Optimal Intervention on Weighted Networks via Edge Centrality

Dongyue Li Tina Eliassi-Rad Hongyang R. Zhang

Abstract

We consider the problem of diffusion control via interventions that change network topologies. We study this problem for general weighted networks and present an iterative algorithm, Frank-Wolfe-EdgeCentrality, to reduce the spread of a diffusion process by shrinking the network's top singular values. Given an edge-weight reduction budget, our algorithm identifies the near-optimal edge-weight reduction strategy to minimize the sum of the largest r eigenvalues of $W^\top W$, where W is the network weight matrix. Our algorithm provably converges to the optimum at a rate of $O(t^{-1})$ after t iterations; each iteration only requires a nearly-linear runtime in the number of edges.

We perform a detailed empirical study of our algorithm on a wide range of weighted networks. In particular, we apply our approach to reduce edge weights on mobility networks (between points of interest and census block groups), which have been used to model the spread of COVID-19. In SEIR model simulations, our algorithm reduces the number of infections by 25.70% more than existing approaches, averaged over three weighted graphs and eight mobility networks. Meanwhile, the largest singular value of the weight matrix W decreases by 25.48% more than existing approaches on these networks. An extension of our algorithm to temporal mobility networks also shows an effective reduction in the number of infected nodes.

Keywords: *Targeted Immunization, Edge Centrality, Graph Algorithms, Epidemic Spreading.*

1 Introduction

The problem of diffusion control has gained recent interest in identifying non-pharmaceutical interventions such as lockdown to slow the spread of the SARS-CoV-2 virus. For example, Chang et al. [3] and Chang et al. [4] introduce mobility-based modeling to study the spread of the COVID-19 pandemic. Their approach consists of two major components.

We study network-based interventions to reduce the number of infected nodes in general weighted and directed networks. Suppose there is an epidemic spreading on the network. How can we slow down the spread of the epidemic in the network, subject to reducing the *edge weights* by a limited amount, due to budget constraints? For example, the weight of an edge from a census block group to a place in a mobility network represents the amount of traffic between them. Reducing the weight of this edge corresponds to mobility reduction.

The spreading rate of a diffusion process is closely related to the spectral properties of a network. An important result from the epidemics literature is that the epidemic threshold–below which a diffusion process will die out quickly–scales linearly with the largest (in module) eigenvalue (denoted as λ_1) of the adjacency matrix of the network [2]. As Prakash et al. [19] proved, this result generalizes to various epidemic models. Thus, a natural strategy for slowing down a diffusion process is to remove nodes or edges to reduce λ_1 of a network's adjacency matrix. However, minimizing λ_1 subject to removing a fixed number of nodes or edges is NP-hard via a reduction to the independent set problem [11, 7]. Therefore, various heuristics are proposed to solve this problem in practice. For example, Chen et al. [7] show that by choosing nodes with the highest *node centrality* scores (i.e., a node's value in the eigenvector corresponding to the largest eigenvalue) in a greedy approach, one can reduce the largest eigenvalue and achieve notable reductions in the number of infected nodes. Tong et al. [23] have likewise shown that choosing edges with the highest *edge centrality* scores (i.e., the product of the node centrality scores from both endpoints of an edge) in a greedy algorithm is a scalable and effective approach. Chen et al. [5] further quantified the approximation ratio of these greedy approaches by using techniques from submodular optimization. In light of these works, one natural approach to solving the intervention problem is minimizing the top eigenvalue(s) using edge-weight reduction, and the following questions arise. Does this approach perform well (e.g., compared to greedy algorithms)? Can this problem be solved efficiently in polynomial time? This work provides affirmative answers to these questions by developing an iterative algorithm that provably converges to the optimum of the minimization problem on weighted

Northeastern University, Boston, MA. Correspondence to All Authors ⟨li.dongyu, t.eliassirad, ho.zhang@northeastern.edu⟩.

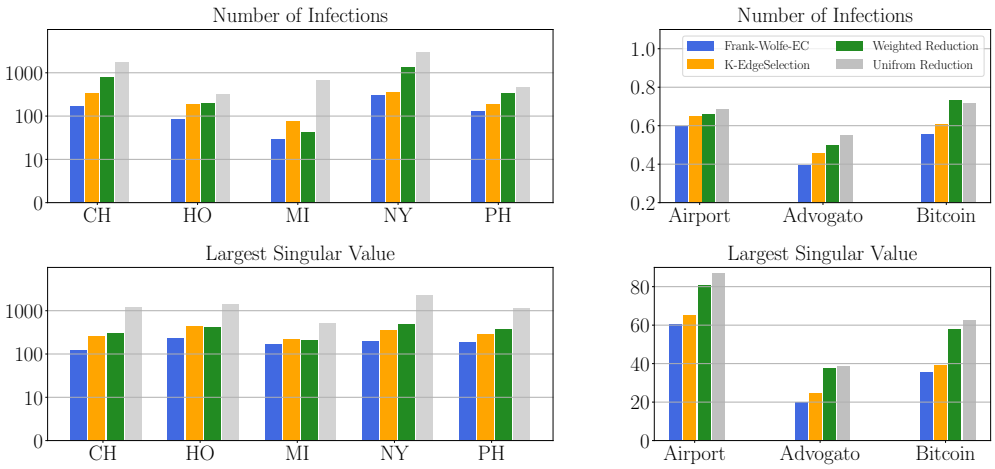


Figure 1: Comparison of our Frank-Wolfe-EC algorithm, K-EdgeSelection [23], and edge-weighted and uniform reduction [3] over eleven weighted graphs. Our approach reduces the number of infections and the largest singular value more significantly than previous approaches on all networks. The number of infections is from simulating an SEIR model over 50 runs.

networks. See Figure 1 for an illustration of our results.

We begin by showing that the edge centrality measures from Tong et al. [23] are equal to the gradient of the largest eigenvalue $\lambda_1(W^\top W)$ (up to scaling), where W is the network weight matrix. We validate that reducing the edge weights of the highest edge centrality scores effectively reduces the number of infected nodes in a weighted network.

Then, we develop a new algorithm, Frank-Wolfe Edge Centrality minimization, to minimize the sum of the largest r eigenvalues of $W^\top W$, subject to an edge-weight reduction budget. At every iteration, our algorithm finds a descent direction that correlates the least with the edge centrality scores, given their gradient interpretation above. A naive approach to finding the descent direction requires solving a linear program (LP). Instead, we present a nearly-linear time algorithm (in the number of edges) by characterizing the LP's optimum as a greedy selection of edges with the highest edge centrality scores. Additionally, we prove that Frank-Wolfe-EC converges to the global minimum at a rate of $O(t^{-1})$ after t iterations.

We evaluate our algorithm by simulating an epidemic model on publicly available weighted graphs and mobility networks. First, on three weighted graphs, our approach achieves on average **10.46**% improvement over baselines during SEIR model simulations. Meanwhile, the largest singular value decreases by an average of **11.42**% more than the baselines. Second, we apply our approach to reduce edge weights on mobility networks. Our algorithm reduces the infected populations by **30.17**% and the largest singular value by **30.75**% more than prior approaches on average. Finally, we extend our algorithm to tackle temporal networks by allocating the weight-reduction budget proportionally to a network's largest singular value. We find that on sequences of temporal mobility networks, this strategy reduces infections by **39.82**% more than other heuristics.

2 Preliminaries

Problem setup. Given an epidemic spreading process on a weighted graph, we are interested in designing algorithms to reduce the number of infected nodes. Let $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a weighted and possibly directed graph. Let $\boldsymbol{V}$ be the set of vertices and $\boldsymbol{E}$ be the set of edges. We use W to denote a non-negative weight matrix over the edges. Let $W_{i,j}$ be the (i,j)-th entry of W. Suppose there is an edge weight reduction budget B. How should we allocate the budget across the graph?

To answer this question, we consider an eigenvalue optimization approach that has been used for unweighted graphs in prior works [2, 19, 23, 7]. The idea behind eigenvalue optimization approaches is to modify the weight matrix W so that its largest eigenvalue is most reduced. We extend the eigenvalue minimization approach to weighted networks as follows. We will state a mathematical optimization formulation of this problem. Let $\lambda_i(W)$ be the i-th largest singular value of W, for i from 1 to r. Notice that the square of $\lambda_i(W)$, denoted as $\lambda_i^2(M)$, is equal to the i-th largest eigenvalue of $M^\top M$. Thus, there is a one-to-one mapping between the singular values of M and the eigenvalues of $M^\top M$, and they can be deduced from each other. Given a rank parameter r, we consider the following

optimization problem:

$$
\begin{aligned}
(2.1)\quad f^{\mathrm{OPT}} \leftarrow \min_{M \in \mathbb{R}^{n \times n}} \quad & f(M) := \sum_{k=1}^{r} \lambda_k^2(M) \\
\text{s.t.} \quad & \sum_{(i,j) \in \boldsymbol{E}} \Big(W_{i,j} - M_{i,j} \Big) \leq B \\
& 0 \leq M_{i,j} \leq W_{i,j}, \ \forall (i,j) \in \boldsymbol{E}, \\
& M_{i,j} = 0, \qquad \forall (i,j) \notin \boldsymbol{E}.
\end{aligned}
$$

After solving the above problem, we get a modified weight matrix M, that will be the solution of our intervention strategy. As a remark, our objective in equation (2.1) generalizes the objective of Tong et al. [23] in that we include the top-r eigenvalues, with $r = 1$ being a special case.

Example. As a motivating example, reducing the weight of an edge between a group and a location in mobility networks corresponds to restricting mobility. A widely used model of epidemic spread is the SEIR compartmental model. An SEIR model uses four compartments to capture a spreading process: Susceptible (S), Exposed (E), Infected (I), and Recovered (R). Every node must belong to one of the four states during the process. A metapopulation SEIR model is introduced in the mobility-based modeling approach of Chang et al. [3]. The metapopulation SEIR model is launched on mobility networks. Mobility networks are bipartite graphs to model the traffic between population groups and locations. One part of the graph includes census block groups (CBGs), which involve a population of individuals in each group. The other part of the graph includes points of interest (POIs), which map to locations. Since there is a population of individuals in each CBG, one SEIR model is instantiated for each CBG. One of their key findings is that fitting the metapopulation dynamics on mobility traffic data results in a surprisingly accurate prediction of the reported number of infected cases.

In Figure 2, we show that reducing the largest singular value of G indeed reduces the number of infections during a SEIR diffusion process. We perform the experiment on a weighted mobility network.

3 Optimization Algorithms and Convergence

We present a new algorithm to optimize problem (2.1) efficiently. To motivate our approach, we start by observing that the gradient of $f(M)$ is equivalent to the "edge centrality scores". Then, we develop an iterative algorithm with an inner loop that reduces edges with the highest edge centrality. We prove that our algorithm converges to the global optimum f^{OPT}, with a nearly-linear runtime in $|\boldsymbol{E}|$ per-iteration.

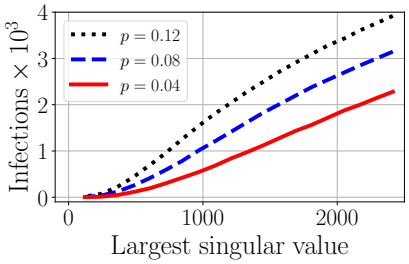


Figure 2: The number of infections strongly correlates with the largest singular value of the graph; for higher values of the latter, more infections will occur. p is the infection rate of the spreading process.

3.1 Edge centrality as gradient

To motivate our approach, we begin by reviewing the algorithm of Tong et al. [23], which considers controlling network diffusion by adding or removing edges. A central notion behind their approach is *edge centrality*, defined as the product of the *eigenvector scores* from both endpoints of an edge. More precisely, let X be any matrix. Let $\vec{u}_1$ and $\vec{v}_1$ be the left and right singular vector of X, corresponding to $\lambda_1(X)$. Then, for any edge $(i,j) \in \boldsymbol{E}$, the edge centrality score of this edge is given by $\vec{u}_1(i) \cdot \vec{v}_1(j)$, where $\vec{u}_1(i)$ denotes the i-th coordinate of $\vec{u}_1$ and $\vec{v}_1(j)$ denotes the j-th coordinate of $\vec{v}_1$.

The edge-weight reduction can be viewed as a "continuous relaxation" of edge removal since the weight of an edge can be reduced by a fraction. Interestingly, we show that the edge centrality scores are equal to the gradient of the largest eigenvalue of $W^\top W$ (up to scaling), $\lambda_1^2(W)$, over decreasing the edge weights. A more general statement holds for a generalized notion of edge centrality scores, including the largest-k singular values of W.

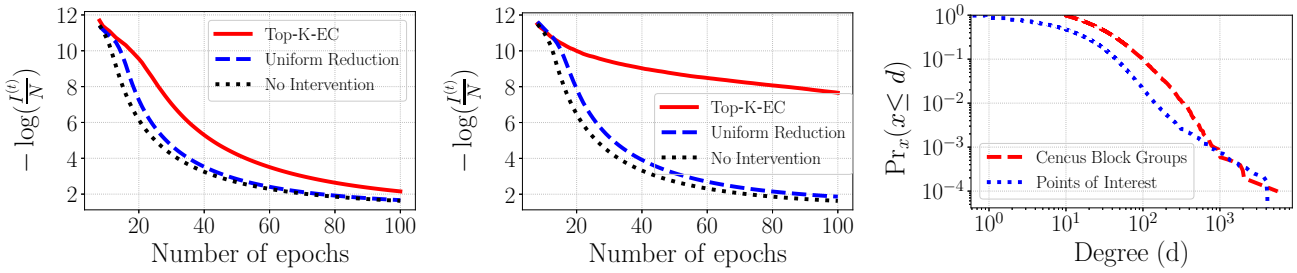
Lemma 3.1. *Assume that the singular values of X are all distinct. Then, the partial derivative of $\lambda_1^2(X)$ w.r.t. $X_{i,j}$ satisfies*

$$
(3.2)\qquad \frac{\partial \lambda_1^2(X)}{\partial X_{i,j}} = 2\lambda_1(X) \cdot \vec{u}_1(i) \cdot \vec{v}_1(j).
$$

More generally, for any $r \geq 1$, we have

$$
(3.3)\qquad \frac{\partial \Big(\sum_{k=1}^{r} \lambda_k^2(X) \Big)}{\partial X_{i,j}} = 2 \sum_{k=1}^{r} \lambda_k(X) \cdot \vec{u}_k(i) \cdot \vec{v}_k(j).
$$

The proof of Lemma 3.1 is presented in Appendix A. Given a weight matrix W of a network, we compute the edge centrality scores via the best rank-r approximation of W as $\tilde{W}_r = U_r D_r V_r^\top$. More precisely, D_r is an r by r square matrix, containing the largest r singular values of W; U_r is an n by r matrix, containing the left singular



(a) Budget is 1% of total edge weights. (b) Budget is 20% of total edge weights. (c) CCDF of CBGs and POIs.

Figure 3: Comparison of greedy selection and uniform edge-weight reduction on a mobility network. Top-K-EC is more effective in reducing the infected proportion throughout the SEIR model simulation. Moreover, the CBGs and the POIs in this network have a heavy-tailed degree distribution, which supports the intuition behind selecting edges by their centrality scores.

vectors corresponding to D_r; $V_r^\top$ is an r by m matrix, containing the right singular vectors corresponding to D_r. For every edge $(i,j) \in \boldsymbol{E}$, let $\tilde{W}_r(i,j)$ be the edge centrality score of this edge.

We validate that removing edges via top edge centrality scores effectively reduces infections. Figure 3 compares Top-K-EC (cf. Algorithm 1) to uniformly reducing every edge's weight by the same ratio. With Top-K-EC, the largest singular value dropped by 57.7% in (3a) and 96.8% in (3b). With uniform reduction, the drop goes down to 1% and 20%, respectively.

3.2 Global optimization via iterative greedy

We now develop the Frank-Wolfe edge centrality minimization algorithm, or Frank-Wolfe-EC, specified in Algorithm 1. The high-level idea is iteratively applying a greedy selection of edges with the highest edge centrality scores while recomputing the scores.

At every iteration t from 1 to T, let $M_t \in \mathbb{R}^{n\times m}$ be the currently modified weight matrix. Let $\nabla f(M_t)$ be the gradient of $f(M_t)$. The Frank-Wolfe algorithm [8, 16] computes a descent direction G_t for M_t, by minimizing the following matrix inner product

$$\langle \nabla f(M_t), G_t \rangle = \operatorname{Tr}\left[\nabla f(M_t)^\top G_t\right],$$

subject to the same constraints as problem (2.1):

$$\begin{aligned} G_t^\star \leftarrow \arg\min_X \quad & \langle X, \nabla f(M_t)\rangle \\ \text{s.t.} \quad & \sum_{(i,j)\in\boldsymbol{E}} \Big(W_{i,j} - X_{i,j}\Big) \le B \\ & 0 \le X_{i,j} \le W_{i,j}, \ \forall (i,j) \in \boldsymbol{E}, \\ & X_{i,j} = 0, \qquad \forall (i,j) \notin \boldsymbol{E}. \end{aligned} \tag{3.4}$$

The core of our approach is to show that the optimal descent direction, $G_t^\star$, is given by a greedy selection of edges based on their edge centrality scores. This procedure, Top-K-EdgeCentrality, or Top-K-EC, is specified as part of Algorithm 1. Let $\tilde{W}_r^{(t)}$ be the best rank-r approximation of M_t for every t. Let $(i_1,j_1),(i_2,j_2),\ldots,(i_m,j_m)$ be the edges in descending order of their edge centrality scores $\tilde{W}_r^{(t)}$, where $m = |\boldsymbol{E}|$ is the number of edges. Consider the first k edges whose total weight exceeds the reduction budget B:

$$\sum_{l=1}^{k-1} W_{i_l,j_l} < B \quad \text{and} \quad \sum_{l=1}^{k} W_{i_l,j_l} \ge B. \tag{3.5}$$

Then, the weight of the first $k-1$ edges is reduced to zero. The weight of the last edge decreases by $\sum_{l=1}^{k} W_{i_l,j_l} - B$. The following result proves that the above greedy selection yields an optimal solution of the inner optimization problem (3.4).

THEOREM 3.1. *The optimal solution $G_t^\star$ (cf. 3.4) is equal to the output of Top-K-EdgeCentrality$(W, B; M_t)$ in Algorithm 1.*

The proof can be found to Appendix A. After finding the descent direction $G_t^\star$, the next step of the Frank-Wolfe algorithm is setting a learning rate by minimizing $f\big((1-\eta_t)M_t + \eta_t G_t\big)$, for η_t in a range H between 0 and 1. Then, we update the weight matrix accordingly.

To recap, each iteration of Frank-Wolfe-EC computes a truncated rank-r SVD of a sparse matrix with at most m nonzeros and sorts an array of size m. The former requires a runtime complexity of $O(mr\log(m))$ (e.g., Theorem 1 of Musco and Musco [15]). The latter can be achieved with runtime $O(m\log(m))$. By comparison, the runtime complexity for solving a general linear program (i.e., problem (3.4)) is at least quadratic in the dimension of W.

Next, we examine the number of iterations that our algorithm needs to converge to f^{OPT}. A well-

Algorithm 1 Frank-Wolfe EC Minimization

Input: A graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ with weight matrix W; Budget B.
Parameters: Rank r; Iterations T; Range of learning rate H.
Output: A weight matrix M modified from W.

procedure FRANK-WOLFE-EDGECENTRALITY($W, B; T, H$)
  Let $M_0 = W$
  for $t = 0, 1, \ldots, T-1$ **do**
    $G_t^\star$ = TOP-K-EDGECENTRALITY($W, B; M_t$)
    Set η_t by minimizing $f\big((1-\eta_t)M_t + \eta_t G_t^\star\big)$ for $\eta_t \in H$
    $M_{t+1} = (1-\eta_t)M_t + \eta_t G_t^\star$
  end for
  if there is unused budget in M_T **then**
    $B' = B - \mathrm{sum}(W - M_T)$
    $M^\star$ = TOP-K-EDGECENTRALITY($M_T, B'; M_T$)
  end if
  return $M^\star$
end procedure

procedure TOP-K-EDGECENTRALITY($W, B; M$)
  Let $\tilde{M}_r$ be the best rank-r approximation of M
  Sort the edges in $\boldsymbol{E}$ by their edge centrality scores from $\tilde{M}_r$: let k be the number of edges obtained from equation 3.5 with weight matrix W
  Reduce the first $k-1$ edges' weight to zero and the last edge's weight by the remaining budget
  return the updated W
end procedure

Algorithm 2 Frank-Wolfe for Time-Varying Networks

Input: A sequence of graphs with weight matrix $\mathcal{W}$ in s steps.
Parameters: Same as the static case.
Output: A sequence of matrices $\mathcal{M}$ modified from $\mathcal{W}$.

procedure FRANK-WOLFE-TIMEVARYING($\mathcal{W}^{(t)}, B; T, H$)
  Let $\mathcal{M}_0 = \mathcal{W}$
  for $t = 0, 1, \ldots, T-1$ **do**
    $\mathcal{G}_t = \{G_t^{\star(i)}\}_{i=1}^s$ = TOP-K-TIMEVARYING($\mathcal{W}, B; \mathcal{M}_t$)
    Set η_t by minimizing $f\big((1-\eta_k)\mathcal{M}_t + \eta_t \boldsymbol{G}_k\big)$ for $\eta_k \in H$
    $\mathcal{M}_{t+1} = \{M_{t+1}^{(i)} = (1-\eta_t)M_t^{(i)} + \eta_t G_t^{\star(i)} : 1 \le i \le s\}$
  end for
  if there is unused budget in $\mathcal{M}_T$ **then**
    $B' = B - \sum_{i=1}^s \mathrm{sum}(W^{(i)} - M_T^{(i)})$
    $\mathcal{M}^\star$ = TOP-K-TIMEVARYING($\mathcal{M}_T, B'; \mathcal{M}_T$)
  end if
  return $\mathcal{M}^\star$
end procedure

procedure TOP-K-TIMEVARYING($\mathcal{W}, B; \mathcal{M}$)
  Let $\tilde{M}_r$ be the best rank-r approximation of $\prod_{i=1}^s M^{(i)}$
  Sort the edges in all of the s graphs by their edge centrality scores (cf. 3.9): let k be the number of edges obtained from equation 3.5 with weight matrix $W^{(1)}, \ldots, W^{(s)}$
  Reduce the first $k-1$ edges' weight to zero and the last edge's weight by the remaining budget
  return the updated $\mathcal{W}$
end procedure

established result (e.g., Jaggi [10]) is that for convex minimization problems, the Frank-Wolfe algorithm will converge to the global minimum under mild conditions. In the following, we will show that the objective $f(M)$ is convex. Based on that, we show that our Frank-Wolfe-EC algorithm will converge to the global minimum of problem (2.1), at a rate of $O(T^{-1})$ after T iterations.

THEOREM 3.2. *Let κ be the minimum of $\lambda_r(M_t) - \lambda_{r+1}(M_t)$ over $t = 0, 1, \ldots, T-1$. Assume that κ is strictly positive. Then, the following holds for M_T:*

$$f(M_T) - f^{\mathrm{OPT}} \le \frac{40\big(\sum_{(i,j)\in\boldsymbol{E}} W_{i,j}^2\big)\alpha_2}{T}, \tag{3.6}$$

where $\alpha_2 = \kappa^{-1}\big(\max_{t=1}^T \lambda_1(M_t)\sqrt{r}\big) + r + C$, for a fixed value $C > 0$.

This result guarantees that our algorithm will converge to global minimum solution under mild conditions. See Appendix A for the proof. While the constants $\alpha_1\alpha_2$ can be large as inherited from previous guarantees of the Frank-Wolfe algorithm, we find that the number of iterations T required for Frank-Wolfe-EC to converge is less than 30 in all cases.

3.3 Optimization on time-varying networks

Our study has focused on mitigating the spread in a static network. Another consideration is that network typologies evolve over time. Therefore, an important question is how to tackle such temporal evolution. Next we show how to extend our optimization algorithm to time-varying networks.

Let $\mathcal{W} = \{W^{(1)}, W^{(2)}, \ldots, W^{(s)}\}$ be the weight matrix of a sequence of graphs. We define the weight matrix that controls the epidemic spreading as $W = \prod_{i=1}^s W^{(i)}$ and minimize the sum of top-r eigenvalues of $W^\top W$ as in problem (2.1). Motivated by Prakash et al. [20] which connect epidemic spreading with time-varying networks, we consider an eigenvalue minimization problem on time-varying networks. Let $\mathcal{M} = \{M^{(1)}, M^{(2)}, \ldots, M^{(s)}\}$ be a sequence of modified weight matrices. We aim to find $\mathcal{M}$ that shrinks the largest eigenvalue(s) of their product:

$$\begin{aligned} \min_{\mathcal{M}} \quad & f(\mathcal{M}) := \sum_{k=1}^r \lambda_k^2\Big(\prod_{t=1}^s M^{(t)}\Big) \\ \text{s.t.} \quad & \sum_{t=1}^s \sum_{(i,j)\in\boldsymbol{E}^{(t)}} \Big(W_{i,j}^{(t)} - M_{i,j}^{(t)}\Big) \le B \\ & 0 \le M_{i,j}^{(t)} \le W_{i,j}^{(t)}, \ \forall (i,j) \in \boldsymbol{E}^{(t)}, t = 1, \ldots, s, \\ & M_{i,j}^{(t)} = 0, \qquad \forall (i,j) \notin \boldsymbol{E}^{(t)}, t = 1, \ldots, s. \end{aligned} \tag{3.7}$$

Above, $\boldsymbol{E}^{(t)}$ represents the set of edges in the t-th graph of the sequence. Following Lemma 3.1 and the chain rule, we show that the gradient of the largest eigenvalue of $W^\top W$ to $W^{(t)}$ is equal to $\tilde{W}_r$ multiplied with the rest

of weight matrices in the sequence except $W^{(t)}$:

$$\frac{\partial\Big(\sum_{k=1}^{r}\lambda_k^2(W)\Big)}{\partial W^{(t)}} = \frac{\partial\Big(\sum_{k=1}^{r}\lambda_k^2(W)\Big)}{\partial W}\frac{\partial W}{\partial W^{(t)}} \tag{3.8}$$

$$= 2U_r D_r V_r^\top \prod_{i\neq t} W^{(i)}$$

$$= 2\tilde{W}_r \prod_{i\neq t} W^{(i)}, \tag{3.9}$$

where $\tilde{W}_r = U_r D_r V_r^\top$ is the best rank-r approximation of W. The gradients can be treated as edge centrality scores on time-varying networks. For any edge $(i,j) \in \boldsymbol{E}^{(t)}$ in $W^{(t)}$, the edge centrality score for the edge is defined as the (i,j)-th entry of $\tilde{W}_r \prod_{i\neq t} W^{(i)}$ from (3.9). Then, we can develop a similar optimization algorithm on time-varying networks following the static case. The complete procedure can be found in Algorithm 2. Similar to Theorem 3.2, one can then prove that Algorithm 2 is guaranteed to converge to the optimum solution of problem (3.7) at the rate of $O(T^{-1})$ after T iterations! We defer the details to Appendix A.

4 Experiments

We evaluate our proposed approaches on a range of mobility networks and weighted graphs. Our experiments seek to address the following questions: First, does our proposed algorithm reduce the infections and the largest singular values well compared to methods from prior works? Second, what are the effects of each component in our algorithm, e.g., setting the rank r, running multiple iterations of greedy selection, and setting the budget? We present positive results to answer these three questions, validating the practical benefit of our algorithm.

4.1 Experimental setup

We follow the procedure described within Chang et al. [3] to construct the mobility networks. We briefly summarize the procedure and defer a comprehensive discussion to their paper. The construction uses the mobility patterns, geometry, and population census datasets from the SafeGraph platform. Additionally, we will use the reported cases of COVID-19 infections from The New York Times to calibrate the SEIR model. We describe the data sources in Section B. We generate the mobility networks based on monthly patterns from March 2, 2020, to May 10, 2020. We report the statistics of the mobility networks from each metropolitan statistical area (MSA) in Table 1. Overall, the mobility patterns cover 25,341 CBGs with over 65 million people and 147,638 POIs. The temporal mobility networks are constructed weekly during the same period mentioned above, including ten networks for every MSA.

Besides mobility networks, we consider three other weighted networks: (i) An Airport traffic network of airports in the world; (ii) A network of trust relationships among users on Advogato; (iii) A network of trust relationships among users on a Bitcoin platform. The statistics of these three networks are listed in Table 2.

Baseline methods. The experiments for spreading on a static network involve the following baseline methods: (1) K-EdgeDeletion: Delete a set of edges with the highest edge centrality scores according to the best rank-1 approximation of W [23]. (2) Edge weighted reduction: Reduce the weight of every edge by a ratio that is proportional to its weight. (3) Uniform reduction: Uniformly reduce the weight of every edge by the same fraction. (4) Max occupancy capping: Reduce the cumulative weights at each POI proportional to its max occupancy. (5) Capping by POI category: Cap the maximum occupancy of a particular category of POIs. The last three baselines are adapted from Chang et al. [3].

For the experiments on a sequence of temporal networks, we will only use Algorithm 1 to modify the network weight matrix while varying the budget allocation scheme. We consider the following list of allocation schemes: (1) First week only: Assign all the edge-weight reduction budget to the first week of the sequence. (2) Uniform allocation: Distribute the budget uniformly among every network in the sequence. (3) Exponential allocation: Distribute the budget proportional to $\exp(-t)$, decaying exponentially over time.

Implementation. In Algorithm 1, we search the rank parameter r in $[1, 50]$ and the number of iterations in $[5, 30]$. For each result reported in Section 4, we search the two hyper-parameters 50 times. For the range of learning rate H, we use 30 values from the range of $[10^{-3}, 10^{-1}]$ as H. In each iteration of the algorithm, we conduct a grid search over the learning rate range and choose the learning rate η_t that leads to the smallest object value $f\big((1-\eta_t)M_t + \eta_t G_t^\star\big)$. All the experiments are conducted on an AMD 24-Core CPU machine.

4.2 Experimental results

- **Results for reducing infection:** Figure 1 compares our algorithm to baseline intervention strategies on three weighted graphs. Overall, we see that Frank-Wolfe-EC reduces the number of infected nodes by **10.46**% more than baseline methods on average. Table 1 compares the total number of infected populations using different intervention strategies on eight networks. We note that Frank-Wolfe-EC—our iterative optimization method—outperforms other baselines by **30.17**% on average

Table 1: **Top:** Basic statistics for eight mobility networks constructed from SafeGraph data. **Middle:** Comparison of the total number of infected populations ($\times 10^3$). **Bottom:** Comparison of the largest singular value. We modify the edge weights using the strategy in each row. Results are averaged over 50 runs.

Graphs	AT	CH	DA	HO	MI	NY	PH	DC
Nodes	11,232	32,390	19,069	38,895	17,858	34,216	18,649	10,590
Edges	154,729	439,262	283,928	671,217	276,109	463,719	260,279	107,733
Avg. node weight	2,400	1,593	2,069	2,395	2,219	1,578	1,568	2,060
Avg. edge weight	5.258	4.659	4.921	4.951	4.833	4.749	4.864	4.848
Infected populations	**AT**	**CH**	**DA**	**HO**	**MI**	**NY**	**PH**	**DC**
No Intevention	48.4±3.1	1858.8±46.5	91.9±21.2	366.5±26.7	752.5±26.9	3146.5±21.4	492.5±20.7	41.1±2.2
Uniform Reduction	46.8±2.3	1762.0±64.3	84.7±11.1	312.1±26.3	671.3±23.7	2996.9±40.1	463.4±12.8	41.1±1.6
Weighted Reduction	43.2±2.7	782.5±86.9	66.1±3.2	194.6±18.5	43.4±12.6	1336.6±60.1	342.1±10.4	40.7±1.3
Max Occ. Capping	44.3±2.7	1741.1±65.3	82.3±8.6	315.3±33.3	675.5±26.5	2990.0±45.2	455.1±15.9	41.5±1.7
POI Category	46.1±3.2	1728.6±62.5	77.3±8.4	283.8±31.6	687.6±25.5	2950.2±38.4	458.3±17.6	41.0±1.4
K-EdgeDeletion	44.9±2.9	346.8±40.6	64.1±2.8	186.5±18.9	78.3±8.9	352.9±27.7	185.2±10.6	39.8±0.9
Top-k-EC	45.8±3.5	355.2±46.5	64.0±2.4	187.2±21.2	78.9±7.9	362.9±36.3	178.6±11.2	39.9±1.2
Ours	**40.4±1.7**	**166.2±16.1**	**62.1±2.4**	**86.4±10.9**	**8.5±2.5**	**301.4±88.4**	**129.2±13.4**	**39.2±1.0**
Largest singular value	**AT**	**CH**	**DA**	**HO**	**MI**	**NY**	**PH**	**DC**
No Intevention	5526.6	1296.2	2093.6	1467.7	555.7	2413.4	1203.2	1406.5
Uniform Reduction	5250.3	1231.4	1988.9	1394.3	527.9	2292.7	1143.0	1336.2
Weighted Reduction	1254.2	302.6	564.7	420.5	213.0	481.8	374.5	365.9
Max Occ. Capping.	5250.3	1231.4	1988.9	1394.3	527.9	2292.7	1143.0	1336.2
POI Category	5526.3	1295.8	2073.7	1467.6	555.6	2270.0	1202.8	1375.4
K-EdgeDeletion	1565.2	257.9	417.2	447.9	216.7	355.9	282.7	227.2
Top-k-EC	1565.2	257.9	417.1	447.9	216.7	355.9	282.7	226.8
Ours	**1191.2**	**125.9**	**308.4**	**235.6**	**169.5**	**197.3**	**190.1**	**188.1**

and up to **80.36**%. Additionally, we observe that the trend is consistent with Table 1 during the entire spreading process.

- **Drop in the largest singular value:** Figure 1 illustrates the largest singular value of the modified weight matrix of the three weighted graphs. Frank-Wolfe-EC reduces the largest singular value more than baselines by **11.42**% on average. Additionally, Table 1 reports the largest singular value of networks modified by each edge-weight reduction strategy on mobility networks. Frank-Wolfe-EC is **30.75**% more effective than the best baseline on average.
- **Results for time-varying networks:** We find that allocating the budget to every network proportional to their largest singular value outperforms all the other allocations. In particular, the number of infected populations is smaller by **39.82**% averaged over both Chicago and New York mobility networks.

4.3 Detailed analysis

- **Runtime report:** Across all eleven graphs, our approach converges within 30 iterations (or 17 on average). Each iteration requires an SVD step that takes less than 3 seconds. The other steps in each iteration requires less than 2.7 seconds. For larger graph instances, we run our method on seven graphs with the number of edges included: com-Orkut (117M), com-LiveJournal (34M), wiki-topcats (28M), web-BerkStan (7.6M), web-Google (5.1M), web-Stanford (2.3M), and web-NotreDame (1.4M) from the SNAP datasets. Figure 4 reports the runtime for one iteration of our algorithm. Notice that the runtime scales are nearly-linear with the number of edges. For example, our algorithm takes 4943 seconds on the largest graph with 117 million edges and 3 million nodes. These results show that our algorithm runs efficiently on large-scale graphs.
- **Benefit of choosing ranks:** Recall that our algorithm requires specifying rank r–the number of top singular values–in Equation 2.1. We hypothesize that varying the rank r would lead to different intervention results. We ablate the performance of our algorithm by using different r in a range of $[1, 50]$. The results show that the performance of the best choice r outperforms using $r = 1$ by 40.27% averaged over all networks. This result justifies our formulation of the network intervention problem as an optimization for the sum of largest-r singular values instead of only the largest single value.
- **Benefit of being iterative:** The greedy selection algorithm Top-k-EC can be viewed as a special case of Frank-Wolfe-EC with $T = 1$. Notice that the iterative approach is necessary to get the observed empirical performance. In Table 1, Frank-Wolfe-EC outperforms Top-k-EC by **31.41**% on average, and the largest singular value is reduced by **33.09**% more in Table 1.

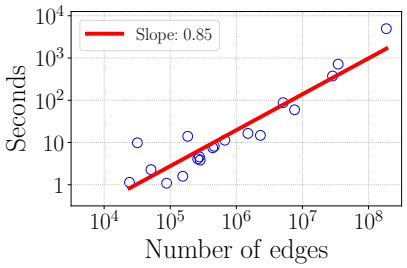


Figure 4: Runtime of Frank-Wolfe-EC in log-log scale for one iteration. The number of edges ranges from 10^4 to 10^8 and the number of nodes ranges from 10^3 to 10^6.

- **Results with different budgets:** We have also observed similar results by varying the budget for mobility reduction. We vary the budget from 1% to 20% using the New York mobility network. We find that our algorithm outperforms the baseline methods consistently using different budget levels, similarly for the largest singular value. Interestingly, when the level of budget is small (e.g., 1%), Frank-Wolfe-EC reduces the largest singular value more significantly than baseline methods.

5 Discussion and Related Work

There is an extensive body of work studying diffusion control on networks. Besides epidemic spreading, network diffusion is also widely studied in social and information networks [14, 9]. We summarize the most relevant research to ours while referring the reader to Pastor-Satorras et al. [18]'s survey for references. A key result in the epidemics literature is that the largest eigenvalue of the adjacency matrix (a.k.a. the spectral radius, denoted as λ_1) characterizes the epidemic threshold for more than 25 propagation models [19].

An important implication of this result is that the epidemic dies out if λ_1 decreases, and this is the basis of many works on epidemic control [24, 13, 6]. Because eigen-optimization problems via edge additions or deletions are NP-hard [23, 12], approximation algorithms are used for diffusion control. A key approach is a greedy algorithm based on some notion of centrality information in the network [17]. There is a connection between this problem and submodular optimization, leading to provable approximation ratios for the greedy algorithm [21, 5].

Besides, there is a line of work studying diffusion control under the name of the Firefighter problem in approximation algorithms [1]. Finally, there are studies on the design of vaccine distribution for pandemic control [26, 22]. These works and their analysis do not lead to direct bounds for weighted networks, which is the focus of our setting.

Besides SEIR compartmental models, there are other ways to model network spreading processes. The critical algorithmic insight of our work is to strategically restrict mobility using spectral properties of a network. While our study focuses on the SEIR model and applications to mobility-based modeling, it is conceivable that our algorithmic insights might apply to different epidemic models and different data-driven modeling of the pandemic. For example, an interesting research question is to examine our approach with different epidemic models such as SIS and SIR. Besides, another interesting question is to study node deletion as the intervention. In the context of mobility networks, reducing the weight of a node means reducing a fraction of the node's mobility. Lastly, it is conceivable that one can combine our approach with existing techniques to better deal with temporal dynamics [20]. These questions are left for future work.

6 Conclusion and Future Work

We studied the problem of controlling the diffusion of an epidemic on weighted networks via reducing edge weights. This problem is motivated by recent studies of mobility-based modeling for the COVID-19. We introduced a constrained optimization problem to reduce edge weights that minimize the network's largest singular values. We designed an iterative procedure for finding the global minimum of the above optimization problem. Our algorithm is guaranteed to converge to the global optimum. Additionally, we theoretically proved and empirically observed that each iteration only requires a nearly-linear runtime in the size of the network. Our experiments demonstrated the superiority of our approaches. Our work highlights the existence of spectral properties in mobility networks and uses them to design practical intervention algorithms.

We remark that implementing this scheme requires knowing the network in the sequence. When such information is not available, one needs first to estimate this information. This is left for future work.

References

[1] E. Anshelevich, D. Chakrabarty, A. Hate, and C. Swamy. "Approximation algorithms for the firefighter problem: Cuts over time and submodularity". In: *ISAAC*. Springer, 2009.

[2] D. Chakrabarti, Y. Wang, C Wang, J. Leskovec, and C. Faloutsos. "Epidemic thresholds in real networks". In: *TISSEC*. 2008.

[3] S. Chang, E. Pierson, P. W. Koh, J. Gerardin, B. Redbird, D. Grusky, and J. Leskovec. "Mobility network models of COVID-19 explain inequities and inform reopening". In: *Nature*. Nature Publishing Group, 2021.

[4] S. Chang, M. Wilson, B. Lewis, Z. Mehrab, K. Dudakiya, E. Pierson, P. W. Koh, J. Gerardin, B. Redbird, D. Grusky, et al. "Supporting COVID-19 policy response with large-scale mobility-based modeling". In: *KDD*. ACM, 2021.

[5] C. Chen, R. Peng, L. Ying, and H. Tong. "Network connectivity optimization: Fundamental limits and effective algorithms". In: *KDD*. 2018.

[6] C. Chen, H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. "Eigen-optimization on large graphs by edge manipulation". In: *TKDD*. ACM, 2016.

[7] C. Chen, H. Tong, B. A. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau. "Node immunization on large graphs: Theory and algorithms". In: *TKDE*. IEEE, 2015.

[8] M. Frank and P. Wolfe. "An algorithm for quadratic programming". In: *Naval Research Logistics Quarterly*. Wiley Subscription Services, 1956.

[9] A. Goel, K. Munagala, A. Sharma, and H. Zhang. "A note on modeling retweet cascades on Twitter". In: *WAW*. 2015.

[10] M. Jaggi. "Revisiting Frank-Wolfe: Projection-free sparse convex optimization". In: *ICML*. 2013.

[11] R. Karp. "Reducibility among combinatorial problems". In: *Complexity of Computer Computations*. 1972.

[12] E. B. Khalil, B. Dilkina, and L. Song. "Scalable diffusion-aware optimization of network topology". In: *KDD*. 2014.

[13] L. Le, T. Eliassi-Rad, and H. Tong. "MET: A fast algorithm for minimizing propagation in large graphs with small eigen-gaps". In: *ICDM*. 2015.

[14] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. "Rise and fall patterns of information diffusion: model and implications". In: *KDD*. 2012.

[15] C. Musco and C. Musco. "Randomized block krylov methods for stronger and faster approximate singular value decomposition". In: *NIPS*. 2015.

[16] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[17] N Parotsidis, E. Pitoura, and P. Tsaparas. "Centrality-aware link recommendations". In: *WSDM*. ACM, 2016.

[18] R. Pastor-Satorras, C. Castellano, P. Mieghem, and A. Vespignani. "Epidemic processes in complex networks". In: *Rev. of Mod. Physics* (2015).

[19] B. A. Prakash, D. Chakrabarti, N. Valler, M. Faloutsos, and C. Faloutsos. "Threshold conditions for arbitrary cascade models on arbitrary networks". In: *Knowledge and information systems* (2012).

[20] B. A. Prakash, H. Tong, N. Valler, M. Faloutsos, and C. Faloutsos. "Virus propagation on time-varying networks: Theory and immunization algorithms". In: *ECML PKDD*. Springer. 2010.

[21] S. Saha, A. Adiga, B. A. Prakash, and A. K. Vullikanti. "Approximation algorithms for reducing the spectral radius to control epidemic spread". In: *SDM*. 2015.

[22] P. Sambaturu, B. Adhikari, B. A. Prakash, S. Venkatramanan, and A. Vullikanti. "Designing effective and practical interventions to contain epidemics". In: *AAMAS*. Springer, 2020.

[23] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. "Gelling, and melting, large graphs by edge manipulation". In: *CIKM*. 2012.

[24] P. Van Mieghem, D. Stevanović, F. Kuipers, C. Li, R. Van De Bovenkamp, D. Liu, and H. Wang. "Decreasing the spectral radius of a graph by link removals". In: *Physical Review E*. APS, 2011.

[25] T. Vu, E. Chunikhina, and R. Raich. "Perturbation expansions and error bounds for the truncated singular value decomposition". In: *Linear Algebra and its Applications*. Elsevier, 2021.

[26] Y. Zhang and B. A. Prakash. "Scalable vaccine distribution in large graphs given uncertain data". In: *CIKM*. 2014.

A Proofs

Proof of Lemma 3.1. Consider a singular value λ_k of X, for any k. Let $\vec{u}_k$ and $\vec{v}_k$ be the left and right singular vectors of X corresponding to λ_k, respectively. By the chain rule, it suffices to show that $\frac{\partial \lambda_k(X)}{\partial X_{i,j}} = \vec{u}_k(i) \cdot \vec{v}_k(j)$. First, we have $\vec{u}_k^\top X = \lambda_k \vec{v}_k^\top$. We differentiate over X on both sides of the above equation:

$$\mathrm{d}(\vec{u}_k^\top)X + \vec{u}_k^\top \,\mathrm{d}(X) = \mathrm{d}(\lambda_k)\vec{v}_k^\top + \lambda_k \,\mathrm{d}(\vec{v}_k^\top). \tag{A.1}$$

Since $\vec{v}_k$ is a unit length vector,

$$\mathrm{d}(\|\vec{v}_k\|^2) = 2\langle \vec{v}_k, \mathrm{d}(\vec{v}_k)\rangle = 2\,\mathrm{d}(\vec{v}_k^\top)\vec{v}_k = 0. \tag{A.2}$$

Thus, by multiplying both sides of equation (A.1) with $\vec{v}_k$, we get

$$\mathrm{d}(\vec{u}_k^\top)X\vec{v}_k + \vec{u}_k^\top \,\mathrm{d}(X)\vec{v}_k = \mathrm{d}(\lambda_k)\vec{v}_k^\top \vec{v}_k + \lambda_k \,\mathrm{d}(\vec{v}_k^\top)\vec{v}_k, \tag{A.3}$$

which is equal to $\mathrm{d}(\lambda_k)$ since equation (A.2) holds and v_k is a unit length vector. Looking at equation (A.3), we observe

$$\mathrm{d}(\vec{u}_k^\top)X\vec{v}_k = \mathrm{d}(\vec{u}_k^\top)\lambda_k\vec{u}_k = \lambda_k \,\mathrm{d}(\vec{u}_k^\top)\vec{u}_k = 0, \tag{A.4}$$

where the last step follows similarly to equation (A.2), since $\vec{u}_k$ is also a unit length vector. In summary, we have shown $\vec{u}_k^\top \,\mathrm{d}(X)\vec{v}_k = \mathrm{d}(\lambda_k)$. This implies that the derivative of λ_k over $X_{i,j}$ is equal to $\vec{u}_k(i) \cdot \vec{v}_k(j)$. Since this holds for any k, we thus conclude that equations (3.2) and (3.3) are both true. □

Proof of Theorem 3.1. By Lemma 3.1, for every edge $(i,j) \in \boldsymbol{E}$, the gradient of $f(M_t)$ over this edge is given by the *edge centrality* scores. Since $X_{i,j} = 0$ for any $(i,j) \notin \boldsymbol{E}$, the optimization objective is:

$$\langle X, \nabla f(M)\rangle = \sum_{(i,j)\in \boldsymbol{E}} 2X_{i,j}\Big(\sum_{k=1}^{r} \lambda_k \cdot \vec{u}_k(i) \cdot \vec{v}_k(j)\Big). \tag{A.5}$$

Above, each variable $X_{i,j}$ is multiplied precisely by the edge centrality of the edge (i,j) (cf. line (15)). Consider minimizing the equivalent objective (A.5) with the constrains of Problem (3.4). The minimizer, $G_t^\star$, is achieved by reducing the weight of the edges with the highest edge centrality to zero until the budget B gets exhausted. This is precisely the procedure of Top-K-EC from lines (15)-(17). Thus, we have proved this result. □

Proof of Theorem 3.2. We complete the convergence analysis of our algorithm. First, we show that the objective function $f(M)$ is convex in M. Second, we invoke the result of Jaggi [10], specifically Lemma 7 and Theorem 1, which show that as long as the gradient $\nabla f(M)$ is Lipschitz-continuous and the constraint set has bounded diameter, the Frank-Wolfe algorithm will converge to the optimum at a rate of $O(\frac{1}{t})$ after t iterations.

We first show that the sum of top singular values $g(M) = \sum_{k=1}^{r} \lambda_k(M)$ is convex. With the variational characterization of singular values, $g(M)$ is equal to

$$g(M) = \max_{U^\top U = V^\top V = \mathrm{Id}_r:\ U\in\mathbb{R}^{n\times r}, V\in\mathbb{R}^{m\times r}} \langle UV^\top, M\rangle. \tag{A.6}$$

Thus, for any n by m matrix M_1, M_2, and any $\alpha \in [0,1]$, let $\tilde{U}$ and $\tilde{V}$ be the maximizer of the above for $f\big(\alpha M_1 + (1-\alpha)M_2\big)$. Therefore,

$$\begin{aligned} g\big(\alpha M_1 + (1-\alpha)M_2\big) &= \langle \tilde{U}\tilde{V}^\top, \alpha M_1 + (1-\alpha)M_2\rangle \\ &\le \alpha\langle \tilde{U}\tilde{V}^\top, M_1\rangle + (1-\alpha)\langle \tilde{U}\tilde{V}^\top, M_2\rangle \\ &\le \alpha g(M_1) + (1-\alpha)g(M_2), \end{aligned}$$

which implies that $g(M)$ is convex. Next, we show that $f(M)$ is convex. For any $\alpha \in [0,1]$,

$$\begin{aligned} &f(\alpha M_1 + (1-\alpha)M_2) = g\Big((\alpha M_1 + (1-\alpha)M_2)^T(\alpha M_1 + (1-\alpha)M_2)\Big) \\ &\le \alpha^2 g(M_1^\top M_1) + (1-\alpha)^2 g(M_2^\top M_2) + 2\alpha(1-\alpha)g(M_1^\top M_2). \end{aligned}$$

Let $\tilde{U}$ and $\tilde{V}$ be the maximizer of (A.6) for $M_1^\top M_2$. We have

$$\begin{aligned}
&2g(M_1^\top M_2) = 2\langle \tilde{U}\tilde{V}^\top, M_1^\top M_2\rangle = 2\langle M_1\tilde{U}, M_2\tilde{V}\rangle \\
\leq &\left\|M_1\tilde{U}\right\|_F^2 + \left\|M_2\tilde{V}\right\|_F^2 = \langle M_1^\top M_1, \tilde{U}\tilde{U}^\top\rangle + \langle M_2^\top M_2, \tilde{V}\tilde{V}^\top\rangle \\
\leq &g(M_1^\top M_1) + g(M_2^\top M_2).
\end{aligned}$$

Therefore, $f(\alpha M_1 + (1-\alpha)M_2)$ is less than $\alpha \cdot g(M_1^\top M_1) = \alpha \cdot f(M_1)$ plus $(1-\alpha)\cdot g(M_2^\top M_2) = (1-\alpha)\cdot f(M_2)$.

Second, we verify that $\nabla f(M)$ is α_2 Lipschitz continuous in the Frobenius norm. The proof is based on matrix perturbation bounds. Let $\tilde{M} = M + E$ be a perturbation of M. Let $M_r = U_r D_r V_r^\top$ be the top-r SVD of M. Let μ_1 be the largest singular value of M. Let $\tilde{M}_r = \tilde{U}_r\tilde{D}_r\tilde{V}_r^\top$ be the top-r SVD of $\tilde{M}$. First, consider $\|E\|_2 \leq \kappa/2$. By matrix perturbation bounds on the truncated SVD of a matrix (e.g., Theorem 1 of Vu et al. [25]; the condition is satisfied since κ is the spectral gap between the r-th and $(r+1)$-th largest singular values), we have

$$\|M_r - \tilde{M}_r\|_F^2 \leq 2\|E\|_F^2 + \frac{4\lambda_1^2}{\kappa^2}\|E\|_F^2 + C\|E\|_F^2.$$

When $\|E\|_2 \geq \kappa/2$, notice that

$$\begin{aligned}
\|M_r - \tilde{M}_r\|_F^2 &= \|U_r D_r V_r^\top - \tilde{U}_r\tilde{D}_r\tilde{V}_r^\top\|_F^2 \\
&\leq 2\|D_r\|_F^2 + 2\|\tilde{D}_r\|_F^2 \\
&\leq 2r\lambda_1^2 + 2r(\lambda_1 + \|E\|_2)^2,
\end{aligned}$$

which is at most $2r(3\lambda_1^2 + 2\|E\|_2^2)$. The step above uses the Weyl's Theorem that $\|D_r - \tilde{D}_r\|_2 \leq \|E\|_2$. Taken together, we conclude that $\nabla f(M)$ must be

$$\sqrt{\max\left(2 + \frac{4\lambda_1^2}{\kappa^2} + C, \frac{24r\cdot\lambda_1^2}{\kappa^2} + 4r\right)}$$

Lipschitz continuous. Lastly, the diameter of the constraint set is at most $\sqrt{\sum_{(i,j)\in \boldsymbol{E}} W_{i,j}^2}$, since for every $(i,j) \in \boldsymbol{E}$, the search space is bounded between 0 and $W_{i,j}$. Taken together, we have proved that: $f(M)$ is convex, $\nabla f(M)$ is α_2 Lipschitz continuous, and the diameter of the constrained space of problem (2.1) is $\sqrt{\alpha_1/8}$. Using Lemma 7 and Theorem 1 of Jaggi [10], the proof is complete. □

B Additional Experimental Setup

At every time t,

- $S^{(t)}$ denotes the set of susceptible nodes at time t. A node may get exposed if its incoming neighbors are infectious. The probability depends on the edge weights and the virus transmission rate.
- $E^{(t)}$ denotes the nodes who have been exposed to the virus but who are not infectious at time t. In expectation, a node remains exposed for δ_E periods.
- $I^{(t)}$ denotes the nodes who are infectious at time t. Each node remains infectious for δ_I periods in expectation.
- $R^{(t)}$ denotes the nodes who have recovered at time t.

Data availability. The mobility data is freely available to researchers, non-profit organizations, and governments through the SafeGraph COVID-19 Data Consortium.[1] The New York Times COVID-19-data is publicly available online.[2] Links to the other weighted networks are included in the references.

[1] https://www.safegraph.com/covid-19-data-consortium
[2] https://github.com/nytimes/covid-19-data

Table 2: Basic statistics for three weighted graphs.

	Airport	Advogato	Bitcoin
Number of Nodes	7,977	6,541	3,783
Number of Edges	30,501	51,127	24,186
Average edge weight	1.45	0.83	1.46

The construction of mobility networks requires the following data sources: (i) Mobility patterns from the Monthly Pattern[3] and Weekly Pattern datasets [4]. (ii) The geometry dataset[5], and (iii) The Open Census Dataset[6]. We also consider three weighted networks in the experiments: Airport[7], Adavogato[8], and Bitcoin[9].

Simulation setup. For the experiments concerning mobility networks, we follow the procedures of Chang et al. [3] to simulate a metapopulation SEIR model in each network. We simulate 100 epochs to be consistent with the simulation of Chang et al. [3]. The results are consistent throughout the simulation. We compare the Frank-Wolfe-EC algorithm with baseline methods using an edge-weight reduction budget as 5% of the total edge weights. Results of using other budget amounts are consistent.

We simulate an SEIR model on each graph for the other weighted networks. To avoid infecting all the graph nodes, we simulate for 50 epochs. We use a slightly higher edge-weight reduction budget as 20% of the total edge weights because the average edge weight in these three graphs is smaller than the mobility networks.

For the temporal mobility networks experiments, we simulate the metapopulation SEIR model on a sequence of ten networks for 70 epochs or seven epochs for every network. We set the edge-weight reduction budget as 5% of the total edge weights of the sequence and allocate the budget to each network by allocation strategies described above.

We calibrate the parameters of the SEIR model following the method presented in Chang et al. [3]. Specifically, we calibrate the following parameters: (i) the transmission constant in POIs, ψ; (ii) the base transmission rate, β_{base}; and (iii) the ratio of initial exposed people, p_0. We use grid search to find the parameters with the smallest root mean square error compared to the reported number of infected cases. We calibrate an SEIR model for every MSA independently. For the weighted graphs, we use a transmission rate $\beta_{\text{Base}} = 0.05$ and a initial exposed ratio $p_0 = 0.01$.

C Model validation

We compare the predicted cases of our simulated SEIR model with the reported cases from New York Times COVID-19 data. The root mean squared error of all the epochs is 295.17 averaged over eight mobility networks. The error is within 3% compared to the overall infected population which is at the scale of 10^4. These results reaffirm the finding of Chang et al. [3].

D Discussion and Future work

We mention two questions for future work. First, although we demonstrated that choosing the rank r larger than one yields superior empirical performance, theoretically justifying the reduction of $f(M)$ to reducing epidemic threading is still lacking, and we leave it for future work. Second, while our algorithm achieved strong empirical performance on mobility networks, theoretically analyzing it within the setting of mobility-based modeling remains an interesting question. In particular, we are not aware of any result on the epidemic threshold of the metapopulation SEIR model. More broadly, we hope our work inspires further algorithmic and theoretical studies on epidemic spreads, which could in turn contribute to the ongoing discussion of pandemic prevention.

[3] https://docs.safegraph.com/docs/monthly-patterns
[4] https://docs.safegraph.com/docs/weekly-patterns
[5] https://docs.safegraph.com/docs/geometry-data
[6] https://docs.safegraph.com/docs/open-census-data
[7] http://opsahl.co.uk/tnet/datasets/openflights.txt
[8] https://downloads.skewed.de/mirror/konect.cc/files/download.tsv.advogato.tar.bz2
[9] http://snap.stanford.edu/data/soc-sign-bitcoinalpha.html