Graph Design via Convex Optimization: Online and Distributed Perspectives

De Meng

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2017

Reading Committee:

Professor Maryam Fazel, Chair

Professor Mehran Mesbahi

Professor Baosen Zhang

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Graph Design via Convex Optimization: Online and Distributed Perspectives

De Meng

Chair of the Supervisory Committee:
Professor Maryam Fazel
Electrical Engineering

Network and graph have long been natural abstraction of relations in a variety of applications, e.g. transportation, power system, social network, communication, electrical circuit, etc. As a large number of computation and optimization problems are naturally defined on graphs, graph structures not only enable important properties of these problems, but also leads to highly efficient distributed and online algorithms. For example, graph separability enables the parallelism for computation and operation as well as limits the size of local problems. More interestingly, graphs can be defined and constructed in order to take best advantage of those problem properties. This dissertation focuses on graph structure and design in newly proposed optimization problems, which establish a bridge between graph properties and optimization problem properties.

We first study a new optimization problem called *Geodesic Distance Maximization Problem (GDMP)*. Given a graph with fixed edge weights, finding the shortest path, also known as the geodesic, between two nodes is a well-studied network flow problem. We introduce the Geodesic Distance Maximization Problem (GDMP): the problem of finding the edge weights that maximize the length of the geodesic subject to convex constraints on the weights. We show that GDMP is a convex optimization problem for a wide class of flow costs, and provide a physical interpretation using the dual. We present applications of the GDMP in various fields, including optical lens design, network interdiction, and resource allocation in the con-

trol of forest fires. We develop an Alternating Direction Method of Multipliers (ADMM) by exploiting specific problem structures to solve large-scale GDMP, and demonstrate its effectiveness in numerical examples.

We then turn our attention to distributed optimization on graph with only local communication. Distributed optimization arises in a variety of applications, e.g. distributed tracking and localization, estimation problems in sensor networks, multi-agent coordination. Distributed optimization aims to optimize a global objective function formed by summation of coupled local functions over a graph via only local communication and computation. We developed a weighted proximal ADMM for distributed optimization using graph structure. This fully distributed, single-loop algorithm allows simultaneous updates and can be viewed as a generalization of existing algorithms. More importantly, we achieve faster convergence by jointly designing graph weights and algorithm parameters.

Finally, we propose a new problem on networks called *Online Network Formation Problem*: starting with a base graph and a set of candidate edges, at each round of the game, player one first chooses a candidate edge and reveals it to player two, then player two decides whether to accept it; player two can only accept limited number of edges and make online decisions with the goal to achieve the best properties of the synthesized network. The network properties considered include the number of spanning trees, algebraic connectivity and total effective resistance. These network formation games arise in a variety of cooperative multiagent systems. We propose a primal-dual algorithm framework for the general online network formation game, and analyze the algorithm performance by the competitive ratio and regret.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I wish to express my gratitude to my advisor, Professor Maryam Fazel, for her mentorship and support, to Professors Aleksandr Aravkin, James Burke, Mehran Mesbahi, Baosen Zhang, and Dr. Lin Xiao for kindly serving on my thesis committee, to Professors Stephen Boyd, Pablo Parrilo and Ting Kei Pong with whom I have had the privilege of working, and to my family and friends whose support has been absolutely invaluable over the years.

DEDICATION

to my mom and dad.

Chapter 1

INTRODUCTION

Graphs have long been natural abstraction of relations in a variety of applications, e.g. transportation, social network, communication, statistics, electrical circuit, etc. It is typical that variables and functions are defined for each node, and each node only connects to its neighbors. This graph structure imposes natural coupling relations between variables: each variable is only connected/coupled with variables associated with immediate neighbors, but separated from others. The implication of this structure can be both positive and negative. On one hand, the separability enables the parallelism for computation and operation, as well as limits the size of local problems. On the other hand, it limits the communication to be local while global communication is often necessary in achieving global objectives, such as global averaging, consensus and exchange.

Chapter 2 studies a new optimization problem called *Geodesic Distance Maximization Problem (GDMP)*, particularly the separability structure over the graph which is exploited in developing the algorithm. Given a graph with fixed edge weights, finding the shortest path, also known as the geodesic, between two nodes is a well-studied network flow problem. We introduce the Geodesic Distance Maximization Problem (GDMP), i.e., the problem of finding the edge weights that maximize the length of the geodesic, subject to convex constraints on the weights. We show that GDMP is a convex optimization problem for a wide class of flow costs, and provide a physical interpretation using the dual. We present applications of the GDMP in various fields, including optical lens design, network interdiction, and resource allocation in the control of forest fires. For the case of propagation on a regular grid, the problem can be cast as a Second-order Cone Program (SOCP); however standard SOCP solvers fail to scale to the large grid sizes of interest. We develop an Alternating Direction

Method of Multipliers (ADMM) by exploiting specific problem structure to solve large-scale GDMP, and demonstrate its effectiveness in numerical examples. We discuss extensions of the problem to upwind discretization and to anisotropic propagation, which can also be handled by ADMM.

Chapter 3 is about distributed optimization on graph with only local communication. Distributed optimization arises in a variety of applications, e.g. distributed tracking and localization, estimation problems in sensor networks, multi-agent coordination. Distributed optimization aims to optimize a global objective function formed by summation of coupled local functions over a graph via only local communication and computation. We developed a weighted proximal Alternating Direction Method of Multipliers (ADMM) for distributed optimization using graph structure. We give a bound on the rate of convergence of the algorithm in terms of the graph parameters. This fully distributed, single-loop algorithm allows simultaneous updates and can be viewed as a generalization of existing algorithms. More importantly, we achieve faster convergence by jointly designing graph weights and algorithm parameters. Numerical examples demonstrate that designing the graph weights and proximal term can considerably improve the algorithm performance.

Chapter 4 introduces an *Online Network Formation Problem*: starting with a base graph and a set of candidate edges, at each round of the game, player one first chooses a candidate edge and reveals it to player two, then player two decides whether to accept it; player two can only accept limited number of edges and make online decisions with the goal to achieve the best properties of the synthesized network. The network properties considered include the number of spanning trees, algebraic connectivity and total effective resistance. These network formation games arise in a variety of cooperative multiagent systems, such as robotic networks establishing a secured network in a changing uncertain environment, or individuals forming teams in social networks. We propose a primal-dual algorithm framework for the general online network formation game. In a nutshell, at each round the algorithm computes the dual solution using all information from previous rounds of the game, and decides the weight on the new edge based on the complementary slackness. We derive the competitive

ratio and regret to evaluate the performance of the algorithm.

1.1 Graph background

A graph G consists of a node set V and an edge set E, where an edge is a pair of distinct nodes of G. The number of nodes is $|V| = n$ and the number of edges is $|E| = m$. G is a directed graph if each edge has a direction. Denote (i, j) a directed edge from node i to node j. G is an undirected graph if there is no direction for edges, and an undirected edge is denoted as $\{i, j\}$. In an undirected graph, node j is said to be a *neighbor* of node i if $\{i, j\} \in E$. Denote $\mathcal{N}(i)$ the set of neighbors of node i and assume that there is no self loop.

The *weight/adjacency* matrix W of an undirected graph G is an $n \times n$ symmetric matrix with $w_{ij} > 0$ if $\{i, j\} \in E$ and $w_{ij} = 0$ otherwise. The *degree* d_i of each node i is defined as $d_i = \sum_{j \in V} w_{ij}$. Assume that $d_i > 0$ and denote $D = \mathbf{diag}\{d_1, \ldots, d_n\}$. The graph *volume* is defined as $\mathrm{vol}(G) = \sum_{i \in V} d_i = \mathbf{1}^T W \mathbf{1}$. The *incidence* matrix $\mathcal{I}$ of a weighted *directed* graph G is an $m \times n$ matrix with $\mathcal{I}_{(i,j),i} = \sqrt{w_{ij}}$ and $\mathcal{I}_{(i,j),j} = -\sqrt{w_{ij}}$ for each $(i, j) \in E$ and 0 otherwise. For an *undirected* graph, we can replace each undirected edge $\{i, j\}$ with two directed edges (i, j) and (j, i) with *opposite* directions and the same weight $w_{ij}/2$, where w_{ij} is the original weight for undirected edge $\{i, j\}$. Then the *incidence* matrix $\mathcal{I}$ of a weighted undirected graph can be defined as a $2m \times n$ matrix with $\mathcal{I}_{(i,j),i} = \sqrt{\frac{w_{ij}}{2}}$, $\mathcal{I}_{(i,j),j} = -\sqrt{\frac{w_{ij}}{2}}$, $\mathcal{I}_{i,(j,i)} = -\sqrt{\frac{w_{ij}}{2}}$, $\mathcal{I}_{j,(j,i)} = \sqrt{\frac{w_{ij}}{2}}$ for each $\{i, j\} \in E$ and 0 otherwise. The *graph Laplacian* of a weighted undirected graph G is defined as $\mathcal{L}(G) = D - W$. The graph Laplacian can also be defined using the incidence matrix defined above

$$\mathcal{L} = \mathcal{I}^T \mathcal{I}. \tag{1.1}$$

$\mathcal{L}$ is symmetric and positive semidefinite. Denote the eigenvalue decomposition of the graph Laplacian, $\mathcal{L} = U \Lambda U^T$, $\Lambda = \mathbf{diag}\{\lambda_1, \ldots, \lambda_n\}$, $\lambda_1 \leq \ldots \leq \lambda_n$. We recall very important property of $\mathcal{I}$ and $\mathcal{L}$: if the graph is connected, the nullspaces of $\mathcal{I}$ and $\mathcal{L}$ are both spanned by the all-one vector $\mathbf{1}$.

Chapter 2

GEODESIC DISTANCE MAXIMIZATION PROBLEM

2.1 Introduction

We begin by reviewing the classic shortest path problem in a graph, where the weights on the edges of the graph define a metric for measuring distances. We then consider our problem of interest: allocating weights to edges in order to *maximize the length of the shortest path*. One simple application of this problem is the classic *network interdiction*: the evader, e.g. smuggler, transverses the network along the shortest path, while the interdictor, e.g. police, maximizes this shortest path by allocating weights in order to intercept the smuggler. Beyond this simple application, our problem of interest actually arises in a number of applications in various fields such as physics, transportation, forest fire and disease control, and etc, which will be surveyed in Section 2.2.

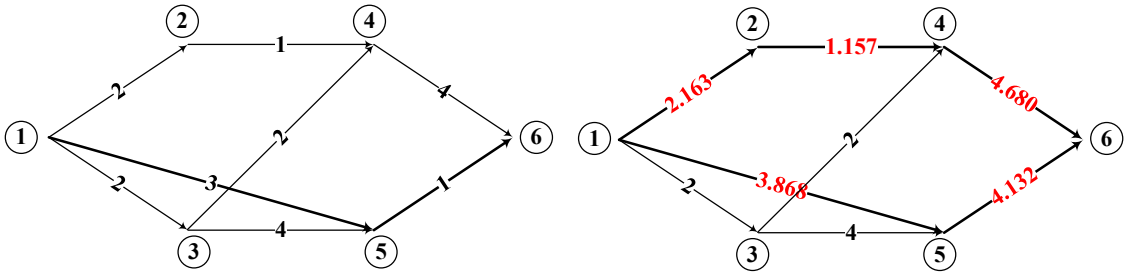
Let us first examine the network flow formulation of the shortest path problem. Let $G = (V, E)$ be a directed graph, where $V = \{1, \ldots, n\}$ is the set of vertices and $E \subseteq V \times V$ with $|E| = m$ is the set of edges. Assign a flow $f_{ij} \in \mathbf{R}$ to each edge $(i, j) \in E$ and let $f \in \mathbf{R}^m$ denote the vector of all flows. Assume flow is conserved at each vertex, i.e.,

$$\sum_{j:(j,i)\in E} f_{ji} - \sum_{j:(i,j)\in E} f_{ij} = \delta_i, \quad i \in V, \tag{2.1}$$

where $\delta_i > 0$ implies i is a source where external flow is injected into the graph, $\delta_i < 0$ implies i is a destination where external flow leaves the graph. At other vertices, we have $\delta_i = 0$. Flow conservation (2.1) can be expressed compactly as

$$\mathcal{A}f = \delta, \tag{2.2}$$

where $\mathcal{A} : \mathbf{R}^m \to \mathbf{R}^n$ is the *incidence matrix* with $\mathcal{A}_{i,(j,i)} = 1$ for all $(j, i) \in E$, $\mathcal{A}_{i,(i,j)} = -1$ for all $(j, i) \in E$. Suppose a flow through the graph incurs a cost given by $h(w, f) =$



(a) (b)

Figure 2.1: A simple graph demonstration of maximizing the shortest path by allocating weights. The source node is 1 and the destination node is 6.

$\sum_{(i,j)} w_{ij}|f_{ij}|$, where w_{ij} is the cost per unit flow or weight for edge (i, j), and $w \in \mathbf{R}^m$ denotes the vector of all weights. The classic *shortest path problem* is to find a path connecting the source s and the destination t with the minimum cost.

The problem we focus on in this paper is the following. In many applications, the weights w are subject to change and we are interested to know how the minimum cost changes in the worst case as w varies. Wherever the goal is to slow down the flow from source to destination, we are interested in allocating the weights w subject to resource limitations denoted by $\mathcal{W}$, to make the cheapest flow as costly as possible (see examples in Section 2.2). Before formally stating the problem, we give an simple example as shown in Fig. 2.1 to demonstrate the idea. This simple graph has 6 nodes and 8 directed edges with fixed edge weights labeled on each edge as shown in Fig. 2.1 (a). The shortest distance between node 1 and 6 is 4 via the path $1 \to 5 \to 6$. If we allow the weights to increase from the labeled number but the total amount of the increase is limited by 5, what is the optimal way to allocate the weights such that the shortest distance is maximized? The optimal weights (obtained by solving an optimization problem to be discussed later) are labeled on the edges in Fig. 2.1 (b): 5 edges have increased weights such that the shortest distance between nodes 1 and 6 increases to 8 (from 4), and there are more than one shortest path ($1 \to 2 \to 4 \to 6$ and $1 \to 5 \to 6$).

More generally, let the cost of flow $f \in \mathbf{R}^m$ in a graph with weights $w \in \mathbf{R}^m,\ w \succeq 0$ be given by a function $h(w, f)$ which is *concave* in w and *convex* in f. Then for a fixed w, the problem of finding the minimum cost flow is

$$\begin{array}{lll} p(w) = & \min\limits_{f} & h(w, f) \\ & \text{subject to} & \mathcal{A}f = \delta, \end{array} \tag{2.3}$$

where $\delta_s = 1$, $\delta_t = -1$ and $\delta_i = 0$ for other vertices. This is a convex optimization problem. If the problem is feasible, it follows from flow conservation that $\sum_{i \in V} \delta_i = 0$. The dual of this problem is

$$d(w) = \max_{\phi} \ \delta^T \phi - h^*(w, \mathcal{A}^T \phi), \tag{2.4}$$

where $\phi \in \mathbf{R}^n$ is the Lagrange multiplier, and $h^*(w, y) = \sup_f (y^T f - h(w, f))$ is the conjugate function of h with respect to the variable f. To consider the worst-case geodesic distance, we need to maximize $p(w)$ with respect to w,

$$\begin{array}{lll} \underset{w}{\text{maximize}} & \min\limits_{f} \ h(w, f) \\ \text{subject to} & \mathcal{A}f = \delta, \\ & w \in \mathcal{W}, \end{array} \tag{2.5}$$

where $\mathcal{W}$ is a convex constraint set. We call this problem the *geodesic distance maximization problem (GDMP)* on a graph. The continuous domain versions will be discussed in Section 2.4.

The max-min form of the GDMP (2.5) does not easily lend itself to optimization over f and w. A more convenient formulation can be obtained by using the dual of (2.3). The dual variable ϕ_i can be viewed as the *potential* at vertex i, $(\mathcal{A}^T \phi)_{(i,j)} = \phi_j - \phi_i$. If optimization problems (2.3) and (2.4) satisfy strong duality, the optimal objective values are the same, and we can maximize $d(w)$ instead of $p(w)$ to get

$$\begin{array}{ll} \underset{w,\phi}{\text{maximize}} & \delta^T \phi - h^*(w, \mathcal{A}^T \phi) \\ \text{subject to} & w \in \mathcal{W}. \end{array} \tag{2.6}$$

We refer to (2.5) as the primal formulation and (2.6) as the dual formulation of the GDMP. We prefer (2.6) which is jointly convex in ϕ and w for all cost functions $h(w, f)$ that are concave in w and convex in f.

In Section 2.2, we survey applications from a variety of areas. In Section 2.3, we discuss the GDMP on a graph. We give an equivalent form for GDMP using Lagrangian duality, and propose several cost functions in Section 2.3.1 for which the GDMP is a convex problem. Then we discuss a special finite difference case of GDMP on a regular grid in Section 2.3.3 and extend it to upwind discretization in Section 2.3.4. In Section 2.4, we consider GDMP in a continuous domain, with isotropic case in Section 2.4.1 and anisotropic case in 2.4.2. We discuss algorithms for GDMP in Section 2.5, and give numerical examples in Section 2.6. Conclusions follow in Section 2.7.

2.2 Applications

The GDMP arises in applications from a wide variety of fields: physics (e.g., lens system design in optics, and more generally material design in wave propagation), transportation (e.g., the traffic congestion problem), network resource allocation (e.g., network interdiction, forest fire control, defensive landscape design), and more. This section catalogs some of these applications.

2.2.1 Network interdiction problems

Special cases of GDMP have been studied in the context of *network interdiction* [8,17,27,39, 43,51,95]. Fulkerson et al. [39] originally considered a simple network interdiction problem: the evader (e.g., smuggler) picks his path between a source and a destination in a graph, and the interdictor (e.g., police) allocate their effort (e.g., personnel, surveillance tools) to each edge, aiming to intercept the evader. The weight on edge (i, j) is $\underline{w}_{ij} + u_{ij}$, increasing with the allocated interdiction effort $u_{ij} \geq 0$, and the total effort is bounded as $\sum_{(i,j)\in E} u_{ij} \leq \tilde{u}$. To optimally allocate its resources, the interdictor should maximize the geodesic with respect

to u,

$$\begin{array}{ll} \underset{u}{\text{maximize}} & \min\limits_{f} \sum\limits_{(i,j)\in E} (\underline{w}_{ij} + u_{ij})|f_{ij}| \\ \text{subject to} & \sum\limits_{j:(i,j)\in E} f_{ij} - \sum\limits_{j:(i,j)\in E} f_{ji} = \delta_i, \quad i \in V \\ & u_{ij} \geq 0, \quad (i,j) \in E \\ & \sum\limits_{(i,j)\in E} u_{ij} \leq \tilde{u}, \end{array} \tag{2.7}$$

which can be written as a GDMP (with variables $w_{ij} = \underline{w}_{ij} + u_{ij}$ satisfying $\tilde{w} = \sum_{(i,j)\in E} \underline{w}_{ij} + \tilde{u}$). A continuous version of this problem is given in [10]. Suppose an invader progresses along the geodesic toward a defender in a continuous field. To slow down the invader, the defender maximizes the geodesic by changing the landscape or constructing fortification. Solving the GDMP gives the optimal landscape design.

Probabilistic interdiction with exponential interdiction model. Suppose the evader can choose a path between a source and a destination, while the interdictor can detect the presence of the evader on each edge with a certain probability. Let p_{ij} denote the probability that the evader passes edge (i,j) undetected. Then the probability that the evader makes it to the destination undetected along a path P is $\prod_{(i,j)\in P} p_{ij}$. Suppose detection failure probability depends exponentially on the interdiction effort $w_{ij} \geq 0$, i.e., $p_{ij} = e^{-w_{ij}}$, and the interdiction effort satisfies $w \preceq \overline{w}$ and $\sum_{(i,j)\in E} w_{ij} \leq \tilde{w}$.

The interdictor aims to minimize the maximum total detection failure probability,

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \max\limits_{P} \prod\limits_{(i,j)\in P} e^{-w_{ij}} \\ \text{subject to} & 0 \preceq w \preceq \overline{w} \\ & \sum\limits_{(i,j)\in E} w_{ij} \leq \tilde{w}. \end{array} \tag{2.8}$$

Problem (2.8) can be equivalently expressed as a GDMP by replacing the objective with its logarithm, noting that the log function is monotonically increasing on $\mathbf{R}_{++}$.

2.2.2 Worst-case traffic planning and traffic congestion

Shipping networks of delivery companies often experience uncertainty caused by unexpected incidents, e.g., traffic accidents, road maintenance and weather conditions, which may cause serious delays. In order to guarantee customers the latest delivery time, companies may need to estimate the *worst-case* delivery delay. In this application, w represents the time cost coefficients, and its uncertainty is modeled by a convex set $\mathcal{W}$ obtained from side information (e.g., historical data, weather forecast). If the cost is proportional to the delivery flow, the worst-case delivery time can be found by solving a GDMP.

Congestion is a practical phenomenon in traffic problems, and can be accounted for in our model if the cost increases faster than linearly as the flow increases. The power cost function described in Section 2.3.1 can be used to model this nonlinear increase. Traffic congestion has been an essential component in the classic *traffic equilibrium problem* [36, 61, 88, 93] [34, Sec. 3.6]. Recently it has been extended to the continuous case in [9, 10], which consider a non-cooperative game where drivers compete to travel in a network with congestion attempting to minimize their own costs. This game admits the well-known *Wardrop equilibrium* [93], where each driver chooses the geodesic (i.e., minimum cost path) and the paths used all have equal cost; paths with costs higher than the minimum will have no flow. The GDMP can be interpreted as a traffic game between one target driver and all others: the target driver navigates a geodesic to minimize the cost, while others compete with this driver by changing the weights.

2.2.3 Maximizing evacuation time in forest fires

The GDMP comes up in applications such as controlling forest fires and managing the spatial spread of disease or a contaminating agent.

Consider a forest fire that starts at a source location and expands outwards. A number of models have been proposed to simulate the propagation of fire [35, 67, 77]. In fire simulation systems, a simple model that is often used is the elliptic firefront model [4, 80] assuming that

fire propagates along the geodesic, which can be modeled by a partial differential equation called the Eikonal equation [49, Ch. 1] derived from Huygens Principle. The Eikonal equation and its generalization will be further discussed in Section 2.4 and Appendix A.2. In this application, the weight w can be interpreted as a "slow down factor" which is the inverse of fire speed and affected by natural conditions such as wetness, density and types of forest vegetation at each point.

In order to slow down the fire and maximize evacuation time, firefighters aim to optimally change the weights w by applying limited amount of resources, e.g., depositing retardants or removing vegetation. To find the optimal resource allocation strategy for firefighters, most existing work heuristically iterates between simulating fire propagation and allocating resources. We approach this problem by formulating it as a GDMP, and the optimal resource allocation plan is achieved by solving a convex optimization problem.

2.2.4 Design of optical lens systems

It is well known in geometric optics that a light ray travels between two points in the least time (*Fermat's principle*), and the propagation of light in the medium follows the Eikonal equation [49, Ch. 1] [42, Sec. 2.2, 2.4]. The weights w have the physical meaning of being the *index of refraction* of the medium at each point, which is defined as the ratio of the speed of light in vacuum to its speed in this medium.

Lens design has long been a research focus in optical engineering [52] [66, Ch. 18] [59, Ch. 17]. A common goal is to change the medium by placing material with different indices of refraction at different points (for example via gradient-index optics, where the index of refraction of a material is designed to vary gradually), to steer propagating rays of light and create a desired focal point. Maximizing the geodesic between source and destination points with respect to w leads to an equilibrium where all paths connecting the two points have the same length. Therefore, all light rays arrive at the destination point simultaneously, creating a focus point as desired. The optimal weights from numerical examples in Section 2.6.2 agree with physical intuition for lens placement from geometric optics.

2.3 Geodesic distance maximization on a graph

As we discuss in Section 2.4, in some applications the graph G arises from discretizing a continuous domain in which flow propagates, and the vectors f and w arise from discretizing the corresponding continuous flow vector field and continuous weight function. Such cases require a cost that allows coupling between flows on different edges, but is *separable* with respect to groups of flow variables. Consider a partition $\{P_1, \ldots, P_l\}$ of the edge set E, and let $f_k \in \mathbf{R}^{|P_k|}$ denote the vector of flows f_{ij} with $(i,j) \in P_k$ ($|P|$ denotes the cardinality of a set P). Let the flow cost function be separable with respect to this partition, i.e.,

$$h(w,f) = \sum_{k=1}^{l} h_k\left(w_k, f_k\right),$$

where w_k denotes the weight for the kth cost function h_k, and $h_k(w_k, f_k)$ are convex in f_k and concave in w_k.

Recall that the conjugate function $h_k^*(w_k, y_k)$ is always convex in y_k. Moreover, if $h_k(w_k, f_k)$ is concave in w_k, then $h_k^*(w_k, y_k)$ is jointly convex in w_k and y_k. Let $\phi_i \in \mathbf{R}$ be the Lagrange multiplier for the ith constraint in (2.3). The dual problem can be written as

$$\begin{aligned} d(w) &= \max_{\phi} \inf_{f} \textstyle\sum_{k=1}^{l} h_k(w_k, f_k) + \phi^T(\delta - \mathcal{A}f) \\ &= \max_{\phi} \phi^T\delta - \sup_{f} \sum_{k=1}^{l} \left\{(\mathcal{A}^T\phi)_k^T f_k - h_k(w_k, f_k)\right\} \\ &= \max_{\phi} \phi^T\delta - \sum_{k=1}^{l} \sup_{f} \left\{(\mathcal{A}^T\phi)_k^T f_k - h_k(w_k, f_k)\right\} \\ &= \max_{\phi} \ (\phi_s - \phi_t) - \sum_{k=1}^{l} h_k^*(w_k, (\mathcal{A}^T\phi)_k), \end{aligned} \tag{2.9}$$

where the fourth equality holds since f_k are decoupled due to disjoint partitions. Several such costs motivated by applications are listed in Section 2.3.1.

2.3.1 Flow cost examples

We present several examples of separable flow costs $h(w, f) = \sum_k h_k(w_k, f_k)$, covering key special cases and applications. If $h_k(w_k, f_k)$ is rotation invariant with respect to f_k, i.e., $h_k(w_k, f_k) = h(w_k, Uf_k)$ for any rotation matrix U, we call the cost h *isotropic* because it depends on the amount of flow but not the direction of the vector f_k. Otherwise h is called *anisotropic*.

1. $h_k(w_k, f_k)$ **has the form** $w_k\|f_k\|$**,** $w_k \geq 0$ Consider the case where the cost of f_k (vector of flow variables in the kth partition) is given by $w_k\|f_k\|$, $w_k \geq 0$. Norms are natural measures for the amount of flow, and $h_k(w_k, f_k)$ is convex in f_k and linear in w_k. Recall that the conjugate function of a norm is the indicator function of the dual norm ball,

$$\begin{aligned} h_k^*(w_k, y_k) &= \sup_{f_k} \; y_k^T f_k - w_k\|f_k\| \\ &= \sup_{f_k} \; \|y_k\|_*\|f_k\| - w_k\|f_k\| = \begin{cases} 0 & \text{if } \|y_k\|_* \leq w_k \\ +\infty & \text{otherwise.} \end{cases} \end{aligned}$$

   Hence problem (2.6) has a linear objective and dual norm ball constraints for each partition,

$$\begin{array}{ll} \underset{\phi, w}{\text{maximize}} & \phi_s - \phi_t \\ \text{subject to} & \|(\mathcal{A}^T\phi)_k\|_* \leq w_k, \quad k = 1, \ldots, l \\ & w \in \mathcal{W}. \end{array} \tag{2.10}$$

   This problem is jointly convex in $w \in \mathbf{R}^l$ and $\phi \in \mathbf{R}^n$. If the norm is Euclidean, the flow cost function is isotropic and the problem (2.10) is a *second-order cone program (SOCP)*. In Section 2.3.3, we will discuss this SOCP for a regular grid, and in Section 2.5.2, we will develop an algorithm to efficiently solve large-scale instances. If the dual norm is the ℓ_1 or the ℓ_∞ norm, the cost function is anisotropic, and problem (2.10) can be expressed as a linear program by standard techniques.

Another special case is when each partition has only one edge. As noted earlier, the cost function becomes $h(w,f) = \sum_{(i,j)\in E} w_{ij}|f_{ij}|$ and problem (2.3) reduces to the classic shortest path problem.

2. $h_k(w_k, f_k)$ **has the form** $w_k\|f_k\|^q$**,** $q > 1$**,** $w_k \geq 0$ The cost is proportional to the qth power of $\|f_k\|$, and arises in applications such as the traffic congestion problem [10], where one seeks a set of paths with minimum total cost for a large number of vehicles moving between a source and a destination. Congestion occurs when the cost increases faster than linear as the traffic flow increases. Although each vehicle aims to choose the geodesic, using the same edges increases congestion and results in a higher cost. This problem was extensively studied in the context of the Wardrop equilibrium, see e.g. [22, 45, 93]. The conjugate function of h_k is

$$\begin{aligned} h_k^*(w_k, y_k) &= \sup_{f_k} \; y_k^T f_k - w_k\|f_k\|^q \\ &= \sup_{f_k} \; \|y_k\|_*\|f_k\| - w_k\|f_k\|^q \\ &= \frac{q-1}{p}\frac{\|y_k\|_*^p}{w_k^{p-1}}. \end{aligned}$$

Since h_k^* is the (scaled) perspective function (see, e.g., [16]) of the convex function $\|\cdot\|_*^p$, $p > 1$, it is jointly convex in w_k and y_k. Problem (2.6) can be written as

$$\begin{array}{ll} \underset{\phi,w}{\text{maximize}} & \phi_s - \phi_t - \dfrac{q-1}{p}\displaystyle\sum_{k=1}^{l}\frac{\|(\mathcal{A}^T\phi)_k\|_*^p}{w_k^{p-1}} \\ \text{subject to} & w \in \mathcal{W}, \end{array} \tag{2.11}$$

which is also jointly convex in $w \in \mathbf{R}_+^l$ and $\phi \in \mathbf{R}^n$.

3. $h_k(W_k, f_k)$ **has the form** $\|W_k^{1/2}f_k\|_2$**,** $W_k \succeq 0$ The anisotropic cost function h_k is a quadratic norm parameterized by a positive semidefinite weight matrix W_k. The cost generated by a flow vector f_k can take any value between $\lambda_{\min}^{1/2}(W_k)\|f_k\|_2$ and $\lambda_{\max}^{1/2}(W_k)\|f_k\|_2$, where $\lambda_{\min}(W_k)$ and $\lambda_{\max}(W_k)$ are the smallest and largest eigenvalues of W_k. When $W_k = w_kI$, the cost reduces to the case in Section 1. More details of

modeling anisotropy will be discussed in Appendix A.2. The conjugate function of h_k is

$$\begin{aligned} h_k^*(W_k, y_k) &= \sup_{f_k} \; y_k^T f_k - \|W_k^{1/2} f_k\|_2 \\ &= \sup_{f_k} \; \|W_k^{-1/2} y_k\|_2 \|W_k^{1/2} f_k\|_2 - \|W_k^{1/2} f_k\|_2 \\ &= \begin{cases} 0 & \text{if } \|W_k^{-1/2} y_k\|_2 \leq 1 \\ +\infty & \text{otherwise.} \end{cases} \end{aligned}$$

It turns out that $\|W_k^{-1/2} y_k\|_2$ is jointly convex in W_k and y_k, since we can write the constraint $\|W_k^{-1/2} y_k\|_2 \leq 1$ as $y_k^T W_k^{-1} y_k \leq 1$, which can in turn be expressed as a linear matrix inequality (LMI) using Schur complements. Hence problem (2.6) can be written as

$$\begin{array}{ll} \underset{W,\phi}{\text{minimize}} & \phi_s - \phi_t \\ \text{subject to} & \begin{bmatrix} W_k & (\mathcal{A}^T\phi)_k \\ (\mathcal{A}^T\phi)_k^T & 1 \end{bmatrix} \succeq 0, \quad k = 1, \ldots, l \\ & W_1, \ldots, W_l \in \mathcal{W}. \end{array} \tag{2.12}$$

This is a *semidefinite program (SDP)* with variables $W_k \in \mathbf{S}_{++}^{|P_k|}$, $k = 1, \ldots, l$ and $\phi \in \mathbf{R}^n$. In Section 2.3.3, we will discuss this SDP on the regular grid.

2.3.2 Multiple sources and destinations

The GDMP can be readily generalized to the case of multiple sources and destinations, where we maximize the geodesic distance between a source set S and a destination set T.

The primal problem can be written with more variables as

$$\begin{aligned} \underset{w}{\text{maximize}} \quad & \min_{f,\delta} \sum_{k=1}^{l} h_k(w_k, f_k) \\ \text{subject to} \quad & \mathcal{A}f = \delta, \\ & \sum_{i \in S} \delta_i = 1, \quad \sum_{i \in T} \delta_i = -1, \\ & \delta_i = 0, \quad i \notin S \cup T \\ & w \in \mathcal{W}. \end{aligned} \tag{2.13}$$

The optimal δ gives a source-destination pair with the smallest geodesic distance, among all pairs. The optimal f gives the geodesic for this pair. The corresponding dual formulation is

$$\begin{aligned} \underset{w,\phi,\phi_S,\phi_T}{\text{maximize}} \quad & (\phi_S - \phi_T) - \sum_{k=1}^{l} h_k^*(w_k, (\mathcal{A}^T\phi)_k) \\ \text{subject to} \quad & \phi_i = \phi_S, \quad i \in S \\ & \phi_i = \phi_T, \quad i \in T \\ & w \in \mathcal{W}. \end{aligned} \quad . \tag{2.14}$$

We can reduce the multiple sources and destinations to the single source and single destination case by simply introducing a new vertex s and connecting s to all sources $i \in S$ by directed edges (s, i) with $w_{si} = 0$ (and similarly, introducing a vertex t and connecting all destinations $i \in T$ to t by directed edges (i, t) with $w_{it} = 0$). Maximizing the geodesic distance between s and t can be expressed as

$$\begin{aligned} \underset{w}{\text{maximize}} \quad & \min_{f} \sum_{k=1}^{l} h_k(w_k, f_k) \\ \text{subject to} \quad & \sum_{j:(i,j) \in E} f_{ij} - \sum_{j:(j,i) \in E} f_{ji} = 0, \quad i \in V \backslash (S \cup T) \\ & \sum_{j \in S} f_{sj} = 1, \quad \sum_{j \in T} f_{jt} = -1 \\ & w \in \mathcal{W}. \end{aligned} \tag{2.15}$$

which can be easily shown to be equivalent to (2.13).

2.3.3 GDMP on a regular grid: simple finite difference discretization

In this section we consider the GDMP on a special graph: the $m \times n$ regular grid in $\mathbf{R}^2$ as shown in Fig. 2.2. This special case commonly arises from discretizing continuous problems which will be discussed in Section 2.4.

With a slight abuse of notation, we index each vertex by its row and column indices (i, j) (previously (i, j) denoted an edge between two vertices). Each interior vertex (i, j) is connected to four neighbors $(i-1, j), (i, j-1), (i+1, j)$ and $(i, j+1)$ by four edges $(i-1, j, 1)$, $(i, j-1, 2)$, $(i, j, 1)$ and $(i, j+1, 2)$. The corner vertices $(1, 1), (1, n), (m, 1)$ and (m, n) each have two neighbors, and other boundary vertices have three neighbors. For this grid, f is an $m \times n \times 2$ array with $f_{i,j} = [f_{i,j,1}, f_{i,j,2}]^T \in \mathbf{R}^2$, ϕ, w, δ are matrices of size $m \times n$. Flow conservation conditions are

$$(\mathcal{A}f)_{i,j} = (f_{i,j,1} - f_{i-1,j,1}) + (f_{i,j,2} - f_{i,j-1,2}) = \delta_{i,j}, \tag{2.16}$$

for $i = 1, \ldots, m$, $j = 1, \ldots, n$, and we set the boundary conditions to be $f_{0,j,1} = f_{m,j,1} = 0$ for $j = 1, \ldots, n$ and $f_{i,0,2} = f_{i,n,2} = 0$ for $i = 1, \ldots, m$. The left-hand side of each flow conservation condition can be seen as the *divergence operator* approximated by *backward difference* (see, e.g., [50, 62]).

Next we define a partition as follows: for each vertex (i, j), group together two edges $(i, j, 1)$ and $(i, j, 2)$ that leave this vertex, i.e., $P_{i,j} = \{(i, j, 1), (i, j, 2)\}$. We will see that this partition corresponds to the *forward difference* discretization of the *gradient operator* (see, e.g., [50, 62]). Other partitions corresponding to different discretization schemes will be discussed in Section 2.3.4. Let $\mathcal{D} : \mathbf{R}^{m \times n} \to \mathbf{R}^{m \times n \times 2}$ be the forward difference operator

$$(\mathcal{D}\phi)_{i,j} = (\mathcal{A}^T\phi)_{i,j} = \begin{bmatrix} \phi_{i+1,j} - \phi_{i,j} \\ \phi_{i,j+1} - \phi_{i,j} \end{bmatrix}. \tag{2.17}$$

If the norm in problem (2.10) is the Euclidean norm, the dual norm ball constraints are second-order cone constraints involving the forward operator,

$$\|(\mathcal{D}\phi)_{i,j}\|_2 = \left\| \begin{bmatrix} \phi_{i+1,j} - \phi_{i,j} \\ \phi_{i,j+1} - \phi_{i,j} \end{bmatrix} \right\|_2 \leq w_{i,j}, \quad \text{for } i = 1, \ldots, m,\ j = 1, \ldots, n. \tag{2.18}$$

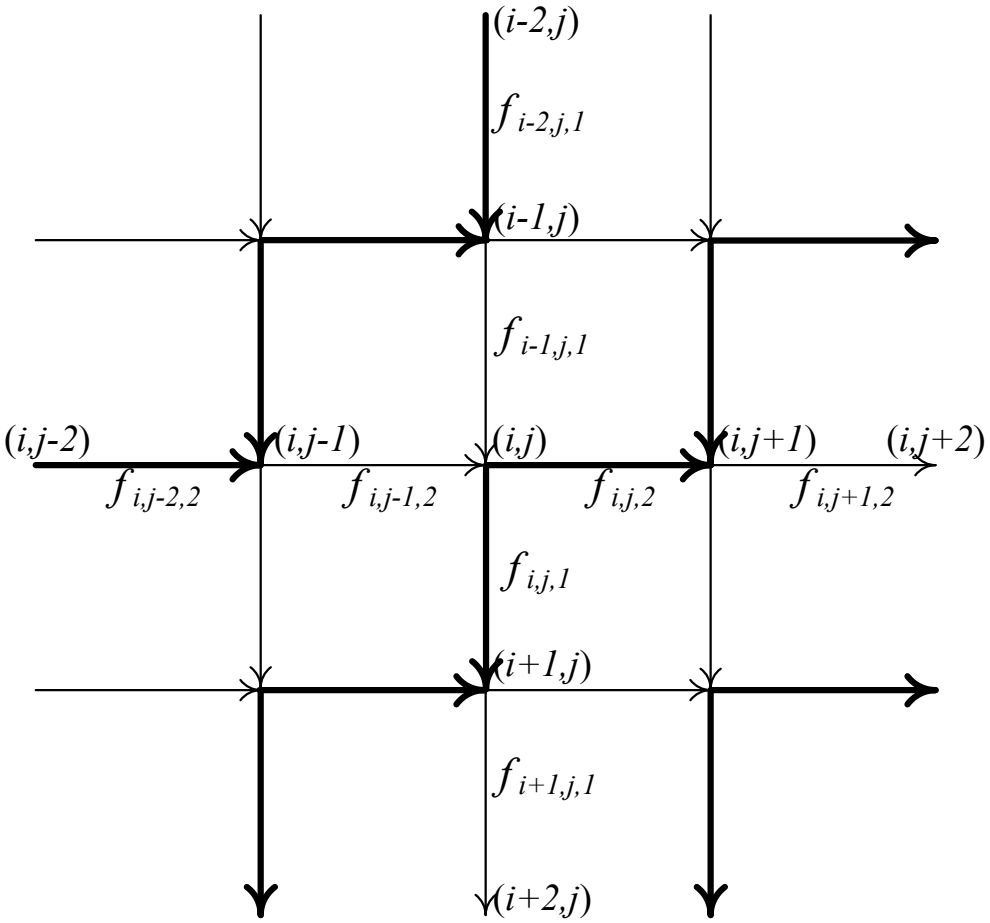


Figure 2.2: Finite difference partition structure.

We can also express the constraints as $(w_{i,j}, (\mathcal{D}\phi)_{i,j}) \in K_3$, where K_3 denotes the second-order cone of size 3, $K_3 = \{(t, y) \in \mathbf{R}^3 \mid \|y\|_2 \leq t\}$. The boundary conditions are $\phi_{m+1,j} - \phi_{m,j} = 0$ for $j = 1, \ldots, n$ and $\phi_{i,n+1} - \phi_{i,n} = 0$ for $i = 1, \ldots, m$. and

Note that the dual formulation (2.10) with a source set S and destination set T can be written as

$$\begin{array}{ll} \underset{w,\phi,\phi_S,\phi_T}{\text{maximize}} & \phi_S - \phi_T \\ \text{subject to} & (w_{i,j}, (\mathcal{D}\phi)_{i,j}) \in K_3, \quad i = 1, \ldots, m,\ j = 1, \ldots, n \\ & \phi_{i,j} = \phi_S, \quad (i,j) \in S \\ & \phi_{i,j} = \phi_T, \quad (i,j) \in T \\ & w \in \mathcal{W}. \end{array} \tag{2.19}$$

Similarly, the anisotropic problem (2.12) can be written as

$$\begin{array}{ll} \underset{W,\phi,\phi_S,\phi_T}{\text{maximize}} & \phi_S - \phi_T \\ \text{subject to} & \begin{bmatrix} W_{i,j} & (\mathcal{D}\phi)_{i,j} \\ (\mathcal{D}\phi)_{i,j}^T & 1 \end{bmatrix} \succeq 0, \quad i = 1, \ldots, m,\ j = 1, \ldots, n \\ & \phi_{i,j} = \phi_S, \quad (i,j) \in S \\ & \phi_{i,j} = \phi_T, \quad (i,j) \in T \\ & W \in \mathcal{W}. \end{array} \tag{2.20}$$

Some numerical examples of (2.19) and (2.20) are given in Section 2.6.2. In Section 7.2, we develop a specialized Alternating Direction Methods of Multipliers (ADMM) to efficiently solve the SOCP (2.19) for large-scale instances.

2.3.4 GDMP on a regular grid: upwind discretization

In Section 2.3.3, we chose the edge partition $P_{i,j} = \{(i,j,1), (i,j,2)\}$ for each vertex and the cost function $h_{i,j}(w_{i,j}, f_{i,j}) = w_{i,j}\|f_{i,j}\|_2$. This choice of partitions and cost led to the second-order cone constraint (2.18) involving finite differences (2.17) for ϕ. Finite differences are often used to approximate gradient operators, and the constraint (2.18) is the finite discretization of a partial differential equation (the Eikonal equation) to be discussed in

Section 2.4.1. However, as discussed in the literature on numerical PDEs, more advanced discretization schemes are often needed to guarantee the consistency of the solution; for example, the *upwind discretization* of the gradient operator which is given as

$$\begin{bmatrix} \max\{\max\{\phi_{i,j}-\phi_{i-1,j},0\},\max\{\phi_{i,j}-\phi_{i+1,j},0\}\} \\ \max\{\max\{\phi_{i,j}-\phi_{i,j-1},0\},\max\{\phi_{i,j}-\phi_{i,j+1},0\}\} \end{bmatrix}. \tag{2.21}$$

See, e.g., [81, 84] for more details on the upwind discretization.

Let $\mathcal{B}:\mathbf{R}^{m\times n}\to\mathbf{R}^{m\times n\times 2}$ be the *backward difference* operator used in numerical PDE

$$(\mathcal{B}\phi)_{i,j}=\begin{bmatrix} \phi_{i,j}-\phi_{i-1,j} \\ \phi_{i,j}-\phi_{i,j-1} \end{bmatrix}. \tag{2.22}$$

The second-order cone constraint with upwind discretization can be written as

$$\left\|([(\mathcal{B}\phi)_{i,j}\ (\mathcal{D}\phi)_{i,j}])_+\right\|_{\infty,2}=\left\|\left(\begin{bmatrix} \phi_{i,j}-\phi_{i-1,j} & \phi_{i,j}-\phi_{i+1,j} \\ \phi_{i,j}-\phi_{i,j-1} & \phi_{i,j}-\phi_{i,j+1} \end{bmatrix}\right)_+\right\|_{\infty,2}\le w_{i,j}, \tag{2.23}$$

where $(X_{i,j})_+=\max\{X_{i,j},0\}$ is the positive part of $X_{i,j}$, the norm $\|X\|_{\infty,2}$ is the Euclidean norm of the vector of ℓ_∞ norms of all *rows* of matrix X. In this section, we show how the GDMP covers this discretization.

First, each flow $f_{i,j,k}$ is decomposed as $(f_{i,j,k})_+-(f_{i,j,k})_-$, where $(f_{i,j,k})_+=\max\{f_{i,j,k},0\}$ and $(f_{i,j,k})_-=\max\{-f_{i,j,k},0\}$. As shown in Figure 2.3, we associate four flows to each vertex (i,j): $(f_{i,j,1})_-,(f_{i-1,j,1})_+$ in the vertical direction, and $(f_{i,j,2})_-,(f_{i,j-1,2})_+$ in the horizontal direction. It is convenient to arrange the set of four flows associated with each vertex (i,j) in a 2×2 matrix,

$$f_{i,j}=\begin{bmatrix} (f_{i,j,1})_- & (f_{i-1,j,1})_+ \\ (f_{i,j,2})_- & (f_{i,j-1,2})_+ \end{bmatrix}.$$

We can view $(f_{i,j,k})_+$ and $(f_{i,j,k})_-$ as independent variables, then the $f_{i,j}$ are decoupled, i.e., $f_{i,j}$ does not share any components with $f_{i',j'}$, $i'\neq i$, $j'\neq j$. This property will be used in decomposing the Lagrangian. We consider the following cost function

$$h_{i,j}(w_{i,j},f_{i,j})=w_{i,j}\left\|\begin{bmatrix} (f_{i,j,1})_-+(f_{i-1,j,1})_+ \\ (f_{i,j,2})_-+(f_{i,j-1,2})_+ \end{bmatrix}\right\|_2.$$

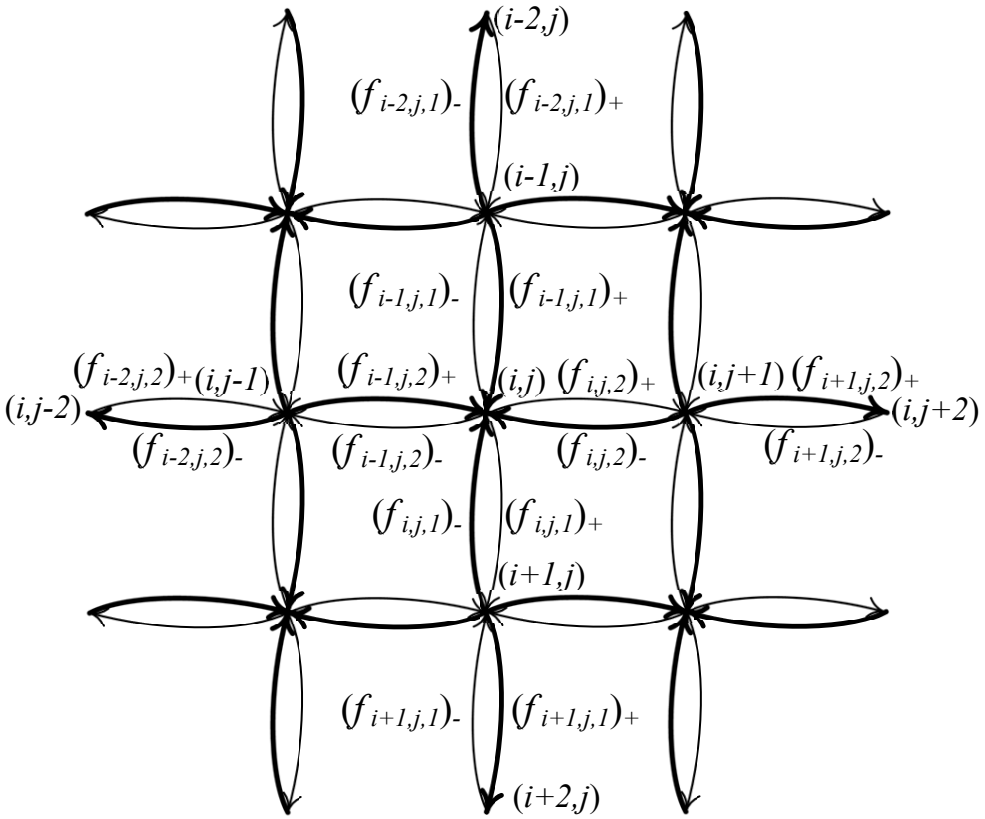


Figure 2.3: Variables and structure for upwind discretization.

Clearly, $h_{i,j}(w_{i,j}, f_{i,j})$ is convex in $f_{i,j}$ and concave (linear) in $w_{i,j}$. The flow conservation constraints are generalized as

$$\begin{aligned}(\mathcal{A}f)_{i,j} &= [(f_{i,j,1})_- - (f_{i+1,j,1})_- + (f_{i-1,j,1})_+ - (f_{i-2,j,1})_+] \\ &\quad +[(f_{i,j,2})_- - (f_{i,j+1,2})_- + (f_{i,j-1,2})_+ - (f_{i,j-2,2})_+] \\ &= \delta_{i,j},\end{aligned}$$

for $i = 1, \ldots, m$, $j = 1, \ldots, n$, where we call $\mathcal{A} : \mathbf{R}^{m\times n\times 2\times 2} \to \mathbf{R}^{m\times n}$ the *upwind divergence* operator. Then the geodesic problem can be written as

$$\begin{array}{ll}\underset{f}{\text{minimize}} & \sum_{i,j} h_{i,j}(w_{i,j}, f_{i,j}) \\ \text{subject to} & \mathcal{A}f = \delta \\ & f \geq 0.\end{array} \tag{2.24}$$

Let $\mathcal{A}^* : \mathbf{R}^{m\times n} \to \mathbf{R}^{m\times n\times 2\times 2}$ be the adjoint mapping of $\mathcal{A}$. It is easy to check that $(\mathcal{A}^*\phi)_{i,j} = [(\mathcal{B}\phi)_{i,j}\ (\mathcal{D}\phi)_{i,j}]$, noting that

$$\begin{aligned}&\sum_{i,j} \phi_{i,j}(\mathcal{A}f)_{i,j} \\ =\ &\sum_{i,j} \phi_{i,j} \{[(f_{i,j,1})_- - (f_{i+1,j,1})_- + (f_{i-1,j,1})_+ - (f_{i-2,j,1})_+] \\ &\qquad +[(f_{i,j,2})_- - (f_{i,j+1,2})_- + (f_{i,j-1,2})_+ - (f_{i,j-2,2})_+]\} \\ =\ &\sum_{i,j} \{(f_{i,j,1})_-(\phi_{i,j} - \phi_{i-1,j}) + (f_{i-1,j,1})_+(\phi_{i,j} - \phi_{i+1,j}) \\ &\qquad +(f_{i,j,2})_-(\phi_{i,j} - \phi_{i,j-1}) + (f_{i,j-1,2})_+(\phi_{i,j} - \phi_{i,j+1})\} \\ =\ &\sum_{i,j} \mathbf{Tr}\left(\begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} & \phi_{i,j} - \phi_{i+1,j} \\ \phi_{i,j} - \phi_{i,j-1} & \phi_{i,j} - \phi_{i,j+1} \end{bmatrix}^T \begin{bmatrix} (f_{i,j,1})_- & (f_{i-1,j,1})_+ \\ (f_{i,j,2})_- & (f_{i,j-1,2})_+ \end{bmatrix}\right) \\ =\ &\sum_{i,j} \mathbf{Tr}\left((\mathcal{A}^*\phi)_{i,j}^T f_{i,j}\right).\end{aligned}$$

The Lagrangian dual is

$$\begin{aligned}
&\min_{f\geq 0} \sum_{i,j} h_{i,j}(w_{i,j}, f_{i,j}) + \sum_{i,j} \phi_{i,j}(\delta_{i,j} - (\mathcal{A}f)_{i,j}) \\
= \;& \sum_{i,j} \phi_{i,j}\delta_{i,j} - \max_{f\geq 0} \left\{ \sum_{i,j} \phi_{i,j}(\mathcal{A}f)_{i,j} - \sum_{i,j} h_{i,j}(w_{i,j}, f_{i,j}) \right\} \\
= \;& \sum_{i,j} \phi_{i,j}\delta_{i,j} - \max_{f\geq 0} \left\{ \sum_{i,j} \mathbf{Tr}\left((\mathcal{A}^*\phi)_{i,j}^T f_{i,j}\right) - \sum_{i,j} h_{i,j}(w_{i,j}, f_{i,j}) \right\} \\
= \;& \sum_{i,j} \phi_{i,j}\delta_{i,j} - \sum_{i,j} \max_{f_{i,j}\geq 0} \left\{ \mathbf{Tr}\left((\mathcal{A}^*\phi)_{i,j}^T f_{i,j}\right) - h_{i,j}(w_{i,j}, f_{i,j}) \right\} \\
= \;& \sum_{i,j} \phi_{i,j}\delta_{i,j} - \sum_{i,j} h_{i,j}^{\circ}(w_{i,j}, (\mathcal{A}^*\phi)_{i,j}).
\end{aligned}$$

The max and sum are exchangeable above since $f_{i,j}$ are decoupled as mentioned before. In Appendix A.1, we show that

$$h_{i,j}^{\circ}(w_{i,j}, (\mathcal{A}^*\phi)_{i,j}) = \begin{cases} 0, & \|(\mathcal{A}^*\phi)_+\|_{\infty,2} \leq w_{i,j} \\ +\infty, & \text{otherwise.} \end{cases}$$

Finally, we obtain the dual formulation of the GDMP with upwind discretization as

$$\begin{aligned}
\underset{w,\phi,\phi_S,\phi_T}{\text{maximize}} \quad & \phi_S - \phi_T \\
\text{subject to} \quad & \left\| ([(\mathcal{B}\phi)_{i,j} \; (\mathcal{D}\phi)_{i,j}])_+ \right\|_{\infty,2} \leq w_{i,j}, \quad i = 1, \ldots, m, \; j = 1, \ldots, n \\
& \phi_{i,j} = \phi_S, \quad (i,j) \in S \\
& \phi_{i,j} = \phi_T, \quad (i,j) \in T \\
& w \in \mathcal{W}.
\end{aligned} \tag{2.25}$$

2.4 Geodesic distance maximization in a continuous domain

2.4.1 Isotropic continuous GDMP

The geodesic distance maximization problem discussed in Section 2.3 can be generalized to a continuous domain. To make our presentation simple, we choose the cost function as proportional to the ℓ_2 norm of the flow; other cost functions can be generalized similarly.

The presentation in this section is parallel to Section 2.3. Let $\Omega \subset \mathbf{R}^n$ be a compact set with a nonempty interior and a smooth boundary. As a disclaimer, here we treat the relevant function spaces and operations on them informally; the discussion of choices of function spaces for this problem is beyond the scope of this paper. A function $w : \Omega \to \mathbf{R}_{++}$ defines a *metric* (analog of weights in a graph) on Ω. Flow is defined by a vector-valued function $f : \Omega \to \mathbf{R}^n$, and the flow cost at each point $x \in \Omega$ is proportional to the ℓ_2-norm of $f(x)$, given by $w(x)\|f(x)\|_2$. The total cost over Ω is $\int_\Omega w(x)\|f(x)\|_2 dx$. Flow has to satisfy the divergence constraints (analog of the divergence constraint (2.1) in a graph),

$$\mathbf{div} f = \sum_{i=1}^{n} \frac{\partial f_i}{\partial x_i} = \delta,$$

where δ is defined on Ω and specifies the distribution of sources and destinations. δ can be decomposed as $\delta = \delta_s + \delta_t$, where $\delta_s > 0$ is the source distribution and $\delta_t < 0$ is the destination distribution.

In the case of single source and single destination, δ_s and δ_t are defined as two Dirac measures concentrated at a source point x_s and a destination point x_t. Given w and two disjoint sets $S,\ T \subset \Omega$, we want to find the geodesic connecting S and T, which can be cast as

$$\begin{array}{rll} p_c(w) = & \underset{f,\delta}{\text{minimize}} & \displaystyle\int_\Omega w(x)\|f(x)\|_2 dx \\ & \text{subject to} & \mathbf{div} f(x) = \delta(x), \quad x \in \Omega \\ & & \displaystyle\int_S \delta(x)dx = 1, \quad \int_T \delta(x)dx = -1, \\ & & \delta(x) = 0, \quad x \notin S \cup T. \end{array} \tag{2.26}$$

which is an infinite dimensional optimization problem. We aim to find the weight function $w : \Omega \to \mathbf{R}$ that yield the maximum geodesic, that is,

$$\begin{array}{ll} \underset{w}{\text{maximize}} & p_c(w) \\ \text{subject to} & w \in \mathcal{W}, \end{array} \tag{2.27}$$

where $\mathcal{W}$ is a convex constraint set. We refer to this problem as the *continuous* geodesic distance maximization problem (GDMP). The function $p_c(w)$ is *concave* in w since it is

the pointwise infimum of a family of linear functions of w, therefore (2.27) is a convex optimization problem. As with the graph case, it is helpful to examine the dual of problem (2.26),

$$\begin{aligned} d_c(w) = \quad & \underset{\phi}{\text{maximize}} \quad \phi_S - \phi_T \\ & \text{subject to} \quad \|\nabla\phi(x)\| \leq w(x), \quad x \in \Omega \\ & \qquad\qquad\quad \phi(x) = \phi_S, \quad x \in S \\ & \qquad\qquad\quad \phi(x) = \phi_T, \quad x \in T. \end{aligned} \tag{2.28}$$

The dual variable ϕ is the *potential function* and $\phi(x)$ gives the geodesic distance from the source set S to point x. So $\phi_S - \phi_T$ is the geodesic distance between the sets S and T. Maximizing $d_c(w)$ with respect to w gives the dual formulation of the continuous GDMP

$$\begin{aligned} & \underset{w}{\text{maximize}} \quad d_c(w) \\ & \text{subject to} \quad w \in \mathcal{W}. \end{aligned} \tag{2.29}$$

The overall objective in problem (2.29) is (jointly) concave in ϕ and w. With appropriate choice of function spaces, strong duality can be shown to hold for problems (2.26) and (2.28), thus (2.27) and (2.29) are equivalent.[1]

The dual problem (2.28) has the convex inequality constraints $\|\nabla\phi(x)\|_2 \leq w(x)$. From the complementary slackness condition, wherever $f^\star(x) \neq 0$, the optimal $\phi^\star$ satisfies the partial differential equation

$$\|\nabla\phi^\star(x)\|_2 = w(x).$$

This nonlinear partial differential equation, called the *Eikonal equation* [33, 83, 91], is a fundamental equation governing wave propagation in a medium. The *viscosity solution* [81, 84] to the Eikonal equation at each point gives the geodesic distance from the source set to that point. In addition, complementary slackness reveals a connection between the optimal flow $f^\star$ and potential $\phi^\star$: $f^\star/\|f^\star\|_2 = -\nabla\phi^\star/\|\nabla\phi^\star\|_2$. That is, the flow proceeds along the direction of negative gradient of the potential function.

[1] Unlike finite dimensional duality seen earlier, strong duality is more subtle in the infinite dimensional case and is beyond our scope.

To numerically solve the dual formulation of the continuous GDMP (2.29), we discretize the continuous domain $\Omega \in \mathbf{R}^2$ as an $m \times n$ regular grid. Correspondingly, the functions ϕ, w are discretized as matrices of size $m \times n$. We define the mapping $\mathcal{D} : \mathbf{R}^{m\times n} \to \mathbf{R}^{m\times n\times 2}$ to approximate the gradient by an appropriate finite dimensional operator. For forward finite difference [50, 62], the gradient approximation is defined as in (2.17). Then problem (2.29) can be written as (2.19), for which we develop a specialized and efficient ADMM algorithm in Section 2.5.2. More sophisticated discretization schemes, e.g., *upwind discretization*, can also be used as shown in Section 2.3.4.

2.4.2 Anisotropic continuous GDMP

In a variety of applications, flow propagation speed at a point x depends on both the location x and the direction of the flow vector, which is called *anisotropic* propagation. Anisotropic propagation has been studied in several contexts, including geophysics [28], crystal growth [82] and medical imaging [60]. For anisotropic propagation, the weight function $W(x) : \Omega \to \mathbf{S}^n_{++}$ defines a positive definite metric on Ω, and the Eikonal constraint in 2.28 is generalized to $\|W^{-1/2}(x)\nabla\phi(x)\|_2 \leq 1$. Note that if $W(x) = w(x)I$, we recover the isotropic Eikonal inequality. In this section, we focus on the optimization problem and defer the of modeling anisotropy to Appendix A.2.

With the generalized Eiknonal constraint, the dual formulation of *anisotropic* geodesic distance maximization problem (GDMP) can be formulated as infinite dimensional optimization problem

$$\begin{array}{ll} \underset{W,\phi}{\text{maximize}} & \phi_S - \phi_T \\ \text{subject to} & \|W^{-1/2}(x)\nabla\phi(x)\|_2 \leq 1, \quad x \in \Omega \\ & \phi(x) = \phi_S, \quad x \in S \\ & \phi(x) = \phi_T, \quad x \in T \\ & W \in \mathcal{W}, \end{array} \tag{2.30}$$

where $\mathcal{W}$ is a convex constraint set. Similar to what we did in Section 3, this problem can

be equivalently expressed as

$$\begin{array}{ll} \underset{W,\phi}{\text{maximize}} & \phi_S - \phi_T \\ \text{subject to} & \begin{bmatrix} W(x) & \nabla\phi(x) \\ \nabla\phi(x)^T & 1 \end{bmatrix} \succeq 0, \quad x \in \Omega \\ & \phi(x) = \phi_S, \quad x \in S \\ & \phi(x) = \phi_T, \quad x \in T \\ & W \in \mathcal{W}. \end{array} \tag{2.31}$$

If we discretize the problem as in Section 2.4, problem (2.31) can be expressed as the semidefinite program (2.20).

2.5 Algorithms

In this section we discuss how to efficiently solve GDMP with the cost function $\sum_k h_k(w_k, f_k) = \sum_k w_k \|f_k\|_2$ in different settings. Throughout this section we consider a simple polyhedral constraint set $\mathcal{W}$

$$\mathcal{W} = \{w \in \mathbf{R}^l \mid \underline{w}_k \leq w_k \leq \overline{w}_k, \sum_{k=1}^{l} w_k \leq \tilde{w}\}, \tag{2.32}$$

where $\sum_{k=1}^{l} \underline{w}_k \leq \tilde{w} \leq \sum_{k=1}^{l} \overline{w}_k$. The constraints capture physical upper and lower bounds on the resource (weight) for each flow group f_k. For example in the lens design application, these are limits on the index of refraction of the material that can be placed at a certain location. The last constraint limits the total resource, e.g., the total volume of available material.

2.5.1 Single-edge partition case as a linear program

As discussed in Section 1, if each P_k is a partition with single edge, the cost function is reduced to $\sum_{(i,j)\in E} w_{ij}|f_{ij}|$, and the dual formulation (2.10) becomes a linear program which can be solved by general linear programming solvers. However, there are far more efficient graph-specific algorithms for network linear programs, such as the shortest path problem and

max-flow/min-cut problem (for classic results, see, e.g., [11, Ch. 4-7] [3, Ch. 9-11]). Can we make use of these efficient algorithms to solve the GDMP? It is known that for polyhedral $\mathcal{W}$ given in (2.32), GDMP can be solved by solving a polynomial number of shortest path problems, and for even more special $\mathcal{W}$, by solving a single max-flow/min-cut problem.

In this section we assume all flows are nonnegative, and adjust our graph setting as follows: any two vertices i, j are connected by two directed edges (i,j) and (j,i) with the same weight $w_{ij} = w_{ji}$. Then GDMP can be written as

$$\begin{array}{ll} \underset{w}{\text{maximize}} & \min\limits_{f} \sum\limits_{i,j} w_{ij} f_{ij} \\ \text{subject to} & \sum\limits_{j:(i,j)\in E} f_{ij} - \sum\limits_{j:(j,i)\in E} f_{ji} = \delta_i, \quad i \in V, \\ & f_{ij} \geq 0, \quad (i,j) \in E \\ & \underline{w}_{ij} \leq w_{ij} \leq \overline{w}_{ij}, \quad (i,j) \in E \\ & \sum\limits_{(i,j)\in E} w_{ij} \leq \tilde{w}. \end{array}$$

To make use of graph algorithms, we take the dual with respect to w to get

$$\begin{array}{ll} \underset{f,\lambda,\nu,\gamma}{\text{minimize}} & -\sum\limits_{(i,j)\in E} \lambda_{ij}\underline{w}_{ij} + \sum\limits_{(i,j)\in E} \nu_{ij}\overline{w}_{ij} + \gamma\tilde{w} \\ \text{subject to} & \sum\limits_{j:(i,j)\in E} f_{ij} - \sum\limits_{j:(j,i)\in E} f_{ji} = \delta_i, \quad i \in V \\ & f_{ij} = -\lambda_{ij} + \nu_{ij} + \gamma, \quad (i,j) \in E \\ & \gamma \geq 0, \quad f, \nu, \lambda \succeq 0, \end{array} \tag{2.33}$$

where $f, \lambda, \nu \in \mathbf{R}^m$, γ is a scalar. Since $\lambda_{ij} \geq 0$, we have $\nu_{ij} \geq f_{ij} - \gamma$ and $\nu_{ij} = \max\{0, f_{ij} - \gamma\}$. Substituting $\lambda_{ij} = -f_{ij} + \nu_{ij} + \gamma$, we obtain

$$\begin{array}{ll} \underset{f,\gamma}{\text{minimize}} & \sum\limits_{(i,j)\in E} f_{ij}\underline{w}_{ij} + \sum\limits_{(i,j)\in E} (\overline{w}_{ij} - \underline{w}_{ij}) \max\{0, f_{ij} - \gamma\} + \gamma \left(\tilde{w} - \sum\limits_{(i,j)\in E} \underline{w}_{ij} \right) \\ \text{subject to} & \sum\limits_{j:(i,j)\in E} f_{ij} - \sum\limits_{j:(j,i)\in E} f_{ji} = \delta_i, \quad i \in V \\ & \gamma \geq 0. \end{array} \tag{2.34}$$

Bertsimas and Sim proposed an algorithm for this type of problem in the context of *robust network flow problem* [12]. For a *fixed* γ, it can be shown that (2.34) is a min-cost flow problem on graph obtained by modifying G as follows: for each edge $(i,j) \in E$, introduce another edge $(i,j)'$, then set $\underline{w}_{i,j}$ and $\overline{w}_{ij}$ as the weights and γ and $+\infty$ as the capacities for edges (i,j) and $(i,j)'$ respectively. The intuition is that to minimize the cost, the flow uses the edge (i,j) with smaller weight $\underline{w}_{ij}$ first until this edge reaches its capacity γ, then extra flow uses edge $(i,j)'$ and incurs cost $(\overline{w}_{ij} - \underline{w}_{ij}) \max\{0, f_{ij} - \gamma\}$. To solve (2.34), one can use bisection on γ and solve a min-cost flow problem for each γ on the modified graph. We can consider several special cases of $\mathcal{W}$.

- *No upper bound, $\overline{w} = +\infty$* This special case was studied by Fulkerson and Harding [39] long before [12]. Since $\overline{w} = +\infty$, there is no need to introduce auxiliary edges $(i,j)'$. One can use bisection on γ and solve a series of min-cost flow problems on G with weights $\underline{w}$ and capacity γ,

$$\begin{array}{ll} \underset{f}{\text{minimize}} & \displaystyle\sum_{(i,j)\in E} f_{ij}\underline{w}_{ij} \\ \text{subject to} & \displaystyle\sum_{j:(i,j)\in E} f_{ij} - \sum_{j:(j,i)\in E} f_{ji} = \delta_i, \quad i \in V \\ & 0 \preceq f \preceq \gamma. \end{array} \tag{2.35}$$

- *Zero lower bound and no upper bound, $\underline{w} = 0$, $\overline{w} = +\infty$.* In this even more special case, the problem reduces to

$$\begin{array}{ll} \underset{f,\gamma}{\text{minimize}} & \gamma\tilde{w} \\ \text{subject to} & \displaystyle\sum_{j:(i,j)\in E} f_{ij} - \sum_{j:(j,i)\in E} f_{ji} = \delta_i, \quad i \in V \\ & 0 \preceq f \preceq \gamma \\ & \gamma \geq 0. \end{array} \tag{2.36}$$

  This problem has a simple interpretation: reduce γ to the least value $\gamma^\star$, where the flow f satisfying the divergence conditions is *just* feasible. The flow f is feasible as

long as the capacity of the minimum cut is at least δ_s. Therefore, this problem can be solved as a classical max-flow problem: fix $\gamma = \gamma_0$, solve the max-flow problem using the capacity γ_0 to find the flow $\tilde{f}$ and max-flow value $\tilde{\delta}$, then the solution to (2.36) is $f^\star = \tilde{f}/\tilde{\delta}$, $\gamma^\star = \gamma_0/\tilde{\delta}$.

2.5.2 Simple discretized GDMP as a second-order cone program

In Section 2.3.3, we discussed the GDMP on the regular grid and expressed it as a second-order cone program(2.10). This SOCP can be solved efficiently by Interior Point Method if the problem size is moderate. However, interior point methods cannot successfully scale to large problems, leading us to consider first-order convex optimization methods. We will see that *alternating direction method of multipliers (ADMM)* [15] is well suited to exploit various structures in our problem. In this section, we customize ADMM to solve GDMP with single source and single destination; the extension to the source set and destination set is straightforward and is given in Appendix A.3.

To apply ADMM, we first introduce new variables $u \in \mathbf{R}^{m\times n}$ and $l \in \mathbf{R}^{m\times n\times 2}$, then impose constraints that $u_{i,j} = w_{i,j}$ and $l_{i,j} = (\mathcal{D}\phi)_{i,j}$. Let K denote the direct product of mn second-order cones. The problem (2.19) can be written as

$$\begin{array}{ll} \underset{w,\phi,u,l}{\text{minimize}} & \phi_t - \phi_s + I_K(u,l) \\ \text{subject to} & u = w \\ & l = \mathcal{D}\phi \\ & w \in \mathcal{W}, \end{array} \tag{2.37}$$

where I_K is the indicator function of K

$$I_K(u,l) = \begin{cases} 0, & (u_{i,j}, l_{i,j}) \in K_3 \quad \text{for } i = 1,\dots,m,\ j = 1,\dots,n \\ +\infty, & \text{otherwise.} \end{cases}$$

We decouple the second-order cone constraints and constraint set $\mathcal{W}$ so that projections onto K and $\mathcal{W}$ are separated into two steps. The augmented Lagrangian using scaled dual

variable is

$$L_\rho(w,\phi,u,l,\lambda,\nu) = \phi_t - \phi_s + I_K(u,l) + \frac{\rho}{2}\|w-u+\lambda\|_F^2 + \frac{\rho}{2}\sum_{i,j}\|(\mathcal{D}\phi)_{i,j} - l_{i,j} + \nu_{i,j}\|_2^2,$$

where $\lambda \in \mathbf{R}^{m\times n}$, $\nu \in \mathbf{R}^{m\times n\times 2}$ are scaled dual variables, $\rho \geq 0$ is an algorithm parameter. Then ADMM consists of iterations

- (w,ϕ)-minimization step

$$(w^{(k+1)},\phi^{(k+1)}) = \operatorname*{argmin}_{\phi,w\in\mathcal{W}} L_\rho(w,\phi,u^{(k)},l^{(k)},\lambda^{(k)},\nu^{(k)}).$$

$$\phi^{(k+1)} = \operatorname*{argmin}_{\phi} \frac{\rho}{2}\sum_{i,j}\|(\mathcal{D}\phi)_{i,j} - l_{i,j}^{(k)} + \nu_{i,j}^{(k)}\|_2^2 \tag{2.38}$$

$$w^{(k+1)} = \mathrm{Pr}_{\mathcal{W}}(u^{(k)} - \lambda^{(k)}) \tag{2.39}$$

  where $\mathrm{Pr}_{\mathcal{W}}$ denotes the Euclidean projection onto set $\mathcal{W}$.

- (u,l)-minimization step

$$\begin{aligned}(u^{(k+1)},l^{(k+1)}) &= \operatorname*{argmin}_{u,l,\phi_S,\phi_T} L_\rho(w^{(k+1)},\phi^{(k+1)},u,l,\lambda^{(k)},\nu^{(k)}) \\ &= \mathrm{Pr}_K(w^{(k+1)}+\lambda^{(k)}, \mathcal{D}\phi^{(k+1)}+\nu^{(k)}).\end{aligned} \tag{2.40}$$

- dual variable update

$$\lambda^{(k+1)} = w^{(k+1)} - u^{(k+1)} + \lambda^{(k)}, \tag{2.41}$$

$$\nu^{(k+1)} = \mathcal{D}\phi^{(k+1)} - l^{(k+1)} + \nu^{(k)}. \tag{2.42}$$

We discuss the details of steps (2.38), (2.39) and (2.40) in Section 2.5.3. The primal and dual residuals are

$$\begin{aligned} r_p^{(k+1)} &= (w^{(k+1)} - u^{(k+1)}, \mathcal{D}\phi^{(k+1)} - l^{(k+1)}), \\ r_d^{(k+1)} &= -\rho(u^{(k+1)} - u^{(k)}, \mathcal{D}^*(l^{(k+1)} - l^{(k)})).\end{aligned}$$

We know that when the primal and dual residuals are small, the objective suboptimality must be small, so we use the termination criterion given in [15, Section 3.3.1].

2.5.3 Algorithm details and problem structure

Let us examine the three steps in the ADMM procedure.

Step (2.38): ϕ-minimization step To minimize the function in step (2.38) with respect to ϕ, we set its gradient to zero and obtain a linear system

$$\mathcal{D}^*\mathcal{D}\phi = \mathcal{D}^*(l^{(k)} - \nu^{(k)}). \tag{2.43}$$

Since $\mathcal{D}$ is the forward difference operator as defined in (2.17), $\mathcal{D}^*\mathcal{D}$ is the *discrete Laplace operator*, and (2.43) is the *Poisson equation*. We store ϕ as a column vector $\tilde{\phi}$ of size mn by stacking columns of ϕ atop one another from left to right. The operator $\mathcal{D}^*\mathcal{D}$ has a matrix realization with rank $mn-1$ [30, 50], whose eigenvalues are

$$4 - 2\cos\left(\frac{i-1}{m}\pi\right) - 2\cos\left(\frac{j-1}{n}\pi\right), \quad i = 1, \ldots, m,\ j = 1, \ldots, n.$$

To show the sparsity pattern, we give the matrix realization of $\mathcal{D}^*\mathcal{D}$ on a 3×3 grid

$$\begin{bmatrix}
2 & -1 & & -1 & & & & & \\
-1 & 3 & -1 & & -1 & & & & \\
 & -1 & 2 & & & -1 & & & \\
-1 & & & 3 & -1 & & -1 & & \\
 & -1 & & -1 & 4 & -1 & & -1 & \\
 & & -1 & & -1 & 3 & & & -1 \\
 & & & -1 & & & 2 & -1 & \\
 & & & & -1 & & -1 & 3 & -1 \\
 & & & & & -1 & & -1 & 2
\end{bmatrix}.$$

The Poisson equation (2.43) can be solved very efficiently by many existing fast algorithms and solvers [30], e.g., the Fast Fourier transform.

Step (2.39): projection onto set $\mathcal{W}$ The projection of $u \in \mathbf{R}^{m\times n}$ onto set $\mathcal{W}$ is

$$(\mathrm{Pr}_{\mathcal{W}}(u))_{i,j} = \max(\min(u_{i,j} + \omega, \overline{w}_{i,j}), \underline{w}_{i,j}),$$

where ω is a scalar such that $\sum_{i,j}(\mathrm{Pr}_{\mathcal{W}}(u))_{i,j} = \tilde{w}$. This projection can be computed in $O(N \log N)$ ($N = mn$) operations based on a root finding procedure similar to that discussed in [92].

Step (2.40): projection onto non-overlaping second-order cones We emphasize that the non-overlapping property of these second-order cones is inherited from the non-overlapping partitions defined in Section 3.1. Therefore, these projections can be performed *in parallel* to speed up this step.

The projection onto each cone $K_{n+1} = \{(t, y) \in \mathbf{R}_+ \times \mathbf{R}^n \mid \|y\|_2 \leq t\}$ has an explicit form (e.g., [38, Proposition 3.3])

$$\mathrm{Pr}_{K_{n+1}}(t, y) = \begin{cases} \frac{\|y\|_2 + t}{2\|y\|_2}(\|y\|_2, y) & \text{if } \|y\|_2 > t,\ \|y\|_2 > -t,\ y \neq 0 \\ (t, y) & \text{if } \|y\|_2 \leq t \\ 0 & \text{if } \|y\|_2 \leq -t. \end{cases}$$

2.5.4 Existing algorithm

To the best of our knowledge, the only existing algorithm that can be applied to GDMP is a projected subgradient method (PSGM) developed in [10] [78, Sec. 2.9]. The bottleneck of this method is to compute the subgradient of geodesic distance with respect to the metric. Benmansour et al. [10] developed a method called *subgradient marching* to compute this subgradient based on the single-pass updating scheme of *fast marching methods* [83, 91]. See [10] [78, Sec. 2.9] for more details of this algorithm.

It is well-known that the subgradient method typically converges far more slowly than ADMM, and we observe this in our numerical experiments. A typical performance is shown in Fig. 2.4, which shows the suboptimality of ADMM with $\rho = 0.2$ and $\rho = 0.5$ as well as the suboptimality of PSGM, for a 100×100 GDMP with single source and single destination. The suboptimality of ADMM is $|p^\star - p^{(k)}|$, absolute difference between the current objective value with the optimal objective value, while the suboptimality of PSGM is $p^\star - p^{\mathrm{best}}$, difference between the best objective value found in the current iteration with the optimal objective

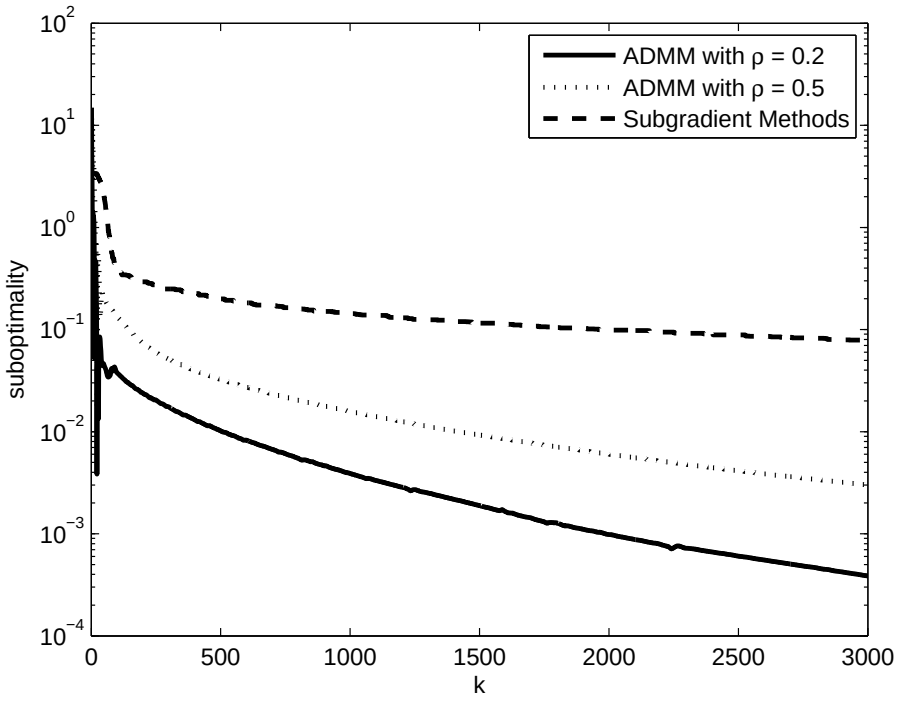
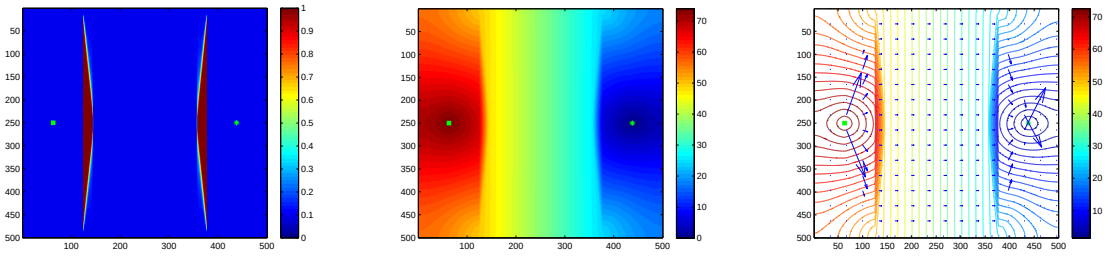


Figure 2.4: Suboptimality of ADMM and PSGM versus iteration

value.

Computational cost In addition, computing the subgradient of the geodesic distance at every iteration of PSGM needs at least, if no more than, $O(N \log N)$ operations on a grid with a total of N points (authors in [10] mention that $O(N^2 \log N)$ operations are needed per iteration). For the ADMM algorithm, the computational bottleneck in each iteration is solving the Poisson equation, which for the setup considered in Section 2.5 takes $O(N \log N)$ operations. We see that ADMM has similar or lower cost per iteration, and also takes far fewer iterations to reach the same level of accuracy in the objective; thus our ADMM outperforms PSGM in these measures.



(a) optimal $w^\star$ (b) optimal $\phi^\star$ (c) optimal $f^\star$ and contours of $\phi^\star$

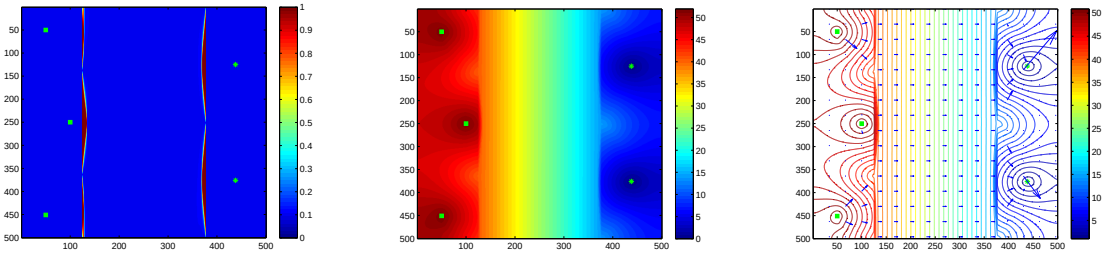
Figure 2.5: Geodesic distance maximization with $m = n = 500$, single source (marked as a green square) and single destination (marked as a green star). The constraints on w are $\underline{w} = 0.1$ in all area, $\overline{w} = 1$ in the middle between columns 125 and 376, and $\overline{w} = 0.1$ on two sides, $\tilde{w} = 0.14mn$.

2.6 Numerical experiments

2.6.1 Illustrative examples

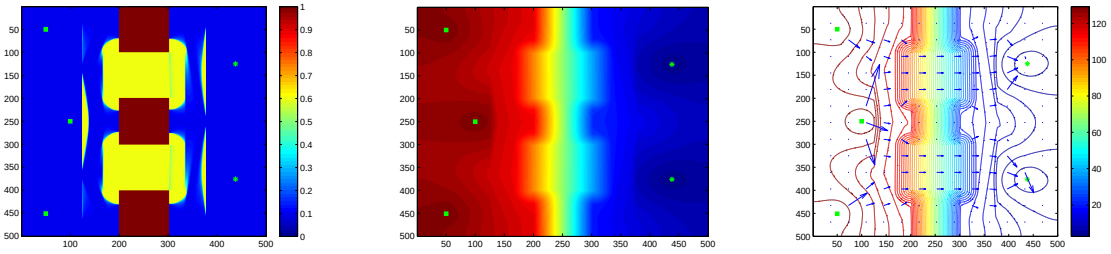
We present illustrative numerical examples for problem (2.19). These small-sized problems are solved with the commercial optimization solver MOSEK 7 [6], and the goal of this section is to illustrate the properties of the solution with different source and destination distributions δ and different constraints $\mathcal{W}$. Larger examples solved by our ADMM algorithm are presented in the next section.

Fig. 2.5 shows an example with a single source and a single destination, and Fig. 2.6 an example with multiple sources and destinations. Fig. 2.8 shows an example with randomly located sources and destinations and demonstrates how $w^\star$ changes as $\tilde{w}$ increases. Some numerical examples of (2.20) are given in Fig. 2.9.



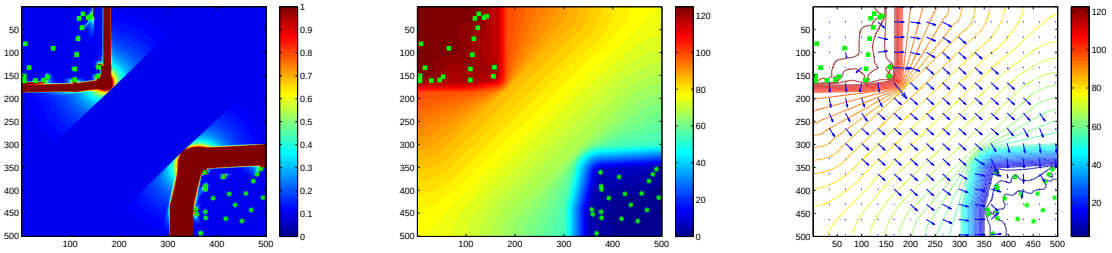
(a) optimal $w^\star$ (b) optimal $\phi^\star$ (c) optimal $f^\star$ and contours of $\phi^\star$

Figure 2.6: Geodesic distance maximization with $m = n = 500$, 3 sources (marked as green squares) and 2 destinations (marked as green stars). The constraints on w are $\underline{w} = 0.1$ in all area, $\overline{w} = 1$ in the middle between columns 125 and 376, and $\overline{w} = 0.1$ on two sides, $\tilde{w} = 0.12mn$.

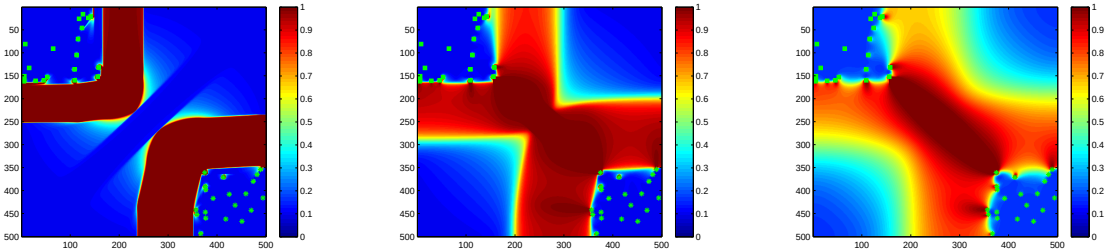


(a) optimal $w^\star$ (b) optimal $\phi^\star$ (c) optimal $f^\star$ and contours of $\phi^\star$

Figure 2.7: Geodesic distance maximization with $m = n = 500$, 3 sources (marked as green squares) and 2 destinations (marked as green stars). Three square-shaped barrier areas are marked red in (a) where $\underline{w}$ and $\overline{w}$ are fixed to 1, a relatively big value. Flows avoid passing through these barrier areas. In rest of area, $\underline{w} = 0.1$, $\overline{w} = 1$ in the middle between columns 125 and 376, and $\overline{w} = 0.1$ on two sides, $\tilde{w} = 0.3mn$.

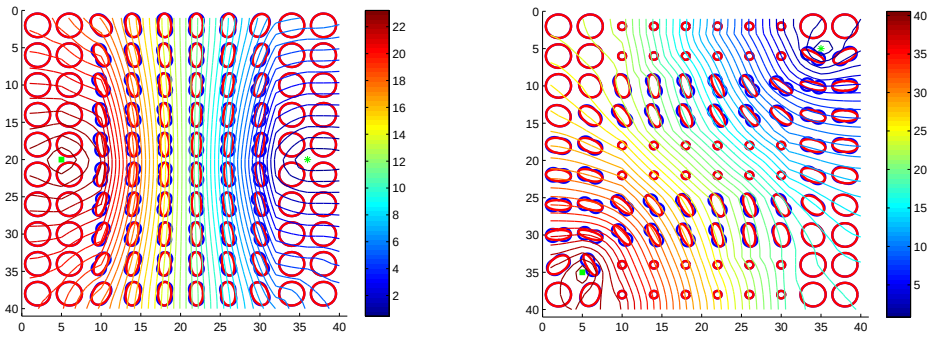


(a) optimal $w^\star$ with $\tilde{w} = 0.2mn$ (b) optimal $\phi^\star$ (c) optimal $f^\star$ and contours of $\phi^\star$



(d) optimal $w^\star$ with $\tilde{w} = 0.4mn$ (e) optimal $w^\star$ with $\tilde{w} = 0.52mn$ (f) optimal $w^\star$ with $\tilde{w} = 0.56mn$

Figure 2.8: Geodesic distance maximization with $m = n = 500$, 20 sources (marked as green squares) randomly located in the left-upper area and 20 destinations (marked as green stars) randomly located in the right-lower area. The constraints on w are $\underline{w} = 0.1$, $\overline{w} = 1$ in all area, $\tilde{w}$ varies.



(a) (b)

Figure 2.9: Anisotropic geodesic distance maximization with $m = n = 40$, single source (marked as green squares) and single destination (marked as green stars). Contours of $\phi^\star$, flow propagation speed profiles $S_v(x)$ (red ellipsoids) and potential expanding speed profile $S_s(x)$ (blue closed curves) for optimal $W^\star$ are plotted. (a) Constraints on W are $\underline{w} = 0.1$ in all area, $\overline{w} = 1$ in the middle between columns 10 and 31, $\overline{w} = 0.1$ in two sides, $\tilde{w} = 0.4$. (b) Constraints on W are $\underline{w} = \overline{w} = 1$ for barrier areas $([1:8, 16:24, 33:40], 10:30)$ and $\underline{w} = 0.1$, $\overline{w} = 1$ for rest of area.

2.6.2 Numerical experiments for ADMM and comparisons

In this section, we give numerical experiments to illustrate the performance of ADMM described in Section 2.5.2, and compare it to the MOSEK which implements a primal-dual interior point method.

The first experiment is a GDMP with single source and single destination. The constraints on w are $\underline{w}(:,:) = 0.1$, $\overline{w}(:,[1:n/4-1, n-n/4+2:n]) = 0.1$, $\overline{w}(:, n/4 : n-n/4+1) = 1$, $\tilde{w} = 0.14mn$. For $m = n = 500$, the optimal $w^\star$, $\phi^\star$ and $f^\star$ are plotted in Fig. 2.5. We initialize ADMM at origin and choose $\epsilon_{\text{abs}} = 2 \times 10^{-5}$, $\epsilon_{\text{rel}} = 10^{-5}$, $\rho^{(0)} = 0.1$. The penalty parameters $\rho^{(k)}$ is updated as

$$\rho^{(k+1)} = \begin{cases} 2\rho^{(k)} & \text{if } \|r_p^{(k)}\|_2 \geq 20\|r_d^{(k)}\|_2 \\ \rho^{(k)}/2 & \text{if } \|r_d^{(k)}\|_2 \geq 20\|r_p^{(k)}\|_2 \\ \rho^{(k)} & \text{otherwise.} \end{cases}$$

We also allow ADMM to run for longer time to see if it can achieve higher accuracy. The codes are run on a PC with Intel Core i7 CPU 2.67GHz and 8GB RAM and Matlab 2013a. The computational results are reported in Table 2.1, where cpu is the CPU runtime in seconds, obj is the objective value at termination, nobj is the normalized objective value of ADMM with respect to the objective value of MOSEK, iter is the number of iterations.

We also run an experiment for GDMP with upwind discretization for 3 sources and 2 destinations. The constraints on w are $\underline{w}(:,:) = 0.1$, $\overline{w}(:,[1:n/4-1, n-n/4+2:n]) = 0.1$, $\overline{w}(:, n/4 : n-n/4+1) = 0.6$, $\tilde{w} = 0.3mn$. To make the example more interesting, we add some barrier areas in which w are fixed to be large values. To be specific, we fix $\overline{w} = \underline{w} = 1$ over the area $([1:m/5, 2m/5:3m/5, 4m/5:m], 2n/5 : n-2n/5+1)$. For $m = n = 500$, the optimal $w^\star$, $\phi^\star$ and $f^\star$ are plotted in Fig. 2.7. We run the ADMM algorithm with the same setup as the previous experiment. The computational results are reported in Table 2.2.

As shown in Table 2.1 and 2.2, ADMM has a fast convergence to a moderate accuracy (2-digit accuracy), but takes a long time to achieve high accuracy as MOSEK does. Due to using the structures discussed above, the computation in each step of ADMM is fairly cheap,

	MOSEK	ADMM	ADMM (higher accuracy)
$m \times n$	cpu/obj/nobj	cpu/obj/nobj/iter	cpu/obj/nobj/iter
100×100	2.1/1.487+1/1	2.8/1.486+1/0.999/355	4.8/1.486+1/0.999/630
300×300	30.9/4.443+1/1	41.3/4.423+1/0.995/485	57.9/4.428+1/0.997/663
500×500	138.6/7.399+1/1	129.6/7.339+1/0.992/428	201.1/7.358+1/0.994/671
700×700	357.9/1.035+2/1	155.8/1.022+2/0.987/246	453.3/1.028+2/0.993/710
1000×1000	885.8/1.479+2/1	125.8/1.453+2/0.982/99	1082.61/1.467+2/0.992/837
1500×1500	3033.4/2.217+2/1	336.3/2.177+2/0.982/113	2510.15/2.194+2/0.990/812
1800×1800	out of memory	688.9/2.611+2/*/124	5755.4/2.637+2/*/1265

Table 2.1: The CPU runtime (cpu), objective values (obj) and normalized objective values (nobj) of MOSEK and ADMM, and the number of iterations (iter) of ADMM (with forward difference discretization).

consequently, ADMM can solve very large scale problems for which MOSEK often runs out of memory.

2.7 Conclusions

The geodesic distance maximization problem arises in a wide variety of fields, e.g., physics, transportation, and resource allocation. In this paper, we set up a general GDMP by defining partitions on graph edges and proposing several general cost functions that are motivated by various applications and physical properties of flows. More importantly, we discovered that the dual formulations of GDMP with those cost functions are joint convex with respect to weight and potential, which enables simultaneous optimization and efficient numerical methods. This optimization framework exhibits a correspondence, shown in Table 2.3, between physical models and optimization models: more general flow propagation models require more advanced optimization problems. We developed an efficient ADMM algorithm by ex-

	MOSEK	ADMM	ADMM (higher accuracy)
$m \times n$	cpu/obj/nobj	cpu/obj/nobj/iter	cpu/obj/nobj/iter
100×100	6.6/2.578+1/1	4.8/2.577/0.999/467	7.4/2.578+1/1/740
300×300	159.6/7.893+1/1	84.9/7.877+1/0.998/764	135.9/7.888+1/0.999/1190
500×500	494.0/1.319+2/1	358.5/1.312+2/0.995/875	545.3/1.317+2/0.998/1311
1000×1000	5325.1/2.646+2/1	1940.4/2.571+2/0.972/1091	3024.3/2.627+2/0.993/1678
1500×1500	out of memory	1687.8/3.215+2/*/405	14815.5/3926+2/*/3355

Table 2.2: The CPU runtime (cpu), objective values (obj) and normalized objective values (nobj) of MOSEK and ADMM, and the number of iterations (iter) of ADMM (with upwind discretization).

ploiting problem structures to solve large-scale SOCP formulation of GDMP on a regular grid. Our algorithm performs much faster than the existing algorithm as well as scales to much larger problems than MOSEK.

There are several directions for future research. One direction is to explore formulations based on the general Hamilton-Jacobi equation, where the eikonal equation is a special case. In the context of games, there are other examples similar to the interdiction game that can be considered; for example, ambush games. On the computational side of GDMP, the structure of the SDP formulation can be exploited to develop fast algorithms for the anisotropic problem. Parallel and distributed implementations for the SDP and SOCP will be helpful to develop. Finally, there are many more potential application areas to explore, including material science (design of meta-materials), machine learning (learning the geometry to the data with metric constraints), controlling the spread of epidemics, as well as spread of information through social networks.

cost function	dual form	optimization
$h_k(W_k, f_k) = \|W_k^{1/2} f_k\|_2$	(2.20)	semidefinite program
$h_k(w_k, f_k) = w_k \|f_k\|_2$	(2.10)	second-order cone program
$h_k(w_k, f_k) = w_k \|f_k\|_1$ or $w_k \|f_k\|_\infty$	(2.10)	linear program

Table 2.3: Optimization framework

Chapter 3

DISTRIBUTED OPTIMIZATION VIA ALTERNATING DIRECTION METHODS OF MULTIPLERS

3.1 Introduction

Distributed optimization arises in a variety of applications, e.g. distributed tracking and localization, estimation problems in sensor networks, multi-agent coordination; see recent surveys [15, 69, 72] and papers [63, 68, 96] for more applications. Recently, there has been significant interests in applying primal-dual algorithms, particularly Alternating Direction Method of Multiplers (ADMM) and its variants, on distributed optimization with strict local communication [23, 53, 56, 87, 94]. These works are motivated by recent progress on convergence analysis on centralized ADMM and multi-block ADMM algorithms. In contrast to subgradient type algorithms [32, 65, 70, 71], ADMM-type algorithms tend to achieve faster convergence rate which is highly preferred when local optimization at each node can be solved efficiently. A unified and extensive literature review is given in Section 3.2.3.

Given an undirected connected graph with n nodes, distributed optimization over the graph is to solve

$$\underset{y}{\text{minimize}} \quad \sum_{i=1}^{n} f_i(y) \tag{3.1}$$

where each $f_i : \mathbf{R}^d \to \mathbf{R}$ is locally known by node i only. Without loss of generality, we suppose $d = 1$ to avoid cumbersome notations. Each node only does local computation and communicate with its immediate neighbors.

The contribution of this paper is as follows. We propose to use (weighted) graph matrices to impose consensus constraints and reformulate distributed optimization as a linear constrained optimization problem. The use of graph matrices naturally brings in graph weights and topology structure. Based on this reformulation, we develop a weighted proximal ADMM

for distributed optimization and link its convergence rate to graph topology. In contrast to existing algorithms, this fully distributed algorithm is simultaneous, single-loop, and more general. More importantly, our weighted setting and proximal term enable us to jointly design the graph weights and algorithm parameters to further improve the convergence.

The rest of the paper is organized as follows. Section 4.2 reformulates distributed optimization (3.1) using weighted graph matrices and provides a unified overview on existing algorithms. In Section 3.3, we develop a weighted proximal ADMM for distributed optimization and present the convergence results, followed by discussion on different choices of parameters and graph design. Section 4.7 presents numerical examples, and Section 3.5 concludes with discussion of future work.

3.2 Background and formulation

3.2.1 Distributed optimization formulation using weighted graph matrices

As we observed, the nullspaces of the incidence matrix and Laplacian have nice properties that can be used to naturally impose consensus constraints

$$x_1 = \ldots = x_n \quad \Leftrightarrow \quad \mathcal{L}x = 0 \quad \Leftrightarrow \quad \mathcal{I}x = 0$$

where $x = [x_1, \ldots, x_n]^T$. $\mathcal{I}x = 0$ essentially defines two weighted constraints for each undirected edge $\sqrt{\frac{w_{ij}}{2}}(x_i - x_j) = 0$ and $\sqrt{\frac{w_{ij}}{2}}(x_j - x_i) = 0$ for $\{i, j\} \in E$. $\mathcal{L}x = 0$ is equivalent to $\sum_{j \in \mathcal{N}(i)} w_{ij}(x_i - x_j) = 0, i \in V$. Then we can reformulate (3.1) using the incidence matrix

$$\begin{aligned} \underset{x_1,\ldots,x_n}{\text{minimize}} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & \mathcal{I}x = 0, \end{aligned} \tag{3.2}$$

or using the graph Laplacian

$$\begin{aligned} \underset{x_1,\ldots,x_n}{\text{minimize}} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & \mathcal{L}x = 0. \end{aligned}$$

Although both reformulations can be used to develop distributed algorithms, we will focus on (3.2) in this paper due to limited space.

3.2.2 Standard and multi-block ADMM

The standard two-block ADMM is suited to solve the optimization problem of the form

$$\begin{array}{ll} \underset{x,z}{\text{minimize}} & f(x)+g(z) \\ \text{subject to} & Ax+Bz=c. \end{array}$$

The augmented Lagrangian is $L_\rho(x,z,\mu)=f(x)+g(z)+\mu^T(Ax+Bz-c)+\frac{\rho}{2}\|Ax+Bz-c\|_2^2$, where μ is the dual variable, ρ is a parameter. Each loop of ADMM consists of alternatingly minimizing the augmented Lagrangian with respect to x and z, and a dual update on μ. The ADMM iterations are

$$\begin{aligned} x^{(k+1)} &= \underset{x}{\text{argmin}}\ L_\rho(x,z^{(k)},\mu^{(k)}), \\ z^{(k+1)} &= \underset{z}{\text{argmin}}\ L_\rho(x^{(k+1)},z,\mu^{(k)}), \\ \mu^{(k+1)} &= \mu^{(k)}+\rho(Ax^{(k+1)}+Bz^{(k+1)}-c). \end{aligned}$$

Separable structure is critical in developing distributed algorithms. A function is called *separable* if it can be written as the sum of terms of independent variables, so that each term can be minimized independently. The augmented Lagrangian is generally not separable even if the objective function and linear terms are separable. This non-separable structure is mainly due to the (quadratic) augmented Lagrangian term $\|Ax+Bz-c\|_2^2$, since A^TA or B^TB are generally not diagonal.

3.2.3 Literature review

Almost all development of distributed ADMM for distributed optimization (3.1) starts with defining local copies x_i of global variable y for each node $i=1,\ldots,n$, and enforcing consensus $x_1=\ldots=x_n$ by imposing some linear constraints $Ax=0$ or $Ax+Bz=0$, where z captures slack variables introduced in some cases. In this way, the objective function is fully decoupled/separable; more importantly, (3.1) is reformulated as a linear constrained optimization problem which has the form better suited for ADMM. Several different ways

of imposing linear constraints have been proposed recently [53, 56, 87, 94]. Although mathematically equivalent, they often lead to very different algorithm routines because of different forms of augmented Lagrangian as well as different techniques applied to decouple the augmented terms. In this section, we carefully review and compare these reformulations and distributed ADMM routines with the goal of providing a systematic and unified view.

The first way to impose consensus constraints is to first assign a pre-determined order to all nodes, then set

$$x_i = x_j, \quad \{i, j\} \in E,\ i < j.$$

Here ordering all nodes essentially turns the undirected graph into a directed one as we can assign directions to edges using node orders, for example, pointing each edge from small-indexed node to large-indexed node. This reformulation was adopted in the papers by Wei and Ozdaglar [94] and Jakovetic et al. [56], which attempted to develop multi-block ADMM for (3.1). The constraints can be compactly written as $Ax = 0$, where $A \in \mathbf{R}^{m\times n}$ is the (unweighted) incidence matrix defined for the directed graph, $A_{(i,j),i} = 1$ and $A_{(i,j),j} = -1$. The augmented Lagrangian for this reformulation is $L_\rho(x, \mu) = \sum_{i=1}^n f_i(x_i) + \mu^T Ax + \frac{\rho}{2}\|Ax\|_2^2$. The first two terms $\sum_{i=1}^n f_i(x_i)$ and $\mu^T Ax$ have separable structures, especially the second term can be decomposed as $\mu^T Ax = \sum_{i=1}^n (\mu^T a_i)x_i$, where a_i is the ith column of A. However, as we mentioned earlier, the augmented term $\|Ax\|_2^2 = x^T A^T Ax$ is a non-separable quadratic term since $A^T A$ is not diagonal.

The existing works [94] and [56] can be viewed as using different techniques to approximate the augmented term by a separable quadratic term: [94] applied the Gauss-Seidel method while [56] attempted the multi-step Jacobi method. In [56], Jakovetic et al. proposed a *double loop* algorithm: $x_1, \ldots, x_n$ are updated simultaneously but multiple times before a dual update. This work can be viewed as approximating the augmented term in a Jacobi way, that is, while x_i is being updated, all coupled variables x_j, $j \in \mathcal{N}(i)$ (i.e. variables at its neighbors) are fixed using some old value $\hat{x}_j$. This can be viewed as approximating the

augmented term at $\hat{x}$ as

$$\begin{aligned} & x^T \mathbf{diag}(A^T A)x + 2x^T(A^T A - \mathbf{diag}(A^T A))\hat{x} + \text{const} \\ = & \sum_{i=1}^{n} \sum_{j \in \mathcal{N}(i)} (x_i - \hat{x}_j)^2 + \text{const} \end{aligned} \tag{3.3}$$

where $\mathbf{diag}(\cdot)$ denotes the diagonal part of a matrix. The approximated quadratic form has separable structure since the matrix of quadratic term, $\mathbf{diag}(A^T A)$, is diagonal. Therefore, minimizing this approximated term can be done simultaneously with respect $x_1, \ldots, x_n$.

In [94], Wei and Ozdaglar proposed a *sequential* distributed ADMM: $x_1, \ldots, x_n$ are updated in a sequential order followed by a dual update. This work can be viewed as approximating the augmented term in a Gauss Seidel way. Denote $\mathbf{ltri}(\cdot)$ and $\mathbf{utri}(\cdot)$ strict lower and upper triangular parts of a matrix, respectively. We replace $(A^T A - \mathbf{diag}(A^T A))\hat{x} = (\mathbf{utri}(A^T A) + \mathbf{ltri}(A^T A))\hat{x}$ in (3.3) by a better approximation $\mathbf{utri}(B)\hat{x} + \mathbf{ltri}(B)\tilde{x}$, where $\tilde{x}$ stores the most recent values of x, which is generated in the same sequential order as x_i are updated. We can write it as the sum of terms of single variable

$$\begin{aligned} & x^T \mathbf{diag}(A^T A)x + 2x^T(\mathbf{ltri}(A^T A)\tilde{x} + \mathbf{utri}(A^T A)\hat{x}) + \text{const} \\ = & \sum_{i=1}^{n} \left(\sum_{j \in \mathcal{N}(i), j<i} (x_i - \tilde{x}_j)^2 + \sum_{j \in \mathcal{N}(i), j>i} (x_i - \hat{x}_j)^2 \right) + \text{const}. \end{aligned}$$

As $A^T A$ is decomposed into upper and lower triangular parts, for each x_i, all coupled variables x_j with smaller index $j < i$ have been updated and are set to more recent values $\tilde{x}_j$, while other coupled variables with $j > i$ are yet to be updated and still set to old values $\hat{x}_j$. The sequential order update is likely to cause long waiting time: node i has to wait until all nodes with smaller indices have been updated.

Second group of works impose consensus constraints by introducing slack variables, and treat original and slack variables as two blocks of variables so that standard two-block ADMM can be applied. One of the early examples is by Boyd et al. [15] which introduces a single slack variable z and set all variables equal to z,

$$x_i = z, \quad i = 1, \ldots, n$$

or compactly written as $Ax + Bz = 0$ with $A = I, B = -\mathbf{1}$. The algorithm routines given by simply applying standard ADMM are not fully distributed since z-update is a global averaging conducted by a central collector.

To the best of our knowledge, the best existing way to introduce slack variables is

$$x_i = z_{ij}, \quad x_j = z_{ji}, \quad z_{ij} = z_{ji}, \quad \{i, j\} \in E$$

or it can be compactly represented as $Ax + Bz = 0$ with $B = -I$ and A has a good structure that is each row of A has only one nonzero entry $A_{(i,j),i} = 1$, so $A^T A$ is a diagonal matrix. This way of imposing constraints has been seen in recent papers [23, 53, 64, 87] and can be traced back to earlier work [68]. The partial augmented Lagrangian is $L_\rho(x, z, \mu) = \sum_i^n f_i(x_i) + \mu^T(Ax - z) + \frac{\rho}{2}\|Ax - z\|_2^2$. Clearly, this partial augmented Lagrangian is separable in x as $A^T A$ is diagonal, and is also separable in z, thus we can just apply standard two-block ADMM and get the routines given in Algorithm 1. As we will see in Section 3.3.2, this algorithm is a special case of our algorithm developed later in Section 3.3, and can be further improved by designing graph weights and algorithm parameters as discussed in Section 2.

3.3 Distributed proximal ADMM

In this section we will use formulation (3.2) to develop a proximal ADMM for distributed optimization (3.1) motivated by recent work [31]. The Lagrangian of (3.2) is

$$L(x, \mu) = \sum_{i=1}^{n} f_i(x_i) + \mu^T \mathcal{I} x \tag{3.4}$$

where $\mu \in \mathbf{R}^{2m}$ is the vector of dual variables. The augmented Lagrangian is

$$\begin{aligned} L_\rho(x, \mu) &= \sum_{i=1}^{n} f_i(x_i) + \mu^T \mathcal{I} x + \frac{\rho}{2}\|\mathcal{I} x\|_2^2 \\ &\overset{(1.1)}{=} \sum_{i=1}^{n} f_i(x_i) + \sum_{i=1}^{n} (\mu^T \mathcal{I})_i x_i + \frac{\rho}{2} x^T \mathcal{L} x, \end{aligned}$$

where $(\mu^T \mathcal{I})_i = \sum_{j \in \mathcal{N}(i)} \sqrt{\frac{w_{ij}}{2}}(\mu_{ij} - \mu_{ji})$. Each node is associated with a primal variable $x_i \in \mathbf{R}$, and two dual variables μ_{ij} and μ_{ji} are defined for each edge $\{i, j\}$.

Algorithm 1 An existing distributed ADMM [23, 53, 64, 68, 87]

input: initial point $x^{(0)}$, $\mu^{(0)}$ with $\mu_{ij}^{(0)} + \mu_{ji}^{(0)} = 0$

for $k = 0, 1, 2, \ldots$ **do**

repeat

each node i updates x_i simultaneously

$$x_i^{(k+1)} = \underset{x_i}{\operatorname{argmin}}\, f_i(x_i) + \sum_{j \in \mathcal{N}(i)} \mu_{ij}^{(k)}(x_i - z_{ij}^{(k)}) + \frac{\rho}{2} \sum_{j \in \mathcal{N}(i)} (x_i - z_{ij}^{(k)})^2.$$

each node i updates $z_{ij}, j \in \mathcal{N}(i)$ simultaneously

$$z_{ij}^{(k+1)} = \frac{1}{2}(x_i^{(k+1)} + x_j^{(k+1)}).$$

each node i computes $\mu_{ij}, j \in \mathcal{N}(i)$

$$\mu_{ij}^{(k+1)} = \mu_{ij}^{(k)} + \frac{\rho}{2}(x_i^{(k+1)} - x_j^{(k+1)}).$$

until

end for

3.3.1 Algorithm design

The coupling relation in the augmented Lagrangian comes from the term $\|\mathcal{I}x\|_2^2 = x^T\mathcal{L}x$. We can approximate the term in a Jacobi way with a proximal term added

$$\begin{aligned} x^T\mathcal{L}x \approx\ & \hat{x}^T\mathcal{L}\hat{x} + 2(x-\hat{x})^T\mathcal{L}\hat{x} \\ & +(x-\hat{x})^T\,\mathbf{diag}(\mathcal{L})(x-\hat{x}) + (x-\hat{x})^T P(x-\hat{x}) \end{aligned}$$

where $\hat{x}$ is a fixed estimate of x and $P = \mathbf{diag}\{p_1, \ldots, p_n\}$ with $p_i \geq 0$ is a diagonal matrix in the proximal term. This approximation gives us

$$\begin{aligned} & \operatorname*{argmin}_x x^T\mathcal{L}x \\ \approx\ & \operatorname*{argmin}_x x^T(\mathbf{diag}(\mathcal{L}) + P)x + 2x^T(\mathcal{L} + P - \mathbf{diag}(\mathcal{L}))\hat{x} \\ =\ & \operatorname*{argmin}_x \sum_{i=1}^{n} \left(\sum_{j\in\mathcal{N}(i)} w_{ij}\left(x_i - \hat{x}_j\right)^2 + p_i\left(x_i - \hat{x}_i\right)^2 \right). \end{aligned}$$

Alternatively, we can interpret the approximation from the graph structure. Recall that $x^T\mathcal{L}x = \sum_{\{i,j\}\in E} w_{ij}(x_i - x_j)^2$. The variable x_i only appears in terms $w_{ij}(x_i - x_j)^2$, $j \in \mathcal{N}(i)$, that is x_i is only coupled with its neighbors. Therefore, when optimizing with respect to x_i, we fix all variables $x_j, j \in \mathcal{N}(i)$ using some values $\hat{x}_j$ from the previous iteration, that is $w_{ij}(x_i - \hat{x}_j)^2$.

For each undirected edge $\{i, j\}$, there are two dual variables μ_{ij} and μ_{ji} defined. The updates for μ_{ij} and μ_{ji} are $\mu_{ij}^{(k+1)} = \mu_{ij}^{(k)} + \rho\sqrt{\frac{w_{ij}}{2}}\left(x_i^{(k+1)} - x_j^{(k+1)}\right)$ and $\mu_{ji}^{(k+1)} = \mu_{ji}^{(k)} + \rho\sqrt{\frac{w_{ij}}{2}}\left(x_j^{(k+1)} - x_i^{(k+1)}\right)$. Therefore, if $\mu_{ij}^{(0)} + \mu_{ji}^{(0)} = 0$, then $\mu_{ij}^{(k)} + \mu_{ji}^{(k)} = 0$. Finally, we also multiply dual variables μ by $\sqrt{2}$ such that constants in the algorithm become simpler. The full algorithm is shown in Algorithm 2.

3.3.2 A special case $P = D$

In Section 3.3.3, we will give a general condition on P that guarantees the convergence, and discuss different choices of P. Here we point out that if p_i of node i is set to be equal

Algorithm 2 A distributed proximal ADMM

input: initial point $x^{(0)}$, $\mu^{(0)}$ with $\mu_{ij}^{(0)} + \mu_{ji}^{(0)} = 0$

for $k = 0, 1, 2, \ldots$ **do**

repeat

each node updates x_i simultaneously

$$x_i^{(k+1)} = \underset{x_i}{\operatorname{argmin}} f_i(x_i) + \left(\sum_{j \in \mathcal{N}(i)} \sqrt{w_{ij}} \mu_{ij}^{(k)} \right) x_i + \frac{\rho}{2} \sum_{j \in \mathcal{N}(i)} w_{ij} \left(x_i - x_j^{(k)} \right)^2 + \frac{\rho p_i}{2} \left(x_i - x_i^{(k)} \right)^2 \quad (3.5)$$

each node updates $\mu_{ij}, j \in \mathcal{N}(i)$ simultaneously

$$\mu_{ij}^{(k+1)} = \mu_{ij}^{(k)} + \rho \sqrt{w_{ij}} \left(x_i^{(k+1)} - x_j^{(k+1)} \right). \quad (3.6)$$

until

end for

to its degree d_i, which is feasible but not optimal as discussed in Section 3.3.4, Algorithm 2 can be viewed as a weighted version of the existing Algorithm 1. To elaborate it, we set $P = D$ and complete the square in (3.5) as $\sum_{j \in \mathcal{N}(i)} w_{ij} \left(\left(x_i - x_j^{(k)} \right)^2 + \left(x_i - x_i^{(k)} \right)^2 \right) = \sum_{j \in \mathcal{N}(i)} 2 w_{ij} \left(x_i - \left(x_i^{(k)} + x_j^{(k)} \right) / 2 \right)^2 + \text{const}$. Then after eliminating constant terms, x_i-update (3.5) turns to be

$$x_i^{(k+1)} = \underset{x_i}{\operatorname{argmin}} \; f_i(x_i) + \left(\sum_{j \in \mathcal{N}(i)} \sqrt{w_{ij}} \mu_{ij}^{(k)} \right) x_i + \rho \sum_{j \in \mathcal{N}(i)} w_{ij} \left(x_i - \frac{x_i^{(k)} + x_j^{(k)}}{2} \right)^2.$$

We can define a new sequence $z^{(k)}$ as $z_{ij}^{(k+1)} = \frac{1}{2}(x_i^{(k+1)} + x_j^{(k+1)})$.

3.3.3 Convergence results

To establish the convergence of Algorithm 2, we make the following assumptions.

Assumption 1. *1. Functions $f_i : \mathbf{R} \to (-\infty, +\infty]$ are closed, proper and convex.*

2. *There exists a saddle point* $(x^\star, \mu^\star)$ *which satisfies the KKT conditions of the Lagrangian given in (3.4) (with scaled dual variable* μ*)*

$$\mathcal{I}x^\star = 0, \quad \textit{i.e. } x_1^\star = \ldots = x_n^\star, \tag{3.7}$$

$$-\frac{1}{\sqrt{2}}\mathcal{I}^T\mu^\star \in \partial f(x^\star). \tag{3.8}$$

3. *Assume that the subgradient set* ∂f *at the optimal* $x^\star$ *is bounded (with respect to* ℓ_2 *norm square), that is, there is* $C \geq 0$*, for any* $g \in \partial f(x^\star)$*,* $\|g\|_2^2 \leq C$*.*

The following lemma follows immediately from Assumption 1 to bound the optimal dual variable $\mu^\star$.

Lemma 1. *Denote* $\mathcal{R}(\mathcal{I})$ *and* $\mathcal{N}(\mathcal{I})$ *the range and nullspace of* $\mathcal{I}$*. If* $\mu^{(0)} = 0$ *or* $\mu^{(0)} \in \mathcal{R}(\mathcal{I})$*, then*

$$\|\mu^\star\|_2^2 \leq \frac{2C}{\lambda_2(\mathcal{L})}. \tag{3.9}$$

All proofs in this section are deferred to the supplement materials due to limited space. Let $x^{(1)}, \ldots, x^{(K)}$ be the sequence generated by updates (3.5). Define the running average after K iterations $\bar{x}^{(K)} = \frac{1}{K}\sum_{k=1}^K x^{(k)}$. In the literature, a variety of quantities have been used to measure the convergence of ADMM. We will adopt two measures here: the duality gap

$$L(\bar{x}^{(K)}, \mu^\star) - L(x^\star, \mu^\star) = f(\bar{x}^{(K)}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}\bar{x}^{(K)} \tag{3.10}$$

which is also the measure introduced in [48]; and the quantity according to [47]

$$\|x^{(K+1)} - x^{(K)}\|_H^2 + \frac{1}{2\rho}\|\mu^{(K+1)} - \mu^{(K)}\|_2^2, \tag{3.11}$$

which measures the change of primal and dual points at two consecutive iterations. We first show that the duality gap (3.10) converges at the rate $O(1/K)$ which gives an ergodic convergence rate for Algorithm 2. Denote $H = P + W$ and $x^\star = \beta\mathbf{1}$.

Theorem 1. *Under Assumption 1, if the diagonal matrix P is chosen such that H is positive semidefinite, then*

$$f(\bar{x}^{(K)}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}\bar{x}^{(K)} \leq \frac{1}{K}\left(\frac{1}{2\rho}\|\mu^\star - \mu^{(0)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(0)}\|_H^2\right).$$

If $x^{(0)} = 0$, $\mu^{(0)} = 0$,

$$\begin{aligned} f(\bar{x}^{(K)}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}\bar{x}^{(K)} &\leq \frac{1}{K}\left(\frac{1}{2\rho}\|\mu^\star\|_2^2 + \frac{\rho}{2}\|x^\star\|_H^2\right) \\ &\leq \frac{1}{K}\left(\frac{C}{\rho\lambda_2(\mathcal{L})} + \frac{\rho\beta^2}{2}\left(\mathbf{Tr}(P) + 2\mathrm{vol}(G)\right)\right). \quad (3.12) \end{aligned}$$

Here we use the fact that $\|x^\star\|_H^2 = \beta^2\mathbf{1}^T H\mathbf{1} = \beta^2\mathbf{1}^T(P+W)\mathbf{1} = \beta^2(\mathbf{Tr}(P) + \mathrm{vol}(G))$.

Next we show that the quantity (3.11), which was used in [47] to analyze the convergence of centralized ADMM, decreases at the rate $O(1/K)$.

Theorem 2. *Under the conditions of Theorem 1,*

$$\|x^{(K+1)} - x^{(K)}\|_H^2 + \frac{1}{2\rho}\|\mu^{(K+1)} - \mu^{(K)}\|_2^2 \leq \frac{1}{K}\left(\frac{1}{2\rho^2}\|\mu^\star - \mu^{(0)}\|_2^2 + \|x^\star - x^{(0)}\|_H^2\right).$$

If $x^{(0)} = 0$, $\mu^{(0)} = 0$,

$$\|x^{(K+1)} - x^{(K)}\|_H^2 + \frac{1}{2\rho}\|\mu^{(K+1)} - \mu^{(K)}\|_2^2 \leq \frac{1}{K}\left(\frac{C}{\rho^2\lambda_2(\mathcal{L})} + \beta^2\left(\mathbf{Tr}(P) + \mathrm{vol}(G)\right)\right).$$

Although the convergence only requires P such that $H = P + W$ positive semidefinite, we next discuss some particular choice of P and how to optimize P and graph weight W in order to achieve improved convergence.

3.3.4 Design of W and P

1. **Fix $P = D$ and optimize W** Clearly, $P = D$ is a feasible choice for P since H is positive semidefinite. It can be easily checked, for any $x \in \mathbf{R}^n$, $x^T(D+W)x = \sum_{\{i,j\}\in E} w_{ij}(x_i + x_j)^2 \geq 0$. Actually, H is called the signless Laplacian in the spectral graph theory. As discussed in Section 3.3.2, if $P = D$, Algorithm 2 reduces to the weighted version of the existing algorithm 1. However, we can further boost the

convergence by designing graph weight W. In order to design W, we minimize the constant part in the convergence rate bound (3.12)

$$\begin{array}{ll} \underset{\mathcal{L},W}{\text{maximize}} & \lambda_2(\mathcal{L}) \\ \text{subject to} & \mathcal{L} = \mathbf{diag}(W\mathbf{1}) - W \\ & W \in \mathcal{W} \end{array} \tag{3.13}$$

where $\mathcal{W}$ is the set of W satisfying graph constraints

$$\begin{aligned} \mathcal{W} \;=\; & \{W \mid W = W^T, w_{ij} = 0 \text{ for } \{i,j\} \notin E, \\ & w_{ij} \geq 0 \text{ for } \{i,j\} \in E, \mathbf{1}^T W \mathbf{1} = \text{vol}(G)\} \end{aligned}$$

This is a convex optimization problem and can be formulated as a semidefinite program similar to [14].

2. **Fix** $P = \alpha I - D, \alpha \geq \lambda_n(\mathcal{L})$**, and optimize** W

We can verify this choice of P is valid since $p_i \geq 0$ and $H \succeq 0$. First $H = \alpha I - D + W = \alpha I - \mathcal{L} = U(\alpha I - \Lambda)U^T$. Since $\alpha \geq \lambda_n(\mathcal{L})$, for any $x \in \mathbf{R}^n$, $x^T H x \geq (\alpha - \lambda_n(\mathcal{L}))\|x\|_2^2 \geq 0$, so H is positive semidefinite as desired. From the literature on bounding the largest Laplacian eigenvalue, we know that $\lambda_n(\mathcal{L}) \geq 1 + \max_{i \in V} d_i$, thus $p_i > 0$. Choosing $\alpha = \lambda_n(\mathcal{L})$, we minimize the constant part in the convergence rate bound (3.12)

$$\begin{array}{ll} \underset{\mathcal{L},W}{\text{minimize}} & \dfrac{C}{\rho\lambda_2(\mathcal{L})} + \dfrac{\rho}{2}\lambda_n(\mathcal{L})\beta^2 n \\ \text{subject to} & \mathcal{L} = \mathbf{diag}(W\mathbf{1}) - W \\ & W \in \mathcal{W} \end{array} \tag{3.14}$$

which is also a convex optimization problem. β controls the trade-off between optimizing $\lambda_2(\mathcal{L})$ and $\lambda_n(\mathcal{L})$. In practice, we can substitute β by an upper bound that may be found by numerical methods, so the objective function remains the constant part of an upper bound on the convergence rate of the algorithm.

3. **Jointly optimize P and W** We can jointly optimize diagonal P and W by solving a convex optimization problem and explicitly impose constraints such that $p_i \geq 0$ and H positive semidefinite,

$$\begin{array}{ll} \underset{\mathcal{L},W,P}{\text{minimize}} & \dfrac{C}{\rho\lambda_2(\mathcal{L})} + \dfrac{\rho\beta^2}{2}\mathbf{Tr}(P) \\ \text{subject to} & \mathcal{L} = \mathbf{diag}(W\mathbf{1}) - W \\ & P + W \succeq 0 \\ & p_i \geq 0, \quad i \in V \\ & W \in \mathcal{W}. \end{array} \tag{3.15}$$

3.4 Numerical examples

In this section, we present numerical examples to demonstrate the algorithm performance with different choices of P and performance improvement by designing P and graph weight W. Consider a simple least square problem

$$f(y) = \frac{1}{n}\sum_{i=1}^{n}(c_i - b_i y)^2$$

where each node only has access to data b_i and c_i. We generate the data $c = By^\star + \xi$, where y and B are generated randomly from standard Gaussian distribution, ξ is Gaussian noise with standard deviation 0.1. We initialize the graph with weights uniformly sampled from $[1, 10]$. The algorithm is tested by four different choices of P and graph weight W:

1. initial weight W and $P = D$,

2. designed weight $W^\star$ by solving optimization (3.13) and $P = D^\star$,

3. designed weight $W^\star$ by solving optimization (3.14) and $P = \lambda_n(\mathcal{L}^\star)I - D^\star$,

4. designed weight $W^\star$ and $P^\star$ by solving optimization (3.15).

We test the algorithm on a graph with 500 nodes. Each pair of nodes are connected by an edge with probability 0.5. The convergence of duality gap (3.10) is shown in Fig. 3.1. As seen from Fig. 3.1, the algorithm with the jointly designed weight W and P achieves the fastest convergence while the choice of initial W and $P = D$ gives the slowest convergence.

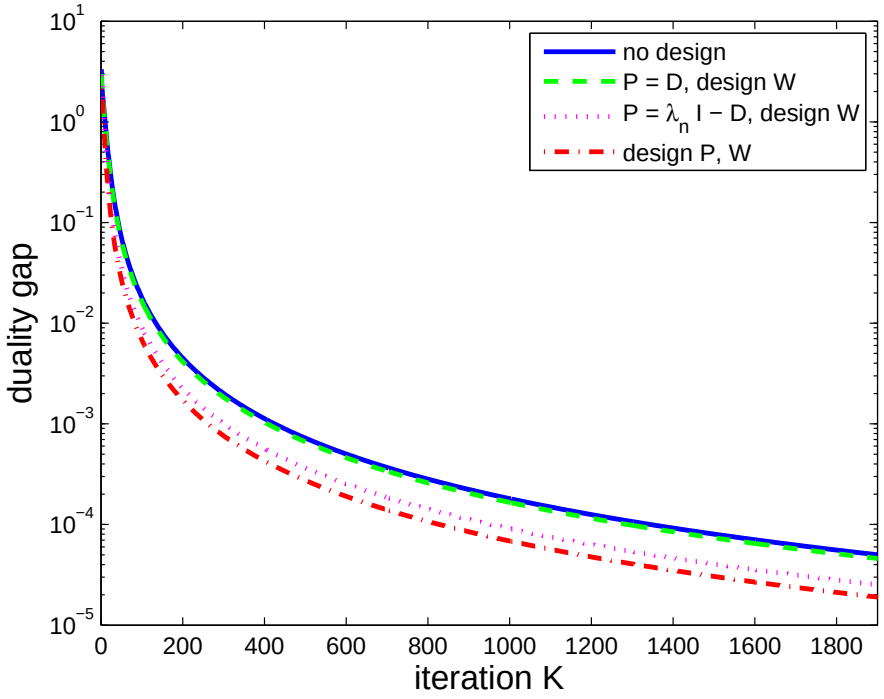


Figure 3.1: The duality gap versus iterations.

3.5 Conclusion and future work

In this paper, we developed a weighted proximal ADMM algorithm to distributed optimization and link the convergence rate to the graph topology. One key in our development is formulating consensus constraints using (weighted) graph matrices. The best existing ADMM-type algorithm for distributed optimization can be viewed a special case of our algorithm with particular choice of parameters. The weighted graph setting and proximal term

enables us to design both graph and algorithm parameters to further improve the convergence. Numerical examples show that jointly designing graph weights and proximal term gives the best performance.

There are several interesting questions that remain to be explored. The proposed algorithm needs global synchronization which may be infeasible in some distributed settings, and the graph may change according to some dynamics. We are interested to extend our algorithm to asynchronous setting where each node is randomly activated, as well as to time-varying graphs. We would also like to gain more insights into design problem (3.14) from graph perspective, and study adaptively adjusting graph weights and P during the algorithm. In addition, it would be interesting to develop a distributed algorithm based on reformulation (3.2.1) using graph Laplacian.

Chapter 4

ONLINE ALGORITHMS FOR NETWORK FORMATION

4.1 Introduction

Network formation problem has been the focus of study in a number of disciplines including economics, computer science and engineering [7, 29, 40, 69]. Most of the existing work consider network formation as a dynamic process of a group of agents forming a network that has certain desirable properties. In this paper, we introduce a new online game setting to network formation: candidate edges are chosen and revealed one by one by the first player, then the other player makes online decisions on whether to accept the edge. The goal of the second player is to choose the best subset of candidate edges within a budget constraint that achieves the best desired properties of the synthesized network. The network properties considered in our paper include the number of spanning trees, algebraic connectivity, total effective resistance, which are respectively measured by the log-determinant [37, 57, 97], the second smallest eigenvalue [14, 25], and the trace of the pseudo inverse [41] of the suitably transformed graph Laplacian. These network formation games arise in a variety of cooperative multiagent systems, such as robotic networks establishing a secured network in a changing uncertain environment, or individuals forming teams in social networks. We propose a primal-dual algorithm framework for the general online network formation game. The algorithm is motivated by recent works on online problems in theoretical computer science such as online matching, covering and packing [18, 21, 55] as well as general online linear and convex programming [1, 2]. The algorithm has deep relations with algorithms in online learning and optimization [5, 19, 20]. The performance of the algorithm is measured by the competitive ratio and regret, which are the ratio and difference of the objective value achieved by the online algorithm to the true offline optimal objective value.

4.1.1 Online network control applications

Online network consensus Consensus systems arise in a variety of settings such as formation control, multi-vehicle control, distributed estimation, swarming [44, 54, 75, 89]. In a more general case when the network has external agents attached, the consensus system dynamics is influenced by those additional agents in a collaborative or malicious way, which is called a semi-autonomous consensus network [25]. In traditional system, performance is often measured by the smallest eigenvalue of the graph Laplacian to quantify the convergence rate, while security is defined by the total effective resistance to measure the rejection ability of the system to external intrusion or disturbance. In semi-autonomous system, both performance and security of a network in response to an external injecting a signal are jointly measured by mean and variance of agents' state, which are motivated by many applications including distributed state estimation and flocking. These metrics can be shown equivalent to the total effective resistance of the graph Laplacian perturbed by external agents. Network performance and security via adaptive topology attracts more recent interests, particularly, the online synthesis of network with budget limit is a largely unexplored area. The instances of systems where the agents dynamics and the underlying interaction protocol are assumed to be fixed: networks with hardwired dynamics and interactions, systems with physically and biologically motivated dynamics and interactions.

Online social and behavioral network Dynamic processes such as diffusion of information and synchronization processes have been the focus of social network studies in recent years [24, 26, 74]. The information diffusion is largely influenced by the dramatic effect of the network topology.

4.2 Background

4.2.1 Graph and notations

Let $G = (V, E)$ be an undirected graph with $|V| = n$ and $|E| = m$. Edges are indexed by $t = 1, \ldots, m$, and each edge is denoted as $\{i_t, j_t\}$ if it connects nodes i_t and j_t. For each edge $\{i_t, j_t\}$, define a vector $a_t \in \mathbf{R}^n$ with $a_{t,i_t} = 1$, $a_{t,j_t} = -1$ and all other entries 0. A binary weight vector $x \in \mathbf{R}^m$ consists of edge weights that are either 1 or 0.

The *graph Laplacian* is the $n \times n$ matrix $L = \sum_{t=1}^m a_t a_t^T x_t$. Let $\lambda_1 \leq \ldots \leq \lambda_n$ be the sorted eigenvalues of L. Clearly, L is positive semidefinite with zero smallest eigenvalue $\lambda_1 = 0$ and the corresponding eigenvector $\mathbf{1}$.

4.2.2 Graph problems

Given a base graph $G_0 = (V, E_0)$ and a set E of m candidate edges on V, (offline) graph problem chooses edges from E with the goal to achieve the best desired property of the graph by adding these edges to G_0. Let $x_1, \ldots, x_m \in \{0, 1\}$ be weights for m candidate edges, where $x_t = 1$ indicates that the edge is chosen, otherwise $x_t = 0$. Denote the graph Laplacian of the base graph and the added graph as L_0 and $\sum_{t=1}^m a_t a_t^T x_t$, respectively. The graph properties are measured by certain *concave* function of the graph Laplacian, $\phi(\sum_{t=1}^m a_t a_t^T x_t)$. We consider two different scenarios in terms of the number of edges that are allowed to be added to the base graph.

Hard budget graph problem The hard budget offline problem can be formulated as an optimization problem

$$\begin{array}{ll} \underset{x_1,\ldots,x_m}{\text{maximize}} & \phi\left(\displaystyle\sum_{t=1}^m a_t a_t^T x_t\right) \\ \text{subject to} & \displaystyle\sum_{t=1}^m x_t \leq b \\ & x_t \in [0,1] \quad t = 1, \ldots, m. \end{array} \tag{4.1}$$

We name the constraint $\sum_{t=1}^{m} x_t \leq b$ the *hard budget constraint*. The goal is to maximize the graph property while the number of added edges is below a given budget b, i.e. no budget violation is allowed. The conjugate function of $\phi(X)$ is defined as $\phi^*(Y) = \inf_{X \succeq 0} \mathbf{Tr}(YX) - \phi(X)$. The dual problem of (4.1) is

$$\begin{array}{ll} \underset{Y,z}{\text{minimize}} & \displaystyle\sum_{t=1}^{m} (a_t^T Y a_t - z)_+ + bz - \phi^*(Y) \\ \text{subject to} & Y \succeq 0, \quad z \geq 0, \end{array} \tag{4.2}$$

where $Y \in \mathbf{R}^{n \times n}$ and $z \in \mathbf{R}_+$ are dual variables, in particular, z corresponds to the hard budget constraint.

If the optimal dual variable $z^\star$ is known, problem (4.1) can be equivalently written as

$$\begin{array}{ll} \underset{x_1,\ldots,x_m}{\text{maximize}} & \displaystyle\phi\left(\sum_{t=1}^{m} a_t a_t^T x_t\right) - z^\star \sum_{t=1}^{m} x_t \\ \text{subject to} & x_t \in [0,1] \quad t = 1, \ldots, m. \end{array} \tag{4.3}$$

However, if exact value of $z^\star$ is hard to compute, or the hard budget b is not required, the problem can be relaxed as a soft budget problem as below.

Soft budget graph problem The soft budget offline problem can be formulated as

$$\begin{array}{ll} \underset{x_1,\ldots,x_m}{\text{maximize}} & \displaystyle\phi\left(\sum_{t=1}^{m} a_t a_t^T x_t\right) - \theta \sum_{t=1}^{m} x_t \\ \text{subject to} & x_t \in [0,1] \quad t = 1, \ldots, m, \end{array} \tag{4.4}$$

where $\theta > 0$ is a given parameter. This problem aims to maximize the graph property function while penalizing the use of budget, and the tradeoff is controlled by the parameter θ. Of course, if parameter θ is chosen as the optimal dual variable of problem (4.1), (4.1) and (4.4) are equivalent. However, estimation of exact optimal dual variable z^* is often very difficult, so the soft budget graph problem with a given parameter is much easier to solve, especially in the online case; therefore, we will develop and analyze separate online algorithms for hard and soft budget graph problems.

Both problems (4.1) and (4.4) can be unified into the form

$$\underset{x_1,\ldots,x_m\in[0,1]}{\text{minimize}}\ \phi\left(\sum_{t=1}^{m} a_t a_t^T x_t\right) + \psi\left(\sum_{t=1}^{m} x_t\right), \tag{4.5}$$

where ψ is named as the *budget function*. For hard budget problem (4.1), $\psi(u)$ is the indicator function, i.e.

$$\psi(u) = I(u \le b) = \begin{cases} 0 & u \le b, \\ -\infty & \text{otherwise} \end{cases}. \tag{4.6}$$

For soft budget problem (4.4), $\psi(u)$ is

$$\psi(u) = \begin{cases} -\theta u & u \ge 0, \\ 0 & \text{otherwise} \end{cases}. \tag{4.7}$$

The conjugate function of the indicator function (4.6) is

$$\psi^*(z) = \inf_u zu - \psi(u) = \begin{cases} bz & z \le 0 \\ -\infty & z > 0 \end{cases}.$$

Therefore the dual problem (4.2) can be written as

$$\underset{Y\succeq 0, z}{\text{minimize}} \quad \sum_{t=1}^{m} (a_t^T Y a_t - z)_+ - \phi^*(Y) - \psi^*(-z). \tag{4.8}$$

The conjugate function of (4.7) is

$$\psi^*(z) = \begin{cases} 0 & -\theta \le z \le 0 \\ -\infty & \text{otherwise} \end{cases}.$$

4.2.3 Graph objective functions

In this paper, we study the following graph properties and the corresponding objective function ϕ.

Log-determinant problem The log-determinant (logdet) of the (perturbed) graph Laplacian measures the number of spanning trees of the graph [37, 57], and it has been used in graph optimization problems [97]. In this case, $\phi(L) = \log\det(L + L_0 + \mathbf{1}\mathbf{1}^T/n)$ and its conjugate function is $\phi^*(Y) = \log\det Y - \mathbf{Tr}((L_0 + \mathbf{1}\mathbf{1}^T/n)Y) + n$. The gradient of this function is $\nabla\phi(L) = (L + L_0 + \mathbf{1}\mathbf{1}^T/n)^{-1}$.

Graph connectivity It is well-known that graph algebraic connectivity can be measured by the second smallest eigenvalue of the graph Laplacian, $\phi(L) = \lambda_2(L + L_0)$. The super gradient of this function is uu^T, where u is the Fielder eigenvector corresponding to $L + L_0$.The algebraic connectivity has been studies in various problems. [14] studies to maximize the algebraic connectivity of a continuous time Markov chain to achieve the fastest mixing rate; [40] studies adding edges from a *given* set of candidate edges to a graph to maximize the algebraic connectivity; [58] studies choosing a configuration of nodes under a proximity constraint that maximizes the algebraic connectivity.

Total effective resistance In an electrical network with edge weights x_t as conductance, the effective resistance R_{ij} between two nodes i and j is the voltage difference between them when one ampere current source is connected between these two nodes. The total effective resistance is the sum of the effective resistance between all pairs of nodes, which has been extensively studied in [41] and can be defined in several forms in terms of the graph Laplacian

$$\begin{aligned} R_{\text{tot}} &= \frac{1}{2}\mathbf{1}^T R\mathbf{1} = n\,\mathbf{Tr}((L+L_0)^\dagger) \\ &= n\,\mathbf{Tr}(L+L_0+\mathbf{1}\mathbf{1}^T/n)^{-1} - n, \end{aligned}$$

where $(L + L_0)^\dagger$ is the pseudo inverse of $L + L_0$. We will adopt the formulation $\phi(L) = -n\,\mathbf{Tr}((L + L_0 + \mathbf{1}\mathbf{1}^T/n)^{-1}) + n$, whose conjugate function is $\phi^*(Y) = 2\,\mathbf{Tr}((nY)^{1/2}) - \mathbf{Tr}((L_0 + \mathbf{1}\mathbf{1}^T/n)Y) - n$. The gradient of this function is $\nabla\phi(L) = n(L + L_0 + \mathbf{1}\mathbf{1}^T/n)^{-2}$.

4.2.4 Nesterov function smoothing and dual regularization

Suppose $\psi^*(z)$ is the conjugate function of a concave function $\psi(u)$. Recall that the Nesterov smoothing [73] adds a proximity function $r(z)$, which is continuous and strongly convex, to the conjugate function,

$$\psi_s^*(z) = \psi^*(z) + \frac{r(z)}{\eta},$$

where $\eta > 0$ is a smoothing parameter. Then the conjugate of $\psi_s^*(z)$ is the smoothed approximation of $\psi(u)$

$$\psi_s(u) \;=\; \inf_z uz - \psi_s^*(z) = \inf_z uz - \psi^*(z) - \frac{r(z)}{\eta}. \tag{4.9}$$

This can also be viewed as adding a strongly convex term in the dual space, which is also called *dual regularization* in some communities such as online optimization. In this paper, we will present in a parallel fashion from both primal smoothing and dual regularization to give a unified view.

Choosing a proximity function is critical in developing algorithms. A basic example of proximity function is quadratic function $r(z) = z^2/2$, however, we will use more advanced proximity functions to smooth the budget function (or say, regularize the dual problem), which leads to improved performance of algorithms to be developed later.

Smoothing hard budget function (4.6) by entropy function Recall that the entropy function on $\mathbf{R}^2$ is defined as $d(v_1, v_2) = v_1 \log v_1 + v_2 \log v_2 + \log 2$, where $v_1, v_2 > 0$ and $v_1 + v_2 = 1$. For our purpose we define an entropy function as the proximity function on the domain $\{z \mid -\theta < z < 0\}$ (where $\theta > 0$)

$$r(z) = -\left(1 + \frac{z}{\theta}\right) \log\left(1 + \frac{z}{\theta}\right) + \frac{z}{\theta} \log\left(-\frac{z}{\theta}\right) - \log 2.$$

Then from (4.9), the smoothed hard budget function is

$$\psi_s(u) = -\frac{1}{\eta} \log\left(\frac{\exp(\eta\theta(u-b)) + 1}{2}\right). \tag{4.10}$$

It can be drived as follows: the entropy function can be viewed as $r(z) = d(-\frac{z}{\theta}, 1 + \frac{z}{\theta})$, where $\{z \mid 1 > -\frac{z}{\theta} > 0\}$. Then $\psi_s(u) = \inf_{-\theta<z<0} \; (u-b)z - r(z)/\eta$. Set the gradient to 0,

$$(u-b) - \frac{1}{\eta\theta}\left(\log(-\frac{z}{\theta}) - \log(1 + \frac{z}{\theta})\right) = 0,$$

then $u - b = -\frac{1}{\eta\theta} \log\left(-\frac{z+\theta}{z}\right)$. Since $\frac{z+\theta}{z} = -\exp(-\eta\theta(u-b))$ and $\frac{z+\theta}{\theta} = \frac{\exp(-\eta\theta(u-b))}{1+\exp(-\eta\theta(u-b))}$, the derivation is done.

Smoothing the soft budget function (4.7) by logarithmic function Define a proximity function on the domain $\{z \mid -\theta < z\}$

$$r(z) = -\left(1 + \frac{z}{\theta}\right) \log\left(1 + \frac{z}{\theta}\right) + \frac{z}{\theta},$$

where $\theta > 0$. Then from (4.9), the smoothed soft budget function is

$$\psi_s(u) = -\theta u + \frac{1}{\eta}(1 - \exp(-\eta\theta u)). \tag{4.11}$$

It can be derived as follows: from $\psi_s(u) = \inf_{-\theta<z<0} uz - r(z)/\eta$, set the gradient to 0,

$$u + \frac{1}{\eta\theta} \log(1 + \frac{z}{\theta}) = 0,$$

then $u = -\frac{1}{\eta\theta} \log(1 + \frac{z}{\theta})$ and $z = \theta(e^{-\eta\theta u} - 1)$.

4.3 Online network formation

In this section, we introduce an *online network formation game*, an online game setting to the graph problems discussed in Section 4.2.2. Starting with a connected base graph $G_0 = (V, E_0)$ and a set E of m candidate edges, the game proceeds to subsequently choose edges and add them to the base graph in m rounds of the game. In each round $t = 1, \ldots, m$, player one first chooses one candidate edge from E and reveals it to player two, then player two chooses to accept it or not, i.e. decides an edge weight $\hat{x}_t \in \{0, 1\}$. In the hard budget scenario, the number of chosen by player two is bounded by b, i.e. $\sum_{t=1}^{m} \hat{x}_t \leq b$; in the soft budget scenario, player two is penalized to use the budget but no hard limited is imposed. Player two has *no* prior knowledge on the candidate edges in the soft budget case, however, it knows the total number of candidate edges for the hard budget case. Player two makes online decisions aiming to form a network with the best properties as those discussed in Section 4.2.2.

We propose online algorithms in Section 4.4 for soft budget network formation, and in Section 4.5 for hard budget network formation. Algorithms are motivated by recent work on online problems in the theoretic computer science such as online matching, covering and packing [18, 21, 55] as well as general online linear and convex programming [1, 2].

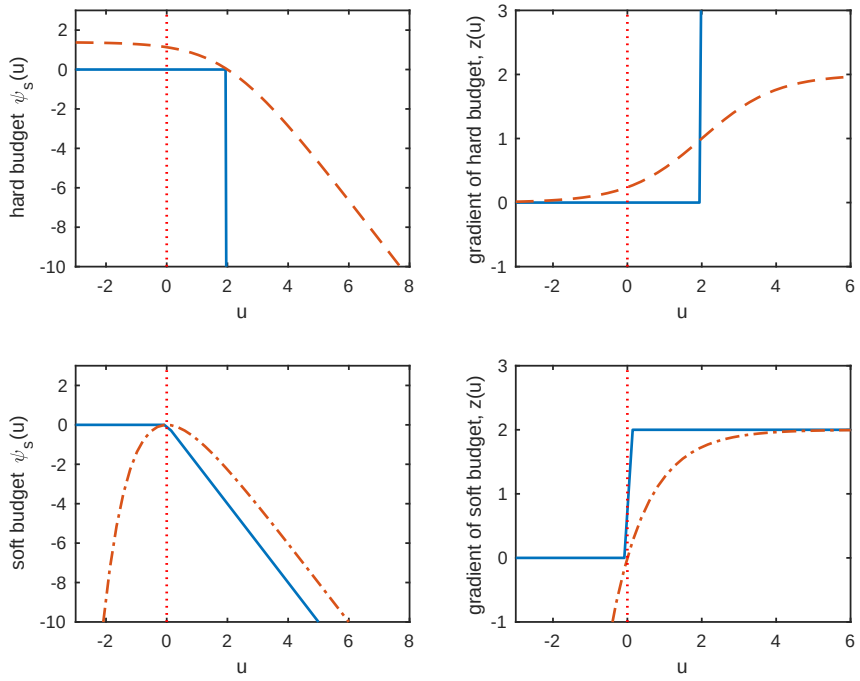


Figure 4.1: Smoothing and shifting of budget functions and their corresponding negative gradients(supergradients)

4.4 Online algorithms for soft budget network formation

4.4.1 Basic algorithm

The basic algorithm given in Algorithm 3 is an online primal-dual framework for soft budget network formation game. In each round, after an edge is revealed by player one, player two first assumes to accept the edge and updates the dual variables $\hat{Y}_t$ and $\hat{z}_t$ using all information $\hat{x}_1, \ldots, \hat{x}_{t-1}$ and $a_1, \ldots, a_{t-1}$ from previous rounds and newly received a_t. Then player two use dual variables to decide whether to accept this edge, i.e. assign a $0-1$ weight $\hat{x}_t$ by comparing $a_t^T \hat{Y}_t a_t$ with $\hat{z}_t$. A complete interpretation on Algorithm 3 will be given in Section 4.4.2.

Algorithm 3 Basic online soft budget network formation

input: L_0

for $t = 1, \ldots m$ **do**

Player one chooses an edge from E and reveals a_t

Player two updates dual variables

$$\hat{Y}_t \in \partial\phi\left(a_t a_t^T + \sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s\right), \tag{4.12}$$

$$\hat{z}_t \in -\partial\psi(1 + \sum_{s=1}^{t-1} \hat{x}_s), \tag{4.13}$$

Player two decides the edge weight

$$\hat{x}_t = \begin{cases} 1 & a_t^T \hat{Y}_t a_t \geq \hat{z}_t \\ 0 & \text{otherwise} \end{cases}. \tag{4.14}$$

end for

When applying this algorithm to graph problems, only $\hat{Y}$ update are problem-dependent and given as follows:

- Logdet problem

$$\hat{Y}_t = \left(a_t a_t^T + \sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s + L_0 + \frac{\mathbf{1}\mathbf{1}^T}{n} \right)^{-1}. \tag{4.15}$$

- Graph connectivity

$$\hat{Y}_t = uu^T, \tag{4.16}$$

  where u is the Fielder eigenvector corresponding to $\lambda_2 \left(a_t a_t^T + \sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s + L_0 \right)$.

- Total effective resistance

$$\hat{Y}_t = n \left(a_t a_t^T + \sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s + L_0 + \frac{\mathbf{1}\mathbf{1}^T}{n} \right)^{-2}. \tag{4.17}$$

Algorithm 3 is the theoretic algorithm which will be used for analysis in Section C.1; more implementable algorithms for graph problems will be given in Section 4.6, where no inversion operation is needed by exploiting the structure.

4.4.2 Interpretation

Tradeoff between graph objective and budget function

Algorithm 3 can be better understood through a trade-off between objective function ϕ and budget function ψ in (4.5): if an edge is accepted and $\hat{x}_t = 1$, the objective function would increase in the expense of the decrease of the budget function. $a_t^T \hat{Y}_t a_t$ and $\hat{z}_t$, which are gradients (or supergradients) of ϕ and ψ respectively with respect to $\hat{x}_t$ and computed for $\hat{x}_t = 1$, measure the increase of the objective function and the decrease of the budget function after adding edge a_t. Therefore, the decision rule (4.23) essentially accepts an edge only if the objective function increases more than the budget function decreases, i.e. when $a_t^T \hat{Y}_t a_t \geq \hat{z}_t$.

However, directly using the linear soft budget function (4.6) as in Algorithm 3 does not lead to a clever strategy: if the *constant* parameter θ is chosen too big, the algorithm will always reject edges, since $a_t^T \hat{Y}_t a_t \leq \hat{z}_t = \theta$. A more desired algorithm would increase θ as more edges are being chosen. Therefore, a better algorithm relies on the design of new budget functions which will be discussed by using smoothing in Section 4.4.3.

Regularized online optimization

Algorithm 3 can be interpreted from a primal-dual perspective. The problem (up to time t in the online setting) can be written as

$$\begin{aligned}
& \sup_{\hat{x}_1,\ldots,\hat{x}_t\in[0,1]} \phi\left(\sum_{s=1}^{t} a_s a_s^T \hat{x}_s\right) + \psi\left(\sum_{s=1}^{t} \hat{x}_s\right) \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_t\in[0,1]} \phi\left(\sum_{s=1}^{t} a_s a_s^T \hat{x}_s\right) - \theta \sum_{s=1}^{t} \hat{x}_s \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_t\in[0,1]} \inf_{\hat{Y}\succeq 0} \langle \sum_{s=1}^{t} a_s a_s^T \hat{x}_s, \hat{Y}\rangle - \phi^*(\hat{Y}) - \theta \sum_{s=1}^{t} \hat{x}_s \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_t\in[0,1]} \inf_{\hat{Y}\succeq 0} \sum_{s=1}^{t} (a_s^T \hat{Y} a_s - \theta)\hat{x}_s - \phi^*(\hat{Y}).
\end{aligned}$$

In the online setting, $\hat{x}_1, \ldots, \hat{x}_t$ are decided one by one, and Algorithm 3 alternatingly optimizes with respect to $\hat{Y}, \hat{z}$ and $\hat{x}$. At time t, Algorithm 3 first assumes $\hat{x}_t = 1$, and optimizes with respect to $\hat{Y}, \hat{z}$ with fixed $\hat{x}_1, \ldots, \hat{x}_t$ as follows

$$\hat{Y}_t \in \operatorname*{argmin}_{\hat{Y}\succeq 0} \; a_t^T \hat{Y} a_t + \sum_{s=1}^{t-1} a_s^T \hat{Y} a_s \hat{x}_s - \phi^*(\hat{Y}),$$

and $\hat{z}_t = \theta$. These $\hat{Y}_t$ and $\hat{z}_t$ are equivalent to those given in Algorithm 3. When maximizing with respect to $\hat{x}_t$, fix $\hat{Y}, \hat{z}$ to $\hat{Y}_t, \hat{z}_t$ from the previous iteration. Therefore, by maximizing $(a_t^T \hat{Y}_t a_t - \theta)\hat{x}_t$ with respect to $\hat{x}_t \in [0, 1]$, set $\hat{x}_t = 1$ if $a_t^T \hat{Y}_t a_t > \theta$, or $\hat{x}_t = 0$ otherwise.

4.4.3 Design budget function by smoothing

As discussed in Section 4.4.2, we smooth the soft budget function (4.7) to (4.11). Using this budget function ψ_s, $\hat{z}_t$ update (4.13) changes to

$$\begin{aligned}
\hat{z}_t &= -\nabla\psi_s(1 + \sum_{s=1}^{t-1} \hat{x}_s), && (4.18) \\
&= \theta\left(1 - \exp\left(-\eta\theta\left(1 + \sum_{s=1}^{t-1} \hat{x}_s\right)\right)\right). && (4.19)
\end{aligned}$$

This $\hat{z}_t$ update of the smoothed function leads to a better strategy: instead of setting $\hat{z}_t$ to be a constant θ, $\hat{z}_t$ increases from 0 to θ as more edges are chosen. Therefore, the algorithm become more and more conservative when more budget is being used.

Note that (4.18) is equivalent to

$$\begin{aligned} \hat{z}_t &= -\operatorname*{argmin}_{\hat{z}} \hat{z}\left(1+\sum_{s=1}^{t-1}\hat{x}_s\right) - \psi_s^*(\hat{z}), \\ &= -\operatorname*{argmin}_{-\theta<\hat{z}<0} \hat{z}\left(1+\sum_{s=1}^{t-1}\hat{x}_s\right) - \frac{r(\hat{z})}{\eta}. \end{aligned} \tag{4.20}$$

In the language of online optimization, the $\hat{z}_t$-update (4.20) can be interpreted as the Follow The Regularized Leader (FTRL) method [20, 46, 86] with a strongly convex regularizer $\frac{r(z)}{\eta}$.

Algorithm 4 Online soft budget network formation with smoothing

input: L_0, η, θ

for $t = 1, \dots m$ **do**

Player one chooses an edge from E and reveals a_t

Player two updates dual variables

$$\hat{Y}_t \in \partial\phi\left(a_t a_t^T + \sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s\right), \tag{4.21}$$

$$\hat{z}_t = \theta\left(1-\exp\left(-\eta\theta\left(1+\sum_{s=1}^{t-1}\hat{x}_s\right)\right)\right), \tag{4.22}$$

Player two decides the edge weight

$$\hat{x}_t = \begin{cases} 1 & a_t^T \hat{Y}_t a_t > \hat{z}_t \\ 0 & \text{otherwise} \end{cases}. \tag{4.23}$$

end for

4.5 Online algorithms for hard budget network formation

In this section we design the algorithm for online hard budget network formation. In order to demonstrate the tricks to achieve the exact budget, first write down the optimization problem (up to time t)

$$\begin{array}{ll} \underset{\hat{x}_1,\ldots,\hat{x}_t}{\text{maximize}} & \frac{t}{m}\phi\left(\frac{m}{t}\sum_{s=1}^{t} a_s a_s^T \hat{x}_s\right) \\ \text{subject to} & \frac{1}{t}\sum_{s=1}^{t}\hat{x}_s \leq \frac{b}{m} \\ & \hat{x}_s \in [0,1] \quad s=1,\ldots,t. \end{array} \tag{4.24}$$

The constraint $\frac{1}{t}\sum_{s=1}^{t}\hat{x}_s \leq \frac{b}{m}$ bounds the *average* use of the budget by b/m. The constraint can be written as $\sum_{s=1}^{t}\hat{x}_s \leq \frac{tb}{m}$ which can be interpreted as releasing budget b gradually, linearly proportional to the current time over the horizon, t/m. At the end of the algorithm when $t=m$, this constraint is exactly the same budget constraint $\sum_{s=1}^{t}\hat{x}_s \leq b$ as the one in problem (4.1). Therefore, the algorithm for the hard budget case requires the additional information on horizon m, which is not needed for the soft budget case.

Using the smoothed hard budget function ψ_s given in (4.10) (replacing b with tb/m), problem (4.24) can be written as

$$\underset{\hat{x}_1,\ldots,\hat{x}_s\in[0,1]}{\text{maximize}} \frac{t}{m}\phi\left(\frac{m}{t}\sum_{s=1}^{t} a_s a_s^T \hat{x}_s\right) + \psi_s\left(\sum_{s=1}^{t}\hat{x}_s\right).$$

To design the algorithm, compute the (super)gradients of ϕ and ψ,

$$\hat{Y}_t \in \partial\frac{t}{m}\phi\left(\frac{m}{t}\sum_{s=1}^{t} a_s a_s^T \hat{x}_s\right),$$

and

$$\begin{aligned} \hat{z}_t &= -\nabla\psi_s\left(\sum_{s=1}^{t}\hat{x}_s\right) \\ &= \frac{\theta\exp\left(\eta\theta(\sum_{s=1}^{t}\hat{x}_s - \frac{tb}{m})\right)}{1+\exp\left(\eta\theta(\sum_{s=1}^{t}\hat{x}_s - \frac{tb}{m})\right)}. \end{aligned}$$

Then set $\hat{x}_{t+1} = 1$ if ϕ increases more than the decrease of ψ, i.e. $a_t^T \hat{Y}_t a_t > \hat{z}_t$, or $\hat{x}_t = 0$ otherwise.

To interpret the algorithm from the primal-dual perspective, write the problem (4.24) as

$$\begin{aligned}
& \sup_{\hat{x}_1,\ldots,\hat{x}_s \in [0,1]} \frac{t}{m}\phi\left(\frac{m}{t}\sum_{s=1}^{t} a_s a_s^T \hat{x}_s\right) + \psi_s\left(\sum_{s=1}^{t}\hat{x}_s\right) \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_s \in [0,1]} \inf_{\hat{Y}\succeq 0,\hat{z}} \frac{t}{m}[\frac{m}{t}\sum_{s=1}^{t} a_s^T \hat{Y} a_s \hat{x}_s - \phi^*(\hat{Y})] + \hat{z}\sum_{s=1}^{t}\hat{x}_s - \psi^*(\hat{z}) - \frac{r(\hat{z})}{\eta} \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_s \in [0,1]} \inf_{\hat{Y}\succeq 0,\hat{z}\leq 0} [\sum_{s=1}^{t} a_s^T \hat{Y} a_s \hat{x}_s - \frac{t}{m}\phi^*(\hat{Y})] + \hat{z}[\sum_{s=1}^{t}\hat{x}_s - \frac{t}{m}b] - \frac{r(\hat{z})}{\eta} \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_s \in [0,1]} \inf_{\hat{Y}\succeq 0,\hat{z}\leq 0} \sum_{s=1}^{t}\left(a_s^T \hat{Y} a_s \hat{x}_s - \frac{\phi^*(\hat{Y})}{m}\right) + \hat{z}\sum_{s=1}^{t}\left(\hat{x}_s - \frac{b}{m}\right) - \frac{r(\hat{z})}{\eta} \\
= & \sup_{\hat{x}_1,\ldots,\hat{x}_s \in [0,1]} \inf_{\hat{Y}\succeq 0,\hat{z}\geq 0} \sum_{s=1}^{t}\left(a_s^T \hat{Y} a_s \hat{x}_s - \hat{z}\right)\hat{x}_s - \frac{t}{m}\phi^*(\hat{Y}) - \frac{tb}{m} - \frac{r(-\hat{z})}{\eta}.
\end{aligned}$$

At time t, Algorithm 5 optimizes with respect to $\hat{Y}$ and $\hat{z}$ with $\hat{x}_1, \ldots, \hat{x}_t$ fixed. The $\hat{Y}_t$-update is

$$\hat{Y}_t = \operatorname*{argmax}_{\hat{Y}\succeq 0} \sum_{s=1}^{t} a_s^T \hat{Y} a_s \hat{x}_s - \frac{t}{m}\phi^*(\hat{Y}),$$

and $\hat{z}_t$-update is

$$\hat{z}_t = -\operatorname*{argmin}_{\hat{z}\leq 0} \hat{z}\sum_{s=1}^{t}(\hat{x}_s - \frac{b}{m}) - \frac{r(\hat{z})}{\eta}.$$

This is the Follow the Regularized Leader with entropy function as regularization. When maximizing with respect to $\hat{x}_t$, fix $\hat{Y}$ and $\hat{z}$. Therefore, by maximizing $\left(a_t^T \hat{Y}_t a_t \hat{x}_t - \hat{z}_t\right)\hat{x}_t$ with respect to $\hat{x}_t \in [0,1]$, set $\hat{x}_t = 1$ if $a_t^T \hat{Y}_t a_t \hat{x}_t > \hat{z}_t$, or $\hat{x}_t = 0$ otherwise.

4.5.1 Estimation of parameter θ

Choosing the right parameter θ is critical in the algorithm: θ in the entropy function is an upper bound on $\hat{z}$, as a result, in $\hat{z}$-update, θ dictate the level of $\hat{z}$ and thus controls the tradeoff between the graph objective and budget constraint. The regret bound also naturally

Algorithm 5 Online hard budget network formation algorithm

input: L_0, η, θ

for $t = 1, \dots m$ **do**

Player one chooses an edge from E and reveals a_t

Player two updates dual variables

$$\hat{Y}_t = \partial \frac{t}{m} \phi \left(\frac{m}{t} \sum_{s=1}^{t} a_s a_s^T \hat{x}_s \right), \tag{4.25}$$

$$\hat{z}_t = \frac{\theta \exp\left(\eta\theta(\sum_{s=1}^{t} \hat{x}_s - \frac{tb}{m})\right)}{1 + \exp\left(\eta\theta(\sum_{s=1}^{t} \hat{x}_s - \frac{tb}{m})\right)}. \tag{4.26}$$

Player two decides the edge weight

$$\hat{x}_t = \begin{cases} 1 & a_t^T \hat{Y}_t a_t > \hat{z}_t \\ 0 & \text{otherwise} \end{cases}. \tag{4.27}$$

end for

depends θ. Clearly the ideal choice of θ is actually the optimal dual variable $z^\star$, however $z^\star$ is not available. In this section, we design dynamic algorithm to estimate an upper bound on $z^\star$, which gives a guidance on choice of θ. This section adapts the derivation by Agrawal and Devanur in [1] to our problem to estimate an upper bound on $z^\star$ dynamically.

The idea in estimating $z^\star$ is to solve sample optimization problems on samples of received edges with a relaxed budget constraint by δ. For instance, for k sampled edges, the optimization problem is

$$\begin{aligned} \underset{\hat{x}_1,\dots,\hat{x}_k}{\text{maximize}} \quad & \frac{k}{m} \phi \left(\frac{m}{k} \sum_{s=1}^{k} a_s a_s^T \hat{x}_s \right) \\ \text{subject to} \quad & \frac{1}{k} \sum_{s=1}^{k} \hat{x}_s \leq \frac{b}{m} + \delta \\ & \hat{x}_s \in [0,1] \quad s = 1, \dots, k. \end{aligned} \tag{4.28}$$

The average budget contraint is relaxed from b/m to $b/m + \delta$. Denote the optimal value of this problem as $P_k^\star(\delta)$. The optimization problem is solved after every 2^{r-1} steps, so the rth

optimization problem uses 2^{r-1} edges. In this case, the estimate θ is

$$\theta = \frac{P^\star_{2^{r-1}}(4\gamma) - P^\star_{2^{r-1}}(\gamma)}{2^{r-1}\gamma}.$$

4.6 Network algorithms

4.6.1 Logdet problem

Define $\hat{L}_t = \sum_{s=1}^{t} a_s a_s^T \hat{x}_s + L_0 + \mathbf{1}\mathbf{1}^T/n$ and $\hat{L}_0 = L_0 + \mathbf{1}\mathbf{1}^T/n$. Note that $\hat{L}_t = (\hat{L}_{t-1} + a_t a_t^T \hat{x}_t)^{-1}$, so it is easy to obtain a rank-one update formular for $\hat{L}_t$ from $\hat{L}_{t-1}$ by applying the Sherman-Morrison formula,

$$\hat{L}_t^{-1} \;=\; \hat{L}_{t-1}^{-1} - \frac{\hat{L}_{t-1}^{-1} a_t a_t^T \hat{L}_{t-1}^{-1} \hat{x}_t}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t \hat{x}_t},$$

where $\hat{L}_{t-1}$ is invertible since the base graph is assumed to be connected and thus $\hat{L}_0$ has full rank. From $\hat{Y}$-update (4.15), $\hat{Y}_t = (a_t a_t^T + \hat{L}_{t-1})^{-1}$, i.e. $\hat{Y}_t = \hat{L}_t^{-1}$ if $\hat{x}_t = 1$, therefore

$$a_t^T \hat{Y}_t a_t = a_t^T \hat{L}_{t-1}^{-1} a_t - \frac{(a_t \hat{L}_{t-1}^{-1} a_t)^2}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t} = \frac{a_t^T \hat{L}_{t-1}^{-1} a_t}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t}. \tag{4.29}$$

Note that $0 < a_t^T \hat{Y}_t a_t < 1$. Since only $a_t^T \hat{Y}_t a_t$ is needed in the algorithm, we do not need to store $\hat{Y}_t$ but only compute $a_t^T \hat{Y}_t a_t$ from $\hat{L}_{t-1}^{-1}$. The online algorithm for logdet problem is given in Algorithm 6.

We can interpret the algorithm on an electrical network: $\hat{x}_t$ is the conductance added *in series* to edge (or branch) $\{i_t, j_t\}$, and the graph Laplacian $\hat{L}_{t-1}$ is the conductance matrix before adding $\hat{x}_t$. Let $v \in \mathbf{R}^n$ be the potential vector of all nodes. Applying one ampere current source between nodes i_t and j_t, $(\sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s + L_0)v = a_t$ gives the relation between node potential vector v and edge current vector a_t. The effective resistance is the voltage difference between nodes i_t and j_t

$$R_{i_t j_t} = v_{i_t} - v_{j_t} = a_t^T (\sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s + L_0)^\dagger a_t = a_t^T \hat{L}_{t-1}^{-1} a_t,$$

as derived in [41, Section 2.4]. Therefore, $a_t^T \hat{L}_{t-1}^{-1} a_t$ and $a_t^T \hat{Y}_t a_t$ are the effective resistance between nodes i_t and j_t *before* and *after* adding the conductance $\hat{x}_t$ to the network. Given

$\hat{L}_{t-1}^{-1}$, computing $a_t^T \hat{L}_{t-1}^{-1} a_t = \hat{L}_{t-1,i_t i_t}^{-1} + \hat{L}_{t-1,j_t j_t}^{-1} - 2\hat{L}_{t-1,i_t j_t}^{-1}$ is very simple. It is interesting to note from (4.29) that the effective resistance of a pair of nodes is always reduced by adding an edge $\hat{x}_t = 1$, which can be interpreted from the electric circuit analysis as *paralleling* one unit resistor with the old effective resistance $a_t^T \hat{Y}_t a_t$.

Algorithm 6 Online algorithm for logdet problem

---

input: $\hat{L}_0^{-1} = L_0^\dagger + \mathbf{1}\mathbf{1}^T/n$, η, θ

for $t = 1, \ldots m$ **do**

Player two receives edge $\{i_t, j_t\}$, updates threshold $\hat{z}_t$ according to (4.22) for the soft budget case or (4.26) for the hard budget case; and computes the effective resistance between nodes i_t and j_t if accepting this edge

$$R_{i_t j_t} = \frac{a_t^T \hat{L}_{t-1}^{-1} a_t}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t}.$$

Player two decides edge weight $\hat{x}_t = 1$ if $R_{i_t j_t} > \hat{z}_t$; otherwise set $\hat{x}_t = 0$.

Player two updates $\hat{L}_t^{-1}$

$$\hat{L}_t^{-1} = \hat{L}_{t-1}^{-1} - \frac{\hat{L}_{t-1}^{-1} a_t a_t^T \hat{L}_{t-1}^{-1} \hat{x}_t}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t \hat{x}_t}.$$

end for

4.6.2 Total effective resistance

From $\hat{Y}$-update (4.17) for total effective resistance,

$$\begin{aligned} a_t^T \hat{Y}_t a_t &= n\|(a_t a_t^T + \hat{L}_{t-1})^{-1} a_t\|_2^2 \\ &= n \left\| \left(\hat{L}_{t-1}^{-1} - \frac{\hat{L}_{t-1}^{-1} a_t a_t^T \hat{L}_{t-1}^{-1}}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t} \right) a_t \right\|_2^2 \\ &= n \left\| \frac{\hat{L}_{t-1}^{-1} a_t}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t} \right\|_2^2 \\ &= \frac{n\|\hat{L}_{t-1}^{-1} a_t\|_2^2}{(1 + a_t^T \hat{L}_{t-1}^{-1} a_t)^2}. \end{aligned} \tag{4.30}$$

The algorithm only needs to maintain $\hat{L}_{t-1}^{-1}$ and update it using Sherman-Morrison formula as Algorithm 6. The online algorithm for total effective resistance is given in Algorithm 7.

In an electric network, $a_t^T \hat{Y}_t a_t$ can be interpreted in terms of the mean-square voltage $\mathbf{E} v_{i_t j_t}^2$ between nodes i_t and j_t after adding edge $\{i_t, j_t\}$. When applying a random current excitation $\xi \in \mathbf{R}^m$ with zero mean and covariance matrix $I - \mathbf{1}\mathbf{1}^T/n$, the voltage across edge $\{i_t, j_t\}$ is $v_{i_t j_t} = a_t^T (a_t a_t^T + \hat{L}_{t-1})^{-1} \xi$, and the expected value of the squared voltage is $\mathbf{E} v_{i_t j_t}^2 = \mathbf{E}(a_t^T (a_t a_t^T + \hat{L}_{t-1})^{-1} \xi \xi^T (a_t a_t^T + \hat{L}_{t-1})^{-1} a_t) = a_t^T (a_t a_t^T + \hat{L}_{t-1})^{-1} (I - \mathbf{1}\mathbf{1}^T/n)(a_t a_t^T + \hat{L}_{t-1})^{-1} a_t = \|(a_t a_t^T + \hat{L}_{t-1})^{-1} a_t\|_2^2$. Therefore, $a_t^T \hat{Y}_t a_t$ can also be interpreted as the average power dissipated on edge $\{i_t, j_t\}$ when random currents are injected. Algorithm 7 adds the conductance $\hat{x}_t = 1$ only if the mean-square voltage across this edge is greater than the threshold $\hat{z}_t$.

4.6.3 Graph connectivity

When choosing the supergradient uu^T, where u is the Fiedler eigenvector of $a_t a_t^T + \hat{L}_{t-1}$, to update $\hat{Y}_t$ in (4.16), we have $a_t^T \hat{Y}_t a_t = (u_{i_t} - u_{j_t})^2$. In this case, the algorithm should compute the Fiedler eigenvector directly from $a_t a_t^T + \hat{L}_{t-1}$ in each round. When λ_2 is isolated, uu^T is the gradient and $(u_{i_t} - u_{j_t})^2$ is the first order approximation of the increase in λ_2. The algorithm allocates the budget to set $\hat{x}_t = 1$ only if the increase $(u_{i_t} - u_{j_t})^2$ is above the

Algorithm 7 Online algorithm for total effective resistance

input: $\hat{L}_0^{-1} = L_0^{\dagger} + \mathbf{1}\mathbf{1}^T/n$, η, θ

for $t = 1, \ldots m$ **do**

Player two receives edge $\{i_t, j_t\}$, updates threshold $\hat{z}_t$ according to (4.22) for soft budget case or (4.26) for the hard budget case; and computes the mean square voltage

$$\mathbf{E}v_{i_t j_t}^2 = \frac{n\|\hat{L}_{t-1}^{-1} a_t\|_2^2}{(1 + a_t^T \hat{L}_{t-1}^{-1} a_t)^2}.$$

Player two sets the edge weight $\hat{x}_t = 1$ for edge $\{i_t, j_t\}$ if $\mathbf{E}v_{i_t j_t}^2 > \hat{z}_t$; otherwise set $\hat{x}_t = 0$.

Player two updates $\hat{L}_t^{-1}$

$$\hat{L}_t^{-1} = \hat{L}_{t-1}^{-1} - \frac{\hat{L}_{t-1}^{-1} a_t a_t^T \hat{L}_{t-1}^{-1} \hat{x}_t}{1 + a_t^T \hat{L}_{t-1}^{-1} a_t \hat{x}_t}.$$

end for

threshold given by $\hat{z}_t$. The online algorithm for graph connectivity is given in Algorithm 8.

Algorithm 8 is related but different with the one proposed by Ghosh and Boyd [40]. Although both algorithms choose one edge at a time, the greedy algorithm presented in [40] assumes the prior knowledge on all m candidate edges, thus allows to choose any edge that gives the *largest* increase in λ_2. In contrary, since the candidate edges in our setting are revealed one by one, our algorithm has to estimate a threshold $\hat{z}_t$ and accepts a candidate edge only if the increase in λ_2 is sufficiently large.

The computation of the Fiedler eigenvector is critical but beyond the scope of this paper. We refer interested readers to [40, Section IIIA].

4.7 Numerical examples

In this section, we conducted the numerical experiments of Algorithms 6, 7 and 8 for both hard and soft budget cases. The graph has 15 nodes, 42 edges in the connected base graph,

Algorithm 8 Online algorithm for graph connectivity

input: $\hat{L}_0 = L_0$, η, θ

for $t = 1, \dots m$ **do**

Player two computes $\hat{z}_t$ according to (4.22) for the soft budget case or (4.26) for the hard budget case; and the Fiedler eigenvector u of $a_t a_t^T + \hat{L}_{t-1}$.

Player two sets the edge weight $\hat{x}_t = 1$ for edge $\{i_t, j_t\}$ if $(u_{i_t} - u_{j_t})^2 > \hat{z}_t$; otherwise $\hat{x}_t = 0$.

Player two updates and updates the graph Laplacian

$$\hat{L}_t = \hat{L}_{t-1} + a_t a_t^T \hat{x}_t.$$

end for

and 63 candidate edges. In each instance, all candidate edges are randomly permuted and revealed to the algorithm one after one. Each experiment was repeated 100 instances.

In the hard budget case, a budget b is given, and parameters θ and η are chosen by the algorithm to satisfy the hard budget constraint. Figure 4.2 demonstrates the experiments of the hard budget case for log-determinant problem. The top plot in the left figure shows the actual number of accepted edges versus the given budget b (red dash line). The bottom part in the left figure plots all objective values of online algorithm (black), offline optimal solution (blue), and random edge selection (red). The right figure in Figure 4.2 shows the competitive ratio in the top plot and regret in the bottom plot of online algorithm (blue) and random selection (red) versus the number of accepted edges. In these plots, dots are values of all instances and solid lines connect averaged value of instances. Figure 4.3 and Figure 4.4 correspond to graph connectivity problem and total effective resistance problem, respectively.

In the soft budget case, parameters θ and η are given to the algorithm, and we tested the algorithm for a large number of θ and η values. Figure 4.5 shows the number of accepted edges for different values of θ and η, and Figure 4.6 shows the competitive ratio and regret

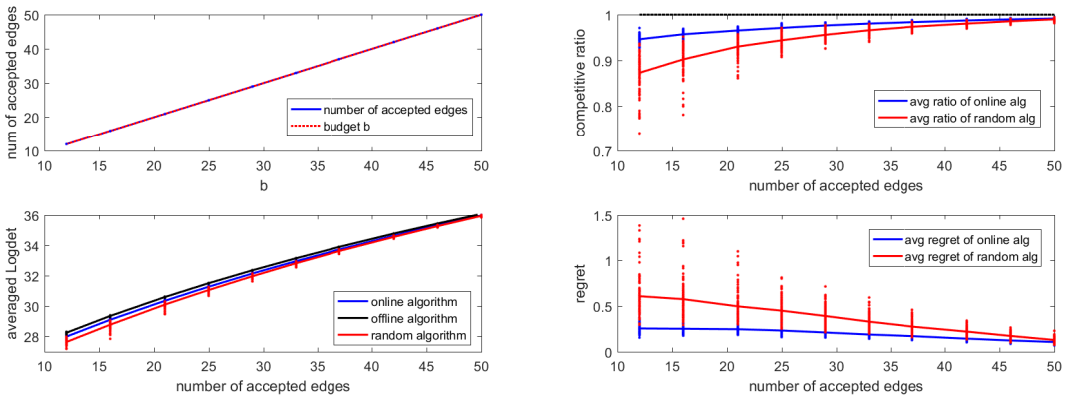


Figure 4.2: Numerical results of the hard budget network formation of the log-determinant problem

versus $\eta\theta$. Figures 4.7, 4.8, and Figures 4.9, 4.10 correspond to graph connectivity problem and total effective resistance problem, respectively.

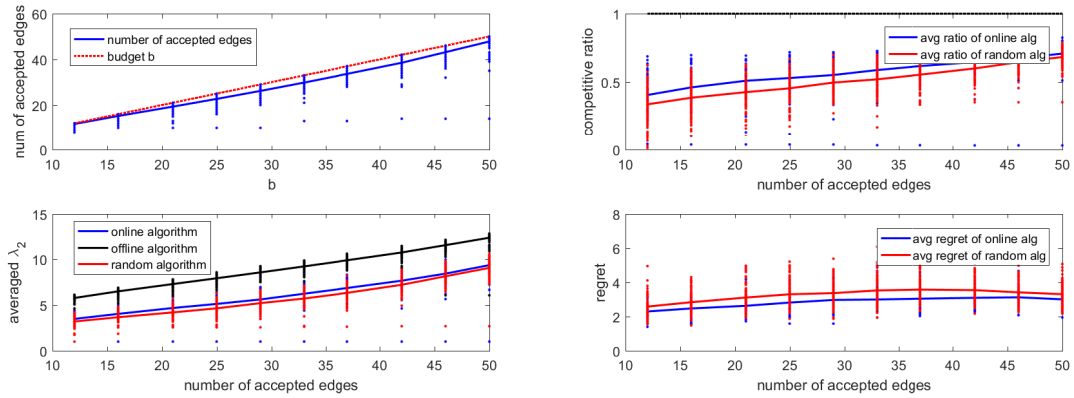


Figure 4.3: Numerical results of the hard budget network formation of the graph connectivity problem

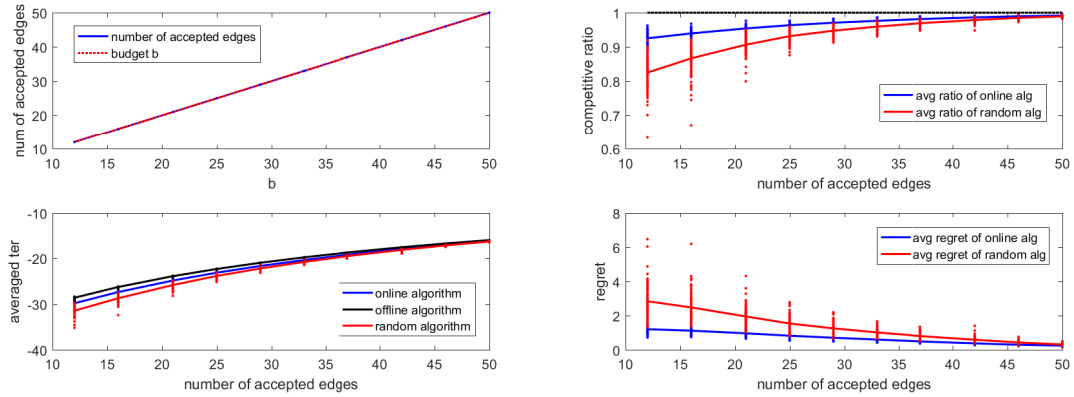


Figure 4.4: Numerical results of the hard budget network formation of the total effective resistance

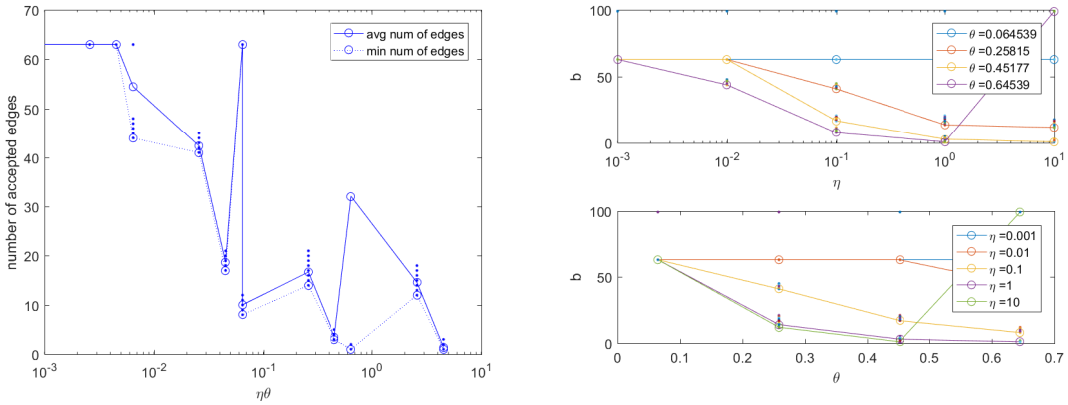


Figure 4.5: The number of accepted edges of the soft budget network formation of the log-determinant problem

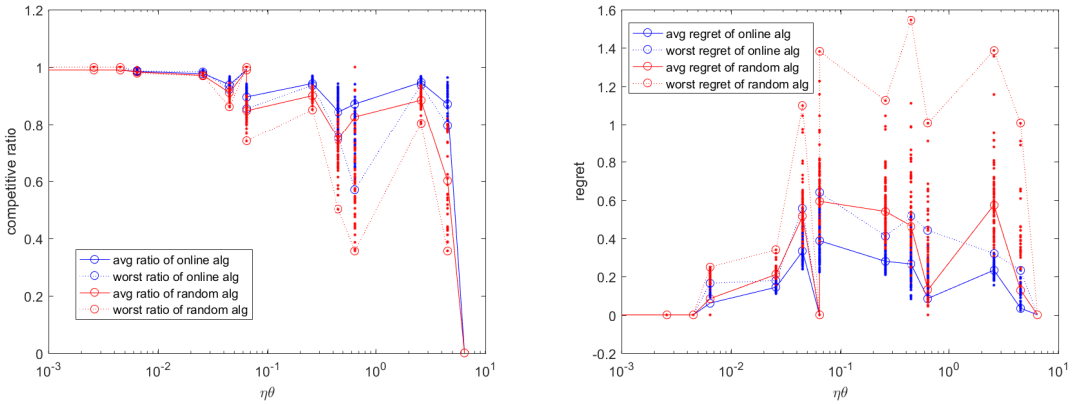


Figure 4.6: Competitive ratio and regret of the soft budget network formation of the log-determinant problem

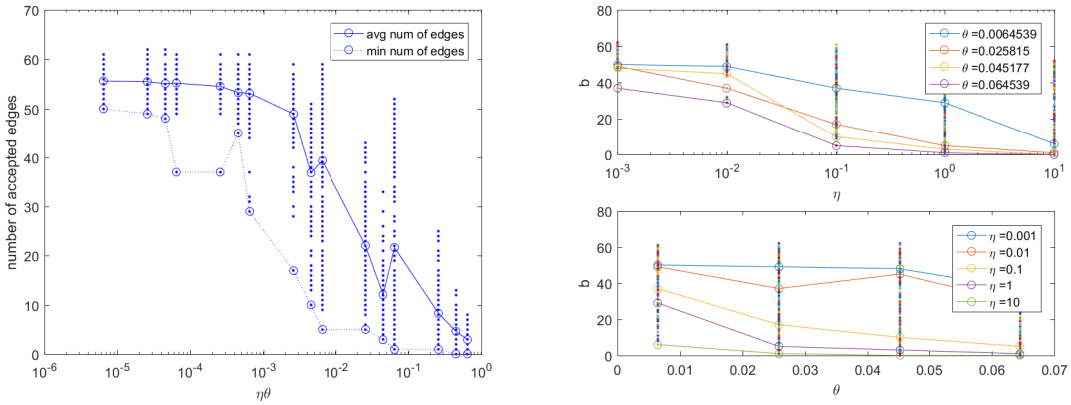


Figure 4.7: The number of accepted edges of the soft budget network formation of the graph connectivity problem

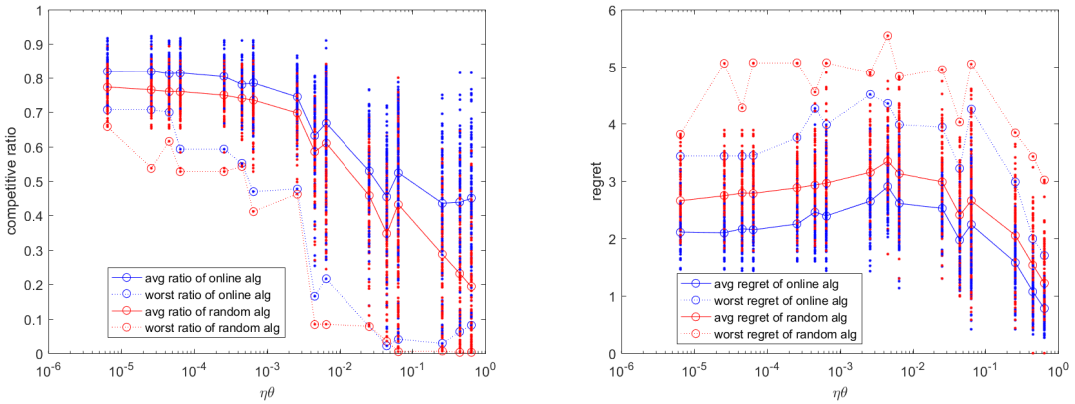


Figure 4.8: Competitive ratio and regret of the soft budget network formation of the graph connectivity problem

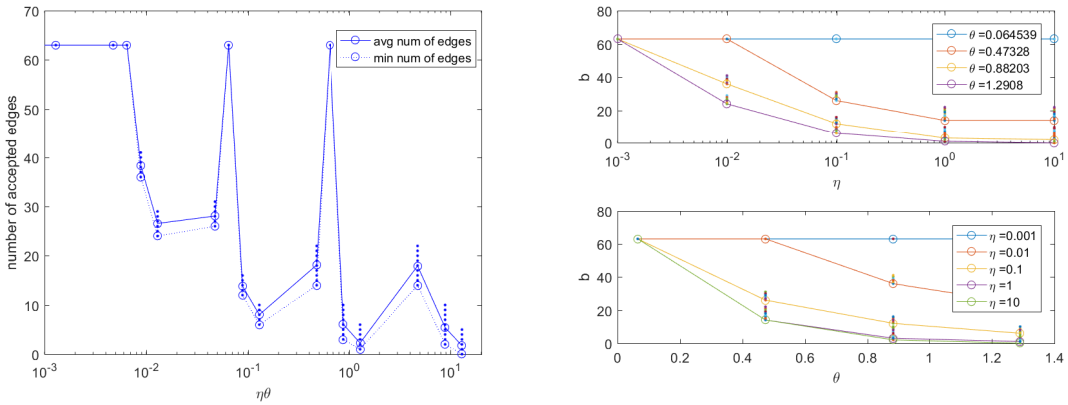


Figure 4.9: The number of accepted edges of the soft budget network formation of the total effective resistance

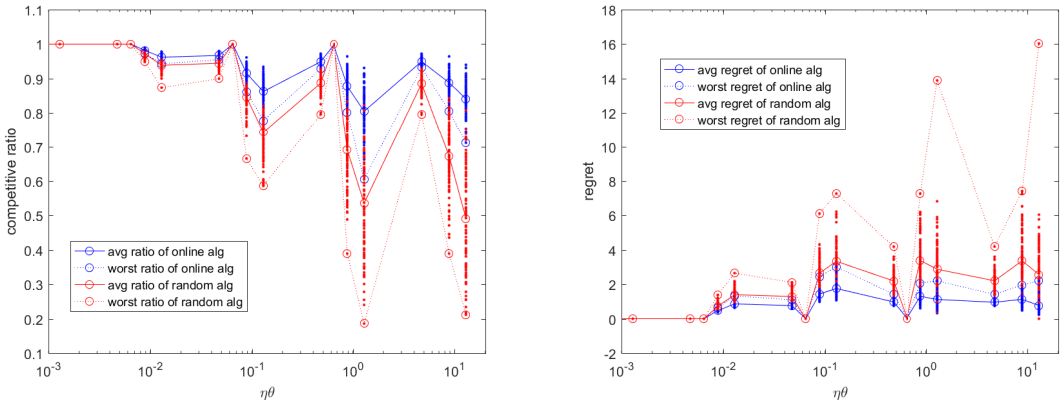


Figure 4.10: Competitive ratio and regret of the soft budget network formation of the total effective resistance

BIBLIOGRAPHY

[1] Shipra Agrawal and Nikhil R Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1405–1424. SIAM, 2015.

[2] Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A dynamic near-optimal algorithm for online linear programming. *arXiv preprint arXiv:0911.2974*, 2009.

[3] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice hall, 1993.

[4] D.H. Anderson, E.A. Catchpole, N.J. De Mestre, and T. Parkes. Modelling the spread of grass fires. *The Journal of the Australian Mathematical Society. Series B. Applied Mathematics*, 23(04):451–466, 1982.

[5] Lachlan Andrew, Siddharth Barman, Katrina Ligett, Minghong Lin, Adam Meyerson, Alan Roytman, and Adam Wierman. A tale of two metrics: Simultaneous bounds on competitiveness and regret. In *ACM SIGMETRICS Performance Evaluation Review*, volume 41, pages 329–330. ACM, 2013.

[6] MOSEK ApS. The mosek optimization toolbox for matlab manual, version 7.0 (revision 103). *Available online from* `http://www.mosek.com/resources/doc`.

[7] Venkatesh Bala and Sanjeev Goyal. A noncooperative model of network formation. *Econometrica*, 68(5):1181–1229, 2000.

[8] H. Bayrak and M.D. Bailey. Shortest path network interdiction with asymmetric information. *Networks*, 52(3):133–140, 2008.

[9] F. Benmansour, G. Carlier, G. Peyré, and F. Santambrogio. Numerical approximation of continuous traffic congestion equilibria. *Networks and Heterogeneous Media*, 4(3):605–623, 2009.

[10] F. Benmansour, G. Carlier, G. Peyré, and F. Santambrogio. Derivatives with respect to metrics and applications: Subgradient marching algorithm. *Numerische Mathematik*, 116(3):357–381, 2010.

[11] D.P. Bertsekas. *Network optimization: continuous and discrete models*. Athena Scientific Belmont, Massachusetts, 1998.

[12] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98(1-3):49–71, 2003.

[13] A.V. Borovskikh. Eikonal equations for an inhomogeneous anisotropic medium. *Journal of Mathematical Sciences*, 164(6):859–880, 2010.

[14] Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest mixing markov chain on a graph. *SIAM Review*, 46(4):667–689, 2004.

[15] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[16] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[17] G. Brown, M. Carlyle, J. Salmerón, and K. Wood. Defending critical infrastructure. *Interfaces*, 36(6):530–544, 2006.

[18] Niv Buchbinder, Shahar Chen, Anupam Gupta, Viswanath Nagarajan, et al. Online packing and covering framework with convex objectives. *arXiv preprint arXiv:1412.8347*, 2014.

[19] Niv Buchbinder, Shahar Chen, Joseph Naor, and Ohad Shamir. Unified algorithms for online learning and competitive analysis. *Mathematics of Operations Research*, 2016.

[20] Niv Buchbinder, Shahar Chen, and Joseph Seffi Naor. Competitive analysis via regularization. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 436–444. SIAM, 2014.

[21] Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research*, 34(2):270–286, 2009.

[22] G. Carlier, C. Jimenez, and F. Santambrogio. Optimal transportation with traffic congestion and wardrop equilibria. *SIAM Journal on Control and Optimization*, 47(3):1330–1350, 2008.

[23] Tsung-Hui Chang, Mingyi Hong, and Xiangfeng Wang. Multi-agent distributed optimization via inexact consensus admm. *IEEE Transctions on Signal Processing*, pages 482–497.

[24] Vineet Chaoji, Sayan Ranu, Rajeev Rastogi, and Rushi Bhatt. Recommendations to boost content spread in social networks. In *Proceedings of the 21st International Conference on World Wide Web*, pages 529–538. ACM, 2012.

[25] Airlie Chapman and Mehran Mesbahi. Semi-autonomous consensus: network measures and adaptive trees. *IEEE Transactions on Automatic Control*, 58(1):19–31, 2013.

[26] Hanool Choi, Sang-Hoon Kim, and Jeho Lee. Role of network structure and network effects in diffusion of innovations. *Industrial Marketing Management*, 39(1):170–177, 2010.

[27] R.L. Church, M.P. Scaparra, and R.S. Middleton. Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of American Geographers*, 94(3):491–502, 2004.

[28] J.A. Dellinger. *Anisotropic seismic wave propagation*. PhD thesis, Stanford University, 1991.

[29] Gabrielle Demange and Myrna Wooders. *Group Formation in Economics: Networks, Clubs, and Coalitions*. Cambridge University Press, 2005.

[30] J.W. Demmel. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics, 1997.

[31] Wei Deng, Ming-Jun Lai, Zhimin Peng, and Wotao Yin. Parallel multi-block admm with $o(1/k)$ convergence. *arXiv preprint arXiv:1312.3040*, 2013.

[32] John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57(3):592–606, 2012.

[33] L.C. Evans. *Partial differential equations*. American Mathematical Society, 2010.

[34] M.C. Ferris and J.S. Pang. Engineering and economic applications of complementarity problems. *Siam Review*, 39(4):669–713, 1997.

[35] M.A. Finney. Fire growth using minimum travel time methods. *Canadian Journal of Forest Research*, 32(8):1420–1424, 2002.

[36] M. Florian. *Nonlinear cost network models in transportation analysis*. Springer, 1986.

[37] Robin Forman. Determinants of Laplacians on graphs. *Topology*, 32(1):35–46, 1993.

[38] M. Fukushima, Z.Q. Luo, and P. Tseng. Smoothing functions for second-order-cone complementarity problems. *SIAM Journal on Optimization*, 12(2):436–460, 2002.

[39] D.R. Fulkerson and G.C. Harding. Maximizing the minimum source-sink path subject to a budget constraint. *Mathematical Programming*, 13(1):116–118, 1977.

[40] Arpita Ghosh and Stephen Boyd. Growing well-connected graphs. In *45th IEEE Conference on Decision and Control*, pages 6605–6611, 2006.

[41] Arpita Ghosh, Stephen Boyd, and Amin Saberi. Minimizing effective resistance of a graph. *SIAM Review*, 50(1):37–66, 2008.

[42] M. Giaquinta and S. Hildebrandt. *Calculus of variations II: the Hamiltonian formalism*, volume 1. Springer Verlag, 1996.

[43] B. Golden. A problem in network interdiction. *Naval Research Logistics Quarterly*, 25(4):711–713, 1978.

[44] Yuko Hatano and Mehran Mesbahi. Agreement over random networks. *IEEE Transactions on Automatic Control*, 50(11):1867–1872, 2005.

[45] A. Haurie and P. Marcotte. On the relationship between nash-cournot and wardrop equilibria. *Networks*, 15(3):295–308, 1985.

[46] Elad Hazan. The convex optimization approach to regret minimization. *Optimization for machine learning*, page 287, 2012.

[47] Bingsheng He and Xiaoming Yuan. On non-ergodic convergence rate of douglas-rachford alternating direction method of multipliers. 2012.

[48] Bingsheng He and Xiaoming Yuan. On the $o(1/n)$ convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012.

[49] D.D. Holm. *Geometric mechanics: Dynamics and symmetry*, volume 1. Imperial College Press, 2008.

[50] A. Iserles. *A first course in the numerical analysis of differential equations*, volume 44. Cambridge University Press, 2009.

[51] E. Israeli and R.K. Wood. Shortest-path network interdiction. *Networks*, 40(2):97–111, 2002.

[52] M. Isshiki, H. Ono, K. Hiraga, J. Ishikawa, and S. Nakadate. Lens design: global optimization with escape function. *Optical Review*, 2(6):463–470, 1995.

[53] Franck Iutzeler, Pascal Bianchi, Philippe Ciblat, and Walid Hachem. Asynchronous distributed optimization using a randomized alternating direction method of multipliers. pages 3671–3676, 2013.

[54] Ali Jadbabaie, Jie Lin, and A Stephen Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[55] Patrick Jaillet and Xin Lu. Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research*, 2013.

[56] Dusan Jakovetic, Jose MF Moura, and Joao Xavier. Distributed augmented lagrangian algorithms: convergence rate. pages 563–566, 2013.

[57] Richard Kenyon. The Laplacian and Dirac operators on critical planar graphs. *Inventiones Mathematicae*, 150(2):409–439, 2002.

[58] Yoonsoo Kim and Mehran Mesbahi. On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian. *IEEE Transactions on Automatic Control*, 51(1):116–120, 2006.

[59] R. Kingslake and R.B. Johnson. *Lens design fundamentals*. academic press, 2009.

[60] E. Konukoglu, O. Clatz, B.H. Menze, B. Stieltjes, M.A. Weber, E. Mandonnet, H. Delingette, and N. Ayache. Image guided personalization of reaction-diffusion type tumor growth models using modified anisotropic eikonal equations. *IEEE Transactions on Medical Imaging*, 29(1):77–95, 2010.

[61] L.J. LeBlanc, E.K. Morlok, and W.P. Pierskalla. An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research*, 9(5):309–318, 1975.

[62] R.J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady State and Time Dependent Problems*. SIAM, 2007.

[63] Dan Li, Kerry D Wong, Yu Hen Hu, and Akbar M Sayeed. Detection, classification, and tracking of targets. *IEEE Signal Processing Magazine*, 19(2):17–29, 2002.

[64] Qing Ling and Alejandro Ribeiro. Decentralized dynamic optimization through the alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 62(5):1185–1197, 2014.

[65] Ilan Lobel and Asuman Ozdaglar. Distributed subgradient methods for convex optimization over random networks. *IEEE Transactions on Automatic Control*, 56(6):1291–1306, 2011.

[66] D. Malacara-Hernández and Z. Malacara-Hernández. *Handbook of optical design*. CRC Press, 2013.

[67] V. Mallet, D.E. Keyes, and F.E. Fendell. Modeling wildland fire propagation with level set methods. *Computers & Mathematics with Applications*, 57(7):1089–1101, 2009.

[68] Gonzalo Mateos, Juan Andrés Bazerque, and Georgios B Giannakis. Distributed sparse linear regression. *IEEE Transactions on Signal Processing*, 58(10):5262–5276, 2010.

[69] Mehran Mesbahi and Magnus Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 2010.

[70] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[71] Angelia Nedić and Asuman Ozdaglar. Convergence rate for consensus with delays. *Journal of Global Optimization*, 47(3):437–456, 2010.

[72] Angelia Nedic and Asuman Ozdaglar. Cooperative distributed multi-agent optimization. *Convex Optimization in Signal Processing and Communications*, page 340, 2010.

[73] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

[74] V Nicosia, F Bagnoli, and V Latora. Impact of network structure on a model of diffusion and competitive interaction. *EPL (Europhysics Letters)*, 94(6):68009, 2011.

[75] Reza Olfati-Saber, Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[76] D. Peng, S. Osher, B. Merriman, and H.K. Zhao. The geometry of wulff crystal shapes and its relations with riemann problems. *Nonlinear partial differential equations (Evanston, IL, 1998)*, 238:251–303, 1999.

[77] G.L.W. Perry. Current approaches to modelling the spread of wildland fire: a review. *Progress in Physical Geography*, 22(2):222–245, 1998.

[78] G. Peyré, M. Péchaud, R. Keriven, and L.D. Cohen. Geodesic methods in computer vision and graphics. *Foundations and Trends® in Computer Graphics and Vision*, 5(3–4):197–397, 2010.

[79] A.J. Pullan, K.A. Tomlinson, and P.J. Hunter. A finite element method for an eikonal equation model of myocardial excitation wavefront propagation. *SIAM Journal on Applied Mathematics*, 63(1):324–350, 2002.

[80] G.D. Richards. An elliptical growth model of forest fire fronts and its numerical solution. *International Journal for Numerical Methods in Engineering*, 30(6):1163–1179, 1990.

[81] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM Journal on Numerical Analysis*, 29(3):867–884, 1992.

[82] T.P. Schulze and R.V. Kohn. A geometric model for coarsening during spiral-mode growth of thin films. *Physica D: Nonlinear Phenomena*, 132(4):520–542, 1999.

[83] J.A. Sethian. Fast marching methods. *SIAM review*, 41(2):199–235, 1999.

[84] J.A. Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*. Number 3. Cambridge University Press, 1999.

[85] J.A. Sethian and A. Vladimirsky. Ordered upwind methods for static hamilton-jacobi equations: theory and algorithms. *SIAM Journal on Numerical Analysis*, pages 325–363, 2004.

[86] Shai Shalev-Shwartz and Yoram Singer. A primal-dual perspective of online learning algorithms. *Machine Learning*, 69(2-3):115–142, 2007.

[87] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *arXiv preprint arXiv:1307.5561*, 2013.

[88] M.J. Smith. The existence, uniqueness and stability of traffic equilibria. *Transportation Research Part B: Methodological*, 13(4):295–304, 1979.

[89] Herbert G Tanner, George J Pappas, and Vijay Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, 2004.

[90] Y.H.R. Tsai, L.T. Cheng, S. Osher, and H.K. Zhao. Fast sweeping algorithms for a class of hamilton–jacobi equations. *SIAM journal on numerical analysis*, 41(2):673–694, 2003.

[91] J.N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, 40(9):1528–1538, 1995.

[92] E. Van Den Berg and M.P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

[93] J.G. Wardrop. Some theoretical aspects of road traffic research. In *ICE Proceedings: Engineering Divisions*, volume 1, pages 325–362. Thomas Telford, 1952.

[94] Ermin Wei and Asuman Ozdaglar. Distributed alternating direction method of multipliers. In *IEEE 51st Annual Conference on Decision and Control (CDC)*, pages 5445–5450, 2012.

[95] R Kevin Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18, 1993.

[96] Lin Xiao, Stephen Boyd, and Seung-Jean Kim. Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33–46, 2007.

[97] By Michael M Zavlanos, Magnus B Egerstedt, and George J Pappas. Graph-theoretic connectivity control of mobile robot networks. *Proceedings of the IEEE*, 99(9):1525–1540, 2011.

Appendix A

SUPPLEMENT TO CHAPTER 2

A.1 Proof in Section 2.3.4

In this appendix we show that

$$h^{\circ}_{i,j}(w_{i,j},(\mathcal{A}^*\phi)_{i,j}) = \begin{cases} 0, & \|(\mathcal{A}^*\phi)_+\|_{\infty,2} \le w_{i,j} \\ +\infty, & \text{otherwise.} \end{cases}$$

$$\begin{aligned} & h^{\circ}_{i,j}(w_{i,j},(\mathcal{A}^*\phi)_{i,j}) \\ = & \max_{f_{i,j}\ge 0} \left\{ \mathbf{Tr}\left((\mathcal{A}^*\phi)^T_{i,j} f_{i,j}\right) - h_{i,j}(w_{i,j},f_{i,j}) \right\} \\ = & \max_{f_{i,j}\ge 0} \left\{ \mathbf{Tr}\left(\begin{bmatrix} \phi_{i,j}-\phi_{i-1,j} & \phi_{i,j}-\phi_{i+1,j} \\ \phi_{i,j}-\phi_{i,j-1} & \phi_{i,j}-\phi_{i,j+1} \end{bmatrix}^T \begin{bmatrix} (f_{i,j,1})_- & (f_{i-1,j,1})_+ \\ (f_{i,j,2})_- & (f_{i,j-1,2})_+ \end{bmatrix} \right) \right. \\ & \left. -w_{i,j} \left\| \begin{bmatrix} (f_{i,j,1})_- & (f_{i-1,j,1})_+ \\ (f_{i,j,2})_- & (f_{i,j-1,2})_+ \end{bmatrix} \right\|_2 \right\}. \end{aligned}$$

We fix $(f_{i,j,k})_- + (f_{i-1,j,k})_+ = \alpha_{i,j,k}$, $\alpha_{i,j,k} \geq 0$, $k = 1, 2$.

$$
\begin{aligned}
& h^{\circ}_{i,j}(w_{i,j}, (\mathcal{A}^*\phi)_{i,j}) \\
= & \max_{\alpha_{i,j,k}\geq 0} \left\{ \alpha_{i,j,1} \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} \\ \phi_{i,j} - \phi_{i+1,j} \end{bmatrix} \right)_+ \right\|_\infty + \alpha_{i,j,2} \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i,j-1} \\ \phi_{i,j} - \phi_{i,j+1} \end{bmatrix} \right)_+ \right\|_\infty - w_{i,j} \left\| \begin{bmatrix} \alpha_{i,j,1} \\ \alpha_{i,j,2} \end{bmatrix} \right\|_2 \right\} \\
= & \max_{\alpha_{i,j,k}\geq 0} \left\{ \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} & \phi_{i,j} - \phi_{i+1,j} \\ \phi_{i,j} - \phi_{i,j-1} & \phi_{i,j} - \phi_{i,j+1} \end{bmatrix} \right)_+ \right\|_{\infty,2} \left\| \begin{bmatrix} \alpha_{i,j,1} \\ \alpha_{i,j,2} \end{bmatrix} \right\|_2 - w_{i,j} \left\| \begin{bmatrix} \alpha_{i,j,1} \\ \alpha_{i,j,2} \end{bmatrix} \right\|_2 \right\} \\
= & \begin{cases} 0, & \text{if } \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} & \phi_{i,j} - \phi_{i+1,j} \\ \phi_{i,j} - \phi_{i,j-1} & \phi_{i,j} - \phi_{i,j+1} \end{bmatrix} \right)_+ \right\|_{\infty,2} \leq w_{i,j} \\ +\infty, & \text{otherwise.} \end{cases}
\end{aligned}
$$

The second equality holds if

$$
\frac{\alpha_{i,j,1}}{\alpha_{i,j,2}} = \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} \\ \phi_{i,j} - \phi_{i+1,j} \end{bmatrix} \right)_+ \right\|_\infty \Big/ \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i,j-1} \\ \phi_{i,j} - \phi_{i,j+1} \end{bmatrix} \right)_+ \right\|_\infty .
$$

The first equality follows the fact that

$$
\begin{aligned}
& \max_{(f_{i,j,1})_-\geq 0, (f_{i-1,j,1})_+\geq 0, (f_{i,j,1})_- + (f_{i-1,j,1})_+ = \alpha_{i,j,1}} \begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} \\ \phi_{i,j} - \phi_{i+1,j} \end{bmatrix}^T \begin{bmatrix} (f_{i,j,1})_- \\ (f_{i-1,j,1})_+ \end{bmatrix} \\
= & \max_{(f_{i,j,1})_-\geq 0, (f_{i-1,j,1})_+\geq 0, (f_{i,j,1})_- + (f_{i-1,j,1})_+ \leq \alpha_{i,j,1}} \begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} \\ \phi_{i,j} - \phi_{i+1,j} \end{bmatrix}^T \begin{bmatrix} (f_{i,j,1})_- \\ (f_{i-1,j,1})_+ \end{bmatrix} \\
= & \alpha_{i,j,1} \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i-1,j} \\ \phi_{i,j} - \phi_{i+1,j} \end{bmatrix} \right)_+ \right\|_\infty
\end{aligned}
$$

The equality is achieved as follows: if $\phi_{i,j} - \phi_{i-1,j} \geq 0$ and $\phi_{i,j} - \phi_{i-1,j} \geq \phi_{i,j} - \phi_{i+1,j}$,

$$
(f_{i,j,1})_- = \alpha_{i,j,1}, \quad (f_{i-1,j,1})_+ = 0, \quad (f_{i,j,1})_- + (f_{i-1,j,1})_+ = \alpha_{i,j,1};
$$

if $\phi_{i,j} - \phi_{i-1,j} \geq 0$ and $\phi_{i,j} - \phi_{i-1,j} \geq \phi_{i,j} - \phi_{i+1,j}$,

$$
(f_{i,j,1})_- = 0, \quad (f_{i-1,j,1})_+ = \alpha_{i,j,2}, \quad (f_{i,j,1})_- + (f_{i-1,j,1})_+ = \alpha_{i,j,1};
$$

if $\phi_{i,j} - \phi_{i-1,j} \leq 0$ and $\phi_{i,j} - \phi_{i+1,j} \leq 0$, then $(f_{i,j,1})_-$ and $(f_{i-1,j,1})_+$ might be anything such that $(f^1_{i,j,1})_- + (f^2_{i-1,j,1})_+ = \alpha_{i,j,1}$. However, maximization with respect to $\alpha_{i,j,1}$ will set $\alpha_{i,j,1} = 0$, so we still have $(f^1_{i,j,1})_- + (f^2_{i-1,j,1})_+ = \alpha_{i,j,1}$. The similar argument holds for

$$\begin{aligned} & \max_{(f_{i,j,2})_- \geq 0, (f_{i,j-1,2})_+ \geq 0, (f_{i,j,2})_- + (f_{i,j-1,2})_+ = \alpha_{i,j,2}} \begin{bmatrix} \phi_{i,j} - \phi_{i,j-1} \\ \phi_{i,j} - \phi_{i,j+1} \end{bmatrix}^T \begin{bmatrix} (f_{i,j,2})_- \\ (f_{i,j-1,2})_+ \end{bmatrix} \\ = \quad & \alpha_{i,j,2} \left\| \left(\begin{bmatrix} \phi_{i,j} - \phi_{i,j-1} \\ \phi_{i,j} - \phi_{i,j+1} \end{bmatrix} \right)_+ \right\|_\infty \end{aligned}$$

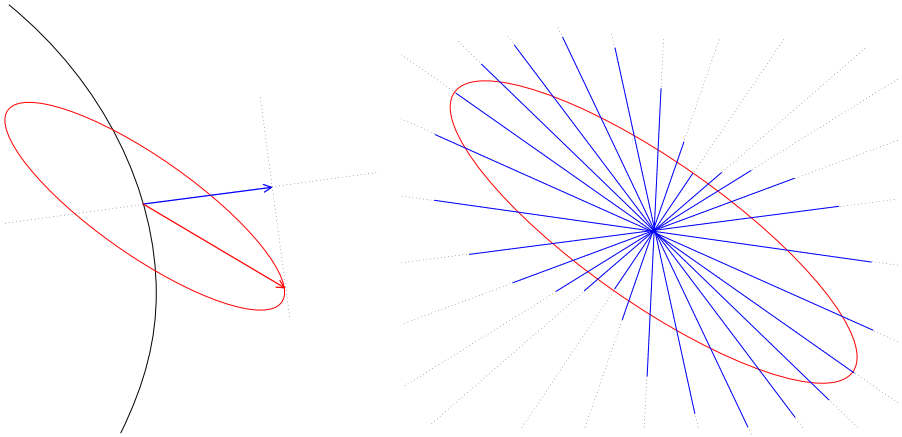
A.2 Modeling anisotropic flow propagation

In physics, flow propagation can be viewed from either the *primal* or *dual* perspectives. First define the *potential front* as any level set of the potential function, $\{x \mid \phi(x) = \phi_0\}$ for some constant ϕ_0. The primal perspective is that each point x on the potential front moves with the *flow propagation speed* $v(x) = 1/w(x)$ along the normal direction of ϕ at x, $\nabla\phi(x)/\|\phi(x)\|_2$ (assume ϕ is differentiable here). The alternative dual perspective is that the potential front expands outward with the *potential expanding speed* $s(x) = 1/w(x)$ along the direction of the normal of ϕ. This propagation is called *isotropic* because the flow propagation speed $v(x)$ is independent of the direction of the flow, and the flow propagation speed and potential expanding speed are the same (in both value and direction).

A natural way to capture anisotropy is to define a direction-dependent speed function $v(x, d)$, where d is a direction vector of unit norm, and an expanding speed function $s(x, \nabla\phi/\|\nabla\phi\|_2)$. It is known that the speeds v and s are related by the following relations [76, 85]

$$s\left(x, \frac{\nabla\phi}{\|\nabla\phi\|_2}\right) = \max_{\|d\|_2=1} \left\{ \left(-\frac{\nabla\phi}{\|\nabla\phi\|_2} \right)^T dv(x, d) \right\}, \tag{A.1}$$

From (A.1), we see that the expanding speed of the potential front is the largest projection of the flow velocity vector $v(x, d)d$ onto the direction of negative potential gradient, $-\nabla\phi(x)/\|\nabla\phi(x)\|_2$, as shown in Fig. A.1(a). Therefore, although the direction of the



(a) (b)

Figure A.1: (a) The relation between the potential expanding speed $s(x, \nabla\phi/\|\nabla\phi\|_2)$ (in the direction of the blue arrow) with the flow propagation speed profile $S_v(x) = \{v(x,d)d \mid \|d\|_2 = 1\}$ (red ellipsoid). (b) Flow propagation speed profile $S_v(x)$ (red ellipsoid) and expanding speed profile $S_s(x) = \{s(x,\tilde{d})\tilde{d} \mid \|\tilde{d}\| = 1\}$ (envelope of blue lines). The expanding speed is between the highest speed $1/\sqrt{\lambda_{\min}(W(x))}$ and the lowest speed $1/\sqrt{\lambda_{\max}(W(x))}$ along the direction of the corresponding eigenvector.

expanding speed $s(x, \nabla\phi/\|\nabla\phi\|_2)$ is fully determined by the gradient of the potential ϕ at x, the flow propagation direction d is determined by the largest projection. Consequently, $v(x,d)$ is generally not identical to $s(x, \nabla\phi/\|\nabla\phi\|_2)$. Only if the two speeds are not direction-dependent in the isotropic case and $s(x) = v(x)$, the optimal direction is $d = -\nabla\phi(x)/\|\nabla\phi(x)\|_2$ and two vectors coincide with each other.

Given the potential expanding speed s, the *anisotropic Eikonal equation*

$$\|\nabla\phi(x)\|_2 s\left(x, \frac{\nabla\phi(x)}{\|\nabla\phi(x)\|_2}\right) = 1, \tag{A.2}$$

has been seen in literatures [13, 79, 90]. Especially, if the potential expanding speed is a

matrix-parameterized norm

$$s\left(x, \frac{\nabla\phi(x)}{\|\nabla\phi(x)\|_2}\right) = \frac{\|W^{-1/2}(x)\nabla\phi(x)\|_2}{\|\nabla\phi(x)\|_2}, \tag{A.3}$$

where $W(x) \in \mathbf{S}^n_{++}$, then we have

$$\|W^{-1/2}(x)\nabla\phi(x)\|_2 = 1. \tag{A.4}$$

Note that it reduces to the isotropic case if $W(x)$ is proportional to the identity matrix, $W(x) = w^2(x)I$.

As an example for the resource constraints set $\mathcal{W}$, we consider

$$\mathcal{W} = \left\{ W \in \mathbf{S}^n_{++} \;\middle|\; \underline{w}(x)I \preceq W(x) \preceq \overline{w}(x)I, \int_\Omega \|W(x)\|dx \leq \tilde{w} \right\},$$

where $\|\cdot\|$ is a matrix norm, for instance, the nuclear norm $\|W(x)\|_* = \sum_i \lambda_i(W(x))$ or the Frobenius norm $\|W(x)\|_F = \sum_i (\lambda_i^2(W(x)))^{1/2}$. The anisotropy at x is captured by the condition number of $W(x)$. Since the eigenvalues are lower bounded by $\underline{w}(x)$ and upper bounded by $\overline{w}(x)$, the condition number is upper bounded by $\overline{w}(x)/\underline{w}(x)$. The constraint $\int_\Omega \|W(x)\|dx \leq \tilde{w}$ denotes the total budget limit over Ω.

A.3 ADMM extension to multiple sources and destinations

We extend the ADMM discussed in Section 2.5.2 to the source set and destination case. The problem (2.19) with a source set S and destination set T can be written in ADMM form as

$$\begin{array}{ll} \underset{w,\phi,\phi_S,\phi_T,u,l}{\text{minimize}} & \phi_T - \phi_S + I_K(u,l) \\ \text{subject to} & u = w \\ & l = \mathcal{D}\phi \\ & \phi_{i,j} = \phi_S, \quad (i,j) \in S \\ & \phi_{i,j} = \phi_T, \quad (i,j) \in T \\ & w \in \mathcal{W}. \end{array} \tag{A.5}$$

The augmented Lagrangian using scaled dual variable is

$$\begin{aligned} & L_\rho(w,\phi,u,l,\phi_S,\phi_T,\lambda,\nu,\xi,\eta) \\ = \;& \phi_T - \phi_S + I_K(u,l) + \frac{\rho}{2}\|w-u+\lambda\|_F^2 + \frac{\rho}{2}\sum_{i,j}\|(\mathcal{D}\phi)_{i,j} - l_{i,j} + \nu_{i,j}\|_2^2 \\ & + \frac{\rho}{2}\sum_{(i,j)\in S}(\phi_{i,j} - \phi_S + \xi_{i,j})^2 + \frac{\rho}{2}\sum_{(i,j)\in T}(\phi_{i,j} - \phi_T + \eta_{i,j})^2, \end{aligned}$$

where $\xi \in \mathbf{R}^{|S|}$ and $\eta \in \mathbf{R}^{|T|}$ are additional scaled dual variables. Then ADMM consists of iterations

- (w,ϕ)-minimization step

$$(w^{(k+1)},\phi^{(k+1)}) = \operatorname*{argmin}_{\phi,w\in\mathcal{W}} L_\rho(w,\phi,u^{(k)},l^{(k)},\phi_S^{(k)},\phi_T^{(k)},\lambda^{(k)},\nu^{(k)},\xi^{(k)},\eta^{(k)}).$$

$$\begin{aligned} \phi^{(k+1)} = \operatorname*{argmin}_{\phi} \quad & \frac{\rho}{2}\sum_{i,j}\|(\mathcal{D}\phi)_{i,j} - l_{i,j}^{(k)} + \nu_{i,j}^{(k)}\|_2^2 + \frac{\rho}{2}\sum_{(i,j)\in S}(\phi_{i,j} - \phi_S^{(k)} + \xi_{i,j}^{(k)})^2 \\ & + \frac{\rho}{2}\sum_{(i,j)\in T}(\phi_{i,j} - \phi_T^{(k)} + \eta_{i,j}^{(k)})^2, \end{aligned} \tag{A.6}$$

$$w^{(k+1)} = \mathrm{Pr}_{\mathcal{W}}(u^{(k)} - \lambda^{(k)}), \tag{A.7}$$

  where $\mathrm{Pr}_{\mathcal{W}}$ denotes the Euclidean projection onto set $\mathcal{W}$.

- (u,l,ϕ_S,ϕ_T)-minimization step

$$(u^{(k+1)},l^{(k+1)},\phi_S^{(k+1)},\phi_T^{(k+1)}) = \operatorname*{argmin}_{u,l,\phi_S,\phi_T} L_\rho(w^{(k+1)},\phi^{(k+1)},u,l,\phi_S,\phi_T,\lambda^{(k)},\nu^{(k)},\xi^{(k)},\eta^{(k)}).$$

$$(u^{(k+1)},l^{(k+1)}) = \mathrm{Pr}_K(w^{(k+1)} + \lambda^{(k)}, \mathcal{D}\phi^{(k+1)} + \nu^{(k)}), \tag{A.8}$$

$$\phi_S^{(k+1)} = \frac{1}{\rho|S|} + \frac{\sum_{(i,j)\in S}\phi_{i,j}^{(k+1)}}{|S|} + \frac{\sum_{(i,j)\in S}\xi_{i,j}^{(k)}}{|S|},$$

$$\phi_T^{(k+1)} = -\frac{1}{\rho|T|} + \frac{\sum_{(i,j)\in T}\phi_{i,j}^{(k+1)}}{|T|} + \frac{\sum_{(i,j)\in T}\eta_{i,j}^{(k)}}{|T|}.$$

- dual variable update

$$\lambda^{(k+1)} = w^{(k+1)} - u^{(k+1)} + \lambda^{(k)},$$

$$\nu^{(k+1)} = \mathcal{D}\phi^{(k+1)} - l^{(k+1)} + \nu^{(k)},$$

$$\xi_{i,j}^{(k+1)} = \phi_{i,j}^{(k+1)} - \phi_S^{(k+1)} + \xi_{i,j}^{(k)}, \quad (i,j) \in S,$$

$$\eta_{i,j}^{(k+1)} = \phi_{i,j}^{(k+1)} - \phi_T^{(k+1)} + \eta_{i,j}^{(k)}, \quad (i,j) \in T.$$

The projections in steps (A.7) and (A.8) remain the same as in Section 2.5.3 while the step (A.6) needs to solve a slightly difference linear equation.

To minimize the function in step (A.6) with respect to ϕ, we set its gradient to zero and obtain a linear system

$$(\mathcal{D}^*\mathcal{D} + \mathcal{I}_S + \mathcal{I}_T)\phi = \mathcal{D}^*(l^{(k)} - \nu^{(k)}) + h_S^{(k)} + h_T^{(k)}, \tag{A.9}$$

where $\mathcal{I}_S, \mathcal{I}_T : \mathbf{R}^{m\times n} \to \mathbf{R}^{m\times n}$ are operators that zero out entries outside the support S and T respectively, and

$$(h_S)_{i,j}^{(k)} = \begin{cases} \phi_S^{(k)} - \xi_{i,j}^{(k)} & (i,j) \in S \\ 0 & \text{otherwise,} \end{cases}$$

and

$$(h_T)_{i,j}^{(k)} = \begin{cases} \phi_T^{(k)} - \eta_{i,j}^{(k)} & (i,j) \in T \\ 0 & \text{otherwise.} \end{cases}$$

The operators $\mathcal{I}_S$ and $\mathcal{I}_T$ turn to be a diagonal matrix with ones on diagonal entries corresponding to the support S and T respectively. Therefore, the operator $\mathcal{D}^*\mathcal{D} + \mathcal{I}_S + \mathcal{I}_T$ has a positive definite matrix realization $\kappa \in \mathbf{S}_{++}^{mn}$. We solve the sparse linear system (A.9) using the sparse Cholesky factorization with minimum degree pivoting. The sparse Cholesky factorization only needs to be carried out once.

Appendix B

SUPPLEMENT TO CHAPTER 3

B.1 Proof of Theorem 1

From (3.5),

$$-\sum_{j\in\mathcal{N}(i)}\left(\sqrt{w_{ij}}\mu_{ij}^{(k)}+\rho w_{ij}(x_i^{(k+1)}-x_j^{(k)})\right)-\rho p_i(x_i^{(k+1)}-x_i^{(k)})\in\partial f_i(x_i^{(k+1)}).$$

Due to 3.6) and $\mu_{ij}^{(k)}=-\mu_{ji}^{(k)}$, we have

$$\begin{aligned}
& \sqrt{w_{ij}}\mu_{ij}^{(k)}+\rho w_{ij}(x_i^{(k+1)}-x_j^{(k)}) \\
= \;& \sqrt{w_{ij}}\mu_{ij}^{(k+1)}+\rho w_{ij}(x_j^{(k+1)}-x_j^{(k)}) \\
= \;& \frac{\sqrt{w_{ij}}}{2}\left(\mu_{ij}^{(k+1)}-\mu_{ji}^{(k+1)}\right)+\rho w_{ij}(x_j^{(k+1)}-x_j^{(k)}).
\end{aligned}$$

Since f_i are convex

$$\begin{aligned}
& f(x^{(k+1)})-f(x) \\
= \;& \sum_{i=1}^{n}\left(f_i(x_i^{(k+1)})-f_i(x_i)\right) \\
\leq \;& \sum_{i=1}^{n}\left(\sum_{j\in\mathcal{N}(i)}\frac{\sqrt{w_{ij}}}{2}\left(\mu_{ij}^{(k+1)}-\mu_{ji}^{(k+1)}\right)(x_i-x_i^{(k+1)})\right) \\
& +\sum_{i=1}^{n}\sum_{j\in\mathcal{N}(i)}\rho w_{ij}(x_j^{(k+1)}-x_j^{(k)})(x_i-x_i^{(k+1)})+\sum_{i=1}^{n}\rho p_i(x_i^{(k+1)}-x_i^{(k)})(x_i-x_i^{(k+1)}) \\
= \;& \frac{1}{\sqrt{2}}\mu^{(k+1)T}\mathcal{I}(x-x^{(k+1)})+\rho(x-x^{(k+1)})^T(W+P)(x^{(k+1)}-x^{(k)}). \qquad \text{(B.1)}
\end{aligned}$$

Then

$$\begin{aligned} & f(x^{(k+1)}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}x^{(k+1)} \\ \overset{(3.7)}{\leq} & \frac{1}{\sqrt{2}}(\mu^\star - \mu^{(k+1)})^T\mathcal{I}x^{(k+1)} + \rho(x^\star - x^{(k+1)})^T\left(W+P\right)(x^{(k+1)} - x^{(k)}) \\ \overset{(3.6)}{=} & \frac{1}{2\rho}(\mu^\star - \mu^{(k+1)})^T(\mu^{(k+1)} - \mu^{(k)}) + \rho(x^\star - x^{(k+1)})^T H(x^{(k+1)} - x^{(k)}). \end{aligned}$$

Note that

$$\begin{aligned} & (\mu^\star - \mu^{(k+1)})^T(\mu^{(k+1)} - \mu^{(k)}) \\ = & \frac{1}{2}\|\mu^\star - \mu^{(k)}\|_2^2 - \frac{1}{2}\|\mu^\star - \mu^{(k+1)}\|_2^2 - \frac{1}{2}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2, \end{aligned} \tag{B.2}$$

and

$$\begin{aligned} & (x^\star - x^{(k+1)})^T H(x^{(k+1)} - x^{(k)}) \\ = & \frac{1}{2}\|x^\star - x^{(k)}\|_H^2 - \frac{1}{2}\|x^\star - x^{(k+1)}\|_H^2 - \frac{1}{2}\|x^{(k+1)} - x^{(k)}\|_H^2. \end{aligned} \tag{B.3}$$

Then

$$\begin{aligned} & \sum_{k=0}^{K-1}\left(f(x^{(k+1)}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}x^{(k+1)}\right) \\ \leq & \sum_{k=0}^{K-1}\left(\frac{1}{4\rho}\|\mu^\star - \mu^{(k)}\|_2^2 - \frac{1}{4\rho}\|\mu^\star - \mu^{(k+1)}\|_2^2 - \frac{1}{4\rho}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(k)}\|_H^2\right. \\ & \left. -\frac{\rho}{2}\|x^\star - x^{(k+1)}\|_H^2 - \frac{\rho}{2}\|x^{(k+1)} - x^{(k)}\|_H^2\right) \\ = & \frac{1}{4\rho}\|\mu^\star - \mu^{(0)}\|_2^2 - \frac{1}{4\rho}\|\mu^\star - \mu^{(K)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(0)}\|_H^2 - \frac{\rho}{2}\|x^\star - x^{(K)}\|_H^2 \\ & -\frac{1}{4\rho}\sum_{k=0}^{K-1}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2 - \frac{\rho}{2}\sum_{k=0}^{K-1}\|x^{(k+1)} - x^{(k)}\|_H^2 \\ \leq & \frac{1}{4\rho}\|\mu^\star - \mu^{(0)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(0)}\|_H^2. \end{aligned}$$

The last inequality is due to the condition on $p_1, \ldots, p_n$ such that H is positive semidefinite.

Since f_i convex, $f_i(\bar{x}_i) \leq \frac{1}{K}\sum_{k=0}^{K-1} f_i(x_i^{(k+1)})$. Then

$$\begin{aligned} & f(\bar{x}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}\bar{x} \\ \leq\ & \frac{1}{K}\sum_{k=0}^{K-1}\left(f(x^{(k+1)}) - f(x^\star) + \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}x^{(k+1)}\right) \\ \leq\ & \frac{1}{K}\left(\frac{1}{4\rho}\|\mu^\star - \mu^{(0)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(0)}\|_H^2\right). \end{aligned}$$

B.2 Proof of Theorem 2

From (B.1), we have

$$\begin{aligned} & f(x^{(k+1)}) - f(x^\star) \\ \leq\ & \frac{1}{\sqrt{2}}\mu^{(k+1)T}\mathcal{I}(x^\star - x^{(k+1)}) + \rho(x^\star - x^{(k+1)})^T(W+P)(x^{(k+1)} - x^{(k)}). \end{aligned}$$

Since $-\mathcal{I}^T\mu^\star/\sqrt{2} \in \partial f(x^\star)$,

$$f(x^\star) - f(x^{(k+1)}) \leq \frac{1}{\sqrt{2}}\mu^{\star T}\mathcal{I}(x^{(k+1)} - x^\star).$$

Add together

$$\begin{aligned} 0 \quad \leq\ & \frac{1}{\sqrt{2}}(\mu^\star - \mu^{(k+1)})^T\mathcal{I}(x^{(k+1)} - x^\star) + \rho(x^\star - x^{(k+1)})^T H(x^{(k+1)} - x^{(k)}) \\ \overset{(3.6)}{=}\ & \frac{1}{2\rho}(\mu^\star - \mu^{(k+1)})^T(\mu^{(k+1)} - \mu^{(k)}) + \rho(x^\star - x^{(k+1)})^T H(x^{(k+1)} - x^{(k)}). \end{aligned}$$

Use (B.2) and (B.3),

$$\begin{aligned} 0 \quad \leq\ & \frac{1}{4\rho}\|\mu^\star - \mu^{(k)}\|_2^2 - \frac{1}{4\rho}\|\mu^\star - \mu^{(k+1)}\|_2^2 - \frac{1}{4\rho}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(k)}\|_H^2 \\ & -\frac{\rho}{2}\|x^\star - x^{(k+1)}\|_H^2 - \frac{\rho}{2}\|x^{(k+1)} - x^{(k)}\|_H^2. \end{aligned}$$

Then,

$$\begin{aligned} & \frac{1}{4\rho}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2 + \frac{\rho}{2}\|x^{(k+1)} - x^{(k)}\|_H^2 \\ \leq\ & \frac{1}{4\rho}\|\mu^\star - \mu^{(k)}\|_2^2 - \frac{1}{4\rho}\|\mu^\star - \mu^{(k+1)}\|_2^2 + \frac{\rho}{2}\|x^\star - x^{(k)}\|_H^2 - \frac{\rho}{2}\|x^\star - x^{(k+1)}\|_H^2. \end{aligned}$$

Since $\frac{1}{4\rho}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2 + \frac{\rho}{2}\|x^{(k+1)} - x^{(k)}\|_H^2$ is non-increasing,

$$\begin{aligned}
& \frac{1}{2\rho^2}\|\mu^{(K+1)} - \mu^{(K)}\|_2^2 + \|x^{(K+1)} - x^{(K)}\|_H^2 \\
\leq\ & \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{2\rho^2}\|\mu^{(k+1)} - \mu^{(k)}\|_2^2 + \|x^{(k+1)} - x^{(k)}\|_H^2\right) \\
=\ & \frac{1}{K}\left(\frac{1}{2\rho^2}\|\mu^\star - \mu^{(0)}\|_2^2 - \frac{1}{2\rho^2}\|\mu^\star - \mu^{(K)}\|_2^2 + \|x^\star - x^{(0)}\|_H^2 - \|x^\star - x^{(K)}\|_H^2\right) \\
\leq\ & \frac{1}{K}\left(\frac{1}{2\rho^2}\|\mu^\star - \mu^{(0)}\|_2^2 + \|x^\star - x^{(0)}\|_H^2\right).
\end{aligned}$$

Appendix C

SUPPLEMENT TO CHAPTER 4

C.1 Analysis of Algorithm 4

In this section, we analyze the performance of Algorithm 4. Introduce the following notations

- $\hat{P}_t = \phi(\sum_{s=1}^{t} a_s a_s^T \hat{x}_s)$ is the primal objective value of first t online decisions $\hat{x}_1, \ldots, \hat{x}_t$, thus $\hat{P}_0$ and $\hat{P}_m$ are the online primal objective values at the beginning and end of the algorithm,

- $P_m^\star = \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star)$ and $D_m^\star = \sum_{t=1}^{m} a_t^T Y^\star a_t x^\star - \phi^*(Y^\star)$ is the true optimal primal and dual objective value of problem (4.1) with the budget equal to the number of edges chosen in the online algorithm, thus $\sum_{t=1}^{m} x_t^\star = \sum_{t=1}^{m} \hat{x}_t$. Note that $P_m^\star \leq D_m^\star$.

The performance is measured by the *competitive ratio* c defined as

$$\hat{P}_m - \hat{P}_0 \geq c(P_m^\star - \hat{P}_0).$$

The competitive ratio has long been used in the theoretic computer science community to evaluate the algorithms for online problems such as online matching [55], online covering and packing [18, 21], or general online linear and convex programming [1, 2].

Define

$$\hat{z}_{m+1} = \theta\left(1 - \exp\left(-\eta\theta \sum_{t=1}^{m} \hat{x}_t\right)\right), \tag{C.1}$$

$$\hat{z}_{m+1} = \frac{\theta \exp(\eta\theta \sum_{t=1}^{m} \hat{x}_t)}{1 + \exp(\eta\theta \sum_{t=1}^{m} \hat{x}_t)}, \tag{C.2}$$

$$\hat{Y}_{m+1} \in \partial\phi\left(\sum_{t=1}^{m} a_t a_t^T \hat{x}_t\right). \tag{C.3}$$

The first lemma is to lower bound the change of the objective function in each round by using the gradient.

Lemma 2.

$$\hat{P}_t - \hat{P}_{t-1} \geq a_t^T \hat{Y}_t a_t \hat{x}_t. \tag{C.4}$$

Proof. Since $\hat{Y}_t \in \partial\phi(a_t a_t^T + \sum_{s=1}^{t-1} a_s a_s^T \hat{x}_s)$ and $\phi(\cdot)$ is concave,

$$\hat{P}_t - \hat{P}_{t-1} \geq a_t^T \hat{Y}_t a_t,$$

if $\hat{x}_t = 1$, and $\hat{P}_t = \hat{P}_{t-1}$ if $\hat{x}_t = 0$. □

Next lemma attempts to quantify the increase of the objective value, $\hat{P}_t - \hat{P}_{t-1}$, in terms of the decrease of the budget function, $\psi_s(\sum_{s=1}^{t-1} \hat{x}_s) - \psi_s(\sum_{s=1}^{t} \hat{x}_s)$, in each round. Recall from (4.18) that $-\hat{z}_t = \nabla\psi_s(1 + \sum_{s=1}^{t-1} \hat{x}_s)$, and

$$\hat{z}_t \hat{x}_t \geq \psi_s(\sum_{s=1}^{t-1} \hat{x}_s) - \psi_s(\sum_{s=1}^{t} \hat{x}_s). \tag{C.5}$$

Recall that the assigning rule in Algorithm 4 implies that

$$a_t^T \hat{Y}_t a_t \hat{x}_t \geq \hat{z}_t \hat{x}_t. \tag{C.6}$$

Lemma 3. *If* $\psi_s(0) = 0$,

$$\hat{P}_m - \hat{P}_0 \geq -\psi_s\left(\sum_{t=1}^{m} \hat{x}_t\right).$$

Proof. From (C.4) of Lemma 2,

$$\begin{aligned} \hat{P}_t - \hat{P}_{t-1} &\geq a_t^T \hat{Y}_t a_t \hat{x}_t \overset{(C.6)}{\geq} \hat{z}_t \hat{x}_t \\ &\overset{(C.5)}{\geq} \psi_s(\sum_{s=1}^{t-1} \hat{x}_s) - \psi_s(\sum_{s=1}^{t} \hat{x}_s) \end{aligned}$$

Then sum over $t = 1, \ldots, m$

$$\begin{aligned}
\hat{P}_m - \hat{P}_0 &\geq \sum_{t=1}^{m} (\hat{P}_t - \hat{P}_{t-1}) \\
&\geq \sum_{t=1}^{m} \left(\psi_s(\sum_{s=1}^{t-1} \hat{x}_s) - \psi_s(\sum_{s=1}^{t} \hat{x}_s) \right) \\
&\geq \psi_s(0) - \psi_s(\sum_{t=1}^{m} \hat{x}_t)
\end{aligned}$$

□

Lemma 4. *Assume that* $b' \leq \sum_{t=1}^{m} \hat{x}_t$ *and* $\psi_s(0) = 0$,

$$-\psi_s \left(\sum_{t=1}^{m} \hat{x}_t \right) \geq \frac{1 + \eta\theta b' - \exp(-\eta\theta b')}{\eta\theta b'} \theta \sum_{t=1}^{m} \hat{x}_t. \tag{C.7}$$

Proof. From (4.11), we have

$$-\psi_s \left(\sum_{t=1}^{m} \hat{x}_t \right) = \theta \sum_{t=1}^{m} \hat{x}_t + \frac{1}{\eta} \left(\exp \left(-\eta\theta \sum_{t=1}^{m} \hat{x}_t \right) - 1 \right).$$

Since $\sum_{t=1}^{m} \hat{x}_t \geq b'$,

$$\exp \left(-\eta\theta \sum_{t=1}^{m} \hat{x}_t \right) - 1 \geq \frac{1 - \exp(-\eta\theta b')}{\theta b'} \theta \sum_{t=1}^{m} \hat{x}_t,$$

then

$$-\psi_s \left(\sum_{t=1}^{m} \hat{x}_t \right) \geq \frac{1 + \eta\theta b' - \exp(-\eta\theta b')}{\eta\theta b'} \theta \sum_{t=1}^{m} \hat{x}_t.$$

□

Finally we show the competitive ratio of Algorithm 4.

Theorem 3. *Assume that* $b' \leq \sum_{t=1}^{m} \hat{x}_t$ *and* $\psi_s(0) = 0$. *Then*

$$\hat{P}_m - \hat{P}_0 \geq c(P_m^\star - \hat{P}_0),$$

where

$$c = \left(2 + \frac{\eta\theta b'}{1 + \eta\theta b' - \exp(-\eta\theta b')} \right)^{-1}.$$

Proof.

$$\begin{aligned}
D_m^\star &= \sum_{t=1}^m a_t^T Y^\star a_t x_t^\star - \phi^*(Y^\star) \\
&\le \sum_{t=1}^m a_t^T \hat{Y}_{m+1} a_t x_t^\star - \phi^*(\hat{Y}_{m+1}) \\
&\le \sum_{t=1}^m a_t^T \hat{Y}_t a_t x_t^\star - \phi^*(\hat{Y}_{m+1}) \\
&= \sum_{t=1}^m (a_t^T \hat{Y}_t a_t - \hat{z}_t) x_t^\star + \sum_{t=1}^m \hat{z}_t x_t^\star - \phi^*(\hat{Y}_{m+1}) \\
&\le \sum_{t=1}^m (a_t^T \hat{Y}_t a_t - \hat{z}_t) \hat{x}_t + \sum_{t=1}^m \hat{z}_t x_t^\star - \phi^*(\hat{Y}_{m+1}) \\
&\le \sum_{t=1}^m a_t^T \hat{Y}_t a_t \hat{x}_t + \sum_{t=1}^m \hat{z}_t x_t^\star - \phi^*(\hat{Y}_{m+1}) \\
&\le \sum_{t=1}^m a_t^T \hat{Y}_t a_t \hat{x}_t + \theta \sum_{t=1}^m x_t^\star - \phi^*(\hat{Y}_{m+1}) \\
&\le \sum_{t=1}^m a_t^T \hat{Y}_t a_t \hat{x}_t + \theta \sum_{t=1}^m \hat{x}_t - \phi^*(\hat{Y}_{m+1}).
\end{aligned}$$

The third inequality is due to $\hat{Y}_t \preceq \hat{Y}_{m+1}$ for $t = 1, \ldots, m$. From Lemma 2,

$$\hat{P}_t - \hat{P}_{t-1} \ge a_t^T \hat{Y}_t a_t \hat{x}_t.$$

Therefore,

$$\sum_{t=1}^m a_t^T \hat{Y}_t a_t \hat{x}_t \le \hat{P}_m - \hat{P}_0. \tag{C.8}$$

Next we bound $\theta \sum_{t=1}^m \hat{x}_t$. From Lemmas 3 and 4,

$$\theta \sum_{t=1}^m \hat{x}_t \le \frac{\eta\theta b'}{1 + \eta\theta b' - \exp(-\eta\theta b')} (\hat{P}_m - \hat{P}_0). \tag{C.9}$$

Last, we bound

$$-\phi^*(\hat{Y}_{m+1}) \le \hat{P}_m. \tag{C.10}$$

Plugging (C.8)(C.9)(C.10) into $D_m^\star - \hat{P}_0$, we get

$$\hat{P}_m - \hat{P}_0 \ge c(D_m^\star - \hat{P}_0) \ge c(P_m^\star - \hat{P}_0),$$

where

$$c = \left(2 + \frac{\eta\theta b'}{1 + \eta\theta b' - \exp(-\eta\theta b')}\right)^{-1}.$$

□

C.2 Analysis of Algorithm 5

Assume a random permutation model: edges $e_1, \ldots, e_m$ are chosen in a uniformly random order, i.e. given a random permutation π, edges come in a sequence $e_{\pi(1)}, \ldots, e_{\pi(m)}$. Vectors $a_1, \ldots, a_m$ are corresponding to this sequence of edges.

The *regret* is defined as

$$\begin{aligned} R(m) &= P_m^\star - \mathbf{E}[\hat{P}_m] \\ &= \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) - \mathbf{E}[\phi(\sum_{t=1}^{m} a_t a_t^T \hat{x}_t)], \end{aligned}$$

where $x_1^\star, \ldots, x_m^\star$ are the true optimal solution of problem (4.1) with hard budget b. Note that $P_m^\star = \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) = \phi(m\,\mathbf{E}[a_t a_t^T x_t^\star])$.

Lemma 5.

$$P_m^\star \leq \sum_{t=1}^{m} \mathbf{E}[(a_t^T \hat{Y}_t a_t \hat{x}_t - \frac{\phi^*(\hat{Y}_t)}{m}) - \hat{z}_t(\hat{x}_t - \frac{b}{m})].$$

Proof. In each round,

$$(a_s^T \hat{Y}_s a_s x_s^\star - \frac{\phi^*(\hat{Y}_s)}{m}) - \hat{z}_s(x_s^\star - \frac{b}{m}) \leq (a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m}) - \hat{z}_s(\hat{x}_s - \frac{b}{m}),$$

where the inequality holds from the definition of $\hat{x}_s$. Then apply the expectation

$$\begin{aligned} \frac{1}{m}\phi(m\,\mathbf{E}[a_t a_t^T x_t^\star]) &= \frac{1}{m}\phi(m\,\mathbf{E}[a_t a_t^T x_t^\star]) + I(\mathbf{E}[x_s^\star] \leq \frac{b}{m}) \\ &= \inf_{Y \succeq 0} \left(\langle Y, \mathbf{E}[a_s a_s^T x_s^\star]\rangle - \frac{\phi^*(Y)}{m}\right) + \inf_{z \leq 0} z(\mathbf{E}[x_s^\star] - \frac{b}{m}) \\ &\leq \langle \hat{Y}_s, \mathbf{E}[a_s a_s^T x_s^\star]\rangle - \frac{\phi^*(\hat{Y}_s)}{m}) - \hat{z}_s(\mathbf{E}[x_s^\star] - \frac{b}{m}) \\ &= \mathbf{E}[(a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m}) - \hat{z}_s(\hat{x}_s - \frac{b}{m})], \end{aligned}$$

where the first equality come from the fact that the optimal solution is always feasible $\sum_{s=1}^{m} x_s^\star \leq b$. Sum up $s = 1, \ldots, m$

$$\phi(m\,\mathbf{E}[a_s a_s^T x_s^\star]) \;\leq\; \sum_{s=1}^{m} \mathbf{E}[(a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m}) - \hat{z}_s(\hat{x}_s - \frac{b}{m})].$$

□

Denote the regret

$$\begin{aligned} R_1(m) &= \sum_{s=1}^{m}(a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m}) - \inf_{Y \succeq 0} \sum_{s=1}^{m}(a_s^T Y a_s \hat{x}_s - \frac{\phi^*(Y)}{m}) \\ &= \sum_{s=1}^{m}(a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m}) - \phi(\sum_{s=1}^{m} a_s a_s^T \hat{x}_s). \end{aligned}$$

and

$$\begin{aligned} R_2(m) &= \sum_{s=1}^{m}(-\hat{z}_s)(\hat{x}_s - \frac{b}{m}) - \inf_{\hat{z}<0} \sum_{s=1}^{m} \hat{z}(\hat{x}_s - \frac{b}{m}) \\ &= \sum_{s=1}^{m}(-\hat{z}_s)(\hat{x}_s - \frac{b}{m}) - I(\sum_{s=1}^{m} \hat{x}_s \leq b) \\ &= \sum_{s=1}^{m}(-\hat{z}_s)(\hat{x}_s - \frac{b}{m}). \end{aligned}$$

where the last equality comes from the fact that the algorithm terminates when the budget is reached which guarantees $\sum_{s=1}^{m} \leq b$.

Theorem 4. *For online mirror descent with entropy regularization, the regret bound is*

$$R_2(m) \leq \theta L_\psi \sqrt{2m \log 2},$$

where $L_\psi = \max\{1 - \frac{b}{m}, \frac{b}{m}\}$.

Proof. Denote $f_t(z) = \sum_{s=1}^{t} z(\hat{x}_s - \frac{b}{m}) + \frac{1}{\eta} r(z)$. Remind that $-\hat{z}_{t+1} = \operatorname{argmin}_{-\theta<\hat{z}<0} f_t(\hat{z})$, therefore, $f_t'(-\hat{z}_{t+1}) = 0$. Note that $f_t''(z) = \frac{r''(z)}{\eta} = -\frac{1}{\eta z(\theta+z)}$, then $f_t''(z) \geq \frac{4}{\eta\theta^2}$. Therefore,

$$\begin{aligned} f_{t-1}(-\hat{z}_{t+1}) &\geq f_{t-1}(-\hat{z}_t) + \frac{f_{t-1}''(-\hat{z}_t)}{2}(\hat{z}_{t+1} - \hat{z}_t)^2 \\ &\geq f_{t-1}(-\hat{z}_t) + \frac{2}{\eta\theta^2}(\hat{z}_{t+1} - \hat{z}_t)^2. \end{aligned}$$

DERIVATION: $r'(z) = \frac{1}{\theta}(\log(1+\frac{z}{\theta}) - \log(-\frac{z}{\theta}))$, $r''(z) = -\frac{1}{z(\theta+z)}$. Since $-\theta < z < 0$, $r''(z) \geq \frac{4}{\theta^2}$ with equality achieved when $z = -\frac{\theta}{2}$. This can be computed by $(-\frac{1}{z(\theta+z)})' = \frac{2z+\theta}{z^2(z+\theta)^2} = 0$.

Similarly,

$$\begin{aligned} f_t(-\hat{z}_t) &\geq f_t(-\hat{z}_{t+1}) + \frac{f_t''(-\hat{z}_{t+1})}{2}(\hat{z}_{t+1} - \hat{z}_t)^2 \\ &\geq f_t(-\hat{z}_{t+1}) + \frac{2}{\eta\theta^2}(\hat{z}_{t+1} - \hat{z}_t)^2. \end{aligned}$$

Therefore,

$$\frac{4}{\eta\theta^2}(\hat{z}_{t+1} - \hat{z}_t)^2 \leq (\hat{z}_{t+1} - \hat{z}_t)\left(\hat{x}_t - \frac{b}{m}\right).$$

Since $\hat{x}_s \in \{0,1\}$, $|\hat{x}_t - \frac{b}{m}| = L_\psi \leq \max\{1 - \frac{b}{m}, \frac{b}{m}\}$

$$(\hat{z}_{t+1} - \hat{z}_t)\left(\hat{x}_t - \frac{b}{m}\right) \leq L_\psi |\hat{z}_{t+1} - \hat{z}_t|.$$

Therefore

$$|\hat{z}_{t+1} - \hat{z}_t| \leq \frac{\eta L_\psi \theta^2}{4}.$$

Denote $-\hat{z}^\star = \text{argmin}_{-\theta<\hat{z}<0} \sum_{t=1}^m \hat{z}\left(\hat{x}_t - \frac{b}{m}\right)$. It can be shown by induction that

$$\sum_{t=1}^m (-\hat{z}_{t+1})(\hat{x}_t - \frac{b}{m}) + \min_{-\theta<\hat{z}<0} \frac{r(\hat{z})}{\eta} \leq \sum_{t=1}^m (-\hat{z}^\star)(\hat{x}_t - \frac{b}{m}) + \frac{r(\hat{z}^\star)}{\eta}.$$

Therefore,

$$\begin{aligned} \sum_{t=1}^m (-\hat{z}_t)(\hat{x}_t - \frac{b}{m}) &= \sum_{t=1}^m (-\hat{z}_t)(\hat{x}_t - \frac{b}{m}) - \sum_{t=1}^m (-\hat{z}^\star)(\hat{x}_t - \frac{b}{m}) \\ &\leq \sum_{t=1}^m (\hat{z}_{t+1} - \hat{z}_t)(\hat{x}_t - \frac{b}{m}) + \frac{r(\hat{z}^\star)}{\eta} - \min_{-\theta<\hat{z}<0} \frac{r(\hat{z})}{\eta}. \end{aligned}$$

The regret is

$$\begin{aligned} R_2(m) &\leq \frac{1}{\eta}(r(\hat{z}^\star) - \min_{-\theta<\hat{z}<0} r(\hat{z})) + \sum_{t=1}^{m}(\hat{z}_{t+1} - \hat{z}_t)(\hat{x}_t - \frac{b}{m}) \\ &= \frac{r(\hat{z}^\star)}{\eta} + \sum_{t=1}^{m}(\hat{z}_{t+1} - \hat{z}_t)(\hat{x}_t - \frac{b}{m}) \\ &\leq \frac{r(\hat{z}^\star)}{\eta} + \frac{\eta m L_\psi^2 \theta^2}{4} \\ &\leq \frac{2\log 2}{\eta} + \frac{\eta m L_\psi^2 \theta^2}{4}. \end{aligned}$$

Set $\eta = \frac{2\sqrt{2\log 2}}{\theta L_\psi \sqrt{m}}$, then

$$R_2(m) = \theta L_\psi \sqrt{2m\log 2}.$$

□

Theorem 5.

$$R(m) \leq R_1(m) + R_2(m).$$

Proof.

$$\begin{aligned} R(m) &= \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) - \mathbf{E}\left[\phi(\sum_{t=1}^{m} a_t a_t^T \hat{x}_t)\right] \\ &\leq \sum_{s=1}^{m} \mathbf{E}[(a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m}) - \hat{z}_s(\hat{x}_s - \frac{b}{m})] - \mathbf{E}\left[\phi(\sum_{t=1}^{m} a_t a_t^T \hat{x}_t)\right] \\ &= \sum_{s=1}^{m} \mathbf{E}[(a_s^T \hat{Y}_s a_s \hat{x}_s - \frac{\phi^*(\hat{Y}_s)}{m} - \phi(\sum_{t=1}^{m} a_t a_t^T \hat{x}_t)) + \sum_{s=1}^{m} \mathbf{E}[(-\hat{z}_s)(\hat{x}_s - \frac{b}{m})] \\ &\leq R_1(m) + R_2(m). \end{aligned}$$

□

Concentration bound

Lemma 6. *Let $\{x_1^\star, \ldots, x_m^\star\}$ be a finite population of m points and $x_i^\star \in [0, c]$, $\hat{x}_1, \ldots, \hat{x}_k$ are k random samples without replacement. Then for any $\epsilon > 0$*

$$\mathbf{Prob}\left[\left|\frac{1}{k}\sum_{i=1}^{k}\hat{x}_i - \frac{1}{m}\sum_{i=1}^{m}x_i^\star\right| \geq \epsilon\right] \leq 2\exp\left(\frac{-2k\epsilon^2}{c^2}\right).$$

Therefore, let $\gamma = c\sqrt{\frac{\log(2/\rho)}{2k}}$, with probability $1 - O(\rho)$,

$$\left| \frac{1}{k} \sum_{i=1}^{k} \hat{x}_i - \frac{1}{m} \sum_{i=1}^{m} x_i^\star \right| \leq \gamma.$$

Theorem 6.

$$\theta \leq 2L_\phi + 5z^\star.$$

Proof. For any $\delta \geq \gamma\sqrt{\frac{\log(2/\rho)}{2k}}$, with probability $1 - O(\rho)$

$$P_k^\star(\gamma) \geq \frac{k}{m} P_m^\star - kL_\phi\gamma,$$
$$P_k^\star(4\gamma) \leq \frac{k}{m} P_m^\star + k\gamma(L_\phi + 5z^\star).$$

Then set $k = 2^\gamma - 1$

$$\begin{aligned} \theta &= \frac{P_k^\star(4\gamma) - P_k^\star(\gamma)}{\gamma k} \\ &\leq \frac{\frac{k}{m}P_m^\star + k\gamma(L_\phi + 5z^\star) - (\frac{k}{m}P_m^\star - kL_\phi\gamma)}{\gamma k} \\ &= 2L_\phi + 5z^\star. \end{aligned}$$

□

Lemma 7.

$$P_k^\star(\gamma) + kL_\phi\gamma \geq \frac{k}{m} P_m^\star.$$

Proof. Assume $x_1^\star, \ldots, x_m^\star$ are optimal offline solution of problem (4.1) with budget constraint $\frac{1}{m}\sum_{t=1}^{m} x_t^\star \leq \frac{b}{m} + \delta - \gamma$. From the optimal solution, sample $\hat{x}_1, \ldots, \hat{x}_k$. From the Chernoff-Hoeffding bounds,

$$\left| \frac{1}{k} \sum_{t=1}^{k} \hat{x}_t - \frac{1}{m} \sum_{t=1}^{m} x_t^\star \right| \leq \gamma.$$

Then

$$\begin{aligned} \frac{1}{k}\sum_{t=1}^{k} \hat{x}_t &= \left(\frac{1}{k} \sum_{t=1}^{k} \hat{x}_t - \frac{1}{m} \sum_{t=1}^{m} x_t^\star \right) + \frac{1}{m} \sum_{t=1}^{m} x_t^\star \\ &\leq \gamma + \frac{b}{m} + (\delta - \gamma) \\ &= \frac{b}{m} + \delta. \end{aligned}$$

Therefore, $\hat{x}_1, \ldots, \hat{x}_k$ are feasible to problem (4.28).

$$\begin{aligned}
\frac{m}{k}P_k^\star(\gamma) &= \phi(\frac{m}{k}\sum_{t=1}^{k} a_t a_t^T \hat{x}_t) \\
&\geq \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) + m\langle Y^\star, \frac{1}{m}\sum_{t=1}^{m} a_t a_t^T x_t^\star - \frac{1}{k}\sum_{t=1}^{k} a_t a_t^T \hat{x}_t\rangle \\
&= \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) + m\left(\frac{1}{m}\sum_{t=1}^{m} a_t^T Y^\star a_t x_t^\star - \frac{1}{k}\sum_{t=1}^{k} a_t^T Y^\star a_t \hat{x}_t\right) \\
&\geq \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) - m(\max_t a_t^T Y^\star a_t)\left|\frac{1}{m}\sum_{t=1}^{m} x_t^\star - \frac{1}{k}\sum_{t=1}^{k}\hat{x}_t\right| \\
&= \phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star) - mL_\phi\gamma \\
&= P_m^\star - mL_\phi\gamma,
\end{aligned}$$

where $L_\phi = \max_t a_t^T Y^\star a_t$, $Y^\star \in \partial\phi(\sum_{t=1}^{m} a_t a_t^T x_t^\star)$. Therefore,

$$P_k^\star(\gamma) = \frac{k}{m}P_m^\star - kL_\phi\gamma.$$

□

Lemma 8.

$$P_k^\star(4\gamma) \leq \frac{k}{m}P_m^\star + k\gamma(L_\phi + 5z^\star).$$

Proof. The dual problem of (4.28) can be written as

$$\begin{aligned}
&\max_{x_1,\ldots,x_k\in[0,1]} \min_{Y\succeq 0, z\geq 0} \sum_{s=1}^{k}(a_s^T Y a_s x_s - \frac{\phi^*(Y)}{m}) - z\sum_{s=1}^{k}(x_s - \frac{b}{m}) + zk\delta \\
= &\max_{x_1,\ldots,x_k\in[0,1]} \min_{Y\succeq 0, z\geq 0} \sum_{s=1}^{k}(a_s^T Y a_s - z)x_s - \frac{t}{m}\phi^*(Y) + z\frac{kb}{m} + zk\delta \\
= &\min_{Y\succeq 0, z\geq 0} \sum_{s=1}^{t}(a_s^T Y a_s - z)_+ - \frac{t}{m}\phi^*(Y) + zk\left(\frac{b}{m} + \delta\right).
\end{aligned}$$

Let $Y^\star$ and $z^\star$ are true dual optimal value of problem (4.1)

$$P_k^\star(\delta) \leq \sum_{s=1}^{k}(a_s^T Y^\star a_s - z^\star)_+ - \frac{k}{m}\phi^*(Y^\star) + z^\star k\left(\frac{b}{m} + \delta\right)$$

From the concentration bound, with probability $1-O(\rho)$, let $\gamma = \sqrt{\frac{\log(\frac{2}{\rho})}{2k}}$ and $\max_s(a_s^T Y^\star a_s - z^\star)_+ \leq c = \max_s(a_s^T Y^\star a_s) + z^\star = L_\phi + z^\star$,

$$\sum_{s=1}^{k}(a_s^T Y^\star a_s - z^\star)_+ \leq \frac{k}{m}\sum_{s=1}^{m}(a_s^T Y^\star a_s - z^\star)_+ + k\gamma(L_\phi + z^\star).$$

□

Then

$$\begin{aligned} P_k^\star(k) &\leq \frac{k}{m}\sum_{t=1}^{m}(a_s^T Y^\star a_s - z^\star)_+ + k\gamma(L_\phi + z^\star) - \frac{k}{m}\phi^\star(Y^\star) + z^\star\left(\frac{b}{m} + k\delta\right) \\ &= \frac{k}{m}P_m^\star + k\gamma(L_\phi + z^\star) + z^\star k\delta. \end{aligned}$$

Set $\delta = 4\gamma$, then $P_k^\star(k) \leq P_m^\star + k\gamma(L_\phi + 5z^\star)$.

VITA

De Meng received a Bachelor of Engineering Degree in Information and Electronic Engineering from Zhejiang University, Hangzhou, China in 2009, and Master of Science Degree in Applied Mathematics from University of Washington, Seattle in 2014. He is currently a Ph.D. Candidate in Electrical Engineering at University of Washington.