frac{CW}{2n}\mathbf{V}\mathbf{M}^{-1}\mathbf{U}$ only approximately but then we have to compute an inverse of this approximate quantity. In the proposition we used the assumptions that $\mathbf{V}\mathbf{M}^{-1}\mathbf{U}$ exists and that $\left\|\mathbf{V}\mathbf{M}^{-1}\mathbf{U}\right\|_2 \leq 0.99 \cdot \frac{2n}{CW}$, to ensure that this can be done without obtaining too much error. In the proof we will also show that these assumptions imply that the inverse $\mathbf{S}^{-1}$ used in the algorithm exists. See the full version [31] for details. □

The input of the algorithm are the innate opinions $\mathbf{s}$, the user–topic matrix $\mathbf{X}$, the influence–topic matrix $\mathbf{Y}$, the fraction of weight parameter C, and the approximation error parameter ϵ. The algorithm returns the approximated expressed opinions $\widetilde{\mathbf{z}}_{\mathbf{X}}$. Note that if we consider the practical scenario of $k = \operatorname{poly}\log(n)$ and $W \leq \operatorname{poly}(n)$, the running time of the algorithm is $\widetilde{O}(m \log(1/\epsilon))$.

Proposition 3 also allows us to efficiently evaluate the disagreement–polarization index after adding the edges in $\mathbf{A}_{\mathbf{X}}$. More concretely, in the following corollary we show that we can efficiently evaluate our objective function $f(\mathbf{X}) = \mathbf{s}^T(\mathbf{I}+\mathbf{L}+\mathbf{L}_{\mathbf{X}})^{-1}\mathbf{s}$ with small error.

Corollary 4. *Let $\epsilon > 0$. Suppose $\left(-\frac{2n}{CW}\mathbf{I} + \mathbf{V}\mathbf{M}^{-1}\mathbf{U}\right)^{-1}$ exists and $\left\|\mathbf{V}\mathbf{M}^{-1}\mathbf{U}\right\|_2 \leq 0.99\frac{2n}{CW}$. We can compute a value $\widetilde{f}$ such that $\left|\widetilde{f} - f(\mathbf{X})\right| \leq \epsilon$ in expected time $\widetilde{O}((mk + nk^2 + k^3)\log(W/\epsilon))$.*

5.2 Gradient descent-based polarization minimization

Next, we present our gradient descent-based polarization minimization (GDPM) algorithm. We start by presenting basic facts about the gradient of our problem in the following proposition.

Proposition 5. *The following three facts hold for the gradient of $f(\mathbf{X})$ with respect to $\mathbf{X}$:*

(1) The gradient $\nabla_{\mathbf{X}} f(\mathbf{X})$ is given by

$$\nabla_{\mathbf{X}} f(\mathbf{X}) = \frac{CW}{2n}(2 \cdot \mathbf{z}_{\mathbf{X}} \cdot \mathbf{z}_{\mathbf{X}}^\top \cdot \mathbf{Y}^\top - (\mathbf{z}_{\mathbf{X}} \odot \mathbf{z}_{\mathbf{X}}) \cdot \mathbf{1}_k^\top - \mathbf{1}_n \cdot (\mathbf{z}_{\mathbf{X}}^\top \odot \mathbf{z}_{\mathbf{X}}^\top) \cdot \mathbf{Y}^\top). \tag{4}$$

(2) The function $f(\mathbf{X})$ is L-smooth with $L = \frac{8CW}{\sqrt{n}} \cdot \|\mathbf{s}\|_2 \cdot \|\mathbf{Y}\|_2^2$, i.e., for all $\mathbf{X}_1, \mathbf{X}_2 \in Q$ it holds that

$$\|\nabla_{\mathbf{X}} f(\mathbf{X}_1) - \nabla_{\mathbf{X}} f(\mathbf{X}_2)\|_F \leq \frac{8CW}{\sqrt{n}} \cdot \|\mathbf{s}\|_2 \cdot \|\mathbf{Y}\|_2^2 \cdot \|\mathbf{X}_1 - \mathbf{X}_2\|_F.$$

(3) Let $\epsilon > 0$. Suppose the conditions of Proposition 3 hold, then we can compute an approximate gradient $\widetilde{\nabla}_{\mathbf{X}} f(\mathbf{X})$ such that $\left\|\widetilde{\nabla}_{\mathbf{X}} f(\mathbf{X}) - \nabla_{\mathbf{X}} f(\mathbf{X})\right\|_F \leq \epsilon$ in expected time $\widetilde{O}((mk + nk^2 + k^3)\log(W/\epsilon))$.

The gradient of our problem is given in Eq. (4) and in the second point we show that it is Lipschitz continuous. Computing the gradient exactly involves computing $\mathbf{z}_{\mathbf{X}}$ exactly; however, this requires to compute the matrix inverse $(\mathbf{I} + \mathbf{L})^{-1}$, which is expensive for large graphs. Hence, in the third point we show that an approximate gradient can be computed highly efficiently and with error guarantees.

Since we only have an approximate gradient, GDPM is an implementation of the gradient descent method by d'Aspremont [8], who analyzed a method of Nesterov [23] with approximate gradient. We use Kiwiel's algorithm [16] to compute the orthogonal projections $\operatorname{Proj}_Q(\cdot)$ on our set of feasible solutions Q in linear time, where we exploit that our constraints are independent across different rows of $\mathbf{X}$. The pseudocode of GDPM is given in the full version [31].

The algorithm takes as input the innate opinions $\mathbf{s}$, the user–topic matrix $\mathbf{X}$, the influence–topic matrix $\mathbf{Y}$, the budget θ, and the extra *weight* parameter C. It returns $\mathbf{X}^{(T)}$ after a number of iterations T.

In the following theorem we present error and running-time guarantees for GDPM, which show that it converges to the optimal solution given enough iterations.

Theorem 6. *Let $\epsilon > 0$. Suppose at each iteration of GDPM the conditions of Proposition 3 are satisfied. Then GDPM computes a solution $\mathbf{X}^{(T)}$ such that $f(\mathbf{X}) - f(\mathbf{X}^*) \leq \epsilon$ in expected time*

$$\widetilde{O}\left(\sqrt{\epsilon^{-1} \cdot CWkn} \cdot (mk + nk^2 + k^3)\log(W/\epsilon)\right),$$

where $\mathbf{X}^*$ is the optimal solution for Problem 2.

We note that in parameter settings that are realistic in practice, GDPM computes a solution with multiplicative error at most $(1+\epsilon')$ in time $\widetilde{O}(m\sqrt{n}\log(1/\epsilon'))$. More concretely, this is the case when the number of topics $k = \operatorname{poly}\log(n)$ is small, the fraction of additional edges $C = O(1)$ is small, and the network is sparse with $W = \widetilde{O}(n)$. Additionally, it is realistic to assume that the optimal solution still has a large amount of polarization and disagreement since at least a constant fraction of the users will differ from the average opinion by at least 0.01; this argument implies that the polarization is at least $\mathsf{LB} = \Omega(n)$, which in turn implies that $f(\mathbf{X}^*) \geq \mathsf{LB} = \Omega(n)$. Hence, if in the theorem we set $\epsilon = \epsilon' \,\mathsf{LB}$, we get the bound above.

5.3 Baselines

Next, we introduce two greedy baseline algorithms. The baselines proceed in iterations and, intuitively, in each iteration they update the user timelines such that some topics are penalized and others are favored; the choice of these topics depends on the baseline.

More concretely, the baselines obtain as input the original graph and the matrices $\mathbf{X}, \mathbf{X}^{(L)}, \mathbf{X}^{(U)}, \mathbf{Y}$ and a number $T_{\max}$ of iterations to perform. First, we set $\mathbf{X}^{(0)} \leftarrow \mathbf{X}$. Now the algorithm performs $T_{\max}$ iterations. In each iteration T, we initialize $\mathbf{X}^{(T)} \leftarrow \mathbf{X}^{(T-1)}$. Then we manipulate the timeline of each user i by redistributing the weights in row i of $\mathbf{X}^{(T)}$. We pick two topics j and j' and transfer as much weight as possible from topic j' to topic j. Intuitively, one can think of j as a topic that we want to strengthen and j' as a topic that we want to penalize; how these topics are picked depends on the implementation of the baseline (see below). To denote how much weight we can transfer, we set $\delta \leftarrow \min\{\mathbf{X}_{ij}^{(U)} - \mathbf{X}_{ij}^{(T)}, \mathbf{X}_{ij'}^{(T)} - \mathbf{X}_{ij'}^{(L)}\}$, i.e., δ corresponds to the weight that we can transfer from topic j' to j without violating the constraints of Problem 2. Then we set $\mathbf{X}_{ij}^{(T)} \leftarrow \mathbf{X}_{ij}^{(T)} + \delta$ and $\mathbf{X}_{ij'}^{(T)} \leftarrow \mathbf{X}_{ij'}^{(T)} - \delta$. As stated before, we do this for each user i. Then the next iteration $T+1$ starts.

Baseline 1: Strengthening non-controversial topics (BL-1). We introduce our first baseline (BL-1), which aims to penalize controversial topics and to strengthen non-controversial topics. We build upon the meta-algorithm above and state how to pick the topics j and j' for the current user i. First, we compute $\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}$ using Proposition 3 and set $\bar{z} = \frac{1}{n}\sum_{u\in V}\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}(u)$ to the average user opinion. Also, for each topic j we set $\tau_j = \sum_{u\in V}\mathbf{Y}_{ju}\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}(u)$ to the weighted average of the opinions of influential users for topic j. Since this does not depend on the user i, this can be done at the beginning of each iteration T. In BL-1, we set j to a controversial topic that is "far away" from the average opinion and j' to a non-controversial topics which is "close" to the average opinion. More concretely, we let j be the topic with $\mathbf{X}_{ij}^{(T)} < \mathbf{X}_{ij}^{(U)}$ that minimizes $|\tau_j - \bar{z}|$; and we let j' be the topic with $\mathbf{X}_{ij'}^{(T)} > \mathbf{X}_{ij'}^{(L)}$ that maximizes $|\tau_{j'} - \bar{z}|$.

Baseline 2: Strengthening opposing topics (BL-2). Our second baseline (BL-2) can be viewed as a reverse of the above strategy and is inspired by the experimental outputs that we observed from GDPM: it penalizes non-controversial topics and strengthens topics that are opposing to user i's opinion. More concretely, we compute $\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}$ and $\bar{z}$ as before. However, then we strengthen the topic j with $\mathbf{X}'_{ij} < \mathbf{X}_{ij}^{(U)}$ that maximizes $-\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}(i)\tau_j$. For instance, if $\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}(i) > 0$ then the algorithm will pick the topic $\tau_j < 0$ of largest absolute value; note that since $\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}$ and τ_j must have different signs, this corresponds to connecting user i to a topic that opposes its own opinion. Also, we let j' be the topic with $\mathbf{X}'_{ij'} > \mathbf{X}_{ij'}^{(L)}$ and $\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}(i)\tau_j > 0$ that minimizes $|\tau_{j'} - \bar{z}|$; this corresponds to our choice of non-controversial topics in BL-1 assuming that τ_j has the same sign as $\widetilde{\mathbf{z}}_{\mathbf{X}^{(T)}}(i)$.

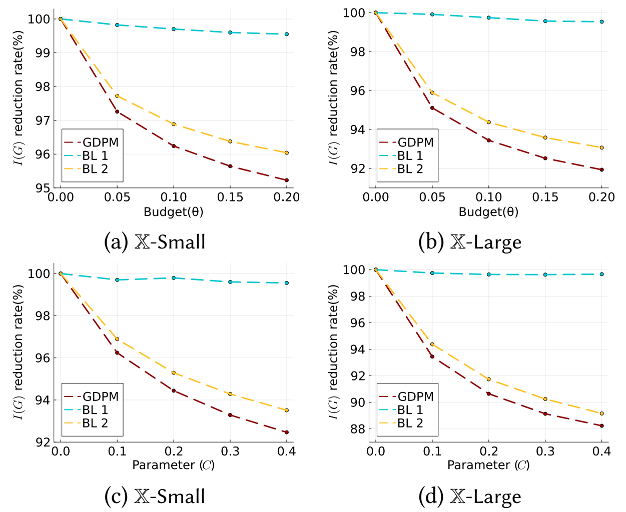
The pseudocode is presented in the full version [31].

6 EXPERIMENTAL EVALUATION

We evaluate our algorithms on 27 real-world datasets. To conduct realistic experiments, we collect two novel real-world datasets from $\mathbb{X}$, which we denote $\mathbb{X}$-Small ($n = 1\,011$, $m = 1\,960$) and $\mathbb{X}$-Large ($n = 27\,058$, $m = 268\,860$). These two datasets contain ground-truth opinions and we use retweet-information to obtain the aggregate information for the interest–topic and influence–topic matrices $\mathbf{X}$ and $\mathbf{Y}$. We provide details on how this data was obtained in the full version [31]. We make our novel datasets available online [30] and we release them for research purposes; we note that $\mathbb{X}$-Large contains more than 27 000 nodes and is thus almost 50 times larger than the previoulsy largest publicly available dataset with ground-truth opinions (which contains less than 550 nodes) [9]. The full version [31] also provides details for the remaining 25 datasets.

We experimentally compare GDPM against the greedy baselines BL-1 and BL-2. We also compare GDPM against the off-the-shelf solver Convex.jl which uses the SCS solver internally. In our experiments, given $\mathbf{X}$ and a parameter $\theta \in [0,1]$, we set $\mathbf{X}_{ij}^{(U)} = \min\{1, \mathbf{X}_{ij} + \theta\}$ and $\mathbf{X}_{ij}^{(L)} = \max\{0, \mathbf{X}_{ij} - \theta\}$, when not mentioned otherwise. We run GDPM with learning rate $L = 10$ (see the full version [31] for a justification).

We conduct our experiments on a Linux workstation with a 2.90 GHz Intel Core i7-10700 CPU and 32 GB of RAM. Our code is written in Julia v1.7.2 and available online [30].



(a) $\mathbb{X}$-Small (b) $\mathbb{X}$-Large

(c) $\mathbb{X}$-Small (d) $\mathbb{X}$-Large

Figure 1: Reduction of the disagreement–polarization index on two datasets for all of our algorithms ($L = 10$). The y-axis shows the reduction ratio $f(\mathbf{X}_{\mathrm{ALG}})/f(\mathbf{X})$. In (a)-(b) we set $C = 0.1$ and vary $\theta \in \{0.05, 0.1, 0.15, 0.2\}$. In (c)-(d) we set $\theta = 0.1$ and vary $C \in \{0.1, 0.2, 0.3, 0.4\}$.

Comparison with greedy baselines and varying parameters. We compare GDPM against the baselines BL-1 and BL-2 on $\mathbb{X}$-Small and $\mathbb{X}$-Large, and vary the parameters θ and C. Note that since GDPM is guaranteed to converge to an optimal solution, we expect it to outperform both baselines.

We report the results of all algorithms with varying $\theta \in \{0.05, 0.1, 0.15, 0.2\}$ in Figures 1(a)–(b) for $\mathbb{X}$-Small and $\mathbb{X}$-Large. As expected, GDPM obtains the largest reduction of the objective. Furthermore, BL-2 outperforms BL-1 by a large margin. This is not surprising, since we designed BL-2 based on insights from analyzing the behavior of GDPM (see below); the observed behavior thus suggests that our intuition about GDPM is correct. In addition, we observe that the reduction in disagreement and polarization increases with θ. This behavior aligns with our expectation, as larger values of θ enlarge the feasible space and allow for more flexibility in recommending interesting topics.

In Figures 1(c)–(d) we report the results of all algorithms with varying $C \in \{0.1, 0.2, 0.3, 0.4\}$ for $\mathbb{X}$-Small and $\mathbb{X}$-Large. The behavior of all algorithms remains consistent: GDPM achieves the largest reduction, while BL-2 outperforms BL-1. As expected, the reduction in disagreement and polarization increases with C, since larger values of C allow for more impact of the timeline algorithm.

Finally, we note that GDPM achieves a larger reduction on $\mathbb{X}$-Large than on $\mathbb{X}$-Small throughout all experiments. This is perhaps a bit surprising since on both datasets we increase the total edge weight by a C-fraction. However, the average node degree of $\mathbb{X}$-Large is larger than for $\mathbb{X}$-Small. Furthermore, the user–topic matrix $\mathbf{X}$ and influence–topic matrix $\mathbf{Y}$ have different structure for $\mathbb{X}$-Small and $\mathbb{X}$-Large, which results in the low-rank adjacency matrix $\mathbf{A}_{\mathbf{X}}$ containing 25% and 33% of non-zero entries, respectively. We believe that both of these characteristics of the datasets lead to higher connectivity in $\mathbb{X}$-Large, which results in better averaging of the opinions and thus ultimately in less polarization and disagreement.

Table 2: Comparison of the reduction ratio $\frac{f(\mathbf{X}^{(T)})}{f(\mathbf{X})}$(%) of GDPM, BL-1, and BL-2 on real-world graphs. In the experiments, we set $\theta = 0.1$ and $C = 0.1$. We used $L = 10$ and $T = 100$ for GDPM and we set $T = 10$ for the greedy baselines. We use synthetic opinion vectors, user-interest, and influence-topic matrices for all datasets except 𝕏-Small and 𝕏-Large.

Graph	GDPM	BL-1	BL-2
𝕏-Small	**96.24**	99.70	96.88
𝕏-Large	**93.44**	99.74	94.37
Erdos992	**94.34**	100	94.97
Advogato	**91.36**	100	92.59
PagesGovernment	**87.82**	100	88.62
WikiElec	**87.30**	100	89.72
HepPh	**86.13**	100	88.69
Anybeat	**92.17**	100	93.21
PagesCompany	**92.41**	100	93.28
AstroPh	**88.20**	100	89.74
CondMat	**91.75**	100	94.36
Gplus	**93.91**	100	94.48
Brightkite	**93.02**	100	93.99
Themarker	**85.62**	100	89.30
Slashdot	**92.23**	100	93.43
WikiTalk	**92.82**	100	93.47
Gowalla	**91.79**	100	92.63
Academia	**92.04**	100	93.37
GooglePlus	**86.43**	100	88.28
Citeseer	**90.53**	100	91.48
MathSciNet	**93.55**	100	93.93
𝕏-Follows	**94.22**	100	95.59
YoutubeSnap	**93.58**	100	94.76

Performance of the optimization algorithms. We report the optimization results of GDPM, BL-1, and BL-2 in Table 2. We used different real-world graphs with synthetically generated polarized opinions and synethically generated matrices $\mathbf{X}$ and $\mathbf{Y}$. We run the greedy baselines 10 iterations due to the high computation cost and choose the best $\mathbf{X}^{(T)}$ as output in our experiments.

We observe that GDPM outperforms the two baselines on all graphs. This is the expected behavior, since GDPM guarantees decreasing the objective function constantly and converges to the optimal solution; the two baselines have no such property. Across all datasets, GDPM decreases the polarization and disagreement by at least 5.6% and by up to 14.4%. Furthermore, BL-2 outperforms BL-1 by a large margin and its results are often not much worse than those of GDPM. Interestingly, BL-1 cannot reduce the polarization and disagreement on all graphs.

Comparison with off-the-shelf convex solver. Since Problem 3 is convex, we compare our gradient-descent based algorithm GDPM against Convex.jl. Convex.jl is a popular off-the-shelf convex optimization tool written in Julia and we use it with the SCS solver. Our experiments show that GDPM is orders of magnitude more efficient than Convex.jl. In particular, even though for this experiment we used 102 GB of RAM, running Convex.jl on graphs with more than 500 nodes exceeds the memory constraint. In contrast, GDPM scales up to graphs with millions of nodes and edges (see Table 2). For the details of these experiments, see the full version [31]. Here, one of the bottlenecks for Convex.jl is that it cannot access our efficient opinion estimation routine from Proposition 3. In the full version [31] we show that the routine from the proposition is indeed orders of magnitude more efficient than estimating the opinions using naïve matrix inversion.

Understanding the behavior of GDPM. Next, we perform experiments to obtain further insights into which topics are favored by GDPM and which ones are penalized.

To answer this question, we consider the initial interest matrix $\mathbf{X}$ and the matrix $\mathbf{X}^{(T)}$ obtained after GDPM converged. To quantify the behavior of GDPM, we consider the column changes among $\mathbf{X}$ and $\mathbf{X}^{(T)}$. Specifically, for each topic j, we measure the change of its weight given by $\delta_j = \sum_i \mathbf{X}_{ij}^{(T)} - \sum_i \mathbf{X}_{ij}$. Note that $\delta_j > 0$ indicates that topic j has more weight in $\mathbf{X}_j^{(T)}$ than in $\mathbf{X}$, i.e., GDPM "favors" it; similarly, $\delta_j < 0$ indicates that topic j has less weight in $\mathbf{X}_j^{(T)}$ than in $\mathbf{X}$, i.e., GDPM "penalizes" it.

In Figure 2(a) we plot tuples $(\tau_{j,\mathbf{s}}, \delta_j)$ for each topic j, where δ_j is the change in importance for topic j, as defined in the previous paragraph, and $\tau_{j,\mathbf{s}} = \sum_{u \in V} \mathbf{Y}_{ju}\mathbf{s}(u)$ is the weighted average of the innate opinions of the influencers for topic j. We also color-code the topics based on their content. We observe that GDPM clearly favors topics with large absolute values $|\tau_{j,\mathbf{s}}|$ and it penalizes non-controversial topics with $|\tau_{j,\mathbf{s}}|$ close to 0. We explain this behavior as a consequence of the FJ model opinion dynamics: more controversial topics have a larger impact on the polarization, and to reduce the polarization one has to bring together people from opposing sides.

We note that in all plots, the most favored topics are political. This is surprising, as the algorithm is not aware of the topic labeling. However, we believe this is a consequence of the fact that political topics are among the most controversial (see also below).

In Figures 2(b) and 2(c), we again show the δ_j values but this time plotted against $\tau_{j,\mathbf{z}_\mathbf{X}}$ using the original expressed opinions $\mathbf{z}_\mathbf{X}$ (before optimization) and $\tau_{j,\mathbf{z}_{\mathbf{X}^{(T)}}}$ using the final expressed opinions $\mathbf{z}_{\mathbf{X}^{(T)}}$ (after optimization). Qualitatively, we observe the same behavior as before: more controversial topics are favored and non-controversial topics are penalized. Now the x-axes have smaller scales, since the expressed opinions are contractions of the innate opinions. It is important to observe that before the optimization (Figure 2(b)) the average topic opinions were in $[-0.066, 0.118]$ and after the optimization (Figure 2(c)) they are in $[-0.028, 0.076]$. Thus, the algorithm brought all topics closer together.

Next, we study the behavior of GDPM when we do not allow to make any changes on the accounts' interests in political topics, i.e., we set $\mathbf{X}_{ij}^{(U)} = \mathbf{X}_{ij}$ and $\mathbf{X}_{ij}^{(L)} = \mathbf{X}_{ij}$ for all political topics j and all accounts i. In Figures 2(d)–(f) we show the same plots as in Figures 2(a)–(c), when weight changes for political topics are not allowed. We obtain the same qualitative outcome as before: controversial topics are favored and non-controversial topics are penalized. We used these qualitative insights to develop BL-2.

As expected, when weight changes for political topics are not allowed, we obtain a restricted version of the problem which limits the disagreement-polarization reduction. For reference, in the

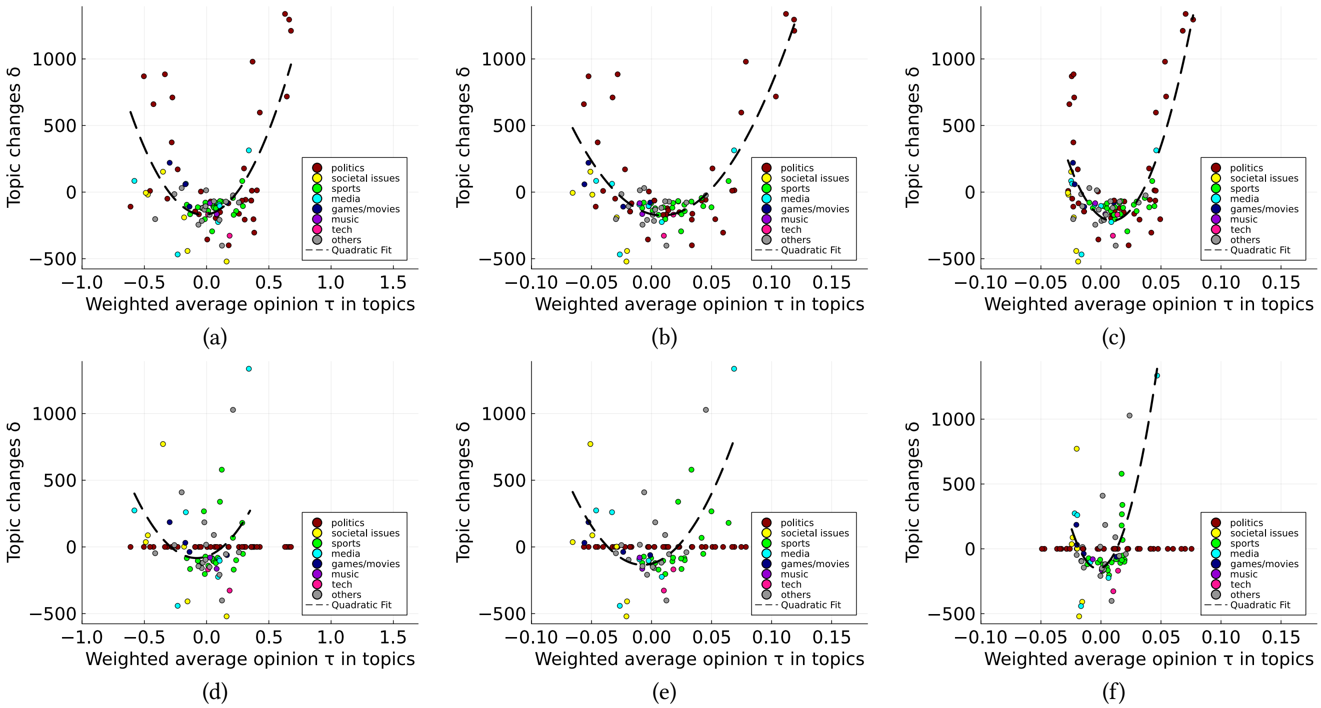


Figure 2: Behavior of GDPM on $\mathbb{X}$-Large ($\theta = 0.1$, $C = 0.1$, $L = 10$). We report the change of topic importance (y-axis) and the weighted average of the opinions of influential users for each topic (x-axis): (a) weighted innate opinions $\tau_{j,s}$; (b) weighted expressed opinions before optimization τ_j; (c) weighted expressed opinions after optimization τ_j. (d)—(f) repeat the same plots when the algorithm must not change interest in political topics. For reference, the results are fitted with a quadratic function.

setting of Figures 2(a)–(c), when weight changes for all topics are allowed, the disagreement-polarization index is reduced to 93.44% of its original value. In contrast, in the setting of Figures 2(d)–(f), with no changes on political topics, the disagreement-polarization index is reduced to only 97.69% of its original value.

We stress that the above fine-grained analysis, that gives insights on which should topics be penalized and favored to reduce the polarization and disagreement in the FJ model, has only become possible due to the introduction of our model from Section 4. We believe that in the future it is interesting to compare these insights with results from political science and computational social sciences.

We include additional experiments, including a running-time analysis, in the full version [31].

7 CONCLUSION

We showed how to augment the popular FJ model to take into account aggregate information of timeline algorithms. This allows us to bridge between network-level opinion dynamics and user-level recommendations. We then considered the problem of optimizing the timeline algorithm, so as to minimize polarization and disagreement in the network, and developed an efficient gradient-descent algorithm, GDPM, which computes an $(1+\epsilon)$-approximate solution in time $\widetilde{O}(m\sqrt{n}\log(1/\epsilon))$ under realistic parameter settings. We presented an algorithm that provably approximates our model, including the measures of polarization and disagreement, in near-linear time. Our experiments confirm the efficiency and effectiveness of the proposed methods and showed that our gradient-descent algorithm is orders of magnitude faster than an off-the-shelf solver. We also release the largest graph datasets with ground-truth opinions.

We believe that our work provides several directions for future research. First, extensions to directed graphs (and, hence, non-symmetric matrices) are highly interesting. Second, inventing algorithms that are even more efficient than GDPM is intriguing (for instance, in a setting with sublinear time/space). Third, it will be valuable to consider other opinion-formation models, beyond the FJ model, and compare the results. Fourth, it will be intriguing to design more complex models, capturing real-world nuances, that allow us to bridge between opinion dynamics and properties of present-day timeline algorithms.

ACKNOWLEDGMENTS

This research has been funded by the ERC Advanced Grant REBOUND (834862), the EC H2020 RIA project SoBigData++ (871042), the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, and the Vienna Science and Technology Fund (WWTF) [Grant ID: 10.47379/VRG23013]. The computation was enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

REFERENCES

[1] Stefano Balietti, Lise Getoor, Daniel G Goldstein, and Duncan J Watts. Reducing opinion polarization: Effects of exposure to similar people with differing political views. *Proceedings of the National Academy of Sciences*, 118(52):e2112552118, 2021.

[2] Michael Barber, Nolan McCarty, Jane Mansbridge, and Cathie Jo Martin. Causes and consequences of polarization. *Political negotiation: A handbook*, 37:39–43, 2015.

[3] Nikita Bhalla, Adam Lechowicz, and Cameron Musco. Local edge dynamics and opinion polarization. In *WSDM*, pages 6–14, 2023.

[4] David Bindel, Jon M. Kleinberg, and Sigal Oren. How bad is forming your own opinion? *Games Econ. Behav.*, 92:248–265, 2015.

[5] Mayee F Chen and Miklós Z Rácz. An adversarial model of network disruption: Maximizing disagreement and polarization in social networks. *IEEE Transactions on Network Science and Engineering*, 9(2):728–739, 2021.

[6] Uthsav Chitra and Christopher Musco. Analyzing the impact of filter bubbles on social network polarization. In *WSDM*, pages 115–123, 2020.

[7] Federico Cinus, Aristides Gionis, and Francesco Bonchi. Rebalancing social feed to minimize polarization and disagreement. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 369–378, 2023.

[8] Alexandre d'Aspremont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19(3):1171–1183, 2008.

[9] Abir De, Sourangshu Bhattacharya, Parantapa Bhattacharya, Niloy Ganguly, and Soumen Chakrabarti. Learning linear influence models in social networks from transient opinion dynamics. *ACM Trans. Web*, 13(3):16:1–16:33, 2019.

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

[11] Jason Gaitonde, Jon Kleinberg, and Eva Tardos. Adversarial perturbations of opinion dynamics in networks. In *EC*, pages 471–472, 2020.

[12] Kiran Garimella, Gianmarco De Francisc iMorales, Aristides Gionis, and Michael Mathioudakis. Mary, mary, quite contrary: Exposing twitter users to contrarian news. In *WWW*, pages 201–205, 2017.

[13] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. Reducing controversy by connecting opposing views. In *WSDM*, pages 81–90, 2017.

[14] Eduardo Graells-Garrido, Mounia Lalmas, and Ricardo Baeza-Yates. Data portraits and intermediary topics: Encouraging exploration of politically diverse profiles. In *IUI*, pages 228–240, 2016.

[15] Shanto Iyengar and Sean J Westwood. Fear and loathing across party lines: New evidence on group polarization. *American Journal of Political Science*, 59(3): 690–707, 2015.

[16] Krzysztof Kiwiel. Breakpoint searching algorithms for the continuous quadratic knapsack problem. *Mathematical Programming*, 112(2):473–491, 2008.

[17] Ioannis Koutis, Gary L. Miller, and Richard Peng. Approaching optimality for solving SDD linear systems. *SIAM J. Comput.*, 43(1):337–354, 2014.

[18] Simon A Levin, Helen V Milner, and Charles Perrings. The dynamics of political polarization, 2021.

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

[20] Nolan McCarty. Reducing polarization by making parties stronger. *Solutions to political polarization in America*, pages 136–45, 2015.

[21] Sean A Munson and Paul Resnick. Presenting diverse political opinions: how and how much. In *CHI*, pages 1457–1466, 2010.

[22] Cameron Musco, Christopher Musco, and Charalampos E. Tsourakakis. Minimizing polarization and disagreement in social networks. In *WWW*, pages 369–378. ACM, 2018.

[23] Yu E Nesterov. A method for solving the convex programming problem with convergence rate $O(\frac{1}{k^2})$. In *Dokl. Akad. Nauk SSSR,*, volume 269, pages 543–547, 1983.

[24] Kenneth Nordström. Convexity of the inverse and moore–penrose inverse. *Linear algebra and its applications*, 434(6):1489–1512, 2011.

[25] Eli Pariser. *The filter bubble: How the new personalized web is changing what we read and how we think.* Penguin, 2011.

[26] Miklós Z. Rácz and Daniel E Rigobon. Towards consensus: Reducing polarization by perturbing social networks. *IEEE Transactions on Network Science and Engineering*, 2023.

[27] Sijing Tu and Stefan Neumann. A viral marketing-based model for opinion dynamics in online social networks. In *WWW*, pages 1570–1578. ACM, 2022.

[28] Sijing Tu, Stefan Neumann, and Aristides Gionis. Adversaries with limited information in the friedkin-johnsen model. In *KDD*, pages 2201–2210, 2023.

[29] Wanyue Xu, Qi Bao, and Zhongzhi Zhang. Fast evaluation for relevant quantities of opinion dynamics. In *WebConf*, pages 2037–2045, 2021.

[30] Tianyi Zhou, Stefan Neumann, Kiran Garimella, and Aristides Gionis. Code and dataset. https://doi.org/10.5281/zenodo.10666350. https://github.com/tianyichow-tcs/GDPM.

[31] Tianyi Zhou, Stefan Neumann, Kiran Garimella, and Aristides Gionis. Modeling the impact of timeline algorithms on opinion dynamics using low-rank updates. https://arxiv.org/abs/2402.10053, 2024. Full version of this paper.

[32] Liwang Zhu, Qi Bao, and Zhongzhi Zhang. Minimizing polarization and disagreement in social networks via link recommendation. *NeurIPS*, 34, 2021.