Better Inference with Graph Regularization

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Harlin Lee

B.S., Electrical Engineering and Computer Science, Massachusetts Institute of Technology

M.Eng., Electrical Engineering and Computer Science, Massachusetts Institute of Technology

Carnegie Mellon University

Pittsburgh, PA

August 2021

Acknowledgments

First of all, I am grateful for the generous support from the Office of Naval Research (ONR) N00014-19-1-2404, the Army Research Office (ARO) W911NF-18-1-0303, the National Institute Of Biomedical Imaging And Bioengineering of the National Institutes of Health (NIH) R01EB025018, the National Science Foundation (NSF) CCF-1826519, as well as the David H. Barakat and LaVerne Owen-Barakat Carnegie Institute of Technology (CIT) Dean's Fellowship and the CIT Dean's Fellowship from Carnegie Mellon University.

My thesis committee (Dr. Yuejie Chi, Dr. Jelena Kovačević, Dr. José Moura, and Dr. Andrea Bertozzi) has been instrumental in shaping this body of work, and I am fortunate to have received their guidance. In particular, I admire José for his endless energy and brilliance, and I am glad that a week-long workshop led to a long-term collaboration with Andrea, which I am very excited to continue during my postdoc at UCLA. Finally, my advisors, Yuejie— who is also the chair of the committee— and Jelena, are the best research advisors *and* life mentors one could ever have. They are everything I aspire to be, and I am thankful for all the times they put up with my whining.

I can't thank everyone here because that would be longer than the actual dissertation. Still, I'd like to acknowledge Jelena's group (Chaojing Duan, Dr. Anuva Kulkarni, Dr. Rohan Varma, Dr. Siheng Chen, Dr. Filipe Condessa) and Yuejie's group (Vince Monardo, Tian Tong, Boyue Li, Laixi Shi, Shicong Cen, Pedro Valdeira, Diogo Cardoso, Dr. Maxime Ferreira Da Costa) for helpful discussions and office camaraderie. Friends from Boston/MIT and Pittsburgh/CMU/ECE, none of this would have been possible without you— and a shoutout to Carmel Fiscko, Vince Monardo and Puppycat for getting me through the absurdity that was 2020. Finally, I'd like to thank my family for being the biggest proponents of my education through the years. Thank you all.

Harlin Lee

Abstract

Improved storage, sensing, and automated data collection technology has resulted in a world full of data that are noisy, incomplete, high-dimensional, and of astronomical size. This abundance of data motivates *data-driven* approaches to signal processing, while their messiness calls for more *robust* and *accurate* inference methods, e.g. by leveraging the structure of the data. *Graphs* are a natural choice to represent a diversity of structures inherent in data. Many physical signals are generated from graph-structured objects such as road and sensor networks, and time-series signals and images can be generalized to graph signals. Moreover, graphs can encode complex relationships and interactions between objects, e.g. via similarity graphs.

This thesis studies how *graph regularization* can help solve challenging inference problems more accurately and quickly. Graph regularization is a flexible technique that drives solutions of optimization problems to have desired properties with respect to a graph. We incorporate graph regularization into various learning tasks (denoising, matrix factorization, and distributed multitask learning) as well as signals of varying data complexity (scalars, vectors, and matrices). We first analyze the performance of non-convex regularizers in denoising, and observe both theoretically and experimentally that the power of graph regularization is bounded by how accurately the graph captures the underlying structure in the data. Next, we propose a fast algorithm to solve matrix factorization with a total-variation-based regularizer, and illustrate its application to hyperspectral unmixing. Finally, we derive a simple fusion framework for distributed multitask learning that linearly combines local estimates based on the task similarities and difficulties. This circumvents the complications around data sharing, e.g. privacy, and requires only one round of communication. The proposed method is instantiated to linear regression and principal component analysis (PCA), and is verified on simulated data.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation for Better Inference with Graph Regularization

Why better inference? We are surrounded by *a lot* of messy data. Technological advances— such as improved sensors (e.g. phones can collect our locations every second) and enhanced storage technology (e.g. even noisy coordinates don't have to be discarded)— have resulted in a world full of data that are noisy, incomplete, high-dimensional, and of astronomical size. This abundance of data offers great potential for data-driven approaches to signal processing, but it also poses a challenge, as traditional methods for understanding data were developed for much smaller and more curated data. Therefore, we need more *robust* and *accurate* inference methods, e.g. by leveraging the graph structure in data.

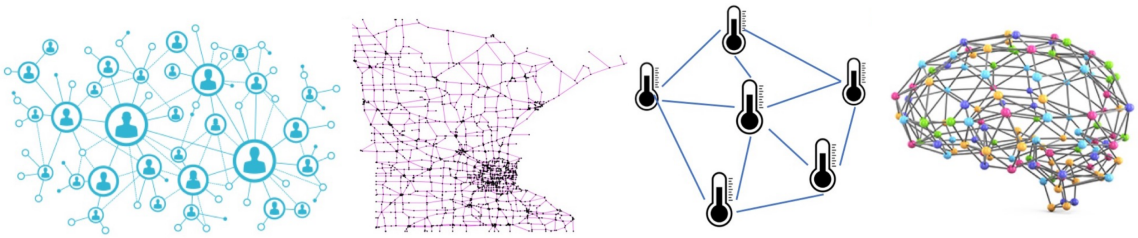


Figure 1.1: Examples of graph-structured objects.

Why graphs? First of all, many physical signals are generated from graph-structured objects to begin with, including social networks, citation networks, biological networks, and physical infrastructure; see Fig. 1.1. This includes time series signals and images, as their structures can be abstracted to a line graph and a 2-dimensional grid graph, respectively. In other situations, graphs can be an intuitive way to represent complex relationships and interactions between objects, e.g. via similarity graph. The emerging field of *graph signal processing* [1, 2, 3, 4] generalizes concepts and tools from classical discrete signal processing to these graph-structured signals.

Why regularization? Regularization is a widely-used, flexible framework that drives the solutions of an optimization problem to have desired properties. Graph regularization, in particular, focuses on properties *with respect to a graph*. In many engineering problems, we have access to prior information or intuition about the ground truth signal based on domain knowledge, and would like to incorporate it into the problem formulation. Or sometimes, the problem is ill-posed and simply impossible to solve without extra assumptions such as sparsity. Additionally, if the output of our inference problem is meant for downstream or generalized tasks, it could be helpful for the estimate to have certain characteristics, such as coefficients of small magnitude. All of these information can be embedded into the optimization problem via a carefully designed regularization term.

1.2 Classical Optimization-based Graph Regularization Framework

Classically, inferences with graph regularization are formulated as the optimization problem

$$\boxed{\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \; f(\boldsymbol{x}, \boldsymbol{\beta}) + \lambda R(\mathcal{G}, \boldsymbol{\beta}), \quad s.t. \; \boldsymbol{\beta} \in \Omega,} \tag{1.1}$$

where we aim to find a signal (or estimator or model) $\hat{\boldsymbol{\beta}}$ that is close to the unknown ground truth $\boldsymbol{\beta}^\star$ by achieving a careful balance between the data fidelity term $f(\boldsymbol{x}, \boldsymbol{\beta})$ and the graph regularization term $R(\mathcal{G}, \boldsymbol{\beta})$. See Fig. 1.2 for a more high-level description.

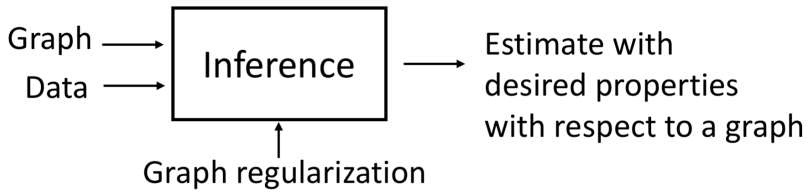


Figure 1.2: High-level block diagram of the classical graph regularization framework as the optimization problem in (1.1). The framework takes in graph ($\mathcal{G}$) and data ($\boldsymbol{x}$) as inputs, and outputs an estimate ($\hat{\boldsymbol{\beta}}$).

Data fidelity $f(\boldsymbol{x}, \boldsymbol{\beta})$: Given some observed data $\boldsymbol{x}$ and some constraint set Ω, we want a solution in Ω that explains the data well according to the function f.

Graph regularization $R(\mathcal{G}, \boldsymbol{\beta})$: The function R is designed such that the candidate $\boldsymbol{\beta}$s that are more consistent with the given assumptions will output a lower value. By taking graph $\mathcal{G}$ as an input along with candidate $\boldsymbol{\beta}$s, R measures how well $\boldsymbol{\beta}$ aligns with the structure described by $\mathcal{G}$. In general, $\boldsymbol{\beta}_i$, the ith element of $\boldsymbol{\beta}$, is associated with the ith node of the graph $\mathcal{G}$.

Parameter λ: The non-negative regularization parameter λ smoothly controls how much emphasis to put on data fidelity f versus regularization R.

1.3 Thesis Contributions: Graph Regularization and Beyond

This thesis tells two connected stories about graph regularization. First, **the optimization-based graph regularization framework (Section 1.2) can help us make more**

accurate inference from data. Chapters 2 and 3 further our understanding on this topic in three directions with two concrete examples:

1. **Different tasks** $f(\boldsymbol{x}, \boldsymbol{\beta})$: denoising and matrix factorization.

2. **Different regularizers** $R(\mathcal{G}, \boldsymbol{\beta})$: (non-)convex and (non-)smooth functions.

3. **Real-world applications**: traffic data analysis and hyperspectral unmixing.

Secondly, **graph regularization leads to a new privacy-preserving framework in distributed inference**. Chapter 4 starts off parallel to Chapters 2 and 3, by applying the classical graph regularization framework from Section 1.2 to the multitask learning problem; see Table 1.1. However, we observe that a different framework that does not require data sharing (Fig. 1.3) can achieve the same set of solutions as the classical graph regularization framework. This new perspective on graph regularization motivates a general privacy-preserving approach to distributed multitask learning.

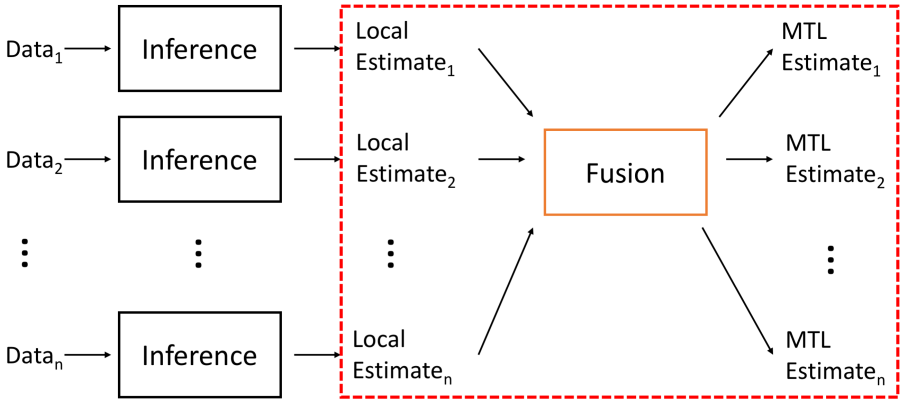


Figure 1.3: High-level block diagram of the new privacy-preserving framework for distributed multitask learning. Unlike the optimization-based graph regularization framework in Section 1.2, datasets are not shared between machines, nor are graphs given as an input, but the two frameworks achieve the same solutions under certain settings. MTL is multitask learning.

	$\boldsymbol{\beta}$	$f(\boldsymbol{x}, \boldsymbol{\beta})$	$\lambda R(\mathcal{G}, \boldsymbol{\beta})$	Reference
Denoising	$\boldsymbol{\beta}$	$\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{\beta}\|_2^2$	$\sum_i \rho\left(i\text{th element of } \boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta};\ \lambda\right)$	Ch. 2 [5, 6]
	$\boldsymbol{B}$	$\frac{1}{2}\|\boldsymbol{X}-\boldsymbol{B}\|_{\mathrm{F}}^2$	$\sum_i \rho\left(\left\|i\text{th row of } \boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}\right\|_2;\ \lambda\right)$	Ch. 2 [6]
Matrix Factorization	$\boldsymbol{S}, \boldsymbol{A}$	$\|\boldsymbol{X}-\boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2$	$\lambda \sum_{i,j} \left\|\boldsymbol{a}_i/\sqrt{D_{ii}} - \boldsymbol{a}_j/\sqrt{D_{jj}}\right\|_2^2 W_{ij}$	Ch. 3 [7]
			$\lambda \sum_{i,j} \left\|\boldsymbol{a}_i/\sqrt{D_{ii}} - \boldsymbol{a}_j/\sqrt{D_{jj}}\right\|_1 W_{ij}$	Ch. 3 [8]
Multitask Linear Regression	$\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n$	$\sum_i \|\boldsymbol{x}_i - \boldsymbol{A}_i\boldsymbol{\beta}_i\|_2^2$	$\lambda \sum_{i,j} \left\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\right\|_2^2 W_{ij}$	Ch. 4
Multitask PCA	$\boldsymbol{P}_1, \ldots, \boldsymbol{P}_n$	$\sum_i \left\|\boldsymbol{P}_i - \hat{\boldsymbol{P}}_i\right\|_{\mathrm{F}}^2$	$\lambda \sum_{i,j} \|\boldsymbol{P}_i - \boldsymbol{P}_j\|_{\mathrm{F}}^2 W_{ij}$	Ch. 4

Table 1.1: Summary of each inference problem formulated with graph regularization (1.1). PCA is Principal Component Analysis; $\boldsymbol{W}$ is the adjacency matrix of graph $\mathcal{G}$; $\boldsymbol{\Delta}$ is the oriented incidence matrix of $\mathcal{G}$; ρ is MCP, SCAD, or absolute value function; $\boldsymbol{a}_i$ is ith column of $\boldsymbol{A}$; D_{ii} is degree of node i; $\hat{\boldsymbol{P}}_i$ is a projection matrix onto a subspace of $\boldsymbol{X}_i$.

	$\mathcal{G}$	$\boldsymbol{\beta}$	Reference
Denoising Theory	Any given unweighted graph.	Any $\boldsymbol{\beta}, \boldsymbol{B}$.	Ch. 2 [5, 6]
Denoising Simulation	2-dimensional grid.	Piecewise constant $\boldsymbol{\beta}$, $\boldsymbol{B}$.	Ch. 2 [5, 6]
	Minnesota road network.	Piecewise constant $\boldsymbol{\beta}$, $\boldsymbol{B}$.	
Traffic Trend Filtering	Manhattan road network.	Taxi pick up/drop off counts.	Ch. 2 [6]
Semi-supervised Classification	Sample similarity graph, derived from pairwise feature distance.	Partially observed class labels.	Ch. 2 [5, 6]
Hyperspectral Unmixing	Pixel similarity graph, derived from pairwise spectra distance.	Abundance map ($\simeq$ material composition).	Ch. 3 [7, 8]
Multitask Learning	Similarity graph between tasks.	Estimates for each task.	Ch. 4

Table 1.2: Summary of graphs and graph signals studied for each application in this thesis.

1.4 Thesis Outline

Chapter 1 motivates and introduces graph regularization, and defines both graph and non-graph notations that are used throughout the document. Chapters 2, 3, and 4 present three works on inference with graph regularization:

- Chapter 2 examines **denoising with graph regularization** with a focus on piecewise smooth graph signals. We show theoretically and experimentally that adding graph regularization with a non-convex penalty function can improve the accuracy.
- Chapter 3 studies **matrix factorization in remote sensing**, specifically in blind hyperspectral unmixing. We apply a total-variation-based graph regularization and propose a fast approximation algorithm to solve it. We demonstrate experimentally that the algorithm can improve computational efficiency without sacrificing accuracy.
- Chapter 4 analyzes **distributed multitask learning**, where full data cannot be shared between local machines. First, we derive that graph regularization yields solutions that are linear combinations of the local estimates. We then build on that intuition to propose a general fusion framework that takes linear combinations of local estimates based on task similarity and difficulty, and illustrate how it can improve accuracy for multitask linear regression and multitask PCA.

We conclude and suggest future works in Chapter 5.

Table 1.1 summarizes how each project relates to the optimization-based graph regularization framework shown in (1.1), and Table 1.2 describes the graphs and graph signals used in each application. These chapters are organized to show a clear trend of increase in task complexity, as well as in data complexity (i.e. $\boldsymbol{\beta}_i$ is a scalar $\rightarrow$ vector $\rightarrow$ matrix). Furthermore, a unifying thread for the theoretical results on the classical graph regularization

framework is that the error between $\hat{\boldsymbol{\beta}}$ and $\boldsymbol{\beta}^\star$ can be bounded using $R(\mathcal{G}, \boldsymbol{\beta}^\star)$, confirming our intuition that the more accurate the graph, the better the inference that uses that graph.

1.5 Notation

This thesis considers graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the set of nodes, $\mathcal{E} = \{e_1, \ldots, e_m\}$ is the set of edges, and $\boldsymbol{W} = [W_{ij}] \in \mathbb{R}^{n \times n}$ is the adjacency matrix – also known as the *graph shift operator* [2]. The edge set $\mathcal{E}$ represents the connections of the graph $\mathcal{G}$, and the non-negative edge weight W_{ij} measures the underlying relation between the ith and the jth node, such as a similarity, a dependency, or a communication pattern. Depending on the application, we assume $\boldsymbol{W}$ is given to us, or derive it ourselves from the dataset, as listed in Table 1.2. The graph-structured data associated with $\mathcal{G}$ are referred to as *graph signals*. Let a scalar-valued graph signal be defined as

$$\boldsymbol{\beta} \;=\; \Big[\beta_1, \beta_2, \ldots, \beta_n\Big]^\top \in \mathbb{R}^n,$$

where β_i denotes the signal coefficient at the ith node. See Fig. 1.4 for an example graph and its graph signal. A vector-valued graph signal is defined as

$$\boldsymbol{B} = \Big[\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_d\Big] \in \mathbb{R}^{n \times d},$$

such that the ith row of the matrix $\boldsymbol{B}$ corresponds to the ith node of the graph.

Here, we define important graph notations that can be derived from the adjacency matrix $\boldsymbol{W}$. The degree matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix, where the degree of the ith node is defined as $D_{ii} = \sum_{j=1}^{n} W_{ij}$. Also, let $\boldsymbol{\Delta} \in \mathbb{R}^{m \times n}$ be the oriented incidence matrix of $\mathcal{G}$, where each row corresponds to an edge. That is, if the edge $e_i = (j, k) \in \mathcal{E}$ connects

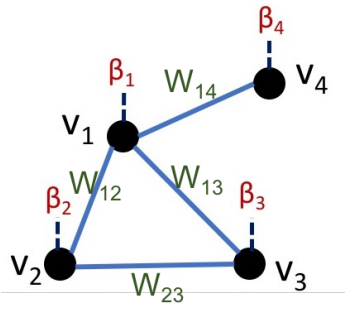


Figure 1.4: This graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W})$ has 4 nodes ($|\mathcal{V}| = 4$) and 4 edges ($|\mathcal{E}| = 4$). We call $\boldsymbol{\beta} = [\beta_1, \beta_2, \beta_3, \beta_4]$ the graph signal of $\mathcal{G}$. For $i = 1, \ldots, 4$, v_i is the ith node, and $\boldsymbol{\beta}_i$ is the signal value defined on v_i. Edge weight W_{ij} indicates the strength of the connection between v_i and v_j. For example, $W_{14} > 0$, and $W_{34} = 0$.

the jth node to the kth node ($j < k$), the entries in the ith row of $\boldsymbol{\Delta}$ is then given as

$$\Delta_{i\ell} = \begin{cases} -\sqrt{W_{jk}}, & \ell = j; \\ \sqrt{W_{jk}}, & \ell = k; \\ 0, & \text{otherwise} \end{cases} .$$

Building on these graph properties, the graph Laplacian matrix $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ and the normalized graph Laplacian matrix $\boldsymbol{L}_s \in \mathbb{R}^{n \times n}$ are

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} = \boldsymbol{\Delta}^\top \boldsymbol{\Delta}$$
$$\boldsymbol{L}_s = \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{W} \boldsymbol{D}^{-\frac{1}{2}}.$$

Throughout the thesis, we use boldface letters $\boldsymbol{a}$ and $\boldsymbol{A}$ to represent vectors and matrices respectively. The transpose of $\boldsymbol{A}$ is denoted as $\boldsymbol{A}^\top$, and the pseudo-inverse of $\boldsymbol{A}$ is defined as $\boldsymbol{A}^\dagger$. The ℓ_2 norm of a vector $\boldsymbol{a}$ is defined as $\|\boldsymbol{a}\|_2$. The spectral norm and the Frobenius norm of a matrix $\boldsymbol{A}$ are defined as $\|\boldsymbol{A}\|$ and $\|\boldsymbol{A}\|_{\mathrm{F}}$, respectively. We use the standard inner product on matrices, i.e., $\langle \boldsymbol{A}, \boldsymbol{B} \rangle = \mathrm{tr}(\boldsymbol{A}^\top \boldsymbol{B})$, where $\mathrm{tr}(\cdot)$ is the matrix trace operator that

returns the sum of all the diagonal elements. $\boldsymbol{I}_p$ is the $p \times p$ identity matrix, and $\mathbf{1}_p$ is the p-dimensional all-one vector. The subscript p may be omitted when the dimension is clear from context.

The cardinality of a set T is denoted as $|T|$. For any set $T \subseteq \{1, 2, ..., r\}$ and $\boldsymbol{x} \in \mathbb{R}^r$, we denote $(\boldsymbol{x})_T \in \mathbb{R}^{|T|}$ such that $x_\ell \in (\boldsymbol{x})_T$ if and only if $\ell \in T$ for $\ell \in \{1, 2, ..., r\}$. Similarly, we define a submatrix $\boldsymbol{A}_{T\cdot} \in \mathbb{R}^{|T| \times d}$ of $\boldsymbol{A} \in \mathbb{R}^{r \times d}$ that corresponds to pulling out the rows of $\boldsymbol{A}$ indexed by T. Unless defined otherwise, the ℓth row of a matrix $\boldsymbol{A}$ is denoted as $\boldsymbol{A}_{\ell\cdot}$, and the jth column of a matrix $\boldsymbol{A}$ is denoted as $\boldsymbol{A}_{\cdot j}$.

Expectation is denoted with $\mathbb{E}$, normal distribution with $\mathcal{N}(\mu, \sigma^2)$, and uniform distribution with $\mathcal{U}[a, b]$. $(\cdot)_+$ is a shorthand for element-wise $\max(\cdot,\ 0)$. For a function $h(\boldsymbol{x}) : \mathbb{R}^p \to \mathbb{R}$, we write $\nabla_{\boldsymbol{x}} h(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^*}$ to denote the gradient or subdifferential of $h(\boldsymbol{x})$, if they exist, evaluated at $\boldsymbol{x} = \boldsymbol{x}^*$. When the intention is clear, this may be written concisely as $\nabla h(\boldsymbol{x}^*)$. We also follow the standard asymptotic notations. If for some constants $C, N > 0$, $|f(n)| \leq C|g(n)|$ for all $n \geq N$, then $f(n) = O(g(n))$; if $g(n) = O(f(n))$, then $f(n) = \Omega(g(n))$.

Chapter 2

Denoising with Graph Regularization

2.1 Summary

This chapter studies the denoising of piecewise smooth graph signals that exhibit inhomogeneous levels of smoothness over a graph, where the value at each node can be vector-valued. We extend the graph trend filtering framework to denoising vector-valued graph signals with a family of non-convex regularizers, which exhibit superior recovery performance over existing convex regularizers. Using an oracle inequality, we establish the statistical error rates of first-order stationary points of the proposed non-convex method for generic graphs. Furthermore, we present an ADMM-based algorithm to solve the proposed method and establish its convergence. Numerical experiments are conducted on both synthetic and real-world data for denoising, support recovery, event detection, and semi-supervised classification.

2.2 Introduction

Signal estimation from noisy observations is a classic problem in signal processing and has applications in signal inpainting, collaborative filtering, recommendation systems and other large-scale data completion problems. Since noise can have deleterious, cascading effects

in many downstream tasks, being able to efficiently and accurately filter and reconstruct a signal is of significant importance.

In graph signal processing, a common assumption is that the graph signal is smooth with respect to the graph, that is, the signal coefficients do not vary much over local neighborhoods of the graph. However, this characterization is insufficient for many real-world signals that exhibit spatially inhomogeneous levels of smoothness over the graph. In social networks for example, within a given community or social circle, users' profiles tend to be homogeneous, while within a different social circle they will be of different yet homogeneous values. Consequently, the signal is often characterized by large variations between regions and small variations within regions such that there are localized discontinuities and patterns in the signal. As a result, it is necessary to develop representations and algorithms to process and analyze such *piecewise smooth* graph signals.

In this chapter, we study the denoising of the class of piecewise smooth graph signals (including but not limited to piecewise constant graph signals), which is complementary to the class of smooth graph signals that exhibit homogeneous levels of smoothness over the graph. The reconstruction of smooth graph signals has been well-studied in previous work both within graph signal processing [2, 3, 4, 9, 10, 11, 12, 13] as well as in the context of Laplacian regularization [14, 15].

The Graph Trend Filtering (GTF) framework [16], which applies total variation denoising to graph signals [17], is a particularly flexible and attractive approach that regularizes discrete graph differences using the ℓ_1 norm. Although the ℓ_1 norm-based regularization has many attractive properties [18], the resulting estimates are biased toward zero for large coefficients. To alleviate this bias effect, non-convex penalties such as the Smoothly Clipped Absolute Deviation (SCAD) penalty [19] and the Minimax Concave Penalty (MCP) [20] have been proposed as alternatives. These penalties behave similarly

to the ℓ_1 norm when the signal coefficients are small, but tend to a constant when the signal coefficients are large. Notably, they possess the so-called *oracle property*: in the asymptotics of large dimension, they perform as well as the case where we know in advance the support of the sparse vectors [21, 22, 23, 24, 25].

Work presented in this chapter strengthens the GTF framework in [16] by considering a large family of possibly non-convex regularizers (including SCAD and MCP) that exhibit superior reconstruction performance over ℓ_1 minimization for the denoising of piecewise smooth graph signals. Furthermore, we extend the GTF framework to allow vector-valued signals, e.g. time series [26], on each node of the graph, which greatly broadens the applicability of GTF to applications in social networks [27], gene networks, and semi-supervised classification [28, 29, 30]. Through theoretical analyses and empirical performance, we demonstrate that the use of non-convex penalties improves the performance of GTF in terms of both reduced reconstruction error and improved support recovery, i.e. how accurately we can localize the discontinuities of the piecewise smooth signals. Our contributions can be summarized as follows:

- Theoretically, we derive the statistical error rates of the signal estimates, defined as first-order stationary points of the proposed GTF estimator. We derive the rates in terms of the noise level and the alignment of the ground truth signal with respect to the underlying graph, without making assumptions on the piecewise smoothness of the ground truth signal. The better the alignment, the more accurate the estimates. Importantly, the estimators do not need to be the global minima of the proposed non-convex problem, which are much milder requirements and important for the success of optimization. For denoising vector-valued signals, the GTF estimate is more accurate when each dimension of the signal shares similar patterns across the graph.

- Algorithmically, we propose an ADMM-based algorithm that is guaranteed to converge to a critical point of the proposed GTF estimator.

- Empirically, we demonstrate the performance improvements of the proposed GTF estimators with non-convex penalties on both synthetic and real data for signal estimation, support recovery, event detection, and semi-supervised classification.

Table 2.1 summarizes some key notations used in this chapter for convenience.

Symbol	Description	Dimension
$\boldsymbol{\Delta}$	oriented incidence matrix	$m \times n$
$\boldsymbol{\Delta}^{(k+1)}$	kth order graph difference operator	$r \times n$
$\boldsymbol{\beta}$	scalar-valued graph signal	n
$\boldsymbol{B}$	vector-valued graph signal	$n \times d$
$\boldsymbol{x}$	noisy observation of $\boldsymbol{\beta}$	n
$\boldsymbol{X}$	noisy observation of $\boldsymbol{B}$	$n \times d$
$\boldsymbol{\Delta}_{\ell\cdot}$	ℓth row of $\boldsymbol{\Delta}$	n
$\boldsymbol{B}_{\cdot j}$	jth column of $\boldsymbol{B}$	n
$\|\boldsymbol{\Delta}^{(k+1)}\|$	spectral norm of $\boldsymbol{\Delta}^{(k+1)}$	1

Table 2.1: Key notations used in Chapter 2.

2.3 Related Work and Connections

Estimators that adapt to spatial inhomogeneities have been well studied in the literature via regularized regression, total variation and splines [31, 32, 33]. Most of these methods involve locating change points or knots that denote a distinct change in the behavior of the function or the signal.

Our work is most related to the spatially adaptive GTF estimator introduced in [16] that *smoothens* or *filters* noisy signals to promote piecewise smooth behavior with respect to the underlying graph structure; see also [34]. In the same spirit as [31], the fused LASSO and univariate trend filtering framework developed in [17, 35, 36] use discrete

difference operators to fit a time series signal using piecewise polynomials. The GTF framework generalizes univariate trend filtering by generalizing a path graph to arbitrarily complex graphs. Specifically, by appropriately defining the discrete difference operator, we can enforce piecewise constant, piecewise linear, and more generally piecewise polynomial behaviors over the graph structure. In comparison to previous work [16], in this paper, we have significantly expanded its scope by allowing vector-valued data over the graph nodes and a broader family of possibly non-convex penalties.

We note that while a significant portion of the relevant literature on GTF or the fused LASSO has focused on the sparsistency or support recovery conditions under which we can ensure the recovery of the location of the discontinuities or knots [37, 38], in this work, we study the asymptotic error rates of our estimator with respect to the mean squared error. Our analysis of error rates leverages techniques in [39, 40] that result in sharp error rates of total variation denoising via oracle inequalities, which we have carefully adapted to allow *non-convex* regularizers. The obtained error rates can be translated into bounds on support recovery or how well we can localize the boundary by leveraging techniques in [41].

Employing a graph-based regularizer that promotes similarities between the signal values at connected nodes has been investigated by many communities, such as graph signal processing, machine learning, applied mathematics, and network science. The Network LASSO proposed in [27], which is similar to the GTF framework with multi-dimensional or vector-valued data, focused on the development of efficient algorithms without any theoretical guarantees. The recent works by Jung et al. [29, 30, 42] have analyzed the performance of Network LASSO for semi-supervised learning when the graph signal is assumed to be *clustered* according to the labels using the network null space property and the network compatibility condition inspired by related concepts in compressed sensing [43].

In contrast, our analysis does not make assumptions on the graph signal, and the error rate is adaptive to the alignment of the signal and the graph structure used in denoising.

A well-studied generalization of the sparse linear inverse problem is when there are multiple measurement vectors (MMV), and the solutions are assumed to have a *common sparsity pattern* [44, 45, 46]. Sharing information across measurements, and thereby exploiting the conformity of the sparsity pattern, has been shown to significantly improve the performance of sparse recovery in compressive sensing and sparse coding [47, 48, 49, 50, 51]. Motivated by these works, we consider vector-valued graph signals that are regarded as multiple measurements of scalar-valued graph signals sharing discontinuity patterns.

There are a few variants of non-convex penalties that promote sparsity such as SCAD, MCP, weakly convex penalties, and ℓ_q $(0 \leq q < 1)$ minimization [21, 52, 53, 54, 55]. Here, we develop theory for a family of non-convex penalties parametrized similarly to that in [21, 53] with SCAD and MCP as our prime examples, although it is valid for other non-convex penalties.

2.4 Graph Trend Filtering (GTF)

For ease in derivation and clarity in presentation, theoretical results in this chapter assumes we are given an unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W})$, i.e. the elements of the adjacency matrix $\boldsymbol{W}$ are either 0 or 1. However, the results should be easily extendable to weighted graphs.

Recall the definition of oriented incidence matrix $\boldsymbol{\Delta}$ and scalar-valued graph signal $\boldsymbol{\beta}$ from Section 1.5. The entries of the signal $\boldsymbol{\Delta\beta} = [(\beta_k - \beta_j)]_{(j,k)\in\mathcal{E}}$ specify the unweighted pairwise differences of the graph signal over each edge. As a result, $\boldsymbol{\Delta}$ can be interpreted as a *graph difference operator*. In graph signal processing, a signal is called smooth over a graph $\mathcal{G}$ if $\|\boldsymbol{\Delta\beta}\|_2^2 = \sum_{(j,k)\in\mathcal{E}}(\beta_k - \beta_j)^2$ is small.

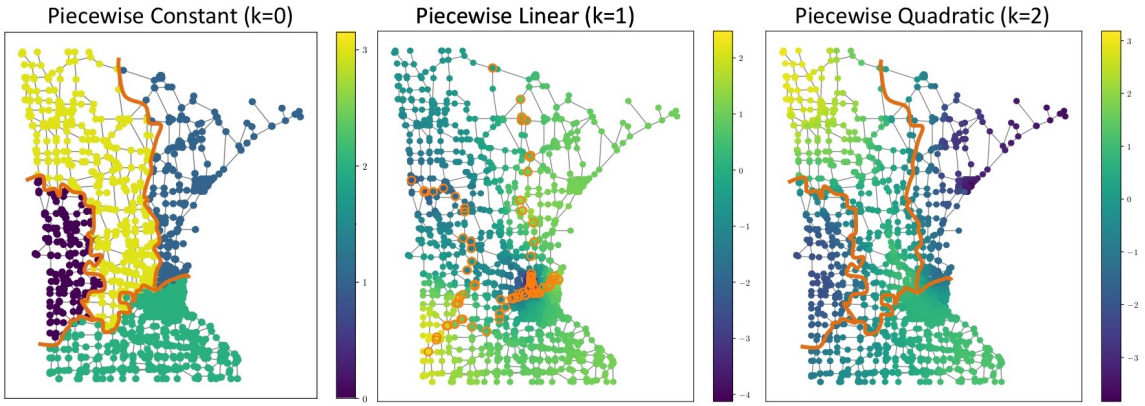


Figure 2.1: Illustration of piecewise smooth signals on the Minnesota road graph. From left to right: piecewise constant ($k = 0$), piecewise linear ($k = 1$), and piecewise quadratic ($k = 2$) graph signals. Note that the highlighted change points, i.e. the support of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star$, are edges for even k and nodes for odd k.

2.4.1 Piecewise Smooth Graph Signals

In practice, the graph signal may not be necessarily smooth over the entire graph, but only locally within different pieces of the graph. To model inhomogeneous levels of smoothness over a graph, we say that a graph signal $\boldsymbol{\beta}$ is piecewise constant over a graph $\mathcal{G}$ if many of the differences $\beta_k - \beta_j$ are zero for $(j, k) \in \mathcal{E}$. Consequently, the difference signal $\boldsymbol{\Delta}\boldsymbol{\beta}$ is sparse and $\|\boldsymbol{\Delta}\boldsymbol{\beta}\|_0$ is small.

We can characterize *piecewise kth order polynomial* signals on a graph, where the piecewise constant case corresponds to $k = 0$, by generalizing the notion of graph difference operators. Specifically, we use the following recursive definition of the kth order graph difference operator $\boldsymbol{\Delta}^{(k+1)}$ [16]. Let $\boldsymbol{\Delta}^{(1)} = \boldsymbol{\Delta}$ for $k = 0$. For $k \geq 1$, let

$$\boldsymbol{\Delta}^{(k+1)} = \begin{cases} \boldsymbol{\Delta}^{(1)\top}\boldsymbol{\Delta}^{(k)} \in \mathbb{R}^{n\times n}, & \text{odd } k \\ \boldsymbol{\Delta}^{(1)}\boldsymbol{\Delta}^{(k)} \in \mathbb{R}^{m\times n}, & \text{even } k \end{cases}. \tag{2.1}$$

The signal $\boldsymbol{\beta}$ is said to be a piecewise kth order polynomial graph signal if $\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}\|_0$ is small. To further illustrate, let us consider the piecewise linear graph signal, corresponding to $k = 1$, as a signal whose value at a node can be linearly interpolated from the weighted average of the values at neighboring nodes. It is easy to see that this is the same as requiring the second-order differences $\boldsymbol{\Delta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}$ to be sparse. Similarly, we say that a signal has a piecewise quadratic structure over a graph if the differences between the second-order differences defined for piecewise linear signals are mostly zero, that is, if $\boldsymbol{\Delta}\boldsymbol{\Delta}^\top\boldsymbol{\Delta}\boldsymbol{\beta}$ is sparse. Fig. 2.1 illustrates various orders of piecewise graph smooth signals over the Minnesota road network graph.

2.4.2 Denoising Piecewise Smooth Graph Signals via GTF

Assume we observe a noisy signal $\boldsymbol{x}$ over the graph under i.i.d Gaussian noise:

$$\boldsymbol{x} = \boldsymbol{\beta}^\star + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I}), \tag{2.2}$$

and seek to reconstruct $\boldsymbol{\beta}^\star$ from $\boldsymbol{x}$ by leveraging the graph structure. When $\boldsymbol{\beta}$ is a smooth graph signal, Laplacian smoothing [14, 15, 56, 57, 58] can be used, which solves the following problem:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\Delta}\boldsymbol{\beta}\|_2^2,$$

where $\lambda > 0$. However, it cannot localize abrupt changes in the graph signal when the signal is piecewise smooth.

Graph trend filtering (GTF) [16] is a flexible framework for estimation on graphs that is adaptive to inhomogeneity in the level of smoothness of an observed signal across

nodes. The kth order GTF estimate is defined as:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}\|_1, \tag{2.3}$$

which can be regarded as applying total variation or fused LASSO with the graph difference operator $\boldsymbol{\Delta}^{(k+1)}$ [17, 59]. The sparsity-promoting properties of the ℓ_1 norm have been well-studied [60]. Consequently, applying the ℓ_1 penalty in GTF sets many of the (higher-order) graph differences to zero while keeping a small fraction of non-zero values. GTF is then *adaptive* over the graph; its estimate at a node adapts to the smoothness in its localized neighborhood.

2.5 Proposed: Vector-valued GTF with Non-convex Penalties

In this section, we first extend GTF to allow a broader family of non-convex penalties, and then extend it to handle vector-valued signals over the graph.

2.5.1 (Non-)convex Penalties

The ℓ_1 norm penalty considered in (2.3) is well-known to produce biased estimates [61], which motivates us to extend the GTF framework to a broader class of sparsity-promoting regularizers that are not necessarily convex. We wish to minimize the following generalized kth order GTF loss function:

$$f(\boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{\beta}\|_2^2 + g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta};\lambda,\gamma), \quad \boldsymbol{\beta}\in\mathbb{R}^n, \tag{2.4}$$

where

$$g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}) \triangleq g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}; \lambda, \gamma) = \sum_{\ell=1}^{r} \rho((\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta})_\ell; \lambda, \gamma)$$

is a regularizer defined as the sum of the penalty function $\rho(\cdot; \lambda, \gamma) : \mathbb{R} \to \mathbb{R}$ applied element-wise to $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}$. Here, $r = m$ for even k and $r = n$ for odd k to account for different dimensions of $\boldsymbol{\Delta}^{(k+1)}$; see (2.1). We will refer to the GTF estimator that minimizes $f(\boldsymbol{\beta})$ as *scalar-GTF*.

Similarly to [21, 23, 53], we consider a family of penalty functions $\rho(\cdot; \lambda, \gamma)$ that satisfies the following assumptions.

Assumption 1. *Assume $\rho(\cdot; \lambda, \gamma)$ satisfies the following:*

(a) $\rho(t; \lambda, \gamma)$ satisfies $\rho(0; \lambda, \gamma) = 0$, is symmetric around 0, and is non-decreasing on the real non-negative line.

(b) For $t \geq 0$, the function $t \mapsto \frac{\rho(t;\lambda,\gamma)}{t}$ is non-increasing in t. Also, $\rho(t; \lambda, \gamma)$ is differentiable for all $t \neq 0$ and sub-differentiable at $t = 0$, with $\lim_{t\to 0^+} \rho'(t; \lambda, \gamma) = \lambda$. This upper bounds $\rho(t; \lambda, \gamma) \leq \lambda|t|$.

(c) There exists $\mu > 0$ such that $\rho(t; \lambda, \gamma) + \frac{\mu}{2}t^2$ is convex.

Many penalty functions satisfy these assumptions. Besides the ℓ_1 penalty, the non-convex SCAD [19] penalty

$$\rho_{\text{SCAD}}(t; \lambda, \gamma) = \lambda \int_0^{|t|} \min\left(1, \frac{(\gamma - u/\lambda)_+}{\gamma - 1}\right) du, \quad \gamma \geq 2,$$

and the MCP [20]

$$\rho_{\text{MCP}}(t; \lambda, \gamma) = \lambda \int_0^{|t|} \left(1 - \frac{u}{\lambda\gamma}\right)_+ du, \quad \gamma \geq 1, \tag{2.5}$$

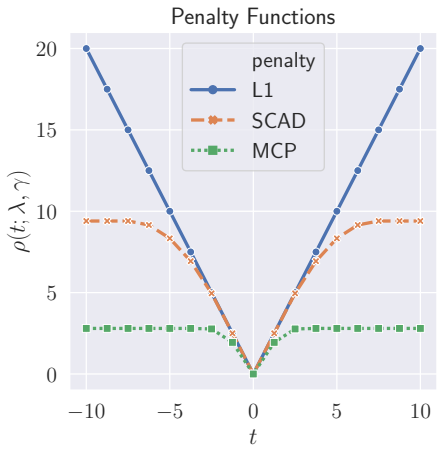


Figure 2.2: Illustration of $\rho(\cdot;\lambda,\gamma)$ for ℓ_1, SCAD ($\gamma = 3.7$), and MCP ($\gamma = 1.4$), where $\lambda = 2$. Both SCAD and MCP move towards ℓ_1 as γ increases.

also satisfy them. We note that Assumption 1 (c) is satisfied for SCAD with $\mu \geq \mu_{\text{SCAD}} = \frac{1}{\gamma-1}$ and for MCP with $\mu \geq \mu_{\text{MCP}} = \frac{1}{\gamma}$. Fig. 2.2 illustrates the ℓ_1, SCAD and MCP penalties for comparison. While the non-convexity means that in general, we may not always find the global optimum of $f(\boldsymbol{\beta})$, it often affords us many other advantages. SCAD and MCP both taper off to a constant value, and hence apply less shrinkage for higher values. As a result, they mitigate the bias effect while promoting sparsity. Further, they are smooth and differentiable for $t \geq 0$ and are both upper bounded by the ℓ_1 penalty for all t.

2.5.2 Vector-valued GTF

In many applications, the signals on each node are in fact multi-dimensional or *vector-valued*, e.g. time series in social networks, multi-class labels in semi-supervised learning, and feature vectors of different objects in feature selection. Therefore, it is natural to consider an extension to the graph signal denoising problem, where the graph signal on each node is a d-dimensional vector instead of a scalar. In this scenario, we define a

vector-valued graph signal to be piecewise smooth if it is piecewise smooth in each of its d dimensions, and assume their discontinuities to coincide over the same small set of edges or nodes. Consistent with the notation introduced in Section 1.5, we denote the vector-valued signal of interest as $\boldsymbol{B}^\star \in \mathbb{R}^{n\times d}$. The noise model for the observation matrix $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ is defined as

$$\boldsymbol{X} = \boldsymbol{B}^\star + \boldsymbol{E},$$

where each element of $\boldsymbol{E} \in \mathbb{R}^{n\times d}$ is drawn i.i.d from $\mathcal{N}(0, \sigma^2)$. A naive approach is to estimate each column $\boldsymbol{B}_{\cdot j}$ of $\boldsymbol{B}$ separately via scalar-GTF:

$$\min_{\boldsymbol{B}\in\mathbb{R}^{n\times d}} \sum_{j=1}^{d} f(\boldsymbol{B}_{\cdot j}). \tag{2.6}$$

However, this formulation does not take full advantage of the multi-dimensionality of the graph signal. Instead, when the columns of $\boldsymbol{B}$ are correlated, coupling them can be beneficial such that we encourage the sharing of information across dimensions or features. For example, if one column $\boldsymbol{B}_{\cdot i}$ exhibits strong piecewise smoothness over the graph, and therefore has compelling evidence about the relationship between nodes, sharing that information to a related column $\boldsymbol{B}_{\cdot j}$ can improve the overall denoising and filtering performance. As a result, we formulate a *vector-GTF* problem as follows:

$$\min_{\boldsymbol{B}\in\mathbb{R}^{n\times d}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{B}\|_{\mathrm{F}}^2 + h(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}; \lambda, \gamma), \tag{2.7}$$

where the new penalty function $h(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}) \triangleq h(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}; \lambda, \gamma) : \mathbb{R}^{r\times d} \to \mathbb{R}$ is the sum of $\rho(\cdot; \lambda, \gamma)$ applied to the ℓ_2 norm of each row of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B} \in \mathbb{R}^{r\times d}$:

$$h(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}\ ; \lambda, \gamma) = \sum_{\ell=1}^{r} \rho\left(\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{\ell\cdot}\|_2; \lambda, \gamma\right).$$

By enforcing sparsity on $\left\{\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{\ell\cdot}\|_2\right\}_{1\le l\le r}$, we are coupling $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}_{\cdot j}$ to have similar sparsity patterns across $j = 1, \ldots, d$. Note the difference from (2.6), where elements of $(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{\ell\cdot}$ can be set to zero or non-zero independently.

2.6 Theoretical Guarantees

In this section, we present the error rates and support recovery guarantees of the generalized GTF estimators, namely scalar-GTF (2.4) and vector-GTF (2.7), under the AWGN noise model. Before continuing, we define a few useful quantities. Let $C_{\mathcal{G}}$ be the number of connected components in the graph $\mathcal{G}$, or equivalently, the dimension of the null space of $\boldsymbol{\Delta}^{(k+1)}$. Further, let r be the number of rows of $\boldsymbol{\Delta}^{(k+1)}$, and ζ_k be the maximum ℓ_2 norm of the columns of $\boldsymbol{\Delta}^{(k+1)\dagger}$.

2.6.1 Error Rates of First-order Stationary Points

Due to non-convexity, global minima of the proposed GTF estimators may not be attainable. Therefore, it is more desirable to understand the statistical performance of any first-order stationary points of the GTF estimators. We call $\hat{\boldsymbol{\beta}} \in \mathbb{R}^n$ a stationary point of $f(\boldsymbol{\beta})$, if it satisfies

$$0 \in \nabla_{\boldsymbol{\beta}} f(\boldsymbol{\beta})|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}}.$$

We further introduce the compatibility factor, which generalizes the notion used in [39] to allow vector-valued signals.

Definition 1 (Compatibility factor)**.** *Let $\boldsymbol{\Delta}^{(k+1)}$ be fixed. The compatibility factor $\kappa_{T,d}$ of a set $T \subseteq \{1, 2, \ldots, r\}$ is defined as $\kappa_{\varnothing,d} = 1$, and for nonempty set T,*

$$\kappa_{T,d}(\boldsymbol{\Delta}^{(k+1)}) = \inf_{\boldsymbol{B}\in\mathbb{R}^{n\times d}} \left\{ \frac{\sqrt{|T|}\cdot\|\boldsymbol{B}\|_{\mathrm{F}}}{\sum_{\ell\in T}\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{\ell\cdot}\|_2} \right\}.$$

To further build intuition, consider $\sqrt{|T|}\kappa_{T,1}(\boldsymbol{\Delta})^{-1} = \sup_{\boldsymbol{\beta}\in\mathbb{R}}\{\|(\boldsymbol{\Delta})_T\boldsymbol{\beta}\|_1/\|\boldsymbol{\beta}\|_2\}$. This is precisely the definition of $\|(\boldsymbol{\Delta})_T\|_{1,2}$, an induced norm of the $|T| \times n$ submatrix of $\boldsymbol{\Delta}$. If we consider signals with fixed power $\|\boldsymbol{\beta}\|_2^2 = 1$, $\|(\boldsymbol{\Delta})_T\boldsymbol{\beta}\|_1$ will depend on how much the T edges are connected to each other. Together with $\|(\boldsymbol{\Delta})_T\boldsymbol{\beta}\|_1 \leq \sqrt{|T|}\|(\boldsymbol{\Delta})_T\boldsymbol{\beta}\|_2$, $\kappa_{T,1}(\boldsymbol{\Delta})$ can be related to the restricted eigenvalue condition, which is often used to bound the performance of LASSO [43]. With slight abuse of notation, we write $\kappa_T := \kappa_{T,d}$.

We have the following oracle inequality that is applicable to the stationary points of the GTF estimators. The proof in Appendix A.1 follows a construction that is similar to Theorem 2 in [39]. The oracle inequality holds for any $\hat{\boldsymbol{\beta}}$ that satisfies the first-order optimality condition, allowing the use of non-convex penalties. This mild condition on $\hat{\boldsymbol{\beta}}$ is a key difference from [16, Theorem 3] and [5, Theorem 1] that are applicable to the global minimizer, which is difficult to guarantee when using non-convex penalties. We also stress that although GTF was motivated by piecewise smooth graph signals, Theorem 1 holds for any graph $\mathcal{G}$ and graph signal $\boldsymbol{\beta}^\star$.

Theorem 1 (Oracle Inequality of GTF Stationary Points)**.** *Assume $\mu < 1/\|\boldsymbol{\Delta}^{(k+1)}\|^2$. Fix $\delta \in (0,1)$. For scalar-GTF (2.4), let $\hat{\boldsymbol{\beta}}$ be a stationary point. Set $\lambda = \sigma\zeta_k\sqrt{2\log\left(\frac{er}{\delta}\right)}$, then*

$$\frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2}{n} \leq \inf_{\boldsymbol{\beta}\in\mathbb{R}^n}\left\{\frac{\|\boldsymbol{\beta} - \boldsymbol{\beta}^\star\|_2^2 + 4g((\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta})_{T^c})}{n}\right\} + \frac{2\sigma^2\left[C_{\mathcal{G}} + 2\sqrt{2C_{\mathcal{G}}\log(\frac{1}{\delta})} + \frac{8\zeta_k^2|T|}{\kappa_T^2}\log(\frac{er}{\delta})\right]}{n(1-\mu\|\boldsymbol{\Delta}^{(k+1)}\|^2)} \tag{2.8}$$

with probability at least $1 - 2\delta$ for any $T \subseteq \{1, 2, ..., r\}$. Similarly, for vector-GTF (2.7),

let $\hat{\boldsymbol{B}}$ *be a stationary point. Set* $\lambda = \sigma\zeta_k\sqrt{2d\log(\frac{edr}{\delta})}$*, then*

$$\frac{\|\hat{\boldsymbol{B}} - \boldsymbol{B}^\star\|_{\mathrm{F}}^2}{dn} \leq \inf_{\boldsymbol{B}\in\mathbb{R}^{n\times d}} \left\{ \frac{\|\boldsymbol{B} - \boldsymbol{B}^\star\|_{\mathrm{F}}^2 + 4h((\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{T^c})}{dn} \right\} + \frac{2\sigma^2\left[C_{\mathcal{G}} + 2\sqrt{2C_{\mathcal{G}}\log(\frac{d}{\delta})} + \frac{8\zeta_k^2|T|}{\kappa_T^2}\log(\frac{edr}{\delta})\right]}{n(1-\mu\|\boldsymbol{\Delta}^{(k+1)}\|^2)} \tag{2.9}$$

with probability at least $1-2\delta$ *for any* $T \subseteq \{1, 2, ..., r\}$*.*

Remark 1. *Recall that* μ *is defined in Assumption 1 (c), which characterizes how "non-convex" the regularizer is, and dictates the inflection point in Fig. 2.2. The assumption* $\mu < 1/\|\boldsymbol{\Delta}^{(k+1)}\|^2$ *in Theorem 1 therefore implicitly constrains the level of non-convexity of the regularizer. Take MCP in (2.5) for example: since* $\mu \geq 1/\gamma$*, we can guarantee the existence of a valid* μ *such that* $\mu < 1/\|\boldsymbol{\Delta}^{(k+1)}\|^2$ *as long as we set* $\gamma > \|\boldsymbol{\Delta}^{(k+1)}\|^2$*.*

Theorem 1 allows one to select $\boldsymbol{\beta}$ and T to optimize the error bounds on the right hand side of (2.8) and (2.9). For example, pick $\boldsymbol{\beta} = \boldsymbol{\beta}^\star$ in (2.8) (hence an "oracle") to have

$$\frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2}{n} \leq \frac{4g((\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star)_{T^c})}{n} + \frac{2\sigma^2\left[C_{\mathcal{G}}^\delta + 8\zeta_k^2\kappa_T^{-2}|T|\log(\frac{er}{\delta})\right]}{n(1-\mu\|\boldsymbol{\Delta}^{(k+1)}\|^2)},$$

where $C_{\mathcal{G}}^\delta = C_{\mathcal{G}} + 2\sqrt{2C_{\mathcal{G}}\log(\frac{1}{\delta})}$.

- By setting T as an empty set, we have

$$\frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2}{n} \leq \frac{4g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star)}{n} + \frac{2\sigma^2 C_{\mathcal{G}}^\delta}{n(1-\mu\|\boldsymbol{\Delta}^{(k+1)}\|^2)}, \tag{2.10}$$

  which suggest that the reconstruction accuracy improves when the ground truth $\boldsymbol{\beta}^\star$ is better aligned with the graph structure, and consequently the value of $g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star)$ is small.

- On the other hand, by setting T as the support of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star$, we achieve

$$\frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2}{n} \leq \frac{2\sigma^2 \left[C_{\mathcal{G}}^{\delta} + 8\zeta_k^2 \kappa_T^{-2} \|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_0 \log(\frac{er}{\delta})\right]}{n(1 - \mu\|\boldsymbol{\Delta}^{(k+1)}\|^2)},$$

  which grows linearly as we increase the sparsity level $\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_0$.

Similar discussions can be conducted for vector-GTF by choosing $\boldsymbol{B} = \boldsymbol{B}^\star$ in (2.9). More importantly, we can directly compare the performance of vector-GTF with scalar-GTF, which was formulated for vector-valued graph signals in (2.6). The error bound of vector-GTF pays a small price in the order of $\log d$, but is tighter than scalar-GTF if $h((\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}^\star)_{T^c}) \ll \sum_{j=1}^d g((\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}^\star_{\cdot j})_{T^c})$. This suggests that vector-GTF is much more advantageous when the support sets of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}^\star_{\cdot j}$ for $j = 1, \ldots, d$ overlap, i.e. when the local discontinuities and patterns in $\boldsymbol{B}^\star_{\cdot j}$ are shared.

2.6.2 Comparison with Scalar-GTF using ℓ_1 Regularization

We compare our error bound for scalar-GTF that is on $\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2/n$ with [16, Theorem 3], which is obtained for GTF with the ℓ_1 penalty, reproduced below for convenience.

Theorem 2 (Basic Error Bound of ℓ_1 GTF Minimizer)**.** *If $\lambda = \Theta(\sigma\zeta_k\sqrt{\log r})$, then $\hat{\boldsymbol{\beta}}$, the minimizer of (2.3), satisfies*

$$\frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2}{n} = O\left(\frac{\lambda\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_1}{n} + \frac{\sigma^2 C_{\mathcal{G}}}{n}\right).$$

The above bound is comparable to our bound in the special case of setting T to an empty set, i.e. (2.10). The first term of the bound in (2.10) is upper bounded by that of Theorem 2. The non-convex regularization yields especially tighter bounds when $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star$

contains large coefficients, so that $g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star) \ll \lambda\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_1$. On the other hand, the second term of (2.10) contains $1-\mu\|\boldsymbol{\Delta}^{(k+1)}\|^2$ in the denominator, which makes it an upper bound of the second term in Theorem 2. This gap can be brought down by choosing a larger γ, which allows one to pick a smaller μ, as mentioned in Remark 1. However, as $\gamma \to \infty$, non-convex SCAD and MCP also tends to ℓ_1, which erases the improvement from using non-convex regularizers in the first term of the bound. This indicates a trade-off in the overall error bound based on γ, or the "non-convexity" of the regularizers chosen for scalar-GTF.

To sum up, despite being non-convex, we can guarantee that any stationary point of the proposed GTF estimator possesses strong statistical guarantees.

2.6.3 Error Rates for Erdős-Rényi Graphs

We next specialize Theorem 1 to the Erdős-Rényi random graphs using spectral graph theory [62]. Let $d_{\max}$ and d_0 respectively be the maximum and expected degree of the graph. It is known that for any graph it holds [16]

$$\zeta_k \leq \lambda_{\min}(\boldsymbol{\Delta}^{(2)})^{-\frac{k+1}{2}},$$

where $\lambda_{\min}(\boldsymbol{\Delta}^{(2)})$ is the smallest *non-zero* eigenvalue of the graph Laplacian matrix $\boldsymbol{\Delta}^{(2)}$. Moreover, we have $\|\boldsymbol{\Delta}^{(k+1)}\|^2 = (\lambda_{\max}(\boldsymbol{\Delta}^{(2)}))^{k+1}$, and $d_{\max}+1 \leq \lambda_{\max}(\boldsymbol{\Delta}^{(2)}) \leq 2d_{\max}$ [63]. Next, we present a simple lower bound on κ_T, which is proved in Appendix A.2.

Proposition 1 (Bound on κ_T)**.** *κ_T is bounded for any T and d as*

$$\kappa_T(\boldsymbol{\Delta}^{(k+1)}) \geq (2d_{\max})^{-\frac{k+1}{2}}.$$

For an Erdős-Rényi random graph, if $d_0 = \Omega(\log(n))$, we have $d_{\max} = O(d_0)$ almost

surely [64, Corollary 8.2] and $C_{\mathcal{G}} = 1$. Furthermore, $\lambda_{\min}(\boldsymbol{\Delta}^{(2)}) = \Omega(d_0 - \sqrt{d_0})$ [16, 62, 65], and $r = n$ for odd k and $r = O(nd_0)$ for even k. Therefore, with probability at least $1 - n^{-10}$, we have

$$\frac{\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2}{n} \lesssim \frac{\sigma^2\sqrt{\log n}}{n} + \min\left\{\frac{g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star)}{n}, \frac{\sigma^2\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_0 \log n}{n}\right\},$$

where $g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star) \lesssim \frac{\sigma\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_1\sqrt{\log n}}{d_0^{(k+1)/2}}$ by plugging in $g(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star) \leq \lambda\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star\|_1$.

These results are also applicable to d_0-regular Ramanujan graphs [65].

2.6.4 Support Recovery

An alternative yet important metric for gauging the success of the proposed GTF estimators is support recovery, which aims to localize the discontinuities in the piecewise smooth graph signals, i.e. the support set of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star$, that is

$$S_k(\boldsymbol{\beta}^\star) = \left\{t \in \{1, \cdots, r\} : (\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star)_t \neq 0\right\}.$$

In particular, for odd k, the discontinuities correspond to graph nodes; and for even k, they correspond to the edges. Let $\hat{\boldsymbol{\beta}}$ be the GTF estimate of the graph signal. The quality of the support recovery can be measured using the *graph screening distance* [41]. For any $t_1 \in S_k(\boldsymbol{\beta}^\star)$ and $t_2 \in S_k(\hat{\boldsymbol{\beta}})$, let $d_{\mathcal{G}}(t_1, t_2)$ denote the length of the shortest path between them. The distance of $S_k(\hat{\boldsymbol{\beta}})$ from $S_k(\boldsymbol{\beta}^\star)$ is then defined as

$$d_{\mathcal{G}}(S_k(\hat{\boldsymbol{\beta}})|S_k(\boldsymbol{\beta}^\star)) = \begin{cases} \max\limits_{t_1 \in S_k(\boldsymbol{\beta}^\star)} \min\limits_{t_2 \in S_k(\hat{\boldsymbol{\beta}})} d_{\mathcal{G}}(t_1, t_2), & \text{if } S_k(\boldsymbol{\beta}^\star) \neq \emptyset \\ \infty & \text{otherwise} \end{cases}.$$

Interestingly, Lin et.al. [41] showed recently that under mild assumptions, one can

translate the error bound into a support recovery guarantee. Specifically, letting R_n be the RHS of (2.8) that bounds the error $\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^\star\|_2^2/n$ in Theorem 1, we have

$$d_{\mathcal{G}}(S_k(\hat{\boldsymbol{\beta}})|S_k(\boldsymbol{\beta}^\star)) = \begin{cases} O\left(R_n H_r^{-2}\right), & k = 0 \\ O\left(R_n^{1/3} H_r^{-2/3}\right), & k = 1 \end{cases},$$

where H_r quantifies the minimum level of discontinuity, defined as the minimum absolute value of the non-zero values of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star$, i.e.

$$H_r = \min_{t \in S_k(\boldsymbol{\beta}^\star)} |(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star)_t|.$$

Consequently, this leads to support recovery guarantees of the proposed GTF estimators. Numerical experiment in Section 2.8.1 verifies the superior performance of the non-convex regularizers over the ℓ_1 regularizer for support recovery.

2.7 ADMM Algorithm and its Convergence

There are many algorithmic approaches to optimize the vector-GTF formulation in (2.7), since scalar-GTF (2.4) can be regarded as a special case with $d = 1$. In this section, we illustrate the approach adopted in this work, which is the Alternating Direction Method of Multipliers (ADMM) framework for solving separable optimization problems [66].

Via a change of variable as $\boldsymbol{Z} = \boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}$, we can transform (2.7) to

$$\min_{\boldsymbol{B} \in \mathbb{R}^{n \times d}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{B}\|_{\mathrm{F}}^2 + h(\boldsymbol{Z}; \lambda, \gamma) \quad \text{s.t.} \quad \boldsymbol{Z} = \boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}.$$

Algorithm 1 ADMM for solving (2.7)

inputs data $\boldsymbol{X}$, graph difference operator $\boldsymbol{\Delta}^{(k+1)}$, and parameters λ, γ, τ.
initialize
  $\boldsymbol{B} \leftarrow \boldsymbol{X}$ or $\boldsymbol{B}_{\texttt{init}}$ if given.
  $\boldsymbol{D} \leftarrow \boldsymbol{\Delta}^{(k+1)}$, $\boldsymbol{Z} \leftarrow \boldsymbol{DB}$, $\boldsymbol{U} \leftarrow \boldsymbol{DB} - \boldsymbol{Z}$
  $\boldsymbol{Y} \leftarrow (\boldsymbol{I} + \tau \boldsymbol{D}^\top \boldsymbol{D})^{-1}$
repeat
  for $j \leftarrow$ `1 to num_cols(`$\boldsymbol{B}$`)` **do**
    $\boldsymbol{B}_{\cdot j} \leftarrow \boldsymbol{Y}(\tau \boldsymbol{D}^\top (\boldsymbol{Z}_{\cdot j} - \boldsymbol{U}_{\cdot j}) + \boldsymbol{X}_{\cdot j})$
  end for
  for $\ell \leftarrow$ `1 to num_rows(`$\boldsymbol{DB}$`)` **do**
    $\boldsymbol{Z}_{\ell \cdot} \leftarrow \texttt{Prox}_\rho(\|\boldsymbol{D}_{\ell \cdot}\boldsymbol{B} + \boldsymbol{U}_{\ell \cdot}\|_2; \lambda/\tau)$
  end for
  $\boldsymbol{U} \leftarrow \boldsymbol{U} + \boldsymbol{DB} - \boldsymbol{Z}$
until termination
outputs signal estimate $\boldsymbol{B}$.

Its corresponding Lagrangian can be written as:

$$\mathcal{L}(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U}) = \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{B}\|_{\mathrm{F}}^2 + h(\boldsymbol{Z}; \lambda, \gamma) + \frac{\tau}{2}\|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B} - \boldsymbol{Z} + \boldsymbol{U}\|_{\mathrm{F}}^2 - \frac{\tau}{2}\|\boldsymbol{U}\|_{\mathrm{F}}^2, \quad (2.11)$$

where $\boldsymbol{U} \in \mathbb{R}^{r \times d}$ is the Lagrangian multiplier, and τ is the parameter. Alg. 1 shows the ADMM updates based on the Lagrangian in (2.11). Recall the proximal operator is defined as $\texttt{Prox}_f(\boldsymbol{v}; \alpha) = \operatorname{argmin}_{\boldsymbol{y}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{v}\|_2^2 + \alpha f(\boldsymbol{y})$ for a function $f(\cdot)$. ℓ_1, SCAD and MCP all admit closed-form solutions of `Prox`, which are simple thresholding operations [67]. Furthermore, we have the following convergence guarantee for Alg. 1, whose proof is provided in Appendix A.3. Theorem 3 implies that the output of Alg. 1 satisfies Theorem 1.

Theorem 3. *Let $\tau \geq \mu$. Then Alg. 1 converges to a stationary point of (2.7).*

In addition, we provide a detailed time complexity analysis of Alg. 1 in Table 2.2. Note that since $\boldsymbol{\Delta}$ is a sparse matrix with exactly $2m$ non-zero entries, Alg. 1 can run

much faster when $k = 0$. As a preprocessing step for each $\boldsymbol{D}$[1], we compute $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{S} \in \mathbb{R}^{n \times n}$, the eigenvectors and eigenvalues of $\boldsymbol{D}^\top \boldsymbol{D}$, exactly once. $\boldsymbol{Y} = \boldsymbol{V}(\boldsymbol{I} + \tau \boldsymbol{S})^{-1} \boldsymbol{V}^\top$ can then be initialized very efficiently for all experiments that use $\boldsymbol{D}$.

	$k \geq 1$	$k = 0$
$\boldsymbol{D}^\top \boldsymbol{D}$ eigen decomposition	$O(rn^2 + n^3)$	$O(m^2 + n^3)$
$\boldsymbol{Z}$ initialization	$O(rnd)$	$O(md)$
$\boldsymbol{Y}$ initialization	$O(n^2)$	$O(n^2)$
$\boldsymbol{B}$ update	$O(d(nr + n^2))$	$O(d(m + n^2))$
$\boldsymbol{DB}$ calculation	$O(rnd)$	$O(md)$
$\boldsymbol{Z}, \boldsymbol{U}$ update	$O(rd)$	$O(rd)$
Total after t iterations	$O(tdrn + tdn^2)$	$O(tdm + tdn^2)$

Table 2.2: Time complexity analysis of vector-GTF in Alg. 1.

2.8 Numerical Experiments

For the following experiments, we fixed $\gamma = 3.7$ for SCAD, and $\gamma = 1.4$ for MCP. The graphs we use in the following experiments satisfy Assumption 1 for this choice of γ. Unless explicitly mentioned, we tuned λ and $\frac{\tau}{\lambda}$ for each experiment using the Hyperopt toolbox [68]. To meet the convergence criteria in Theorem 3, we enforced $\tau \geq 1/\gamma$. SCAD/MCP were warm-started with the GTF estimate with ℓ_1 penalty. Python packages PyGSP [69] and NetworkX [70] were used to construct and plot graphs. The input signal SNR was calculated as $10 \log_{10}(\|\boldsymbol{B}^\star\|_{\mathrm{F}} / \sigma^2 nd)$, while the reconstructed signal SNR was calculated as $10 \log_{10}(\|\boldsymbol{B}^\star\|_{\mathrm{F}} / \|\hat{\boldsymbol{B}} - \boldsymbol{B}^\star\|_{\mathrm{F}})$, where $\hat{\boldsymbol{B}}$ was the reconstruction. Computation time was measured with MacBook Pro 2017 with an 2.9 GHz Intel Core i7 and 16GB RAM. Our code is available at `https://github.com/HarlinLee/nonconvex-GTF-public`.

[1] In this section, we are overloading the variable $\boldsymbol{D}$ by $\boldsymbol{D} := \boldsymbol{\Delta}^{(k+1)}$, since $\boldsymbol{D}$ was first introduced as the degree matrix in Section 1.5. However, this chapter does not mention the degree matrix, so instances of $\boldsymbol{D}$ in Alg. 1 and Appendix A proofs clearly refer to shorthand of $\boldsymbol{\Delta}^{(k+1)}$.

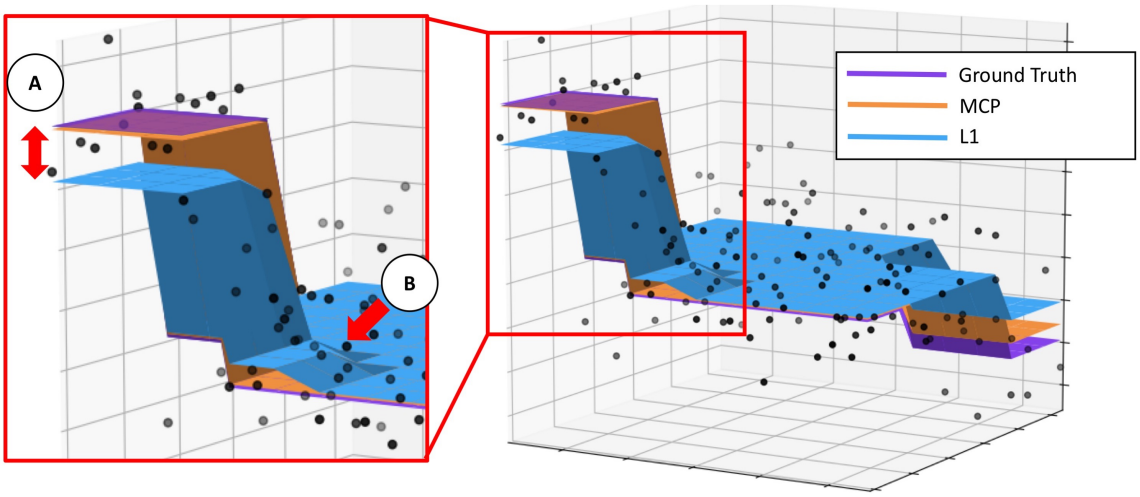


Figure 2.3: Scalar-GTF with MCP (orange) has much lower bias than scalar-GTF with ℓ_1 (blue) when estimating a piecewise constant signal over a 12×12 grid graph. See highlighted regions pointed by red arrows in A and B. The scatter points correspond to a noisy signal with 5dB SNR.

2.8.1 Denoising via GTF with Non-convex Regularizers

We first highlight via synthetic examples two important advantages that non-convex regularizers provide over the ℓ_1 penalty.

- **Bias Reduction:** We demonstrate the reduction in signal bias in Fig. 2.3 for the graph signal defined over a 12×12 2D-grid graph, using both the ℓ_1 penalty and the MCP penalty. Clearly, the MCP estimate (orange) has less bias than the ℓ_1 estimate (blue), and can recover the ground truth surface (purple) more closely.

- **Support Recovery:** We illustrate the improved support recovery performance of non-convex penalties on localizing the boundaries for a piecewise constant signal on the Minnesota road graph, shown in Fig. 2.5. Particularly, we look at how well our estimator localizes the support of $\boldsymbol{\Delta}^{(k+1)}\boldsymbol{\beta}^\star$, that is, the discontinuity of the piecewise constant graph signal by looking at how well we can classify an edge as connecting

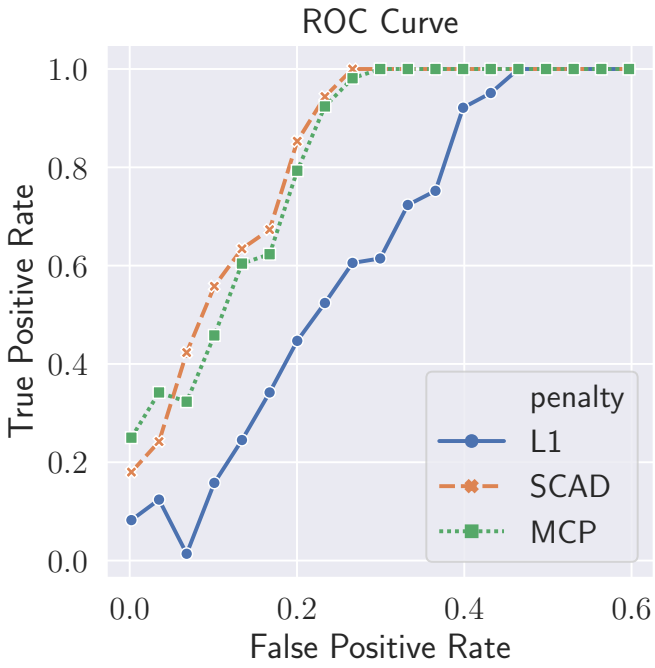


Figure 2.4: The ROC curve for classifying whether an edge lies on a boundary for the Minnesota road graph signal shown in Fig. 2.5. Non-convex penalty SCAD/MCP shows improved support recovery performance compared to ℓ_1. The input SNR of the noisy piecewise constant signal is 7.8dB.

two nodes in the same piece or being a cut edge across two pieces. By sweeping the regularization parameter λ, we obtain the ROC curve in Fig. 2.4, i.e. the true positive rate versus the false positive rate of classifying a cut edge correctly, and see that scalar-GTF with MCP and SCAD consistently outperforms the scalar-GTF with ℓ_1 penalty.

Then, we more rigorously compare the performance of GTF using non-convex regularizers such as SCAD and MCP with that using the ℓ_1 norm. For the ground truth signal $\boldsymbol{\beta}^\star$, we construct a piecewise constant signal on a 20×20 2D-grid graph and the Minnesota road graph [69] as shown in the left panel of Fig. 2.5, and add different levels of

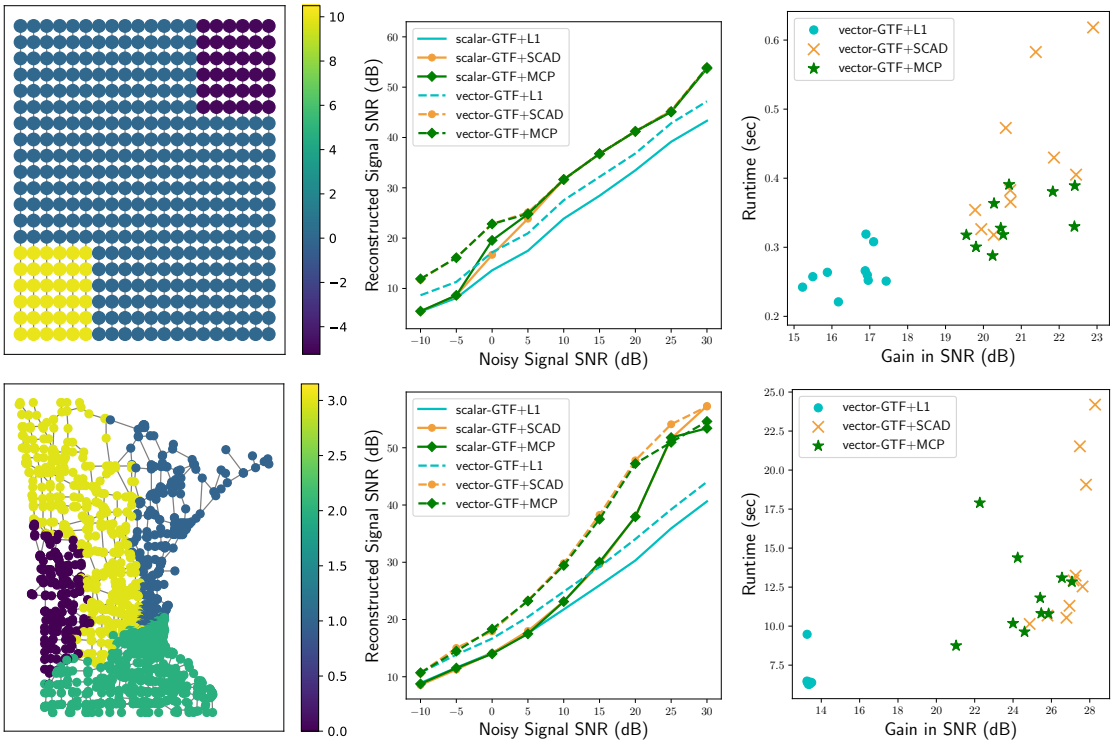


Figure 2.5: The left panel shows the ground truth piecewise constant signals on 20×20 2D-grid graph (top), and Minnesota road graph (bottom). The middle panel shows their corresponding plots of input signal SNR versus reconstructed signal SNR, averaged over 10 and 20 realizations, respectively. The non-convex penalty SCAD/MCP consistently outperforms convex ℓ_1, and vector-GTF denoises better than scalar-GTF. Finally, the right panel plots the computation time against gain in SNR from denoising via vector-GTF. 10 trials were performed for each regularizer, where the input signal SNR was fixed at 20dB.

noise following (2.2). We recover the signal by scalar-GTF with Alg. 1, and plot the SNR of the reconstructed signal versus the SNR of the input signal *in solid lines* in the middle panel of Fig. 2.5, averaged over 10 and 20 realizations, respectively. SCAD/MCP consistently outperforms ℓ_1 in denoising graph signals defined over both regular and irregular structures.

2.8.2 Denoising Vector-valued Signals via GTF

We compare the performance of vector-GTF in (2.7) with (2.6), which applies scalar-GTF to each column of the vector-valued graph signal. The convex ℓ_1 norm and the non-convex SCAD and MCP are employed. We reuse the same ground truth graph signals over the 2D-grid graph and the Minnesota road graph constructed in Section 2.8.1 in Fig. 2.5. d independent noisy realizations of the graph signal are concatenated to construct a noisy vector-valued graph signal with dimension $d = 10$ on the 2D-grid graph and with $d = 20$ on the Minnesota road graph. We recover the vector-valued signal by minimizing vector-GTF (2.7) with Alg. 1.

The middle panel of Fig. 2.5 plots the average SNR of the reconstructed signal versus the average SNR of the input signal *in dotted lines*. We emphasize that the performance of (2.6) is the same as applying scalar-GTF to each realization, which is shown in the middle panel of Fig. 2.5 in solid lines. As before, SCAD/MCP consistently outperforms ℓ_1 in denoising signals over both regular and irregular graphs. Furthermore, as expected, due to the sharing of information across realizations, vector-GTF consistently outperforms scalar-GTF, especially in the low SNR regime.

The right panel of Fig. 2.5 plots the computation time versus the gain in SNR from denoising via vector-GTF. 10 trials are performed for each regularizer with the input signal SNR fixed at 20dB. Parameter tuning and eigen decomposition of $\mathbf{\Delta}^{(2)}$ are preprocessing

steps, and hence they are not included in the time measurement; but for reference, the eigen decomposition took 0.025 and 2.5 seconds for 2D-grid and Minnesota graphs, respectively. Since GTF with non-convex regularizers are warm-started by the ℓ_1 estimate, the runtime for ℓ_1 GTF is added to the SCAD/MCP runtime. Overall, running vector-GTF with SCAD/MCP after once with ℓ_1 takes more time, but with large benefits in the denoising performance. Even with the additional computation time, vector-GTF runs reasonably fast; computation takes less than 25 seconds with the Minnesota road network, where $n = 2642$ and $m = 3304$.

	Average	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**	**#8**
Input SNR (dB)	8.7	-14	0	0	3.5	5.8	12	29	34
Vector-GTF + ℓ_1	29	**10**	**20**	**23**	**26**	**36**	**37**	39	38
Scalar-GTF + ℓ_1	21	0	11	13	16	18	26	41	45
Vector-GTF + SCAD	**32**	**10**	**20**	22	25	**36**	35	**49**	**61**
Scalar-GTF + SCAD	29	0	15	17	25	35	34	47	60
Vector-GTF + MCP	**32**	**10**	**20**	22	25	**36**	35	**49**	**61**
Scalar-GTF + MCP	29	0	15	22	24	30	33	**49**	60

Table 2.3: Noisy input and reconstructed signal SNRs for eight measurements of varying input SNRs, rounded to two significant figures. Highest reconstructed signal SNR for each measurement is in **bold**. Vector-GTF outperforms scalar-GTF, and SCAD/MCP achieves better SNR than ℓ_1 on average. In low SNR settings, information about the boundaries of the graph signal is borrowed from higher SNR signals to improve the estimation.

We further investigate the benefit of sharing information across measurements or realizations in the following experiment, using the same ground truth signal on the 2D-grid graph. We stack eight noisy realizations of this same piecewise constant signal to build a vector-valued signal. We construct these noisy measurements by scaling each one of them differently and randomly such that each will have SNR $\sim \mathcal{U}[-10, 30]$dB under (2.2). This has the effect of rendering some measurements more *informative* than others, and potentially allowing vector-GTF to reap the benefits of sharing information across

measurements. We recover the 8-dimensional graph signal via Alg. 1 using ℓ_1, SCAD, and MCP regularizers, and in Table 2.3, report the input signal and reconstructed signal SNRs for each measurement in addition to the average SNRs. λ is fixed at $0.5\sigma^2$.

First of all, notice that as before, using SCAD/MCP generally achieves results with higher SNR than using ℓ_1, and that on average, minimizing (2.7) outperforms minimizing (2.6). The effect of sharing information across measurements is most apparent in low SNR settings, when information about the boundaries of the graph signal can be borrowed from higher SNR signals to improve the estimation. On the other hand, sharing information with noisier signals does not help denoising signals with high input SNR. However, it is worth noting that, unlike ℓ_1, SCAD/MCP does not see decrease in its performance in the high SNR settings.

2.8.3 Denoising Trends in Real-world Traffic Data

To further illustrate graph trend filtering on a real-world dataset, we consider the road network of Manhattan where the nodes correspond to junctions [71]. We map the pickups and dropoffs of the NYC taxi trip dataset [72, 73] to the nearest road junctions, and define the total count at that junction to be the signal value on the corresponding graph node. The signal of interest, plotted on the top left panel of Fig. 2.6, is the difference between the *event* graph signal on the day of NYC Gay Pride parade, 12-2pm on June 26, 2011, and the *seasonal average* graph signal at the same time during the 8 nearest Sundays. During the event, no pickups and dropoffs could occur in the areas shown in the top right panel of Fig. 2.6 [74]. We denoise the signal via GTF using both ℓ_1 and MCP, where we chose λ such that $\|\boldsymbol{\Delta}\hat{\boldsymbol{\beta}}\|_0 \approx 200$. Once again, we observe the GTF estimate with MCP produces sharper traces around the parade route, indicating better capabilities of event detection and localization.

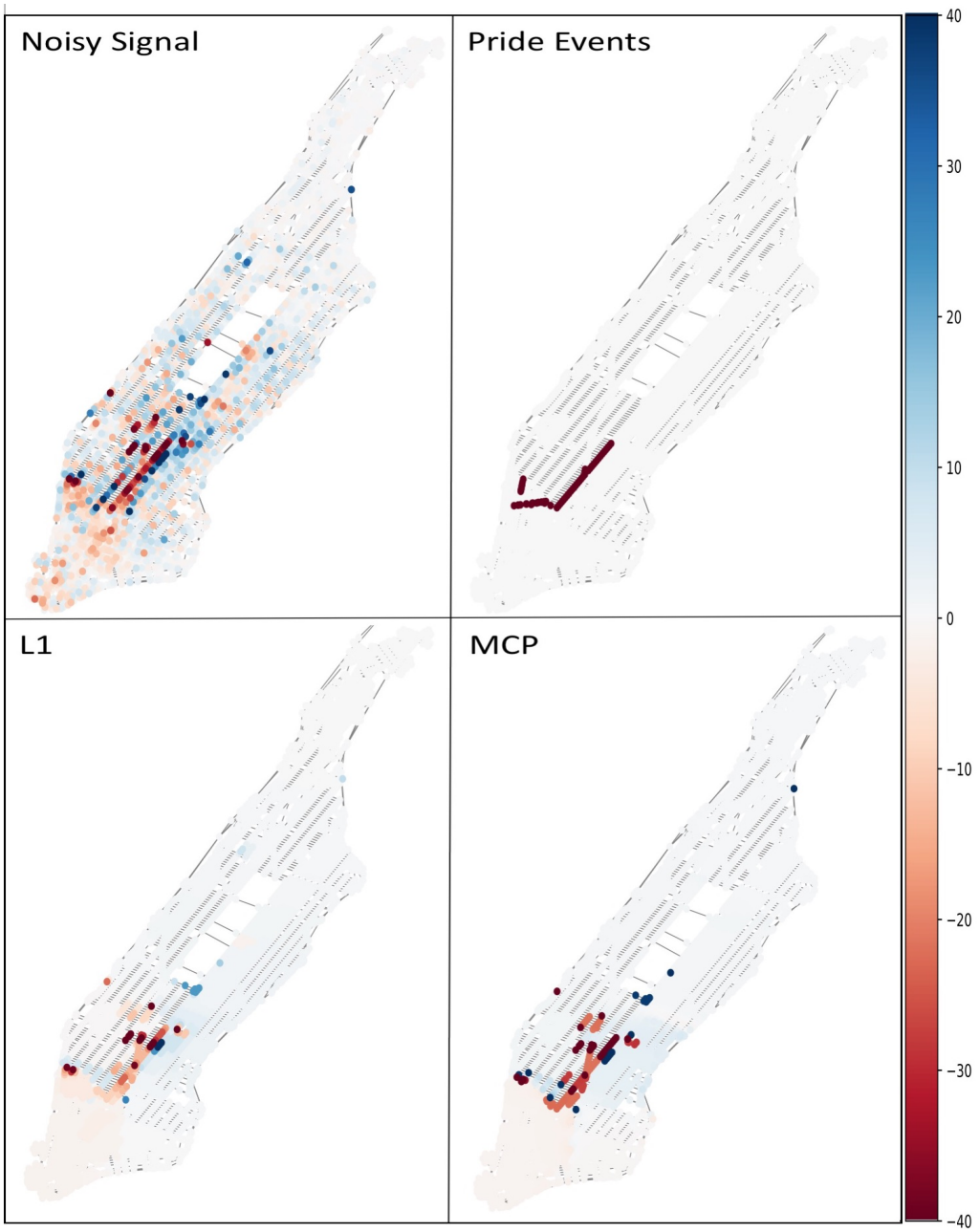


Figure 2.6: Top left: the noisy signal on the Manhattan road network is the change in the taxi pickup and dropoff count during the 2011 NYC Gay Pride. Top right: areas of Pride events, where the traffic was blocked off. Bottom: the GTF estimates using ℓ_1 and MCP. The GTF estimate with MCP better detects and localizes the event, compared to the one using ℓ_1 penalty.

2.8.4 Semi-supervised Classification

		Heart	Wine quality	Wine	Iris	Breast	Car
# of samples (n)		303	1599	178	150	569	1728
# of classes (K)		2	6	3	3	2	4
$k=0$	ℓ_1	0.148	0.346	0.038	0.036	0.042	0.172
	SCAD	0.148	*0.353*	0.038	0.033	0.042	**0.149**
	p-value	1.	*0.06*	1.	0.27	1.	**0.06**
	MCP	0.144	0.351	0.037	0.035	0.040	**0.148**
	p-value	0.23	0.18	0.34	0.34	0.35	**0.05**
$k=1$	ℓ_1	0.143	0.351	0.034	0.039	0.035	0.104
	SCAD	0.144	0.350	0.034	0.039	0.035	0.104
	p-value	0.30	0.43	0.34	1.	0.71	0.66
	MCP	*0.146*	0.350	0.034	0.039	**0.034**	0.103
	p-value	*0.05*	0.44	0.34	1.	**0.02**	0.23

Table 2.4: Misclassification rates averaged over 10 trials, with p-values from running sampled t-tests between SCAD/MCP misclassification rates and the corresponding rates using ℓ_1. Cases where non-convex penalties perform better than ℓ_1 with p-value below 0.1 are highlighted in **bold**, and where they perform worse are in *italic*.

Graph-based learning provides a flexible and attractive way to model data in semi-supervised classification problems when vast amounts of unlabeled data are available compared to labeled data, and labels are expensive to acquire [14, 15, 58]. One can construct a nearest-neighbor graph based on the similarities between each pair of samples, and hope to propagate the label information from labeled samples to unlabeled ones. We move beyond our original problem in (2.4) to a K-class classification problem in a semi-supervised learning setting, where for a given dataset with n samples, we observe a subset of the one-hot encoded class labels, $\boldsymbol{X} \in \mathbb{R}^{n\times K}$, such that $X_{ij} = 1$ if ith sample has been observed to be in jth class, and $X_{ij} = 0$ otherwise. A diagonal indicator matrix $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ denotes samples whose class labels have been observed. Then, we can define the modified absorption

problem [15, 16, 58] using a variation of GTF to estimate the unknown class probabilities $\boldsymbol{B} \in \mathbb{R}^{n \times K}$:

$$\widetilde{\boldsymbol{B}} = \underset{\boldsymbol{B} \in \mathbb{R}^{n \times K}}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{M}(\boldsymbol{X} - \boldsymbol{B})\|_{\mathrm{F}}^2 + \sum_{j=1}^{K} g(\boldsymbol{\Delta}^{(k+1)} \boldsymbol{B}_{\cdot j}; \lambda, \gamma) + \epsilon \|\boldsymbol{R} - \boldsymbol{B}\|_{\mathrm{F}}^2,$$

where $\boldsymbol{R} \in \mathbb{R}^{n \times K}$ (set to be uniform in the experiment) is a fixed prior belief, and $\epsilon > 0$ determines how much emphasis to be given to the prior belief. The labels $\widetilde{\boldsymbol{X}}$ can be estimated using $\widetilde{\boldsymbol{B}}$ such that $\widetilde{X}_{ij} = 1$ if and only if $j = \arg\max_{1 \le \ell \le K} \widetilde{B}_{i\ell}$, and otherwise $\widetilde{X}_{ij} = 0$. Note that this can be completely separated into K scalar-GTF problems, one corresponding to each class.

We applied the algorithm in (2.8.4) to 6 popular UCI classification datasets [75] with $\epsilon = 0.01$. For each dataset, we normalized each feature to have zero mean and unit variance, and constructed a 5-nearest-neighbor graph of the samples based on the Euclidean distance between their features, with edge weights from the Gaussian radial basis kernel. We observed the labels of 20% of samples in each class randomly. Table 2.4 shows the misclassification rates averaged over 10 repetitions, which demonstrates that the performance using non-convex penalties such as SCAD/MCP are at least competitive with, and often better than, those with the ℓ_1 penalty.

2.9 Conclusions

We presented a framework for denoising piecewise smooth signals on graphs that generalizes the graph trend filtering framework to handle vector-valued signals using a family of non-convex regularizers. We provided theoretical guarantees on the error rates of our framework, and derived a general ADMM-based algorithm to solve this generalized graph trend filtering problem. Furthermore, we demonstrated the superior performance of these

non-convex regularizers in terms of reconstruction error, bias reduction, and support recovery on both synthetic and real-world data. In particular, its performance on filtering trend in traffic and semi-supervised classification is investigated.

Chapter 3

Matrix Factorization in Remote Sensing

3.1 Summary

Remote sensing data from hyperspectral cameras suffer from limited spatial resolution, in which a single pixel of a hyperspectral image may contain information from several materials in the field of view. Blind hyperspectral image unmixing identifies the pure spectra of individual materials (i.e., endmembers) and their proportions (i.e., abundances) at each pixel. We propose a novel blind hyperspectral unmixing model based on the graph total variation (gTV) regularization, which can be solved efficiently by the alternating direction method of multipliers (ADMM). To further alleviate the computational cost, we apply the Nyström method to approximate a fully-connected graph from a small subset of sampled points. Furthermore, we adopt the Merriman-Bence-Osher (MBO) scheme to solve the gTV-involved subproblem in ADMM by decomposing a grayscale image into a bit-wise form. A variety of numerical experiments on synthetic and real hyperspectral images are conducted, showcasing the potential of the proposed method in terms of identification accuracy and computational efficiency.

3.2 Introduction

Hyperspectral imaging is an important and useful tool to acquire high resolution data in the electromagnetic spectrum with many applications in remote sensing, including surveillance, agriculture, environmental monitoring, and astronomy. With hundreds to thousands of spectral bands, a hyperspectral image provides a detailed description of a scene. However, due to limited spatial resolution of imaging sensors, the acquired hyperspectral data at each pixel represents a collection of material signatures in the field of view of each pixel. The signature corresponding to one pure material is called an *endmember* in hyperspectral data analysis [76]. Given the endmembers of all materials present in the scene, hyperspectral unmixing aims to estimate the proportions of constituent endmembers at each single pixel, called the *abundance map*. If the spectral information of endmembers is unavailable, then the problem becomes a blind hyperspectral unmixing problem that requires simultaneously identifying the endmembers and estimating the abundance map. There are a large number of hyperspectral mixing and unmixing methods [77, 78], including linear and non-linear models, depending on assumptions about the interaction of the light with the observed scene.

We focus on the linear mixing model in this project. Specifically, by assuming that each light ray interacts with only one endmember in the field of view before reaching the sensor, we model the spectrum at each pixel as a linear combination of all endmembers. Due to the physical interpretation of the hyperspectral mixing model, it is also reasonable to assume that each element of endmembers and abundances is non-negative. Another commonly used constraint is that abundances from all the endmembers at each pixel sum up to one, which implies that all abundance vectors belong to the probability simplex, determined by the standard unit vectors in a Euclidean space. One can remove the sum-

to-one constraint for physically motivated reasons, e.g., when illumination conditions or the topography of the scene change locally in the image [79], but we adopt the sum-to-one constraint for interpretability of the abundances.

Non-negative matrix factorization (NMF) [80], which decomposes a given matrix into a product of two matrices with non-negative entries, is widely used in blind hyperspectral unmixing [81, 82, 83]. Suppose the given hyperspectral image $\boldsymbol{X}$ is of size $w \times n$, where w is the number of spectral bands and n is the number of spatial pixels. One aims to write $\boldsymbol{X}$ as a product of two non-negative matrices $\boldsymbol{S} \in \mathbb{R}^{w\times k}$ and $\boldsymbol{A} \in \mathbb{R}^{k\times n}$ with k being the total number of the endmembers. Note that the rank of the matrix $\boldsymbol{SA}$ is at most k, and k is usually much smaller than w and n. Then the hyperspectral unmixing problem can be formulated as a non-negative least squares problem,

$$\min_{\substack{\boldsymbol{S}\in\Omega_{w\times k}\\ \boldsymbol{A}\in\Omega_{k\times n}}} \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{SA}\|_{\mathrm{F}}^2, \tag{3.1}$$

where $\Omega_{r\times c}$ denotes the set of all non-negative real matrices of size $r \times c$, i.e.,

$$\Omega_{r\times c} := \{\boldsymbol{X} \in \mathbb{R}^{r\times c} \,|\, X_{ij} \geq 0,\ i = 1,\ldots,r,\ \ j = 1,\ldots,c\}. \tag{3.2}$$

However, non-convexity of the objective function in (3.1) may lead to multiple local minima. To address this issue, various regularization techniques have been developed to enforce some desirable properties on the endmembers or abundance matrices. For example, methods based on the spatial sparsity of abundances include the use of the ℓ_0 norm [84], the ℓ_1 norm [85], the ℓ_2 norm in fully constrained least squares unmixing (FCLSU) [86], the $\ell_{1/2}$ norm [87], and the mixed $\ell_{p,q}$ norm for group sparsity [88].

In this work, we propose an efficient framework for blind hyperspectral unmixing based on an approximation of graph total variation (gTV) to exploit the similarity of

spectral information at different pixels, and preserve sharp edges of the abundance map. By treating the spectral vector at each pixel as a vertex, the given hyperspectral data can be modeled as a graph, whose adjacency matrix is determined by the pairwise similarity between any two vertices. Instead of using the incidence matrix to define the discrete graph derivative operator and thereby gTV [3, 89, 90, 91], we approximate gTV by the graph Laplacian. This approach is inspired by a theoretical result in [92]: the TV semi-norm of a binary function defined on a graph is well-approximated by the graph Ginzburg-Landau (GL) functional involving the graph Laplacian and a double-well potential. In order to relax the restriction on binary data, we adopt a bitwise decomposition [93] to deal with grayscale images. Specifically, we decompose the input data into eight bits, solve the optimization problem at each bit channel, and aggregate all bits into grayscale values.

Our framework incorporates several techniques to increase the computational efficiency. To avoid a direct calculation of the graph Laplacian, we adopt the Nyström method [94] in graph clustering to approximate the eigenvalues and eigenvectors of the graph Laplacian. The Nyström method is a low-rank approximation of the weight matrix that does not require the computation of all pairwise comparisons between feature vectors. Rather, it uses random sampling to construct a low rank approximation that is roughly $O(n)$ for the number of feature vectors rather than computing the full matrix which is $O(n^2)$. This is a reasonable assumption in cases where the image is thought to be representable by a relatively small number of features, as would be the case with a modest number of endmembers. This approximation significantly reduces the computational costs in both time and storage, which makes our approach scalable to high-dimensional data. Moreover, we design an efficient numerical algorithm to solve the proposed model via ADMM [66, 95, 96]. In particular, the gTV-related subproblem can be solved efficiently by the Merriman-Bence-Osher (MBO) scheme [97, 98] at each bit channel. We can readily

incorporate an accelerated version [99] of the MBO scheme and the Nyström method into the proposed framework. To demonstrate the effectiveness of these approximations, we conduct extensive experiments on various synthetic and real hyperspectral datasets, showing the great potential of the proposed method in terms of accuracy and computational efficiency.

The main contributions of this work are three-fold:

- We propose a novel data-driven type of graph regularization, i.e., gTV based on the similarity of spectral information, imposed on the abundance map. To the best of our knowledge, this is the first time that the graph total variation regularization has been applied to solve a hyperspectral unmixing problem.
- We apply the Nyström method to efficiently approximate eigenvalues and eigenvectors of a normalized graph Laplacian, which significantly improves the scalability of our approach.
- We present an effective graph-based framework that integrates the Nyström method and the MBO scheme into blind hyperspectral unmixing. We also provide a thorough discussion of computational complexity and parameter selection of the proposed algorithm.

Table 3.1 summarizes some key notations used in this chapter for convenience.

3.3 Related Work

Due to the success of the total variation (TV) [100] in the image processing community, the TV regularization has been applied to hyperspectral unmixing to preserve the piecewise constant structure of the abundance map for each material. For example, sparse unmixing via variable splitting augmented Lagrangian and total variation (SUnSAL-TV) [101]

Symbol	Description	Dimension
n	number of spatial pixels	1
w	number of spectral bands	1
k	number of endmembers	1
$\boldsymbol{X}$	hyperspectral image data	$w \times n$
$\boldsymbol{S}$	endmember spectra	$w \times k$
$\boldsymbol{A}$	abundance map	$k \times n$
$\boldsymbol{a}_i$	ith column of $\boldsymbol{A}$	$k \times 1$
$\widehat{\boldsymbol{a}}_i$	$\boldsymbol{a}_i$ normalized by node degree	$k \times 1$
$\boldsymbol{L}_s$	normalized graph Laplacian	$n \times n$

Table 3.1: Key notations used in Chapter 3.

involves a two-dimensional TV regularization. Other TV-based variants include TV with ℓ_1 [102], TV with sparse NMF [103], TV with non-negative tensor factorization [104], and an improved collaborative NMF with TV (ICoNMF-TV) [105] that combines robust collaborative NMF (R-CoNMF) [106] and TV. A recent work referred to as NMF-QMV [107] considers TV as a quadratic regularization promoting minimum volume in the NMF framework. An extension of TV to non-local spatial operators [108, 109] has led to non-local TV being considered for the blind hyperspectral unmixing problem [110, 111].

TV has also been extended from vectors in Euclidean space to signals defined on a graph. For example, gTV [91] is a special case of the p-Dirichlet form [3, 89] in graph signal processing. Some graph regularization techniques for hyperspectral imaging include graph NMF (GNMF) [112], structured sparse-regularized NMF (SS-NMF) [113], graph-regularized $\ell_{1/2}$-NMF (GLNMF) [114], and graph-regularized multilinear mixing model (G-MLM) based on superpixels [115]. By considering only the sparsity of spatial gradients, the spatial TV regularization has a tendency to oversmooth the abundance map [116]. On the contrary, the proposed gTV regularization considers the similarity of spectral information at different pixels and hence it can preserve fine spatial features in the abundance map.

Most of these graph-based approaches suffer from intensive computation, especially

when computing the pairwise similarity between all pixels. As one of the methods to reduce the computational cost, the Nyström method [94] generates a low-rank approximation of the graph Laplacian, which can be incorporated into unmixing.

3.4 Proposed: Graph Total Variation Regularization for Blind Hyperspectral Unmixing

Let $\boldsymbol{X} \in \mathbb{R}^{w \times n}$ be a hyperspectral image, where w is the number of spectral bands and n is the number of pixels in the image. We denote the spectral signature of pure materials, called *endmembers*, as $\{\boldsymbol{s}_j\}_{j=1}^k$ with k being the number of endmembers. Assume that the spectral signature at each pixel, namely each column of $\boldsymbol{X}$, follows the standard linear mixing model, i.e.,

$$\boldsymbol{x}_i = \sum_{j=1}^{k} a_{ji} \boldsymbol{s}_j, \quad i = 1, \ldots, n, \tag{3.3}$$

where a_{ji} is the proportion of the jth material at the ith pixel. By concatenating all spectral signatures $\boldsymbol{s}_j$'s, we obtain a matrix $\boldsymbol{S} \in \mathbb{R}^{w \times k}$, which is called the *mixing matrix*. Similarly, by assembling all weights a_{ji}'s, we obtain a matrix $\boldsymbol{A} \in \mathbb{R}^{k \times n}$, which is called the *abundance map*. Thus we can rewrite (3.3) as $\boldsymbol{X} = \boldsymbol{S}\boldsymbol{A}$. Different from [117], our method does not require the presence of pure pixels, rather just to assume the linear unmixing model (3.3).

By taking the noise into consideration, the blind unmixing problem is to estimate both $\boldsymbol{S}$ and $\boldsymbol{A}$ simultaneously from the noisy hyperspectral data $\boldsymbol{X}$, i.e.,

$$\boldsymbol{X} = \boldsymbol{S}\boldsymbol{A} + \boldsymbol{E},$$

where $\boldsymbol{E} \in \mathbb{R}^{w \times n}$ is an additive noise term typically assumed to be Gaussian. This is

a highly ill-posed problem, and hence additional assumptions and regularizations are required. First, due to the physical interpretation of (3.3), both $\boldsymbol{S}$ and $\boldsymbol{A}$ are assumed to be non-negative matrices, i.e., $\boldsymbol{S} \in \Omega_{w\times k}$ and $\boldsymbol{A} \in \Omega_{k\times n}$ with Ω defined in (3.2). In addition, since each element of $\boldsymbol{A}$ is the proportion of one of the pure materials in a single pixel, it is natural to impose the sum-to-one assumption, i.e., $\mathbf{1}_k^\top \boldsymbol{A} = \mathbf{1}_n^\top$, where $\mathbf{1}_m$ denotes the all-one (column) vector of length m. The sum-to-one constraint on the abundance map is commonly used in hyperspectral unmixing [77]; it implicitly enforces sparsity because it is related to the ℓ_1 norm. We use the above two assumptions as constraints to refine the solution space.

In order to apply graph regularization, we use the given hyperspectral data $\boldsymbol{X}$ to generate a weighted graph by assuming that spectral signatures and abundance maps share the same spatial smoothness. In particular, we adopt the cosine similarity as the distance function for hyperspectral data in (3.6), which is physically motivated by the fact that illumination effects change the scaling of spectra but not their overall shape in the spectral domain [98, 99, 118].

In our previous work [7], we considered a graph Laplacian regularization for hyperspectral unmixing, i.e.,

$$J_{H_1}(\boldsymbol{A}) = \frac{1}{2}\sum_{i,j=1}^{n} \|\widehat{\boldsymbol{a}}_i - \widehat{\boldsymbol{a}}_j\|_2^2 W_{ij}, \tag{3.4}$$

where $\boldsymbol{a}_i$ is the i-th column of $\boldsymbol{A}$, D_{ii} is the degree of node i, and $\widehat{\boldsymbol{a}}_i = \boldsymbol{a}_i/\sqrt{D_{ii}}$. However, the graph Laplacian regularization usually causes oversmoothing due to the presence of ℓ_2 norm in (3.4). To mitigate the oversmoothing artifacts, we propose graph total variation (gTV) regularization on the abundance map, i.e.,

$$J_{TV}(\boldsymbol{A}) = \frac{1}{2}\sum_{i,j=1}^{n} \|\widehat{\boldsymbol{a}}_i - \widehat{\boldsymbol{a}}_j\|_1 W_{ij}.$$

Minimizing J_{TV} can preserve edges of the abundance map for each material in a non-local fashion. The proposed gTV-regularized model for blind hyperspectral unmixing can be formulated as

$$\min_{\substack{\boldsymbol{S}\in\Omega_{w\times k}\\ \boldsymbol{A}\in\Omega_{k\times n},\mathbf{1}_k^\top\boldsymbol{A}=\mathbf{1}_n^\top}} \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2+\lambda J_{TV}(\boldsymbol{A}), \tag{3.5}$$

where λ is a positive tuning parameter.

3.5 Preliminaries

In this section, we provide preliminary knowledge for a set of building blocks that are used in this work, including the graph construction, the Nyström method for efficiently approximating the similarity weight matrix, and the GL functional with a fast solver to find its minimizer via MBO.

3.5.1 Graph Construction and Nyström Method

Recall the definitions of adjacency matrix $\boldsymbol{W}$, degree matrix $\boldsymbol{D}$, graph Laplacian matrix $\boldsymbol{L}$, and *(symmetric) normalized graph Laplacian* $\boldsymbol{L}_s$ from Section 1.5.

Similarity graphs are an important mathematical tool to describe directed/undirected pairwise connections between objects. Consider a collection of data points $\{\boldsymbol{x}_i\}_{i=1}^n \subseteq \mathbb{R}^w$. One simple way to construct a graph is to treat each point as a vertex of the graph. Then the adjacency matrix $\boldsymbol{W}\in\mathbb{R}^{n\times n}$ is defined by

$$W_{ij}=e^{-d(\boldsymbol{x}_i,\boldsymbol{x}_j)^2/\sigma},\quad i,j=1,\dots,n, \tag{3.6}$$

where $d(\boldsymbol{x}_i,\boldsymbol{x}_j)$ is the distance between the two vertices $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and $\sigma>0$ controls how similar they are. There are two distance metrics widely used in graph-based applications:

1. Euclidean distance: $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$;

2. Cosine similarity: $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - \frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle}{\|\boldsymbol{x}_i\|_2 \|\boldsymbol{x}_j\|_2}$.

Computing and storing pairwise similarities of a fully-connected graph is usually a bottleneck of many graph-based algorithms. In order to reduce the time/space complexity, we apply the Nyström method [94] to approximate the eigenvalues and eigenvectors of $\boldsymbol{W}$ by using only p sampled data points with $p \ll n$. Up to permutations, the similarity matrix $\boldsymbol{W}$ can be expressed in a block-matrix form,

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_{11} & \boldsymbol{W}_{12} \\ \boldsymbol{W}_{21} & \boldsymbol{W}_{22} \end{bmatrix},$$

where $\boldsymbol{W}_{11} \in \mathbb{R}^{p \times p}$ is the similarity matrix of the sampled data points, $\boldsymbol{W}_{12} = \boldsymbol{W}_{21}^\top$ is the one of the sampled points and the unsampled points, and $\boldsymbol{W}_{22}$ is the one of the unsampled points. Assume that the symmetric matrix $\boldsymbol{W}_{11}$ has the eigen decomposition $\boldsymbol{W}_{11} = \boldsymbol{U}\widetilde{\boldsymbol{\Lambda}}\boldsymbol{U}^\top$, where $\boldsymbol{U}$ has orthonormal eigenvectors as columns and $\widetilde{\boldsymbol{\Lambda}}$ is a diagonal matrix whose diagonal entries are eigenvalues of $\boldsymbol{W}_{11}$. The Nyström extension gives an approximation of $\boldsymbol{W}$ by using $\boldsymbol{U}$ and $\widetilde{\boldsymbol{\Lambda}}$ as follows,

$$\boldsymbol{W} \approx \widetilde{\boldsymbol{U}}\widetilde{\boldsymbol{\Lambda}}\widetilde{\boldsymbol{U}}^\top, \quad \text{where} \quad \widetilde{\boldsymbol{U}} = \begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{W}_{21}\boldsymbol{U}\widetilde{\boldsymbol{\Lambda}}^{-1} \end{bmatrix}. \tag{3.7}$$

Note that the columns of $\widetilde{\boldsymbol{U}}$ require further orthogonalization. See [94, 99] for more details.

In this work, we apply the Nyström method to calculate the weight matrix for the sampled data and then use the approximated eigen decomposition (3.7) to approximate

the normalized graph Laplacian, i.e.,

$$\boldsymbol{L}_s \approx \boldsymbol{D}^{-\frac{1}{2}}\widetilde{\boldsymbol{U}}(\boldsymbol{I}-\widetilde{\boldsymbol{\Lambda}})\widetilde{\boldsymbol{U}}^\top \boldsymbol{D}^{-\frac{1}{2}} := \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top, \tag{3.8}$$

where $\boldsymbol{V} = \boldsymbol{D}^{-\frac{1}{2}}\widetilde{\boldsymbol{U}} \in \mathbb{R}^{n\times p}$ and $\boldsymbol{\Lambda} = \boldsymbol{I} - \widetilde{\boldsymbol{\Lambda}} \in \mathbb{R}^{p\times p}$. This way, computation of pairwise similarities is significantly reduced from the whole dataset to a small portion. We chose to use $\boldsymbol{L}_s$ over $\boldsymbol{L}$ due to its outstanding performance in the graph-based data classification [98, 119]. Using $\boldsymbol{L}_s$, we can rewrite the graph Laplacian regularizer (3.4) as

$$J_{H_1}(A) = \frac{1}{2}\sum_{i,j=1}^{n} \|\widehat{\boldsymbol{a}}_i - \widehat{\boldsymbol{a}}_j\|_2^2 W_{ij} = \frac{1}{2}\langle \boldsymbol{A}^\top, \boldsymbol{L}_s\boldsymbol{A}^\top\rangle.$$

3.5.2 Ginzburg-Langdau Functional and MBO Scheme

The classic Ginzburg-Landau (GL) energy [119, 120] for diffuse interface models is

$$\frac{\epsilon}{2}\int_\Omega |\nabla u|^2 dx + \frac{1}{\epsilon}\int_\Omega \Phi(u)dx,$$

where $\Phi(u) := \frac{1}{4}u^2(u-1)^2$ is a double-well potential to enforce u to take binary values of $\{0,1\}$ on a domain Ω. The term "diffuse interface" refers to a smooth transition between two phases of u, where the smoothness is modeled by the H_1-semi norm and the scale of the transition is controlled by the parameter $\epsilon > 0$. It is proven in [121] that the GL functional Γ-converges to the TV semi-norm, i.e., as $\epsilon \to 0$,

$$\frac{\epsilon}{2}\int_\Omega |\nabla u|^2 dx + \frac{1}{\epsilon}\int_\Omega \Phi(u)dx \to C\int_\Omega \|\nabla u\| dx,$$

for some constant $C > 0$.

In a series of works including [98, 118, 122, 123, 124], the GL functional has been

extended to graphs, defined as

$$GL(\boldsymbol{u}) = \epsilon\langle \boldsymbol{u}, \boldsymbol{L}\boldsymbol{u}\rangle + \frac{1}{\epsilon}\Phi(\boldsymbol{u}), \tag{3.9}$$

where $\boldsymbol{u} = [u_1, \ldots, u_n]^\top \in \mathbb{R}^n$ is a signal defined on a graph $\mathcal{G}$ with u_i being the state of vertex i and $\boldsymbol{L}$ is the graph Laplacian of $\mathcal{G}$ or its variant. Here $\Phi(\boldsymbol{u}) = \sum_{i=1}^n \Phi(u_i)$, which can be extended to the matrix case, i.e., $\Phi(\boldsymbol{U}) = \sum_{i,j} \Phi(U_{ij})$ for any matrix $\boldsymbol{U}$. Thanks to the double-well potential, the GL functional has been successfully applied to binary data classification [98] and multiclass classification [99, 122]. We employ the binary model here.

By adding a fidelity term to the GL energy, one obtains the following minimization problem

$$E(\boldsymbol{u}) = GL(\boldsymbol{u}) + \lambda F(\boldsymbol{u}),$$

where $F(\boldsymbol{u})$ is a differentiable functional that fits the unknown variable $\boldsymbol{u}$ to the given data $\boldsymbol{x}$, e.g., $F(\boldsymbol{u}, \boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{u}\|_2^2$. The parameter $\lambda > 0$ balances the contributions between the GL regularization term and the data fidelity term. When $\boldsymbol{u}$ is binary, the energy E can be efficiently minimized via the MBO scheme [97, 98]. In particular, the MBO scheme alternates a gradient descent step that minimizes $\langle \boldsymbol{u}, \boldsymbol{L}\boldsymbol{u}\rangle + \lambda F(\boldsymbol{u})$ and a hard thresholding that minimizes the double-well potential term. More precisely, the updated solution $\boldsymbol{u}^{t+1}$ from the tth iteration is given by

$$\begin{cases} \boldsymbol{u}^{t+1/2} = \boldsymbol{u}^t - dt\big(\boldsymbol{L}\boldsymbol{u}^t + \lambda \nabla F(\boldsymbol{u}^t)\big) \\ \boldsymbol{u}^{t+1} = \mathcal{H}_{1/2}(\boldsymbol{u}^{t+1/2}), \end{cases} \tag{3.10}$$

where ∇F is the gradient of F, $dt > 0$ is a time stepsize, and $\mathcal{H}_{1/2}(\cdot)$ is a hard thresholding

operator defined as

$$(\mathcal{H}_{1/2}(\boldsymbol{u}))_i = \begin{cases} 1, & \text{if } u_i \geq 1/2 \\ 0, & \text{if } u_i < 1/2, \end{cases}$$

for $i = 1, \ldots, n$. To circumvent the restriction on binary solutions in the MBO scheme, we use a bitwise scheme to deal with grayscale images in Section 3.4.

3.6 gtvMBO: ADMM Algorithm

We are now ready to optimize our gTV-regularized model (3.5), reproduced below:

$$\begin{aligned} &\min_{\substack{\boldsymbol{S}\in\Omega_{w\times k} \\ \boldsymbol{A}\in\Omega_{k\times n}, \mathbf{1}_k^\top \boldsymbol{A} = \mathbf{1}_n^\top}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2 + \lambda J_{TV}(A) \\ = &\min_{\substack{\boldsymbol{S}\in\Omega_{w\times k} \\ \boldsymbol{A}\in\Omega_{k\times n}, \mathbf{1}_k^\top \boldsymbol{A} = \mathbf{1}_n^\top}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2 + \frac{\lambda}{2}\sum_{i,j=1}^{n} \|\widehat{\boldsymbol{a}}_i - \widehat{\boldsymbol{a}}_j\|_1 W_{ij}. \end{aligned}$$

In order to apply the ADMM framework, we rewrite the constraints using indicator functions. In general, the indicator function χ_Δ of a set Δ is defined as

$$\chi_\Delta(\boldsymbol{Z}) = \begin{cases} 0, & \boldsymbol{Z} \in \Delta; \\ \infty, & \text{otherwise.} \end{cases}$$

By denoting $\Pi := \{\boldsymbol{Z} \in \mathbb{R}^{k\times n} : \boldsymbol{Z} \in \Omega_{k\times n}, \mathbf{1}_k^\top \boldsymbol{Z} = \mathbf{1}_n^\top\}$, we can rewrite the model (3.5) as an unconstrained problem,

$$\min_{\boldsymbol{S},\boldsymbol{A}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2 + \lambda J_{TV}(\boldsymbol{A}) + \chi_{\Omega_{w\times k}}(\boldsymbol{S}) + \chi_\Pi(\boldsymbol{A}). \tag{3.11}$$

We introduce two auxiliary variables $\boldsymbol{B} \in \mathbb{R}^{k\times n}$, $\boldsymbol{C} \in \mathbb{R}^{w\times k}$ and rewrite the objective function (3.11) as its equivalent form,

$$\begin{aligned}\min_{\boldsymbol{S},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}} \ & \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{C}\boldsymbol{A}\|_{\mathrm{F}}^2+\lambda J_{TV}(\boldsymbol{B})+\chi_{\Omega_{w\times k}}(\boldsymbol{S})+\chi_{\Pi}(\boldsymbol{A}) \\ \text{s.t.} \quad & \boldsymbol{A}=\boldsymbol{B},\ \boldsymbol{S}=\boldsymbol{C}.\end{aligned}$$

The corresponding augmented Lagrangian is

$$\begin{aligned}\mathcal{L}(\boldsymbol{C},\boldsymbol{S},\boldsymbol{A},\boldsymbol{B}) = & \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{C}\boldsymbol{A}\|_{\mathrm{F}}^2+\lambda J_{TV}(\boldsymbol{B})+\chi_{\Omega_{w\times k}}(\boldsymbol{S})+\chi_{\Pi}(\boldsymbol{A}) \\ & +\frac{\rho}{2}\|\boldsymbol{A}-\boldsymbol{B}+\widetilde{\boldsymbol{B}}\|_{\mathrm{F}}^2+\frac{\gamma}{2}\|\boldsymbol{S}-\boldsymbol{C}+\widetilde{\boldsymbol{C}}\|_{\mathrm{F}}^2,\end{aligned}$$

where $\widetilde{\boldsymbol{B}}$, $\widetilde{\boldsymbol{C}}$ are dual variables and ρ,γ are two positive parameters. Then the ADMM algorithm requires solving four subproblems at each iteration, i.e., minimizing $\mathcal{L}$ with respect to $\boldsymbol{C},\boldsymbol{S},\boldsymbol{A}$ and $\boldsymbol{B}$ individually while fixing the others. The $\boldsymbol{C}$-subproblem reads as

$$\operatorname*{argmin}_{\boldsymbol{C}} \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{C}\boldsymbol{A}\|_{\mathrm{F}}^2+\frac{\gamma}{2}\|\boldsymbol{S}-\boldsymbol{C}+\widetilde{\boldsymbol{C}}\|_{\mathrm{F}}^2,$$

which has a closed-form solution. The $\boldsymbol{S}$-subproblem seeks the projection of $\boldsymbol{C}-\widetilde{\boldsymbol{C}}$ onto the set of all non-negative matrices, which can be solved by hard thresholding. As for the $\boldsymbol{A}$-subproblem, the solution can be obtained by projecting a least squares solution onto the convex set Π, i.e.,

$$\boldsymbol{A}=\mathcal{P}_{\Pi}\Big((\boldsymbol{S}^{\top}\boldsymbol{S}+\rho\boldsymbol{I})^{-1}\big(\boldsymbol{S}^{\top}\boldsymbol{X}+\rho(\boldsymbol{B}-\widetilde{\boldsymbol{B}})\big)\Big),$$

where $\mathcal{P}_{\Pi}$ is the projection operator on the set Π that can be implemented by a fast algorithm [125].

For the $\boldsymbol{B}$-subproblem, we approximate the non-differentiable gTV by the graph GL functional,

$$\underset{\boldsymbol{B}}{\operatorname{argmin}} \quad \lambda\epsilon\langle \boldsymbol{B}^\top, \boldsymbol{L}_s\boldsymbol{B}^\top\rangle + \frac{\lambda}{\epsilon}\Phi\left(\boldsymbol{B}\right) + \frac{\rho}{2}\|\boldsymbol{A} - \boldsymbol{B} + \widetilde{\boldsymbol{B}}\|_{\mathrm{F}}^2,$$

where Φ are defined in Section 3.5. To remove the binary restriction of MBO, we approximate any real number in $[0,1]$ by its best M-bit binary representation [93]. We apply the MBO scheme on each channel separately, which can be implemented in parallel. Finally, we combine all the channels to get an approximated solution with elements in $[0,1]$ for the $\boldsymbol{B}$-subproblem. In all our experiments, we set $M = 8$. Specifically, we approximate the matrix $\boldsymbol{B}$ by a set of M binary matrices $\boldsymbol{B}_m \in \mathbb{R}^{k\times n}$ with $m = 1, \cdots, M$ such that

$$\boldsymbol{B}_{ij} \approx \sum_{m=1}^{M} 2^{-m}[\boldsymbol{B}_m]_{ij},$$

where M is the total number of bits being considered and $\boldsymbol{B}_m$ is the mth bit channel of the matrix $\boldsymbol{B}$, i.e., $[\boldsymbol{B}_m]_{ij} \in \{0,1\}$. Likewise, we approximate $\boldsymbol{A}$ and $\widetilde{\boldsymbol{B}}$ in the same manner and get two sets of binary matrices $\{\boldsymbol{A}_m\}_{m=1}^M$ and $\{\widetilde{\boldsymbol{B}}_m\}_{m=1}^M$. Then for each channel, we approximate the gTV regularization J_{TV} by the graph GL functional (3.9). Hence, we obtain the following minimization problem for each $\boldsymbol{B}_m$,

$$\underset{\boldsymbol{B}_m}{\operatorname{argmin}} \quad \lambda\epsilon\operatorname{tr}\left(\boldsymbol{B}_m\boldsymbol{L}_s\boldsymbol{B}_m^\top\right) + \frac{\lambda}{4\epsilon}\sum_{i=1}^{n}\sum_{\ell=1}^{k}[B_m]_{i\ell}^2\left([B_m]_{i\ell}-1\right)^2 + \frac{\rho}{2}\|\boldsymbol{A}_m - \boldsymbol{B}_m + \widetilde{\boldsymbol{B}}_m\|_{\mathrm{F}}^2. \tag{3.12}$$

Note that we assume that the graph structure at each channel is consistent with the one that is defined by the given hyperspectral data $\boldsymbol{X}$.

We apply the MBO scheme (3.10) to minimize (3.12), which is a two-step iterative

algorithm. The first step requires solving for $\boldsymbol{B}_m^\top$ from

$$\boldsymbol{L}_s\boldsymbol{B}_m^\top + \frac{\rho}{\lambda}(\boldsymbol{B}_m^\top - \boldsymbol{A}_m^\top - \widetilde{\boldsymbol{B}}_m^\top) = 0. \tag{3.13}$$

Motivated by [99], we further accelerate the MBO by taking advantage of the approximated eigendecomposition of $\boldsymbol{L}_s$ given in (3.8). Multiplying both sides of (3.13) with $\boldsymbol{V}^\top$ from the left, we get $\boldsymbol{\Lambda}\boldsymbol{V}^\top\boldsymbol{B}_m^\top + \frac{\rho}{\lambda}\Big(\boldsymbol{V}^\top\boldsymbol{B}_m^\top - \boldsymbol{V}^\top(\boldsymbol{A}_m^\top + \widetilde{\boldsymbol{B}}_m^\top)\Big) = 0$, or equivalently

$$\boldsymbol{B}_m\boldsymbol{V}\boldsymbol{\Lambda} + \frac{\rho}{\lambda}\Big(\boldsymbol{B}_m\boldsymbol{V} - (\boldsymbol{A}_m + \widetilde{\boldsymbol{B}}_m)\boldsymbol{V}\Big) = 0, \tag{3.14}$$

since $\boldsymbol{V}^\top\boldsymbol{V} = \boldsymbol{I}$. As a result, we only need to solve for $\boldsymbol{B}_m\boldsymbol{V} \in \mathbb{R}^{k\times p}$ with a reduced problem size. Denote $\boldsymbol{Z}_m = \boldsymbol{B}_m\boldsymbol{V}$ and $\boldsymbol{N}_m = \frac{\rho}{\lambda}\big(\boldsymbol{B}_m\boldsymbol{V} - (\boldsymbol{A}_m + \widetilde{\boldsymbol{B}}_m)\boldsymbol{V}\big)$. At the $(\tau+1)$th iteration, we have the following algorithm to update $\boldsymbol{B}_m$:

$$\begin{cases} \boldsymbol{Z}_m^{\tau+1} = \boldsymbol{Z}_m^{\tau}(\boldsymbol{I} - d\tau\boldsymbol{\Lambda}) - d\tau\cdot\boldsymbol{N}_m^{\tau} \\ \boldsymbol{B}_m^{\tau+1/2} = \boldsymbol{V}\boldsymbol{Z}_m^{\tau+1} \\ \boldsymbol{N}_m^{\tau+1} = \frac{\rho}{\lambda}\Big(\boldsymbol{B}_m^{\tau+1/2} - (\boldsymbol{A}_m + \widetilde{\boldsymbol{B}}_m)\Big)\boldsymbol{V} \\ \boldsymbol{B}_m^{\tau+1} = \mathcal{H}_{1/2}(\boldsymbol{B}_m^{\tau+1/2}). \end{cases} \tag{3.15}$$

The first three equations in (3.15) are obtained by applying fixed-point iteration to solve (3.14), and the last equation in (3.15) is from the MBO scheme in (3.10). Our numerical experiments show that five iterations of (3.15) for each $\boldsymbol{B}_m$-subproblem are sufficient to produce reasonable results. If the $\boldsymbol{B}$-subproblem can be solved within certain accuracy, then the convergence of ADMM can be guaranteed [126].

In summary, each subproblem in the ADMM algorithm can be solved efficiently either through a closed-form solution or within a few iterations. The entire algorithm is

presented in Alg. 2, which terminates when maximum number of iterations is reached, or either the relative error between two subsequent mixing matrices, i.e., $\|\boldsymbol{S}^{t+1} - \boldsymbol{S}^t\|_{\mathrm{F}}/\|\boldsymbol{S}^t\|_{\mathrm{F}}$, or the relative error between two subsequent abundance maps, i.e., $\|\boldsymbol{A}^{t+1} - \boldsymbol{A}^t\|_{\mathrm{F}}/\|\boldsymbol{A}^t\|_{\mathrm{F}}$, is smaller than a given tolerance.

Algorithm 2 gtvMBO: ADMM algorithm with MBO inner loop to solve (3.5)

inputs data $\boldsymbol{X}$, parameters ρ, λ, γ, $d\tau$.
initialize
  Use the Nyström method on $\boldsymbol{X}$ to get the p-rank eigendecomposition form of the graph Laplacian $\boldsymbol{L}_s = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top$.
  $\boldsymbol{S} \leftarrow \boldsymbol{S}_{init}, \quad \boldsymbol{A} \leftarrow \boldsymbol{A}_{init}$
  $\boldsymbol{B} \leftarrow \boldsymbol{A}, \quad \boldsymbol{C} \leftarrow \boldsymbol{S}$
  $\widetilde{\boldsymbol{B}} \leftarrow \boldsymbol{0}, \quad \widetilde{\boldsymbol{C}} \leftarrow \boldsymbol{0}$
repeat
  $\boldsymbol{C} \leftarrow \left(\boldsymbol{X}\boldsymbol{A}^\top + \gamma\left(\boldsymbol{S} + \widetilde{\boldsymbol{C}}\right)\right)\left(\boldsymbol{A}\boldsymbol{A}^\top + \gamma\boldsymbol{I}_k\right)^{-1}$.
  $\boldsymbol{S} \leftarrow \max(\boldsymbol{C} - \widetilde{\boldsymbol{C}}, 0)$.
  $\boldsymbol{A} \leftarrow \mathcal{P}_{\Pi}\left(\left(\boldsymbol{S}^\top\boldsymbol{S} + \rho\boldsymbol{I}_k\right)^{-1}\left(\boldsymbol{S}^\top\boldsymbol{X} + \rho\left(\boldsymbol{B} - \widetilde{\boldsymbol{B}}\right)\right)\right)$.
  Split $\boldsymbol{B}, \boldsymbol{A}, \widetilde{\boldsymbol{B}}$ bitwise.
  for $m \leftarrow 1, \ldots, 8$ **do**
    $\boldsymbol{Z}_m \leftarrow \boldsymbol{B}_m\boldsymbol{V}$
    $\boldsymbol{N}_m \leftarrow \frac{\rho}{\lambda}\left(\boldsymbol{B}_m - \left(\boldsymbol{A}_m + \widetilde{\boldsymbol{B}}_m\right)\right)\boldsymbol{V}$
    repeat
      $\boldsymbol{Z}_m \leftarrow \boldsymbol{Z}_m(\boldsymbol{I}_p - d\tau\boldsymbol{\Lambda}) - d\tau \cdot \boldsymbol{N}_m$
      $\boldsymbol{B}_m \leftarrow \boldsymbol{V}\boldsymbol{Z}_m$
      $\boldsymbol{N}_m \leftarrow \frac{\rho}{\lambda}\Big(\boldsymbol{B}_m - (\boldsymbol{A}_m + \widetilde{\boldsymbol{B}}_m)\Big)\boldsymbol{V}$
      $\boldsymbol{B}_m \leftarrow \mathcal{H}_{1/2}(\boldsymbol{B}_m)$
    until termination
  end for
  Combine $\boldsymbol{B}_m, \boldsymbol{A}_m, \widetilde{\boldsymbol{B}}_m$ bitwise.
  $\widetilde{\boldsymbol{B}} \leftarrow \widetilde{\boldsymbol{B}} + (\boldsymbol{A} - \boldsymbol{B})$.
  $\widetilde{\boldsymbol{C}} \leftarrow \widetilde{\boldsymbol{C}} + (\boldsymbol{S} - \boldsymbol{C})$.
until termination
outputs endmember spectra estimate $\boldsymbol{S}$, abundance map estimate $\boldsymbol{A}$.

3.6.1 Time Complexity of gtvMBO

Here we discuss the complexity of the proposed algorithm and compare it with related methods. The computational complexity of the Nyström method is $O(wpn + p^2n)$, mainly for computing $\boldsymbol{W}_{12}$ and singular value decomposition in (3.7). This is much smaller than calculating the graph Laplacian matrix directly as described in Section 3.5.1, which is $O(wn^2)$. As for the space complexity, using the approximated graph Laplacian requires storing only $O(pn)$ numbers, while using the full graph Laplacian would need to store $O(n^2)$ numbers.

The time complexity of each step in Alg. 2 is summarized as follows:

- $\boldsymbol{C}$ update: $O(wkn)$;
- $\boldsymbol{S}$ update: $O(wk)$;
- $\boldsymbol{A}$ update: $O(wkn + nk\log k) = O(wkn)$;
- $\boldsymbol{B}$ update per bit channel: $O(kpn)$;
- $\tilde{\boldsymbol{B}}, \tilde{\boldsymbol{C}}$ update: $O(kn)$.

Therefore, the time complexity for our algorithm per iteration is $O(kn(w + p))$ in total. Given $p \ll n$ and $k < w$, this is faster than the other two related methods: SUnSAL-TV [101] and GLNMF [114], which are in the order of $O(wn(w + \log n))$ and $O(kn(w + kn))$, respectively.

3.7 Hyperspectral Unmixing Experiments

In this section, we conduct extensive experiments on synthetic and real data to demonstrate the performance of the proposed approach, referred to as gtvMBO, in comparison with the

state-of-the-art methods in blind and non-blind hyperspectral unmixing. Methods that we compare include FCLSU [86], SUnSAL-TV [101] (denoted by STV), GLNMF [114], fractional ℓ_q norm-regularized unmixing method with $q = 0.1$ (denoted by FRAC) [88], NMF-QMV [107] (denoted by QMV), and our earlier unmixing work based on the graph Laplacian [7] (denoted by GraphL).

To quantitatively measure the performance, we adopt the following metrics to calculate the error between an estimation $\hat{\boldsymbol{Y}} \in \mathbb{R}^{r\times c}$ and the reference $\boldsymbol{Y} \in \mathbb{R}^{r\times c}$.

1. Root-mean-square error (RMSE)

$$RMSE(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{1}{c}\sqrt{\frac{1}{r}\sum_{i=1}^{r} \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|_2^2},$$

   where $\boldsymbol{y}_i \in \mathbb{R}^c$ is the ith row of $\boldsymbol{Y}$.

2. Normalized mean-square error (nMSE)

$$nMSE(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{\|\boldsymbol{Y} - \widehat{\boldsymbol{Y}}\|_{\mathrm{F}}}{\|\boldsymbol{Y}\|_{\mathrm{F}}}.$$

3. Spectral angle mapper (SAM) in degrees

$$SAM(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{1}{c}\sum_{j=1}^{c} \arccos\left(\frac{\boldsymbol{y}_j^\top \hat{\boldsymbol{y}}_j}{\|\boldsymbol{y}_j\|_2 \|\hat{\boldsymbol{y}}_j\|_2}\right),$$

   where $\boldsymbol{y}_j \in \mathbb{R}^r$ is the jth column of $\boldsymbol{Y}$. The index j is skipped in the sum when $\|\boldsymbol{y}_j\|_2\|\hat{\boldsymbol{y}}_j\|_2 = 0$.

In order to make a fair comparison, we use the initialization steps in [88] for all the methods considered in this paper. VCA [127] returns $10k$ endmember candidates that are clustered into k groups. This is directly used as $\boldsymbol{S}$ for FCLSU and FRAC,

while we use the mean spectrum within each group and the sum of the abundances estimated by FCLSU within each group as the initial guesses $\boldsymbol{S}_{init}$ and $\boldsymbol{A}_{init}$, respectively, for all compared methods. We set $\sigma = 5$ in the weight computation (3.6) and randomly select 0.1% samples from the entire pixel list in the Nyström method to approximate the graph Laplacian. As for γ, ρ and λ, we choose the optimal parameters that minimize nMSE$(\boldsymbol{A}, \hat{\boldsymbol{A}})$. We first perform a coarse grid search with parameter candidates evenly spaced over the interval on a log scale, then do a finer grid search around the best parameters, e.g., search for an optimal λ in $\{10^{2.5}, 10^{2.75}, \ldots, 10^{3.5}\}$ given $\lambda = 10^3$ from the coarse grid search. For GraphL and gtvMBO, the coarse grid search is over $\lambda \in \{10^{-5}, 10^{-4}, \ldots, 10^5\}$, $\rho/\lambda \in \{10^{-3}, 10^{-2}, \ldots, 10^3\}$, and $\gamma \in \{10^2, 10^3, \ldots, 10^5\}$. For FRAC, we fix $\rho = 10$ as suggested in [88] and search for λ among $\{10^{-5}, 10^{-4}, \ldots, 10^5\}$. For QMV, we search for λ (denoted by β in [107]) $\in \{10^{-5}, 10^{-4}, \ldots, 10^5\}$. For GLNMF and STV, we search for $\lambda, \mu \in \{10^{-5}, 10^{-4}, \ldots, 10^5\}$. See Section 3.8 for a detailed discussion on parameter selection and sensitivity of our method. Our Matlab source codes are available at `https://github.com/HarlinLee/gtvMBO-public`. All experiments are performed in Matlab 2018b on a MacBook Pro 2017 with an 2.9 GHz Intel Core i7 and 16GB RAM in double precision.

3.7.1 Synthetic Data

To evaluate the performance of all methods, we construct a set of synthetic data $\boldsymbol{X}$ with ground truth mixing matrix $\boldsymbol{S}$ and endmember matrix $\boldsymbol{A}$. Fig. 3.1 shows the ground truth abundance maps. We adopt the same simulation procedure as in [101], where an endmember library is generated by randomly selecting 240 materials from the USGS 1995 library with 224 spectral bands. The noise-free hyperspectral image with 75×75 pixels is generated by a random selection of 5 spectral signatures from the library. The respective

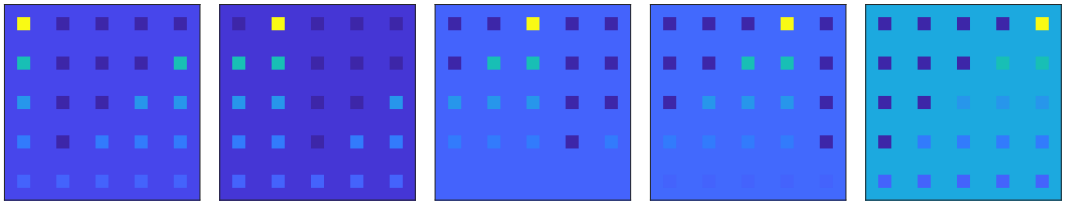


Figure 3.1: Ground truth abundance maps of the synthetic data (five endmembers).

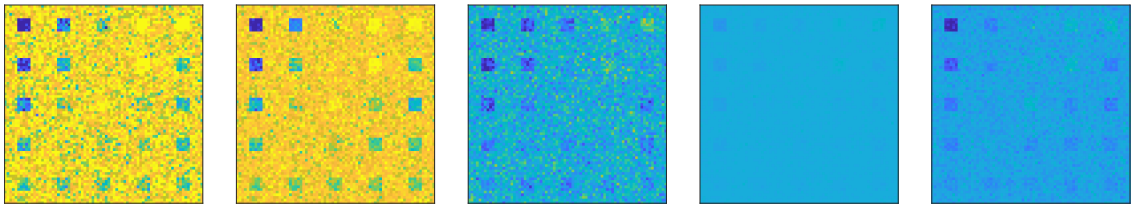


Figure 3.2: Reconstructed abundance maps of the fifth element (i.e. ground truth is the right most image of Fig. 3.1). gtvMBO estimate achieves a balance between high-contrast items in the front and the background noise. Input data had noise added with SNR 10dB. All images are visualized over the range $[0, 1]$.

	FCLSU	**FRAC**	**STV**	**GLNMF**	**QMV**	**GraphL**	**gtvMBO**
				SNR = 10dB			
RMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.242	0.157	0.248	0.24	0.093	0.0513	**0.051**
nMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	1.05	0.696	1.07	1.03	0.435	0.364	**0.327**
RMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	**0.14**	–	–	0.211	0.612	0.16	0.16
nMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	**0.205**	–	–	0.321	0.881	0.244	0.241
SAM($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	10.2	–	–	14.8	40.5	8.65	**8.57**
				SNR= 20dB			
RMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.106	0.106	0.065	0.107	0.048	**0.043**	0.065
nMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.523	0.523	0.314	0.521	**0.227**	0.242	0.314
RMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.055	–	–	0.067	**0.037**	0.095	0.054
nMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.092	–	–	0.104	**0.053**	0.13	0.091
SAM($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	2.67	–	–	3.18	2.69	6.88	**2.65**

Table 3.2: Unmixing results on the synthetic dataset.

ground truth abundances are randomly fixed as 0.1149, 0.0741, 0.2003, 0.2055, and 0.4051. The noisy hyperspectral data is then obtained by adding zero-mean Gaussian noise with a signal-to-noise ratio (SNR) of 10db and 20db, respectively.

Table 3.2 compares all methods on the noisy data quantitatively. To get a visual comparison, we present the case of SNR= 10dB in Fig. 3.2. In particular, we show all the reconstructed abundance maps corresponding to the fifth ground truth abundance in Fig. 3.1. We exclude the results of FCLSU and FRAC in Fig. 3.2, as both fail to recover the abundance maps under such a low SNR scenario. One can see that STV and GLNMF have a different color range on the background comparing to other methods, while the QMV background is still noisy. The proposed gtvMBO achieves a balance between recognizable objects and background noise, while the result of GraphL is slightly oversmoothed. Note that the proposed gtvMBO only considers the regularization on $\boldsymbol{A}$, while QMV uses the minimum-volume-based regularization on $\boldsymbol{S}$, but our method still gives comparable results in recovering $\boldsymbol{S}$ compared to QMV, and has an advantage on reconstructing $\boldsymbol{A}$, especially when the underlying abundance map has spectral geometries. In addition, gtvMBO can reconstruct $\boldsymbol{A}$ well within a few iterations but it takes more iterations to get a good reconstruction of $\boldsymbol{S}$. In the preprocessing step, both GraphL and gtvMBO take less than a second to estimate the eigenvalues and eigenvectors of the low-rank approximation to the graph Laplacian by the Nyström method, while GLNMF typically takes a minute to calculate the graph Laplacian. In terms of running time, gtvMBO is slower than FRAC and GraphL, but much faster than the other competing methods.

3.7.2 Real Data

We use the real hyperspectral data $\boldsymbol{X}$ with the references $\boldsymbol{S}$ and $\boldsymbol{A}$ from [128], including Samson, Jasper Ridge and Urban datasets. In particular, the endmembers are manually

selected from the image data by assuming k distinct materials with one signature per material and neglecting possible spectral variability issues. The reference abundances are obtained via FCLSU. This way of generating references for endmembers/abundances has been widely used for assessing the performance of various unmixing algorithms. As no ground truth is available for the real data, it is common to compare the unmixing results to the reference endmembers/abundances.

Samson

In the first experiment, we use the Samson data with 95×95 pixels and 156 spectral bands after preprocessing, whose reference has three endmembers. The unmixing results are given in Figs. 3.3 and 3.6 and Table 3.3 for abundance maps, endmembers and quantitative metrics, respectively. In Fig. 3.6, all endmember plots can capture the rough shape and discontinuities in the ground truth but with different heights. The gtvMBO result has many endmember elements that are close to zero since we enforce the non-negative constraint on the endmember $\boldsymbol{S}$ by using the hard thresholding operator in the $\boldsymbol{S}$-subproblem. For the abundance maps, the STV results look blurry when trying to preserve spatial smoothness and the GLNMF results are noisy in the homogeneous areas, as its graph Laplacian is based on the entire data that may contain certain amount of noise. Both blurring and noisy artifacts can be mitigated by the low-rank approximation of graph Laplacian in the Nyström method as in GraphL and gtvMBO. On the other hand, gtvMBO yields sharper edges than GraphL, thanks to the graph TV regularization. Table 3.3 reports that GLNMF gives the best estimations in $\boldsymbol{S}$ at the cost of high computational costs, whereas the proposed method is the best in reconstructing the abundance maps. Note that "graph time" in Table 3.3 is referred to as the time needed to compute the adjacency matrix (for GLNMF) and the graph Laplacian matrix (for GraphL and gtvMBO), while "algorithm

time," or "alg. time" in short, refers to the time needed to run the unmixing algorithm after initialization and graph construction. The overall computation time of gtvMBO is the sum of "graph time" and "time," which is comparable to QMV and much faster than GLNMF.

	FCLSU	**FRAC**	**STV**	**GLNMF**	**QMV**	**GraphL**	**gtvMBO**
RMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.044	–	–	**0.036**	0.073	0.052	0.070
nMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.169	–	–	**0.153**	0.302	0.203	0.296
SAM($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	**3.64**	–	–	4.49	12.8	7.86	9.84
RMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.18	0.165	0.165	0.187	0.148	0.139	**0.096**
nMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.455	0.429	0.375	0.502	0.428	0.302	**0.243**
Graph time (sec)	–	–	–	66.4	–	0.082	0.082
Alg. time (sec)	2.34	0.052	4.08	8.73	1.6	0.094	0.609
λ	–	1	0.01	1	$10^{2.75}$	$10^{-5.25}$	$10^{-3.75}$
ρ	–	10	–	–	–	$10^{-1.75}$	$10^{-2.25}$
γ	–	–	–	–	–	10^{5}	10^{4}
μ	–	–	–	1	–	–	–
Iterations	–	2	1000	1000	101	30	30

Table 3.3: Unmixing results on the Samson dataset.

Jasper Ridge

In the second experiment, we test the Jasper Ridge data which has 100×100 pixels and 198 spectral bands. The unmixing results for abundance maps and endmembers are shown in Figs. 3.4 and 3.7. In Fig. 3.4, the FRAC abundance maps have the highest image contrast, while mistakenly identifying trees and roads in some areas, especially the top right part. The STV abundance maps are over-smoothed, especially in the Dirt abundance map. Since only the five nearest neighbors are considered when calculating the pairwise weight of a fully-connected graph, GLNMF may miss some global features while preserving fine details. For example, some variations in the water are captured but some roads are not

identified in the GLNMF abundance maps. One can see that both GraphL and gtvMBO perform very well at identifying Water and Road abundance maps because of the learned graph structure in the Nyström method. Specifically for the road abundance, these two methods can recover the road on the rightmost part of the image. This phenomenon could be explained by the fact that it is a very narrow structure, and the non-local similarity with road pixels across all bands plays an important role, illustrating an advantage of using graph TV over spatial TV. The gtvMBO results are even better than GraphL in preserving the sharpness especially in the Dirt abundance map. The endmember spectral plot in Fig. 3.7 also confirms that the methods failing for the road extract a very poor signatures compared to the reference. Table 3.4 compares all the methods quantitatively. It is true that QMV gives the best results on this dataset, which is probably because that the assumptions made by QMV hold on Jasper, but not on the other datasets. The proposed gtvMBO can recover endmembers and abundance maps in a balanced manner. The comparison results imply that a good RMSE on the reconstructed data can not guarantee a good unmixing performance.

Urban

Lastly, we test a relatively large dataset, i.e. the Urban dataset with 307×307 pixels and 162 spectral bands, whose reference has four endmembers. The results for all methods are presented in Figs. 3.5 and 3.8. In Fig. 3.5, most methods, including FCLSU, FRAC, STV, GLNMF, and QMV, yield abundance maps in low image contrast due to the initial guess, especially in the abundance maps for the asphalt and roof. As a by-product, the proposed gtvMBO can greatly improve the image contrast of the abundance map due to the graph TV regularization. In addition, all the methods have a hard time extracting a good roof endmember, but the graph-based approaches are able to compensate this with more

	FCLSU	**FRAC**	**STV**	**GLNMF**	**QMV**	**GraphL**	**gtvMBO**
RMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.144	–	–	0.133	**0.031**	0.18	0.083
nMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.608	–	–	0.598	**0.107**	0.629	0.288
SAM($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	16.8	–	–	14.9	**3.54**	14.6	12.8
RMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.148	0.109	0.142	0.111	**0.073**	0.145	0.136
nMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.472	0.46	0.47	0.437	**0.221**	0.38	0.353
Graph time (sec)	–	–	–	126	–	0.225	0.225
Alg. time (sec)	4.27	9.52	4.56	10.4	3.89	0.34	3.32
λ	–	1	$10^{-1.25}$	$10^{-0.5}$	$10^{2.25}$	$10^{-4.5}$	$10^{-4.25}$
ρ	–	10	–	–	–	0.1	$10^{-2.75}$
γ	–	–	–	–	–	10^{4}	$10^{3.75}$
μ	–	–	–	$10^{-2.5}$	–	–	–
Iterations	–	300	1000	1000	101	100	100

Table 3.4: Unmixing results on the Jasper Ridge dataset.

features preserved. Also note that because QMV does not enforce non-negativity on $\boldsymbol{S}$, the resulting spectrum for Asphalt in QMV goes below zero. In the Roof abundance maps, only GraphL and gtvMBO can capture those sporadic roof tops since the approximated graph Laplacian considers the pairwise similarity across spectral bands in the original data with dimension w much greater than the dimension k for the column space of the abundance map $\boldsymbol{A}$. In Table 3.5, we list all quantitative metric comparisons where gtvMBO reaches the smallest residual error and get comparable reconstruction errors for the abundance map and endmember with GraphL. Overall, the proposed method can reconstruct abundance maps and endmember matrices with high accuracy in a short time.

3.8 Parameter Selection

Due to heavy computations involved in these tasks, all the results presented in this section are performed on a workstation of DELL R7425 Dual Processor AMD Epyc 32 core 2.2

	FCLSU	**FRAC**	**STV**	**GLNMF**	**QMV**	**GraphL**	**gtvMBO**
RMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.109	–	–	0.188	0.211	**0.099**	**0.099**
nMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	**0.635**	–	–	1.35	1.2	0.636	0.639
SAM($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	19.5	–	–	17.9	46.4	**14.8**	14.9
RMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.145	0.153	0.289	0.175	0.245	**0.134**	0.136
nMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	0.438	0.45	0.756	0.554	0.655	**0.384**	0.393
Graph time (sec)	–	–	–	10800	–	9.09	9.09
Alg. time (sec)	34.7	0.85	142	86.1	29	0.353	22.4
λ	–	$10^{-0.5}$	$10^{-2.25}$	$10^{-1.5}$	$10^{1.75}$	$10^{-3.25}$	10^{-6}
ρ	–	10	–	–	–	$10^{-1.25}$	$10^{-5.5}$
γ	–	–	–	–	–	$10^{4.75}$	$10^{4.75}$
μ	–	–	–	$10^{-5.5}$	–	–	–
Iterations	–	2	1000	1000	101	10	10

Table 3.5: Unmixing results on the Urban dataset.

GHz machines with 512GB RAM each.

There are several tuning parameters in our approach: the filtering parameter σ in computing pairwise weights of the graph, the regularization parameter λ associated with the graph TV in the proposed unmixing model, the penalty parameters ρ and γ in the proposed algorithm based on ADMM, and time step size dt for the diffusion step in the modified MBO scheme. The value of σ could be changed proportionally according to the number of spectral bands w. Since all the test datasets have 100$\sim$200 spectral bands, we find that $\sigma = 5$ typically gives good results, so we fix it throughout the experimental section. To solve the $\boldsymbol{B}$-subproblem, we fix the step size $dt = 0.01$ and run 5 iterations of (3.15) in the modified MBO scheme.

To find optimal or sub-optimal values of λ, ρ, and γ, we consider a skillful strategy which alleviates the time-consuming parameter tuning. If the value of λ increases, the recovered abundance map $\boldsymbol{A}$ has a graph structure more similar to that of the given data $\boldsymbol{X}$ but with larger residual error and vice versa. The penalty parameters ρ and γ both

control the convergence of the proposed algorithm according to the ADMM framework. In other words, λ is a model parameter that affects the performance, and ρ, γ are algorithmic parameters that affect the convergence. Therefore, we suggest a set of default parameters by fixing the ratios as $\rho/\lambda = 1$, $\gamma/\lambda = 10^7$ and only tuning the regularization parameter λ. In fact, the $\boldsymbol{B}$-subproblem is determined by the ratio ρ/λ. Table 3.6 shows that using these default algorithmic parameters still ensures comparable unmixing performance on the datasets to when we tune all the three parameters together. Note that the optimal parameters indeed yield better results than the default parameters in terms of $\text{SAM}(\boldsymbol{S}, \hat{\boldsymbol{S}})$, which is due to the fact that our regularization is formulated on $\boldsymbol{A}$ and the optimal parameters are determined according to $\text{nMSE}(\boldsymbol{A}, \hat{\boldsymbol{A}})$, resulting in more deviations in $\boldsymbol{S}$. In future work, we might consider choosing optimal parameters based on a combination of evaluation metrics on $\boldsymbol{S}$ and $\boldsymbol{A}$.

In addition, learning a graph Laplacian or its low-rank approximation is an important preprocessing step in our proposed method. In the Nyström method, the sampling rate is fixed as 0.1% in all our experiments. Our empirical results show that this is sufficient for preserving the graph structure of the original hyperspectral data. In fact, there is a trade-off between the number of samples corresponding to the rank of the approximated Laplacian and the orthogonality of columns in the approximated eigenvectors: more samples can improve accuracy in approximating the graph Laplacian but may result in loss of orthogonality of the resulting eigenvectors, which is also desired in our modified MBO scheme (3.15). Other adaptive sampling schemes for the Nyström extension [129] will be explored in our future work. For high performance computing applications, the Nyström loop can be optimized for specific architectures as in [130].

	Samson	**Jasper**	**Urban**
RMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.07 / **0.062**	**0.083** / 0.13	**0.099** / 0.10
nMSE($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	0.3 / **0.23**	**0.29** / 0.44	**0.64** / 0.67
SAM($\boldsymbol{S}, \hat{\boldsymbol{S}}$)	**9.84** / 16.1	**12.8** / 17.8	**14.9** / 15.8
RMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	**0.096** / 0.12	0.14 / **0.13**	0.14 / 0.2
nMSE($\boldsymbol{A}, \hat{\boldsymbol{A}}$)	**0.24** / 0.27	0.35 / 0.35	**0.39** / 0.41
λ	$10^{-3.5}$	10^{-8}	$10^{-2.5}$

Table 3.6: Unmixing results of gtvMBO in A/B format, where A is the previous result using optimally tuned λ, ρ, γ, and B is the result of using default ratios $\rho/\lambda, \gamma/\lambda$ and only tuning the λ value (given in the last row.)

3.9 Conclusions

We propose a gTV-regularized approach for blind hyperspectral unmixing to estimate both the abundance map and the mixing matrix under the assumption that the underlying abundance map and the given hyperspectral data share the same graph structure. In particular, we applied the Nyström method to approximate the eigenvalues and eigenvectors of a normalized graph Laplacian. To solve the proposed gTV-regularized unmixing problem with probability simplex constraints, we derived an efficient algorithm based on ADMM. One of the subproblems is decomposed into bits and then solved by the fast MBO scheme at each bit channel. Extensive experiments were conducted to demonstrate that the proposed framework is effective and efficient, especially when the hyperspectral data have similarities across spectral bands. In the future, one could integrate robust graph learning methods and minimum-volume-based regularizations into hyperspectral unmixing.

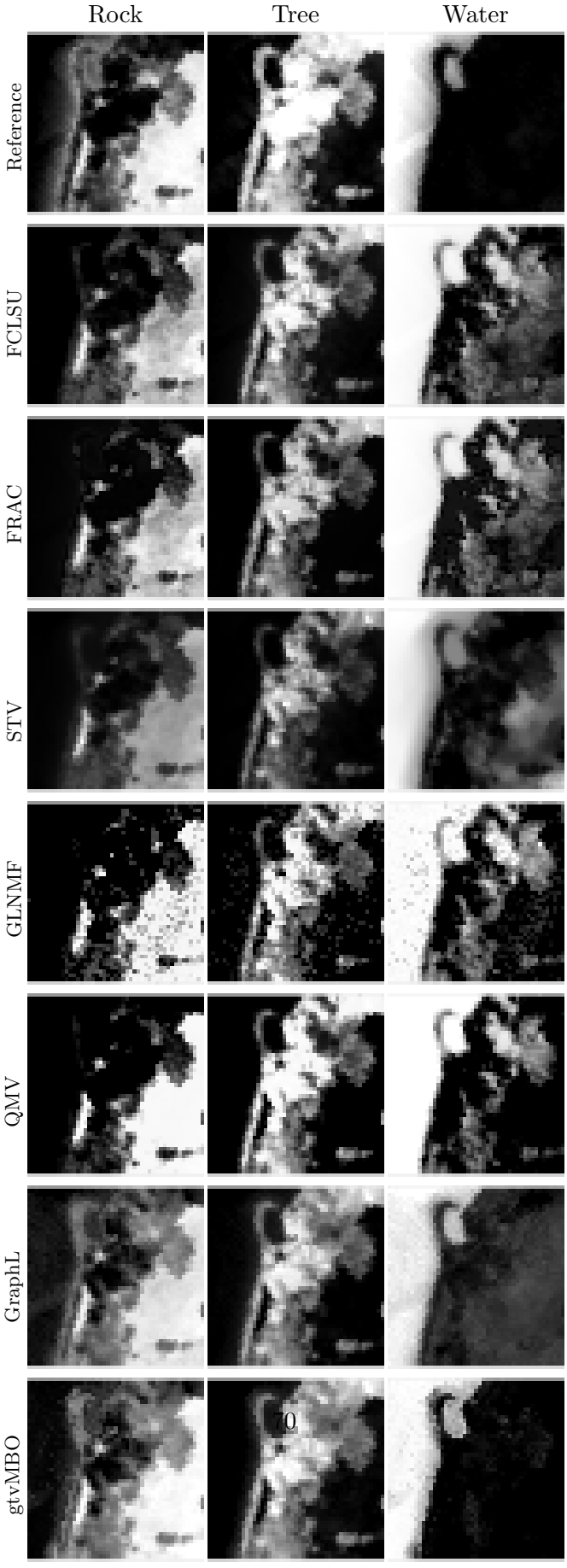


Figure 3.3: Abundance maps ($\boldsymbol{A}$) of the Samson dataset.

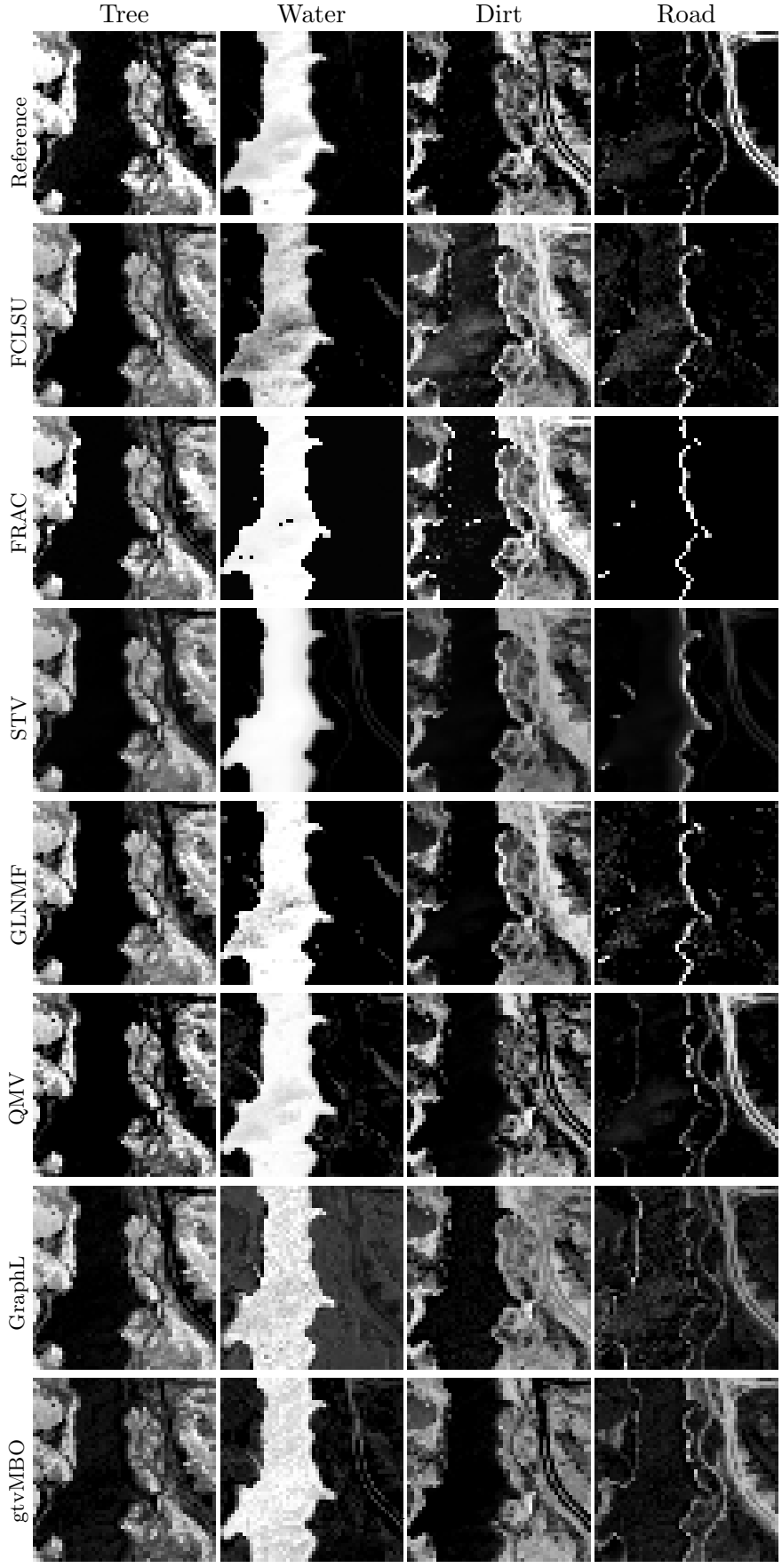


Figure 3.4: Abundance maps ($\boldsymbol{A}$) of the Jasper Ridge dataset.

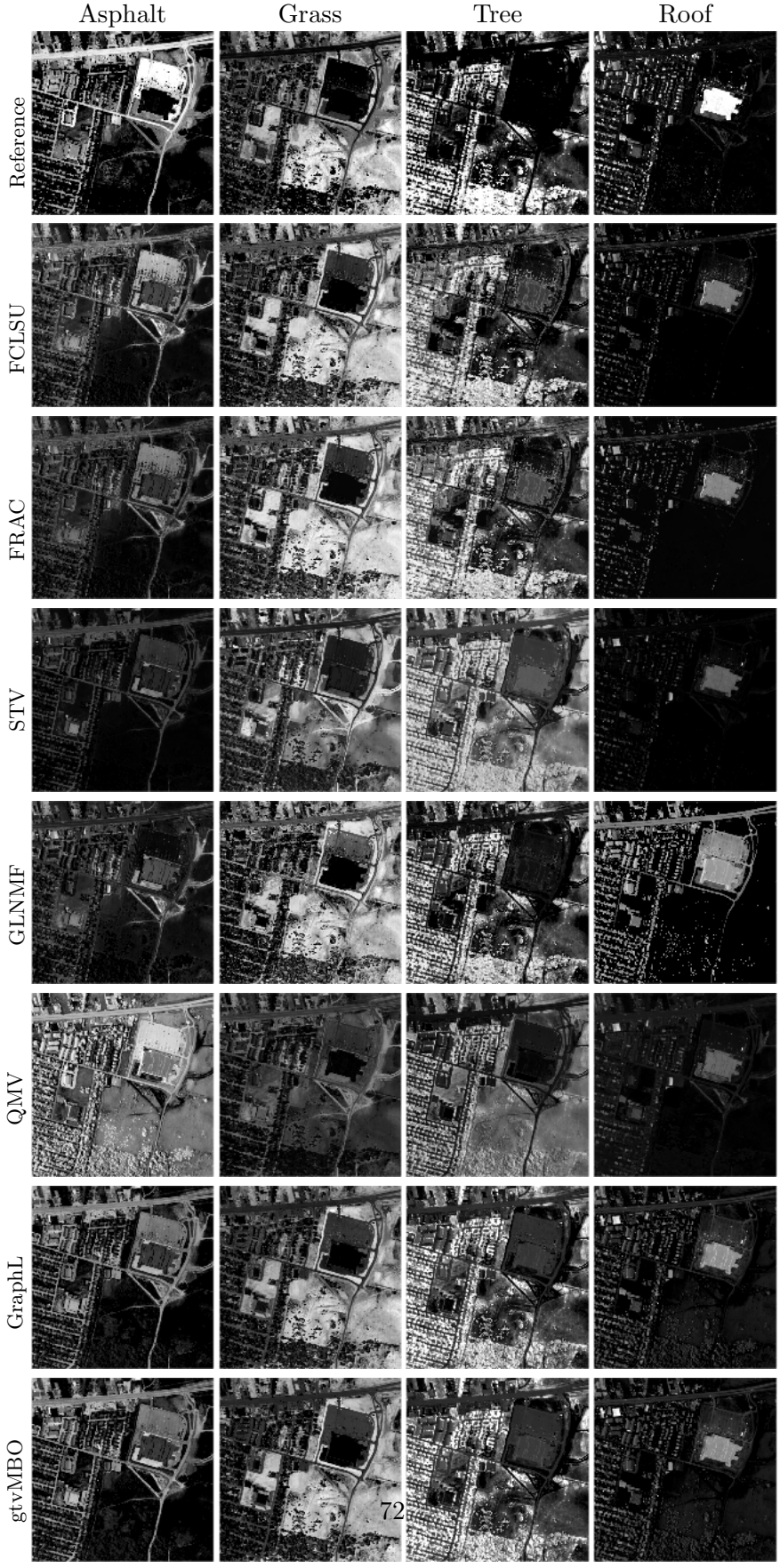


Figure 3.5: Abundance maps ($\boldsymbol{A}$) of the Urban dataset.

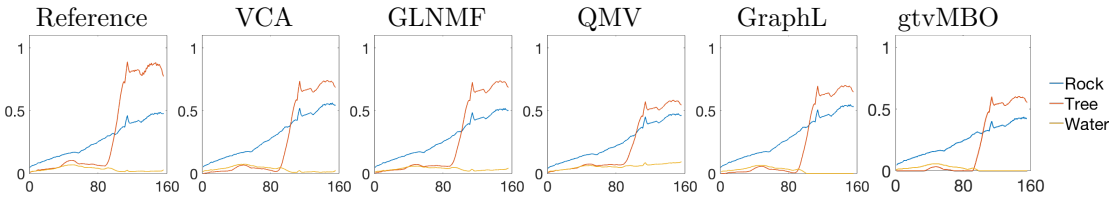


Figure 3.6: Endmember profiles ($\boldsymbol{S}$) of the Samson dataset.

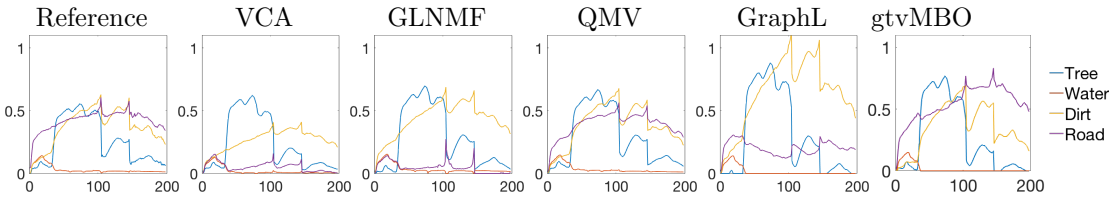


Figure 3.7: Endmember profiles ($\boldsymbol{S}$) of the Jasper Ridge dataset.

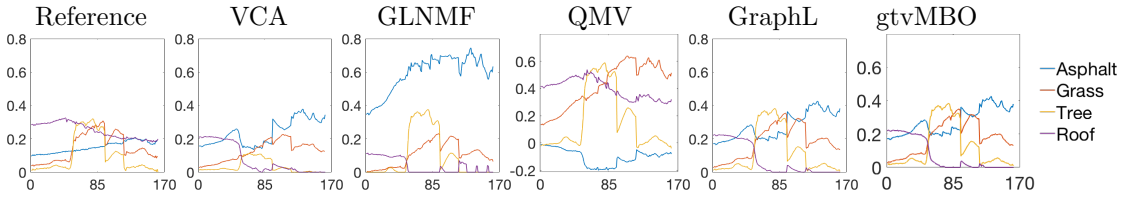


Figure 3.8: Endmember profiles ($\boldsymbol{S}$) of the Urban dataset.

Chapter 4

Distributed Multitask Learning

4.1 Summary

This chapter investigates mulitiask learning (MTL), where multiple learning tasks are performed together rather than separately to leverage their similarities. We focus on the distributed setting, where sharing the full local data between machines is prohibited. Motivated by graph regularization, we propose a novel fusion framework that only requires a one-shot communication of local estimates. Our method linearly combines the local estimates to produce an improved estimate for each task, and the ideal mixing weight for fusion is a function of task similarity and task difficulty. Practical algorithms for multitask linear regression and principal components analysis (PCA) are developed and are shown to significantly reduce mean squared error (MSE) on simulated data.

4.2 Introduction

A learning task rarely exists alone in the void; one often has to solve other related tasks as well. Instead of solving them independently, multitask learning tackles these related tasks together to take advantage of their similarities while respecting their differences. Furthermore, if they have varying levels of difficulty, e.g. sample complexity, it would be advantageous for the harder problem to borrow information from the easier problem. We describe below a few motivating examples where multitask learning can help.

Example 1 (Healthcare System)**.** *A major metropolitan city has multiple neighborhoods that are potentially segregated by race and socioeconomic class. Multiple hospitals, spread across the city, are each building a risk model for predicting heart attack based on their in-house data. Despite the common goal, their optimal risk models may not be identical, since these hospitals serve different subpopulations of the area.*

Example 2 (Housing Price Prediction)**.** *Housing price models reflect people's priority, and therefore cannot be identical across neighborhoods, cities, and countries. For instance, distance to the closest elementary school might majorly affect the housing price at an area for young families, but it will not weigh as much in retirement towns.*

Example 3 (Precision Medicine)**.** *This is a more granular version of Example 1. Traditional machine learning in medicine aims to fit a model that works well for the average target population. However, the emerging field of precision medicine [131] moves away from one-size-fits-all solutions and designs personalized treatment from genetic information.*

Example 4 (Word Prediction in Mobile Phones)**.** *As the user types in a word, the phone should predict and display what the next word might be. For n phone users, there are n extremely personalized datasets, but they share commonalities that can be exploited. Federated learning [132] is a popular edge computing method used for this problem, with some additional constraints such as unreliable hardware.*

In the examples above, we attempt to solve different, yet closely related learning tasks. It is thus natural to hypothesize that some sort of collaboration will help everyone get better estimates. A naive approach would be to give all n sets of full data to one machine for centralized processing. However, this is discouraged or simply impossible, due to HIPAA, privacy concerns, ownership, communication cost, or storage constraints. Therefore, this work focuses on *privacy-preserving distributed multitask learning*. We consider a scenario of

n machines, where the ith machine observes the ith local dataset $\boldsymbol{X}_i$, and $\boldsymbol{X}_i$ in its original form cannot be shared between the machines. Our goal is then to share information from $\{\boldsymbol{X}_i\}_{i=1}^n$ in a meaningful and feasible way such that we can successfully estimate the ground truth signals $\{\boldsymbol{\beta}_i^\star\}_{i=1}^n$.

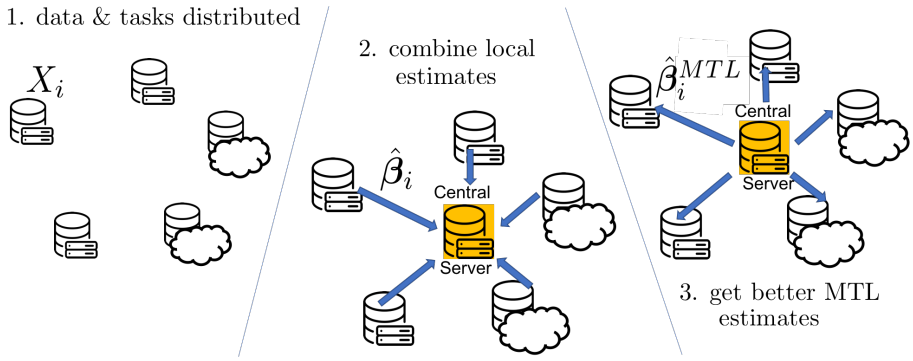


Figure 4.1: Outline of the proposed fusion method. Note that 1) it shares the local estimates and not the full data, and 2) there is only one communication round between the machines.

Our proposed fusion approach, motivated by graph regularization, is summarized in Fig. 4.1. First, n datasets $\{\boldsymbol{X}_i\}_{i=1}^n$ are distributed in a network of n machines. For $i = 1, \ldots, n$, the ith machine calculates a local estimate $\hat{\boldsymbol{\beta}}_i$ and sends it to the central server. The central server then *linearly combines* the local estimates $\{\hat{\boldsymbol{\beta}}_i\}_{i=1}^n$ according to a mixing matrix, and produces the improved multitask learning estimates $\left\{\hat{\boldsymbol{\beta}}_i^{MTL}\right\}_{i=1}^n$, i.e. $\hat{\boldsymbol{\beta}}_i^{MTL} = \sum_{j=1}^n \mathcal{W}_{ij}\hat{\boldsymbol{\beta}}_j$ for some matrix $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{n\times n}$. This approach addresses the aforementioned concerns regarding data sharing, and only calls for a one-shot communication between the machines.

At the heart of this fusion approach is the mixing matrix $\boldsymbol{\mathcal{W}}$. We show that the optimal mixing matrix for MSE reduction depends on exactly three quantities: similarity between the local estimates, how close a given local estimate is to the n different ground truth signals, and the variances of local estimates. Qualitatively, the first two measures relate to task similarity, and the last measure correlates to task difficulty such as noise level

and sample complexity. Additionally, if the local estimates are unbiased, then the first two quantities become exactly $\langle \boldsymbol{\beta}_i^\star, \boldsymbol{\beta}_j^\star \rangle$.

The appeal of our framework lies in its generality and simplicity, explained below:

- It is not limited to a particular local estimate, e.g. ordinary least squares, or even a particular task, e.g. linear regression. For multitask linear regression, we show that under very mild assumptions on the noise (zero-mean, additive, independent between tasks), the expression for ideal mixing matrix holds for any linear estimator $\hat{\boldsymbol{\beta}}_i$. For multitask PCA, we show that this framework applies to fusion of sample covariance matrices.

- It does not rely on any assumptions on the ground truth signal $\boldsymbol{\beta}^\star$. However, it has the potential to be specialized as appropriate for each application. For example, assuming a generative model with fewer parameters for $\boldsymbol{\beta}^\star$, e.g. random-effects model as in [133], will simplify estimation of the mixing matrix.

- It unifies all averaging-based methods in distributed (consensus) learning literature as a special case of distributed multitask learning with identical tasks.

- It is straightforward. Only one round of communication is needed, and the concept of taking a linear combination is easy to understand. No tricky hyper-parameters are introduced. In fact, not much has changed from a single machine's point of view. The fusion step is a layer added on top of an existing network of machines— a "post-hoc" algorithm for the system.

In the following sections, we motivate, define, and demonstrate the framework on multitask linear regression (Section 4.4) and multitask PCA (Section 4.5).

4.3 Related Work and Connections

Taking a linear combination of local estimates is not a new idea in distributed learning. However, existing literature focus on reaching a consensus, which is a special case of our distributed multitask learning setup. They also tend to force the weights to sum to 1 (i.e. weighted average) [134], or simply take the naive average [135, 136, 137]. [133] eliminates the unity constraint, but the scope is still limited to distributed consensus, ridge regression local estimates, and assumes random-effects model for $\boldsymbol{\beta}^\star$, which is integral to their analysis based on random matrix theory. Model interpolation for personalized federated learning in [138] is also related to our method, but the local machine only takes a weighted average of a central model and its own model.

Another popular body of work for multitask learning is shared architecture or shared representation learning [139, 140, 141]. These approaches require joint optimization, and are fundamentally different from our "post-hoc" method. Graph regularization methods also fall under this category [142, 143, 144, 145, 146, 147]. While we use graph regularization to motivate the fusion approach, the resulting method diverges significantly.

Meta-learning-based methods differ from ours in that they assume sequential learning. Their origins are closer to transfer learning, where model from the source task is used to initialize the target task. Similarly, [148] focuses on finding a good initialization model—a central model that can go through a few gradient updates at the local machines.

4.4 Multitask Linear Regression

We define the multitask linear regression problem as follows. Imagine a system of n machines, each trying to solve a similar linear regression problem. In other words, at each

machine $i = 1, \ldots, n$,

$$\boldsymbol{x}_i = \boldsymbol{A}_i \boldsymbol{\beta}_i^\star + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma_i^2 \boldsymbol{I}_m) \tag{4.1}$$

where observation signal $\boldsymbol{x}_i$ and noise $\boldsymbol{\epsilon}_i \in \mathbb{R}^m$, ground truth signal or model $\boldsymbol{\beta}_i^\star \in \mathbb{R}^d$, and the sensing matrix $\boldsymbol{A}_i \in \mathbb{R}^{m \times d}$. Noise level $\sigma_i > 0$, and $\boldsymbol{\epsilon}_i, \boldsymbol{\epsilon}_j$ are independent for $i \neq j$. m can be different for each task, i.e. can be written as m_i, but we assume $m_1 = m_2 = \ldots, = m_n$ for simplicity. The goal of linear regression is to estimate $\boldsymbol{\beta}_i^\star$ given $\boldsymbol{x}_i$ and $\boldsymbol{A}_i$, which is arguably one of the most fundamental problems in signal processing and machine learning.

In Section 4.4.1, we apply graph regularization to multitask linear regression and inspect the resultant solutions. Then, inspired by our findings, we propose a fusion framework for multitask linear regression in Section 4.4.2, and test them with simulation experiments in Section 4.4.4. Table 4.1 summarizes some key notations used in this section for convenience.

Symbol	**Description**	**Dimension**
$\boldsymbol{\beta}_i^\star$	Ground truth signal at ith machine	d
$\boldsymbol{A}_i$	Sensing matrix for $\boldsymbol{\beta}_i^\star$	$m \times d$
$\boldsymbol{x}_i$	Observation at ith machine	m
$\hat{\boldsymbol{\beta}}_i$	Local estimate at ith machine	d
$\mathbb{E}\hat{\boldsymbol{\beta}}_i$	Expectation of $\hat{\boldsymbol{\beta}}_i$ w.r.t. data $\boldsymbol{x}_i$	d
$\boldsymbol{\mathcal{W}}$	Mixing matrix for fusion step	$n \times n$
$\hat{\boldsymbol{\beta}}_i^{MTL}$	The ith MTL estimate $\sum_{j=1}^n \mathcal{W}_{ij} \hat{\boldsymbol{\beta}}_j$	d

Table 4.1: Key notations used in Section 4.4.

4.4.1 Motivation via Graph Regularization

Graph regularization is an intuitive approach to multitask learning as it can easily integrate the task relationship information into the problem formulation [149]. There is also a clear

connection between graphs and communication networks. Let us assume that we have access to (or derived) the similarity information between n tasks as an adjacency matrix $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ with no self-loops, where $\sum_{j=1}^{n} W_{ij} = 1$, $W_{ii} = 0$, and $W_{ij} \geq 0$ for all i, j. Then, given some regularization parameter $\lambda > 0$, the graph-regularized multitask learning problem solves for

$$(\hat{\boldsymbol{\beta}}_1^{\lambda}, \ldots, \hat{\boldsymbol{\beta}}_n^{\lambda}) = \underset{\boldsymbol{\beta}_i}{\operatorname{argmin}} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{A}_i\boldsymbol{\beta}_i\|_2^2 + \lambda \sum_{i,j} W_{ij}\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\|_2^2. \tag{4.2}$$

In this subsection, assume $\boldsymbol{A}_i$ is tall and orthogonal, i.e. $m \geq d$ and $\boldsymbol{A}_i^{\top}\boldsymbol{A}_i = \boldsymbol{I}$, for easier derivation. Then somewhat surprisingly, Theorem 4 states that $\hat{\boldsymbol{\beta}}_i^{\lambda}$s are in fact *convex combinations* of local ordinary least squares (OLS) estimates. In other words, there are two ways to obtain these multitask learning solutions under this setting. On one hand, we can solve (4.2) which uses the classical optimization-based graph regularization framework. On the other hand, we can arrive at the same answers by taking convex combinations of local OLS estimates, which no longer requires data sharing. This new perspective on graph regularization motivates a general privacy-preserving approach to distributed multitask learning.

Theorem 4 (Graph-regularized Multitask Linear Regression: Solutions)**.** *In addition to the observation model (4.1), assume $m \geq d$, $\boldsymbol{A}_i^{\top}\boldsymbol{A}_i = \boldsymbol{I}_d$, and $\boldsymbol{W}$ is a right stochastic matrix with no self-loops, i.e. $\sum_{j=1}^{n} W_{ij} = 1$, $W_{ii} = 0$, and $W_{ij} \geq 0$ for all i, j. Denote the local OLS estimate*

$$\hat{\boldsymbol{\beta}}_i^{OLS} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{x}_i - \boldsymbol{A}_i\boldsymbol{\beta}\|_2^2 = \left(\boldsymbol{A}_i^{\top}\boldsymbol{A}_i\right)^{-1}\boldsymbol{A}_i^{\top}\boldsymbol{x}_i = \boldsymbol{A}_i^{\top}\boldsymbol{x}_i. \tag{4.3}$$

Then $\{\hat{\boldsymbol{\beta}}_i^{\lambda}\}_{i=1}^{n}$, the minimizers of graph-regularized linear regression (4.2) for $\lambda > 0$, are

convex combinations of local OLS estimates $\{\hat{\boldsymbol{\beta}}_i^{OLS}\}_{i=1}^n$ (4.3). *More precisely,*

$$\hat{\boldsymbol{\beta}}_i^{\lambda} = \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda}\ \hat{\boldsymbol{\beta}}_j^{OLS} \tag{4.4}$$

for mixing matrix $\boldsymbol{\mathcal{W}}^{\lambda} \in \mathbb{R}^{n\times n}$*, which is a right stochastic matrix and defined as*

$$\boldsymbol{\mathcal{W}}^{\lambda} = \frac{1}{\lambda+1}\left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1}\boldsymbol{W}\right)^{-1} = \frac{1}{\lambda+1}\sum_{k=0}^{\infty}\left(1-\frac{1}{\lambda+1}\right)^k \boldsymbol{W}^k. \tag{4.5}$$

Proof of Theorem 4 is deferred to Appendix C.1.

It is worth discussing how the convex combination relationship described in (4.5) matches with our intuitions from the original optimization problem (4.2). Since $\boldsymbol{W}$ is an adjacency matrix whose rows sum to 1, we can also consider $\boldsymbol{W}$ to be a transition matrix: that is, an agent moves from node i to node j with W_{ij} probability. Then, $\left[\boldsymbol{W}^k\right]_{ij}$ is the probability that an agent starts at node i and end at node j in k hops. If nodes i and node j are well-connected, it should be easier to reach j from i, so $\hat{\boldsymbol{\beta}}_j^{OLS}$ should have a larger weight on $\hat{\boldsymbol{\beta}}_i^{\lambda}$. This is consistent with what we expect to see with large W_{ij} in (4.2). λ comes into play as the *discount factor* $\frac{1}{\lambda+1}$. For example, if λ is small, information from $\boldsymbol{W}$ is ignored faster as k grows large, which is consistent with the behavior of regularization parameter in (4.2).

Building on the results of Theorem 4, Proposition 2 states a performance bound on multitask linear regression with graph regularization. Proof is deferred to Appendix C.2.

Proposition 2 (Graph-regularized Multitask Linear Regression: Performance)**.** *Assume Theorem 4. Then, the MSE of* $\hat{\boldsymbol{\beta}}_i^{\lambda}$ (4.4) *is bounded by*

$$\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i^{\lambda} - \boldsymbol{\beta}_i^{\star}\right\|_2^2 \le \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda}\left\|\boldsymbol{\beta}_i^{\star} - \boldsymbol{\beta}_j^{\star}\right\|_2^2 + d\sum_{j=1}^{n}(\mathcal{W}_{ij}^{\lambda}\sigma_j)^2. \tag{4.6}$$

The RHS of (4.6) has a nice form to analyze. It decomposes into a roughly bias^2 + variance format, which aligns with task dis-similarity $\|\boldsymbol{\beta}_i^\star - \boldsymbol{\beta}_j^\star\|_2^2$ + task difficulty $d\sigma_j^2$ in this multitask learning problem. Also, note the parallel between $\sum_{j=1}^n \mathcal{W}_{ij}^\lambda \left\|\boldsymbol{\beta}_i^\star - \boldsymbol{\beta}_j^\star\right\|_2^2$ and the graph regularization term $\sum_{j=1}^n W_{ij} \left\|\boldsymbol{\beta}_i - \boldsymbol{\beta}_j\right\|_2^2$ in (4.2). Similar to what we observed in denoising (Chapter 2), this suggests that graph regularization leads to better inference when the graph describes the ground truth task relationship more accurately.

4.4.2 Fusion of Linear Estimators: Proposed Framework

Theorem 4 and Proposition 2 suggest that linearly combining local estimates is a valid approach to combining information without combining data. Building on that intuition, we propose MTL estimates

$$\hat{\boldsymbol{\beta}}_i^{MTL} = \sum_{j=1}^n \mathcal{W}_{ij} \hat{\boldsymbol{\beta}}_j, \quad i = 1, \ldots, n \tag{4.7}$$

which are linear combinations of local estimates $\{\hat{\boldsymbol{\beta}}_j\}_{j=1}^n$ according to a mixing matrix $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{n \times n}$. Theorem 5 specifies the mixing matrix $\boldsymbol{\mathcal{W}}$ with maximum MSE reduction for any linear local estimates $\{\hat{\boldsymbol{\beta}}_j\}_{j=1}^n$, and Corollary 1 then specializes the result to unbiased linear estimates. Details are deferred to Appendix C.3, but the proof of Theorem 5 follows from directly minimizing the MSE of $\hat{\boldsymbol{\beta}}^{MTL}$ with respect to $\boldsymbol{\mathcal{W}}$.

Theorem 5 (Fusion of Linear Estimators for Multitask Linear Regression)**.** *Assume observation model (4.1). Let $\hat{\boldsymbol{\beta}}_i$ be any linear, potentially biased, local estimator of $\boldsymbol{\beta}_i^\star$, which has an expected value $\mathbb{E}\hat{\boldsymbol{\beta}}_i$ and variance $\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2$[1]. For $\hat{\boldsymbol{\beta}}_i^{MTL}$ as defined in (4.7), the MSE $\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i^{MTL} - \boldsymbol{\beta}_i^\star\right\|_2^2$ is minimized for all $i = 1, \ldots, n$ by the mixing matrix*

$$\boldsymbol{\mathcal{W}} = \boldsymbol{K}\left(\boldsymbol{C} + \boldsymbol{V}\right)^{-1}, \tag{4.8}$$

[1]Expectation is taken with respect to randomness in the ith dataset.

where

$$\boldsymbol{K} = \left[\langle \boldsymbol{\beta}_i^\star, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle\right]_{i,j}, \ \boldsymbol{C} = \left[\langle \mathbb{E}\hat{\boldsymbol{\beta}}_i, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle\right]_{i,j}, \ \boldsymbol{V} = diag\left(\left[\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2\right]_{i=1}^n\right).$$

Moreover, the fusion estimate $\hat{\boldsymbol{\beta}}_i^{MTL}$ *will always be at least as accurate as* $\hat{\boldsymbol{\beta}}_i$ *in terms of MSE. For each* $i = 1, \ldots, n$*,*

$$\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i^{MTL} - \boldsymbol{\beta}_i^\star\right\|_2^2 \leq \mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i^\star\right\|_2^2.$$

Theorem 5 states that the ideal mixing weights $\boldsymbol{\mathcal{W}}$ depend on task difficulties and task similarities. For one, $V_{ii} = \mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2$ is precisely the variance of the local estimate $\hat{\boldsymbol{\beta}}_i$. This captures information about the randomness in the ith dataset, e.g. noise level σ_i and number of samples m. Therefore we describe $\boldsymbol{V}$ as the *task difficulty* term. Meanwhile, we categorize $\boldsymbol{K}$ and $\boldsymbol{C}$ as *task similarity* terms. $C_{ij} = \langle \mathbb{E}\hat{\boldsymbol{\beta}}_i, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle$ is clearly proportional to the cosine similarity or angle between $\mathbb{E}\hat{\boldsymbol{\beta}}_i$ and $\mathbb{E}\hat{\boldsymbol{\beta}}_j$, and $K_{ij} = \langle \boldsymbol{\beta}_i^\star, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle$ is negatively correlated with the bias of $\hat{\boldsymbol{\beta}}_j$ with respect to $\boldsymbol{\beta}_i^\star$:

$$\text{bias}^2 = \|\mathbb{E}\hat{\boldsymbol{\beta}}_j - \boldsymbol{\beta}_i^\star\|_2^2 = \|\mathbb{E}\hat{\boldsymbol{\beta}}_j\|_2^2 + \|\boldsymbol{\beta}_i^\star\|_2^2 - 2\langle \boldsymbol{\beta}_i^\star, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle = \|\mathbb{E}\hat{\boldsymbol{\beta}}_j\|_2^2 + \|\boldsymbol{\beta}_i^\star\|_2^2 - 2K_{ij}.$$

In fact, Corollary 1 shows that $K_{ij} = C_{ij} = \langle \boldsymbol{\beta}_i^\star, \boldsymbol{\beta}_j^\star \rangle$ for unbiased local estimates.

Corollary 1 (Fusion of Linear Unbiased Estimators for Multitask Linear Regression)**.** *Assume observation model (4.1). Let* $\hat{\boldsymbol{\beta}}_i$ *be any linear, unbiased, local estimator of* $\boldsymbol{\beta}_i^\star$*, which has an expected value* $\boldsymbol{\beta}_i^\star$ *and variance* $\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i^\star\right\|_2^2$ *by definition. For* $\hat{\boldsymbol{\beta}}_i^{MTL}$ *as defined in (4.7), the MSE* $\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i^{MTL} - \boldsymbol{\beta}_i^\star\right\|_2^2$ *is minimized for all* $i = 1, \ldots, n$ *by the mixing matrix*

$$\boldsymbol{\mathcal{W}} = \boldsymbol{C}\left(\boldsymbol{C} + \boldsymbol{V}\right)^{-1},$$

where

$$\boldsymbol{C} = \left[\langle \boldsymbol{\beta}_i^\star, \boldsymbol{\beta}_j^\star \rangle\right]_{i,j}, \;\; \boldsymbol{V} = diag\left(\left[\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i^\star\right\|_2^2\right]_{i=1}^n\right).$$

$\boldsymbol{C}$ in Corollary 1 is exactly the cosine similarity matrix between the ground truth signals, while $\boldsymbol{V}$ remains the task difficulty term. If $\boldsymbol{C}$ is full rank and $\boldsymbol{V}$ is a zero matrix, then the ideal mixing weights $\boldsymbol{\mathcal{W}} = \boldsymbol{C}\boldsymbol{C}^{-1} = \boldsymbol{I}$. This matches with the noiseless— or zero task difficulty— case where the unbiased estimators should be able to recover the ground truth signals completely.

4.4.3 Fusion of Linear Estimators: Proposed Algorithms

Based on the results of Theorem 5, we designed One-Shot Fusion* (Alg. 3) that calculates the ideal mixing matrix $\boldsymbol{\mathcal{W}}$ and computes the MTL estimates accordingly.

Algorithm 3 One-Shot Fusion* for Mutitask Linear Regression

inputs local datasets $\{\boldsymbol{x}_i\}_{i=1}^n, \; \{\boldsymbol{A}_i\}_{i=1}^n$.
Each machine *locally* calculates a linear estimate $\hat{\boldsymbol{\beta}}_i$ from $\boldsymbol{x}_i$ and $\boldsymbol{A}_i$.
Each machine *locally* estimates $\mathbb{E}\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\|_2^2$, e.g. by bootstrapping.
$\boldsymbol{V} \leftarrow diag\left(\left[\mathbb{E}\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\|_2^2\right]_{i=1}^n\right)$
$\boldsymbol{C} \leftarrow \left[\langle \mathbb{E}\hat{\boldsymbol{\beta}}_i, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle\right]_{i,j}$
$\boldsymbol{K} \leftarrow \left[\langle \boldsymbol{\beta}_i^\star, \mathbb{E}\hat{\boldsymbol{\beta}}_j \rangle\right]_{i,j}$
$\boldsymbol{\mathcal{W}} \leftarrow \boldsymbol{K}\left(\boldsymbol{C} + \boldsymbol{V}\right)^{-1}$
$\hat{\boldsymbol{\beta}}_i^{MTL} \leftarrow \sum_{j=1}^n \mathcal{W}_{ij}\hat{\boldsymbol{\beta}}_j$
outputs MTL estimates $\{\hat{\boldsymbol{\beta}}_i^{MTL}\}_{i=1}^n$.

This algorithm avoids concerns about data sharing by compressing the necessary information from each dataset into $\hat{\boldsymbol{\beta}}_i$ and V_{ii}. For example, if we have tall and orthogonal $\boldsymbol{A}_i$s and decide to combine OLS local estimates, then each machine will estimate σ_i from their dataset, and pass $d\sigma_i^2$ with $\hat{\boldsymbol{\beta}}_i$ to the central server; see Appendix C.4 for more

examples. Note that the communication between the local machines and the central server, as well as the fusion step in the central server, is limited to one round in this algorithm.

However, an obvious shortcoming of One-Shot Fusion* (Alg. 3) is that calculating $\boldsymbol{K}$ requires access to the inner product with the ground truth signal, i.e. $\langle \boldsymbol{\beta}_i^\star, \cdot \rangle$, which is impossible to implement in real life. Therefore, we adopt an iterative approach to address this issue in Iterative Fusion (Alg. 4).

Algorithm 4 Iterative Fusion for Multitask Linear Regression (Practical Version of Alg. 3)

inputs local datasets $\{\boldsymbol{x}_i\}_{i=1}^n$, $\{\boldsymbol{A}_i\}_{i=1}^n$.
Each machine *locally* calculates a linear estimate $\hat{\boldsymbol{\beta}}_i$ from $\boldsymbol{x}_i$ and $\boldsymbol{A}_i$, such that $\mathbb{E}\hat{\boldsymbol{\beta}}_i = \boldsymbol{M}_i\boldsymbol{\beta}_i^\star$ for some matrix $\boldsymbol{M}_i$, e.g. $\boldsymbol{M}_i = \boldsymbol{I}$ for unbiased estimates.
Each machine *locally* estimates $\mathbb{E}\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\|_2^2$, e.g. by bootstrapping.
$\boldsymbol{V} \leftarrow diag\left(\left[\mathbb{E}\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\|_2^2\right]_{i=1}^n\right)$
repeat
    $\boldsymbol{C} \leftarrow \left[\langle \boldsymbol{M}_i\hat{\boldsymbol{\beta}}_i, \boldsymbol{M}_j\hat{\boldsymbol{\beta}}_j\rangle\right]_{i,j}$
    $\boldsymbol{K} \leftarrow \left[\langle \hat{\boldsymbol{\beta}}_i, \boldsymbol{M}_j\hat{\boldsymbol{\beta}}_j\rangle\right]_{i,j}$
    $\boldsymbol{\mathcal{W}} \leftarrow \boldsymbol{K}\left(\boldsymbol{C} + \boldsymbol{V}\right)^{-1}$
    (Optional) Threshold elements of $\boldsymbol{\mathcal{W}}$.
    $\hat{\boldsymbol{\beta}}_i \leftarrow \sum_{j=1}^n \mathcal{W}_{ij}\hat{\boldsymbol{\beta}}_j$.
until termination
(Optional) Project $\hat{\boldsymbol{\beta}}_i$ onto a constraint set.
$\hat{\boldsymbol{\beta}}_i^{MTL} \leftarrow \hat{\boldsymbol{\beta}}_i$.
outputs MTL estimates $\{\hat{\boldsymbol{\beta}}_i^{MTL}\}_{i=1}^n$.

Iterative Fusion (Alg. 4) differs from One-Shot Fusion* (Alg. 3) in three points. First, since we do not have access to $\langle \boldsymbol{\beta}^\star, \cdot \rangle$ nor the true $\langle \mathbb{E}\hat{\boldsymbol{\beta}}, \cdot \rangle$ in real life, we approximate $\boldsymbol{\beta}_i^\star \approx \hat{\boldsymbol{\beta}}_i$, and $\mathbb{E}\hat{\boldsymbol{\beta}}_i = \boldsymbol{M}_i\boldsymbol{\beta}_i^\star \approx \boldsymbol{M}_i\hat{\boldsymbol{\beta}}_i$ to calculate the key matrices $\boldsymbol{C}$ and $\boldsymbol{K}$ for $\boldsymbol{\mathcal{W}}$. Note that such $\boldsymbol{M}_i$ must exist since we are only considering linear estimates for this framework and assume zero mean and additive noise. Secondly, the communication between local machines and the central server is still one-shot, but the fusion step at the central server

alternates between updating the weights $\boldsymbol{\mathcal{W}}$ and the local estimates $\{\hat{\boldsymbol{\beta}}_i\}_{i=1}^n$. The number of iterations for the fusion step can be fixed[2], or chosen for the specific dataset via cross validation. Lastly, we added optional steps to threshold the elements of $\boldsymbol{\mathcal{W}}$ or project the final MTL outputs onto a constraint set.

Remark 2. *We emphasize that while Alg. 4 is designed for the most general case, where we don't have additional information besides $\boldsymbol{x}_i$ and $\boldsymbol{A}_i$, it has the potential to be specialized for each application and problem model as needed. In fact, linear combination of ridge regression estimators proposed in [133] can be considered a special case of our framework. Specifically, [133] assumes a random-effects model[3] on the ground truth signal, and also makes certain random-matrix-theoretic assumptions on the sensing matrix $\boldsymbol{A}_i$. These assumptions allow them to estimate the key matrices in Alg. 4 in the asymptotic regime.*

4.4.4 Simulation Experiments

We test our algorithms by combining local OLS estimates according to our mixing matrix. We simulate tall and orthogonal sensing matrix $\boldsymbol{A}_i \in \mathbb{R}^{m\times d}$ by sampling each element from $\mathcal{N}(0,1)$ i.i.d, and orthogonalizing the matrix. For convenience, we fix $m = d = 20$ unless otherwise specified, and gather observation data via (4.1). The remaining variables $\boldsymbol{\beta}_i^\star$, σ_i, and n are determined as follows for different experiments.

Central model This model assumes that all tasks are similar to each other by roughly the same degree. More concretely put,

$$\boldsymbol{\beta}_i^\star \sim \mathcal{N}(\boldsymbol{\beta}^\star, \sigma_\star^2 \mathbf{I}_d), \quad \boldsymbol{\beta}^\star \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d), \quad \sigma = \sigma_1 = \sigma_2 = \ldots, = \sigma_n \tag{4.9}$$

[2]Simulations suggest that maximum MSE reduction is achieved in less than 5 iterations of the algorithm.

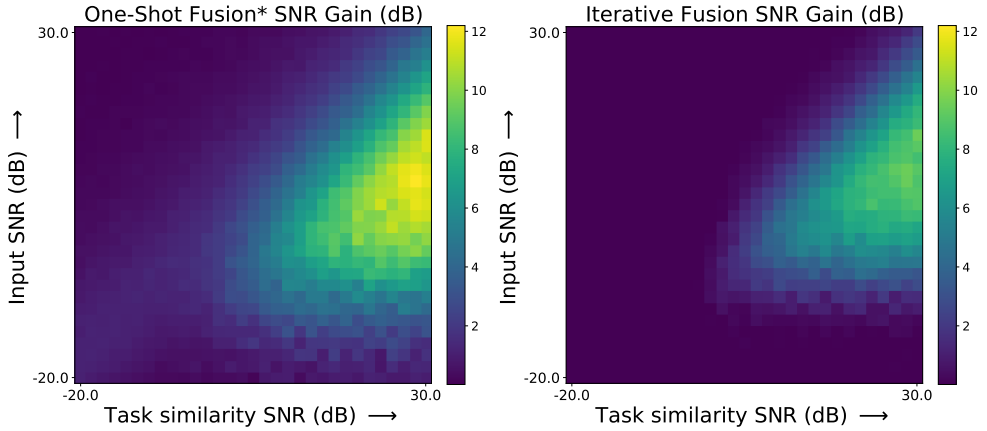
[3]Each coordinate of $\boldsymbol{\beta}_i^\star$ is a random variable of zero-mean and $\sigma_i^2\alpha^2/d$ variance. α^2 is some scalar variable that represents signal-to-noise ratio. These coordinates are independent from each other and noise.

such that ground truth task similiarity is uniformly determined by $\sigma_\star$, and task difficulty σ is identical across i. Note that $\sigma_\star = 0$ reduces the model to a distributed consensus scenario. We define *task similarity SNR* $= 10\log_{10}(\|\boldsymbol{\beta}^\star\|_2^2/d{\sigma_\star}^2) \approx -20\log_{10}(\sigma_\star)$ dB, noisy input SNR $= 10\log_{10}(\|\boldsymbol{\beta}^\star\|_2^2/d\sigma^2) \approx -20\log_{10}(\sigma)$ dB, output MSE $= \frac{1}{nd}\sum_{i=1}^n \|\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i^\star\|^2$, and output SNR $= -10\log_{10}$(output MSE) dB.

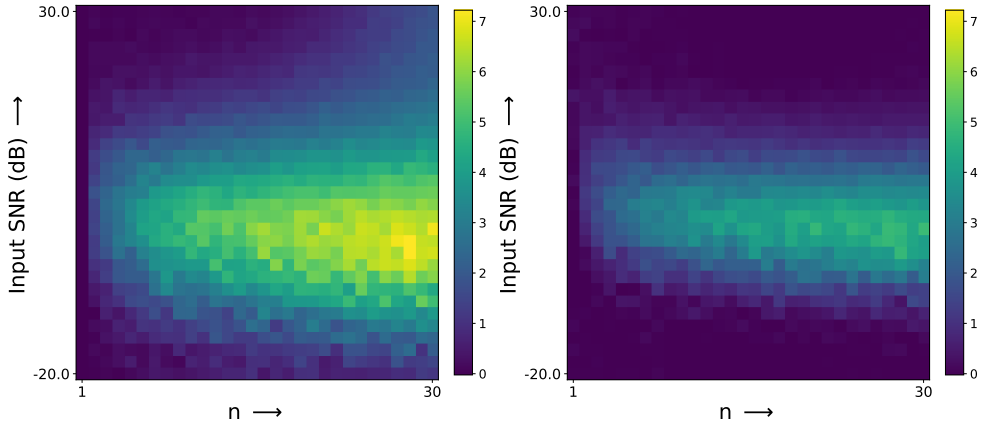
To understand the effect of task similarity SNR, input SNR, and n on the proposed method, we varied $\sigma_\star \in \{10^{-1.5}, \dots 10^1\}, \sigma_i \in \{10^{-1.5}, \dots 10^1\}$, and $n \in \{1, \dots, 30\}$. Then under each set of parameters, we simulated data under the central model, combined *local OLS estimates* via Algs. 3 and 4, and solved *ridge regression* for each local dataset with optimal λ_i (C.2) for comparison. In particular, Iterative Fusion (Alg. 4) is run for 10 steps, and the lowest MSE in hindsight is reported as the output MSE.

The SNR gain, averaged over 10 trials, is summarized as phase transition diagrams in Fig. 4.2. We stress that while our methods combined the local OLS estimates, the output SNRs of our methods are compared against the output SNRs of the *optimal local* ridge regression estimates: when compared to the local OLS estimates, SNR gain is even more substantial. Furthermore, these results suggest that fusion algorithms are beneficial when

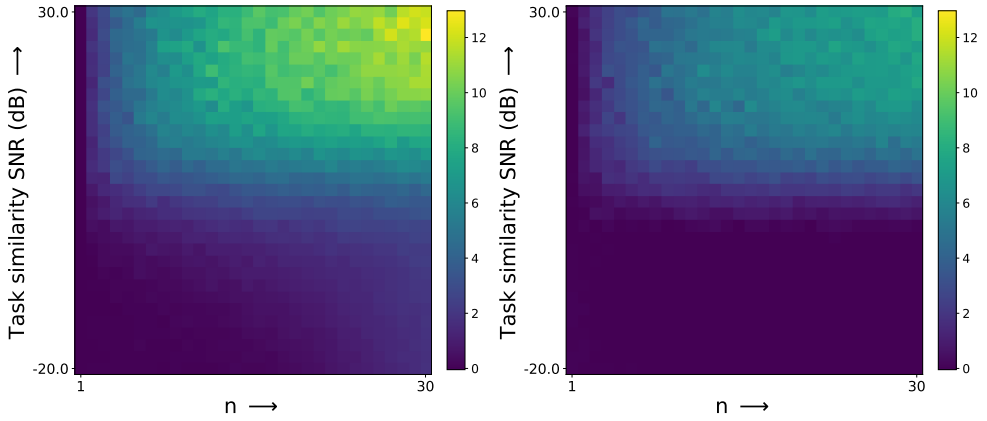
- ground truth signals are more similar to each other than they are to noise (task similarity SNR $\geq$ input SNR);
- tasks are more similar to each other (task similarity SNR $\uparrow$);
- tasks are not too difficult that there is no useful information to be borrowed from others (mid-to-low range input SNR);
- tasks are not too easy that there is *no need to* borrow information from others (mid-to-low range input SNR);
- there are more tasks ($n \uparrow$).



(a) Fusion helps when tasks are more similar to each other (task similarity SNR ↑), and if ground truth signals are more similar to each other than they are to noise (task similarity SNR $\geq$ input SNR). $n = d = 20$.



(b) Fusion helps when tasks are hard enough that collaboration helps, but not too hard that there are no useful information to share (mid-to-low input SNR). Task similarity SNR = 10dB, $\sigma_\star = \sqrt{0.1}$.



(c) Fusion helps when there are more tasks (n ↑). Input SNR = 0dB, $\sigma_i = 1$.

Figure 4.2: SNR gain compared to the *optimal* local ridge regression estimates. Data are simulated under the central model (4.9). These phase diagrams visualize the regimes where the fusion method is effective.

Star player model Building on the central model, we imagine a situation where the first task has significantly more information than the rest of the tasks:

$$\boldsymbol{\beta}_i^\star \sim \mathcal{N}(\boldsymbol{\beta}^\star, \sigma_\star^2 \mathbf{I}_d), \quad \boldsymbol{\beta}^\star \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d), \quad \sigma = 10\sigma_1 = \sigma_2 = \ldots, = \sigma_n. \tag{4.10}$$

For fair comparison, output MSE is updated to $\frac{1}{(n-1)d} \sum_{i=2}^n \|\hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i^\star\|^2$. As expected, Fig. 4.3 confirms that other tasks take advantage of the "star player" during fusion, indicated by the high values in the first column. Fig. 4.4 shows that such strategy leads to reduction in MSE for the other tasks.

Community model In this experiment, we impose a community structure on the tasks. The n tasks are divided into 3 groups (20%, 30%, and 50%), and within each group, data are simulated under the central model (4.9). It is notable that the community structure is captured by the mixing weights in Fig. 4.3, since the algorithm does not make that assumption a priori, nor take in parameters such as the number of clusters. Fig. 4.6 demonstrates the successful MSE reduction by the fusion algorithms.

4.5 Multitask Principal Components Analysis (PCA)

In this section, we tackle multitask PCA, which is a different problem from multitask linear regression. We start by briefly discussing (single-task) PCA and the task of finding the subspace of a single dataset. Consider a zero-mean random variable $\boldsymbol{x} \in \mathbb{R}^d$ drawn from an unknown but fixed distribution D. Given m i.i.d. observations of $\boldsymbol{x}$, $\{\boldsymbol{x}_t\}_{t=1}^m$, we are interested in finding its underlying k-dimensional subspace S, where k is assumed to be known and fixed, and $k < d$. Note that a subspace S can be represented by any orthogonal matrix $\boldsymbol{U}$ such that $\text{span}(\boldsymbol{U}) = S$.

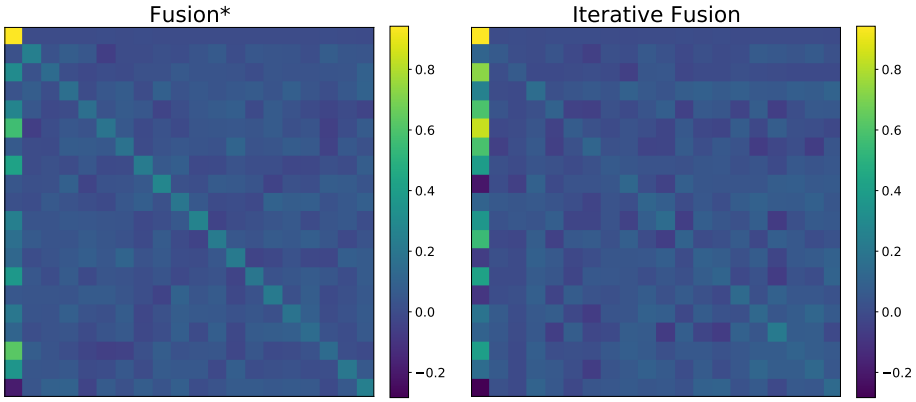


Figure 4.3: The high values in the first column indicate everyone's reliance on the first task (i.e. the "star player"), which is significantly less noisy by design. Mixing weights produced by fusion algorithms under the star player model. MSE of fusion estimates is 0.2 and 0.35, respectively.

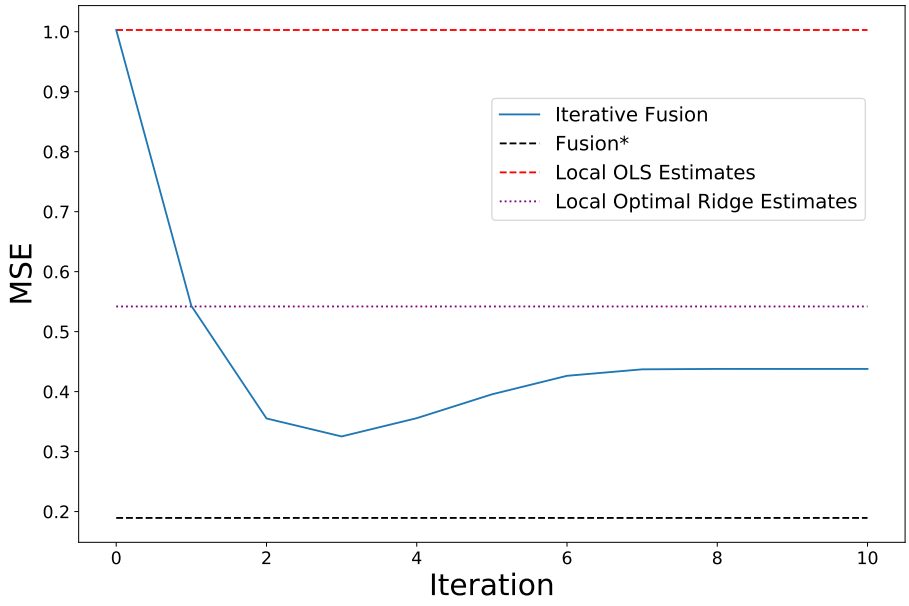


Figure 4.4: Combining OLS estimates can yield better estimates than local optimal ridge regression estimates. MSE reduction by fusion algorithms under the star player model (4.10). $\sigma^\star = 0.5, \sigma = 1, \sigma_1 = 0.1, n = d = 20$. Averaged over 50 trials.

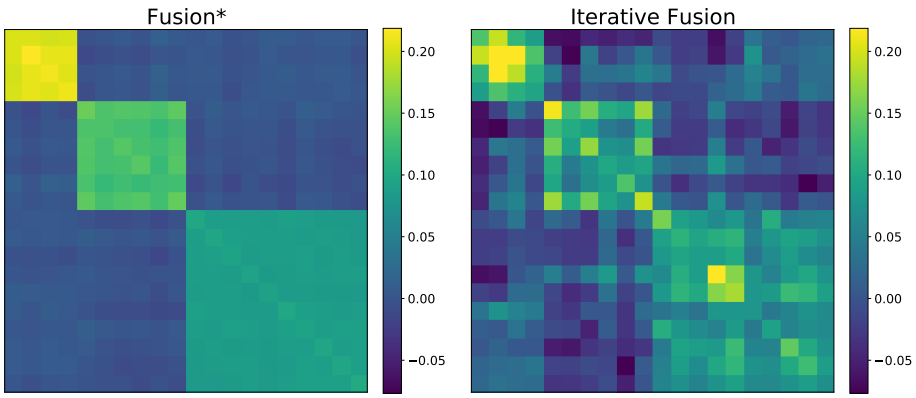


Figure 4.5: Mixing weights produced by fusion algorithms under the community model recover the community structure among the tasks, which were not provided a priori. MSE of fusion estimates is 0.14 and 0.25, respectively.

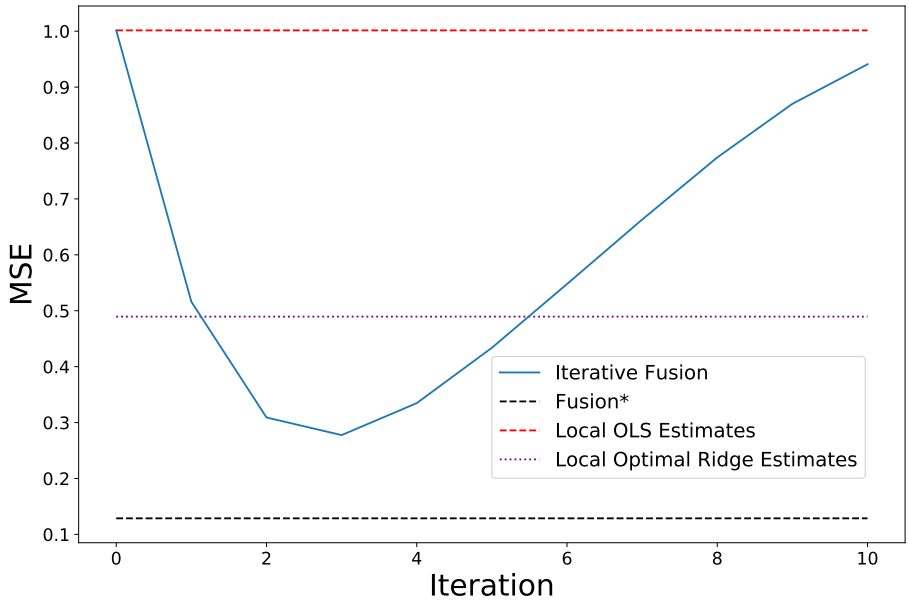


Figure 4.6: Combining OLS estimates can yield better estimates than local optimal ridge regression estimates. MSE reduction by fusion algorithms under the community model. $\sigma^\star = 0.1, \sigma = 1, n = d = 20$. Averaged over 50 trials.

PCA is a celebrated method for finding subspaces under these assumptions. It is an invaluable tool in many applications that require dimensionality reduction, including data visualization, image processing and gene expression analysis. It finds a subspace that maximizes the variation of the projected data samples, or equivalently, minimizes the projection error.

We start by defining multitask PCA and formulating it in the projection matrices space in Section 4.5.1. We then apply graph regularization to multitask PCA in Section 4.5.2, and use insights from that process to propose a fusion framework in Section 4.5.3. Finally, we show some simulation experiment results in Section 4.5.4. Table 4.2 summarizes some key notations used in this section for convenience.

Symbol	Description	Dimension
$\boldsymbol{U}_i^\star$	Ground truth orthogonal matrix at ith machine	$d \times k$
$\boldsymbol{X}_i$	Observation at ith machine	$d \times m$
$\hat{\boldsymbol{\Sigma}}_i$	Sample covariance matrix at ith machine, $\hat{\boldsymbol{\Sigma}}_i = \frac{1}{m}\boldsymbol{X}_i\boldsymbol{X}_i^\top$	$d \times d$
$\boldsymbol{\Sigma}_i$	True covariance matrix, $\mathbb{E}\hat{\boldsymbol{\Sigma}}_i$ w.r.t. data $\boldsymbol{X}_i$	$d \times d$
$\boldsymbol{\mathcal{W}}$	Mixing matrix for fusion step	$n \times n$
$\hat{\boldsymbol{\Sigma}}_i^{MTL}$	The ith MTL estimate $\sum_{j=1}^n \mathcal{W}_{ij}\hat{\boldsymbol{\Sigma}}_j$	$d \times d$
$\hat{\boldsymbol{U}}_i^{MTL}$	Top k eigenvectors of $\hat{\boldsymbol{\Sigma}}_i^{MTL}$	$d \times k$

Table 4.2: Key notations used in Section 4.5.

4.5.1 Multitask Rank-k PCA with Convex Relaxation

Multitask PCA is simply an extension of the problem above to multiple distributions. Finding the subspaces of multiple datasets can be useful in many applications that involve dimensionality reduction and multi-source information fusion, like the analysis of multi-sensor physiological signals or multi-country financial trends. Specifically, we take n

machines and assume the spiked covariance model for each $i = 1, \ldots, n$:

$$\boldsymbol{x}_{i,t} = \boldsymbol{U}_i^\star \boldsymbol{\alpha}_{i,t} + \boldsymbol{\epsilon}_{i,t}, \quad \boldsymbol{\alpha}_{i,t} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_k), \quad \boldsymbol{\epsilon}_{i,t} \sim \mathcal{N}(\boldsymbol{0}, \sigma_i^2 \boldsymbol{I}_d), \tag{4.11}$$

and aim to find the ground truth rank-k subspace $\boldsymbol{U}_i^\star \in \mathbb{R}^{d \times k}$, $\boldsymbol{U}_i^{\star\top} \boldsymbol{U}_i^\star = \boldsymbol{I}_k$ from the observed local dataset $\boldsymbol{X}_i = [\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \ldots, \boldsymbol{x}_{i,m}] \in \mathbb{R}^{d \times m}$. Similar to the linear regression set up, m can be different for each task, i.e. is in fact m_i, but we assume $m_1 = \ldots, = m_n$ for simplicity.

When we perform rank-k PCA on the local observation $\boldsymbol{X}_i$, we solve:

$$\hat{\boldsymbol{U}}_i = \underset{\boldsymbol{U} \in \mathbb{R}^{d \times k}; \boldsymbol{U}^\top \boldsymbol{U} = \boldsymbol{I}_k}{\operatorname{argmax}} \operatorname{tr}(\boldsymbol{U}^\top \hat{\boldsymbol{\Sigma}} \boldsymbol{U}) \tag{4.12}$$

using the sample covariance matrix $\hat{\boldsymbol{\Sigma}}_i = \frac{1}{m} \boldsymbol{X}_i \boldsymbol{X}_i^\top \in \mathbb{R}^{d \times d}$. True covariance matrix $\boldsymbol{\Sigma}_i = \mathbb{E}\hat{\boldsymbol{\Sigma}}_i = \boldsymbol{U}_i^\star \boldsymbol{U}_i^{\star\top} + \sigma_i^2 \boldsymbol{I}_d$. (4.12) is minimized by performing eigenvalue decomposition on $\hat{\boldsymbol{\Sigma}}_i$ and choosing the top k eigenvectors.

One interesting way to re-formulate PCA (4.12) is to lift the problem space from rank-k orthogonal matrices to $d \times d$ projection matrix space, and relax some constraints to be convex. In other words, each local server solves the following optimization problem on projection matrices for each $i = 1, \ldots, n$:

$$\begin{aligned} \hat{\boldsymbol{P}}_i &= \underset{\boldsymbol{P} \in \mathbb{R}^{d \times d},\ \boldsymbol{P} = \boldsymbol{P}^\top}{\operatorname{argmax}} \operatorname{tr}(\boldsymbol{P}^\top \hat{\boldsymbol{\Sigma}}_i) && s.t.\ \operatorname{tr}(\boldsymbol{P}) \le k,\ \|\boldsymbol{P}\| \le 1,\ \boldsymbol{P} \succeq 0 \\ &= \underset{\boldsymbol{P} \in \mathbb{R}^{d \times d},\ \boldsymbol{P} = \boldsymbol{P}^\top}{\operatorname{argmin}} \|\boldsymbol{P} - \hat{\boldsymbol{\Sigma}}_i\|_{\mathrm{F}}^2 && s.t.\ \operatorname{tr}(\boldsymbol{P}) \le k,\ \|\boldsymbol{P}\| \le 1,\ \boldsymbol{P} \succeq 0 \end{aligned} \tag{4.13}$$

and take the top k eigenvectors of $\hat{\boldsymbol{P}}_i$. Interestingly, [136, Lemma 5] has shown that $\hat{\boldsymbol{U}}_i \hat{\boldsymbol{U}}_i^\top$ from (4.12) is an optimal solution of (4.13), $\hat{\boldsymbol{P}}_i$.

4.5.2 Motivation via Graph Regularization

As we saw in previous multitask learning scenarios, if there is evidence that there is some similarity between the true subspaces, it would make sense to take advantage of that relationship. This would be especially helpful if there aren't enough observations in the local dataset to get an accurate estimate on its own. In order to nudge similar tasks to get similar results, one might add graph regularization [147] to (4.13) to get the following formulation:

$$(\hat{\boldsymbol{P}}_1^{\lambda}, \ldots, \hat{\boldsymbol{P}}_n^{\lambda}) = \underset{\boldsymbol{P}_i \in \mathbb{R}^{d\times d},\ \boldsymbol{P}_i = \boldsymbol{P}_i^{\top}}{\operatorname{argmin}} \sum_{i=1}^{n} \|\boldsymbol{P}_i - \hat{\boldsymbol{P}}_i\|_{\mathrm{F}}^2 + \lambda \sum_{i,j} W_{ij} \|\boldsymbol{P}_i - \boldsymbol{P}_j\|_{\mathrm{F}}^2 \tag{4.14}$$

$$s.t.\ \operatorname{tr}(\boldsymbol{P}_i) \le k,\ \|\boldsymbol{P}_i\| \le 1,\ \boldsymbol{P}_i \succeq 0$$

where $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ is the adjacency matrix for the similarity graph and $\sum_{j=1}^{n} W_{ij} = 1$, $W_{ii} = 0$, $W_{ij} \ge 0$ for all i, j. Again, $\hat{\boldsymbol{P}}_i$ is the optimal solution of (4.13). Note that $\hat{\boldsymbol{P}}_i^{\lambda}$ may not be k-rank and require a projection step at the end.

Proposition 3 makes a similar observation as Theorem 4 did in multitask linear regression with graph regularization. This motivates the extension of our proposed fusion method from linear regression to PCA. Proof is deferred to Appendix C.5.

Proposition 3 (Graph-regularized Multitask PCA)**.** *Assume $\boldsymbol{W}$ is a right stochastic matrix with no self-loops, i.e. $\sum_{j=1}^{n} W_{ij} = 1$, $W_{ii} = 0$, $W_{ij} \ge 0$ for all i, j. Then $\hat{\boldsymbol{P}}_i^{\lambda}$, the minimizers of graph-regularized multitask PCA (4.14) for $\lambda > 0$, are convex combinations of local estimates $\hat{\boldsymbol{P}}_j$ (4.13). Specifically,*

$$\hat{\boldsymbol{P}}_i^{\lambda} = \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda}\ \hat{\boldsymbol{P}}_j$$

for mixing matrix $\boldsymbol{\mathcal{W}}^{\lambda} \in \mathbb{R}^{n\times n}$*, which is a right stochastic matrix and defined as*

$$\boldsymbol{\mathcal{W}}^{\lambda} = \frac{1}{\lambda+1}\left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1}\boldsymbol{W}\right)^{-1}.$$

4.5.3 Fusion of Sample Covariance Matrices: Proposed Framework and Algorithms

In light of Proposition 3, we suggest MTL estimates of covariance matrices

$$\hat{\boldsymbol{\Sigma}}_i^{MTL} = \sum_{j=1}^{n} \mathcal{W}_{ij}\hat{\boldsymbol{\Sigma}}_j, \quad i = 1,\ldots,n \tag{4.15}$$

from which we produce $\hat{\boldsymbol{U}}_i^{MTL} \in \mathbb{R}^{d\times k}$ by taking the top k eigenvectors of $\hat{\boldsymbol{\Sigma}}_i^{MTL} \in \mathbb{R}^{d\times d}$. The ideal mixing matrix $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{n\times n}$ for MSE reduction is stated in Theorem 6, parallel to multitask linear regression methods. Proof is deferred to Appendix C.6. We then delineate One-Shot Fusion* (Alg. 5) and Iterative Fusion (Alg. 6) for Multitask PCA. Although we assumed the subspace rank k to be identical across tasks so far, note that these proposed algorithms can handle different ks, i.e. $k_i \neq k_j$ for $i \neq j$.

Remark 3. *One might have expected linear fusion of* $\hat{\boldsymbol{P}}$*— and not* $\hat{\boldsymbol{\Sigma}}$*— to follow Proposition 3. That algorithm needs to be developed more carefully, so we defer it to future works. Furthermore, the sample covariance matrix is a known sufficient statistic that contains more information than the projection matrix, which may help future theoretical analysis.*

Theorem 6 (Fusion of Sample Covariance Matrices for Multitask PCA)**.** *Assume observation model (4.11). For* $\hat{\boldsymbol{\Sigma}}_i^{MTL}$ *as defined in (4.15), the MSE* $\mathbb{E}\left\|\hat{\boldsymbol{\Sigma}}_i^{MTL} - \boldsymbol{\Sigma}_i\right\|_{\mathrm{F}}^2$ *is minimized for all* $i = 1,\ldots,n$ *by the mixing matrix*

$$\boldsymbol{\mathcal{W}} = \boldsymbol{C}\left(\boldsymbol{C}+\boldsymbol{V}\right)^{-1},$$

where

$$\boldsymbol{C} = [\langle \boldsymbol{\Sigma}_i, \boldsymbol{\Sigma}_j \rangle]_{i,j} \in \mathbb{R}^{n \times n}, \text{ and } \boldsymbol{V} = diag\left(\left[\mathbb{E} \left\| \hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j \right\|_{\mathrm{F}}^2 \right]_{j=1}^{n} \right) \in \mathbb{R}^{n \times n}.$$

Moreover, the fusion estimate $\hat{\boldsymbol{\Sigma}}_i^{MTL}$ *will always be at least as accurate as* $\hat{\boldsymbol{\Sigma}}_i$ *in terms of MSE. For each* $i = 1, \ldots, n$*,*

$$\mathbb{E} \left\| \hat{\boldsymbol{\Sigma}}_i^{MTL} - \boldsymbol{\Sigma}_i \right\|_{\mathrm{F}}^2 \leq \mathbb{E} \left\| \hat{\boldsymbol{\Sigma}}_i - \boldsymbol{\Sigma}_i \right\|_{\mathrm{F}}^2.$$

4.5.4 Simulation Experiments

We validate Algs. 5 and 6 on data generated according to the spiked covariance model (4.11). To enforce similarity between the ground truth subspaces, we additionally use the slowly rotating subspace model and code from [150], i.e.

$$\boldsymbol{U}^\star \sim \mathcal{N}(0,1) \; i.i.d, \quad \boldsymbol{U}_1^\star = orth(\boldsymbol{U}^\star), \quad \boldsymbol{U}_i^\star = \exp(\delta_0 \boldsymbol{R}) \boldsymbol{U}_{i-1}^\star \; i = 2, \ldots, n \tag{4.16}$$

where a skew-symmetric matrix $\boldsymbol{R} \in \mathbb{R}^{d \times d}$ is sampled with independent, normally distributed entries, and exp is the matrix exponential. We set the rotation parameter $\delta_0 = 0.01$, which is large enough to prevent our experiment from becoming reduced to the distributed consensus case. Other parameters are set as $n = 20$, $d = 100$, $k = 20$, $\sigma_1 = \sigma_2 = \ldots, \sigma_n = 0.9$, and $m = d/2 = 50$, which makes the local estimation problem quite challenging. We measure the covariance matrix error, which is defined as $\frac{1}{dn} \sum_{i=1}^{n} \|\hat{\boldsymbol{\Sigma}}_i - \boldsymbol{\Sigma}_i\|_{\mathrm{F}}^2$, as well as the subspace error $\frac{1}{dn} \sum_{i=1}^{n} \|\hat{\boldsymbol{U}}_i \hat{\boldsymbol{U}}_i^\top - \boldsymbol{U}_i^\star \boldsymbol{U}_i^{\star\top}\|_{\mathrm{F}}^2$. $\mathbb{E} \left\| \hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j \right\|_{\mathrm{F}}^2$ is estimated with $2k\sigma_j^2 + d\sigma_j^4(1 + \frac{k}{m})$ under the spiked covariance model (4.11).

The left image in Fig. 4.7 corresponds to $\boldsymbol{C}$ in Alg. 5, that is, the true similarity

Algorithm 5 One-Shot Fusion* for Multitask PCA

inputs local datasets $\{\boldsymbol{X}_i\}_{i=1}^n$, (estimated or given) subspace ranks $\{k_i\}_{i=1}^n$.
Each machine *locally* computes sample covariance matrix $\hat{\boldsymbol{\Sigma}} = \frac{1}{m}\boldsymbol{X}_i\boldsymbol{X}_i^\top$.
Each machine *locally* estimates $\mathbb{E}\left\|\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right\|_{\mathrm{F}}^2$, e.g. by matrix perturbation theory.
$\boldsymbol{V} \leftarrow diag\left(\left[\mathbb{E}\left\|\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right\|_{\mathrm{F}}^2\right]_{j=1}^n\right)$
$\boldsymbol{C} \leftarrow \left[\langle \boldsymbol{\Sigma}_i, \boldsymbol{\Sigma}_j\rangle\right]_{i,j}$
$\boldsymbol{\mathcal{W}} \leftarrow \boldsymbol{C}\left(\boldsymbol{C}+\boldsymbol{V}\right)^{-1}$
$\hat{\boldsymbol{\Sigma}}_i^{MTL} \leftarrow \sum_{j=1}^n \mathcal{W}_{ij}\hat{\boldsymbol{\Sigma}}_j$
$\hat{\boldsymbol{U}}_i^{MTL} \leftarrow$ top k_i eigenvectors of $\hat{\boldsymbol{\Sigma}}_i^{MTL}$.
outputs updated fusion estimates $\{\hat{\boldsymbol{U}}_i^{MTL}\}_{i=1}^n$.

Algorithm 6 Iterative Fusion for Multitask PCA (Practical Version of Alg. 5)

inputs local datasets $\{\boldsymbol{X}_i\}_{i=1}^n$, (estimated or given) subspace ranks $\{k_i\}_{i=1}^n$.
Each machine *locally* computes sample covariance matrix $\hat{\boldsymbol{\Sigma}} = \frac{1}{m}\boldsymbol{X}_i\boldsymbol{X}_i^\top$.
Each machine *locally* estimates $\mathbb{E}\left\|\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right\|_{\mathrm{F}}^2$, e.g. by matrix perturbation theory.
$\boldsymbol{V} \leftarrow diag\left(\left[\mathbb{E}\left\|\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right\|_{\mathrm{F}}^2\right]_{j=1}^n\right)$
repeat
    $\boldsymbol{C} \leftarrow \left[\langle \hat{\boldsymbol{\Sigma}}_i, \hat{\boldsymbol{\Sigma}}_j\rangle\right]_{i,j}$
    $\boldsymbol{\mathcal{W}} \leftarrow \boldsymbol{C}\left(\boldsymbol{C}+\boldsymbol{V}\right)^{-1}$
    (Optional) Threshold elements of $\boldsymbol{\mathcal{W}}$
    $\hat{\boldsymbol{\Sigma}}_i \leftarrow \sum_{j=1}^n \mathcal{W}_{ij}\hat{\boldsymbol{\Sigma}}_j$
until termination
$\hat{\boldsymbol{\Sigma}}_i^{MTL} \leftarrow \hat{\boldsymbol{\Sigma}}_i$
$\hat{\boldsymbol{U}}_i^{MTL} \leftarrow$ top k_i eigenvectors of $\hat{\boldsymbol{\Sigma}}_i^{MTL}$.
outputs updated fusion estimates $\{\hat{\boldsymbol{U}}_i^{MTL}\}_{i=1}^n$.

matrix whose (i,j)th element is $\langle \mathbf{\Sigma}_i, \mathbf{\Sigma}_j \rangle$. This confirms that the rotating subspace model (4.16) produces a set of similar but not identical tasks. The mixing weights from fusion algorithms in Fig. 4.7 seem to mirror the trends of the similarity matrix. Finally, Fig. 4.8 summarizes the estimation error reductions by the PCA fusion algorithms.

4.6 Conclusions and Future Works

We proposed a novel fusion framework for distributed multitask learning that linearly combines local estimates to get improved estimates for each task, while bypassing the restrictions on data sharing. Motivated by graph regularization solutions, we developed concrete algorithms for multitask linear regression (with guarantee for any linear estimators), and for multitask PCA. When tested on simulated data, combining local OLS estimates according to our proposed methods significantly surpassed the performance of *optimal* local ridge regression estimates under a wide range of conditions. Fusion of local sample covariance matrices for multitask PCA was also verified on simulated data. The following works will be explored in the future:

- **Applications** - There's a lot left to explore in this direction. First and foremost, we want to test our multitask linear regression and PCA algorithms on real data. Then, we want to validate our linear regression algorithms on biased estimators, e.g. ridge regression estimators, as well as under more complex community data models.

- **Theory** - We want to more *quantitatively* describe when this multitask learning approach helps, and by how much. For example, can we find out how much MSE reduction is guaranteed by our algorithms by plugging in task similarity and task difficulty variables? Can we prove that our approach is more sample efficient?

- **Algorithm** - When $n = O(d)$ (e.g. federated learning setting), the matrix calculation

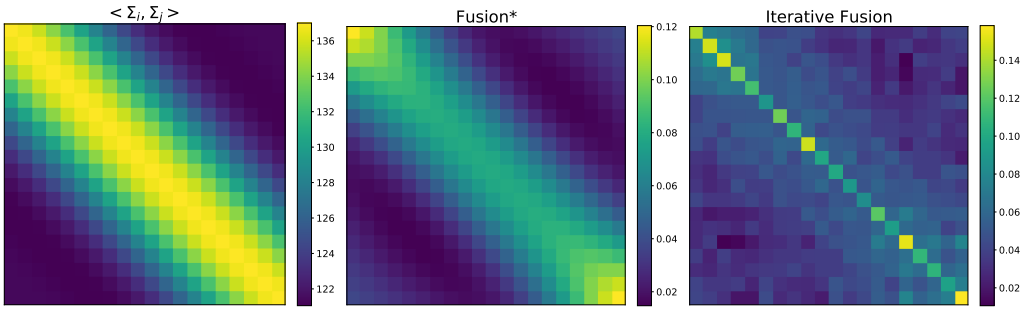


Figure 4.7: (left) Slowly rotating subspace model (4.16) induces similarity between true covariance matrices that taper off as $|i-j|$ become large. Mixing matrices from One-Shot Fusion* (Alg. 5) (middle) and the Iterative Fusion (Alg. 6) (right) mirror the general trend of the task similarity matrix.

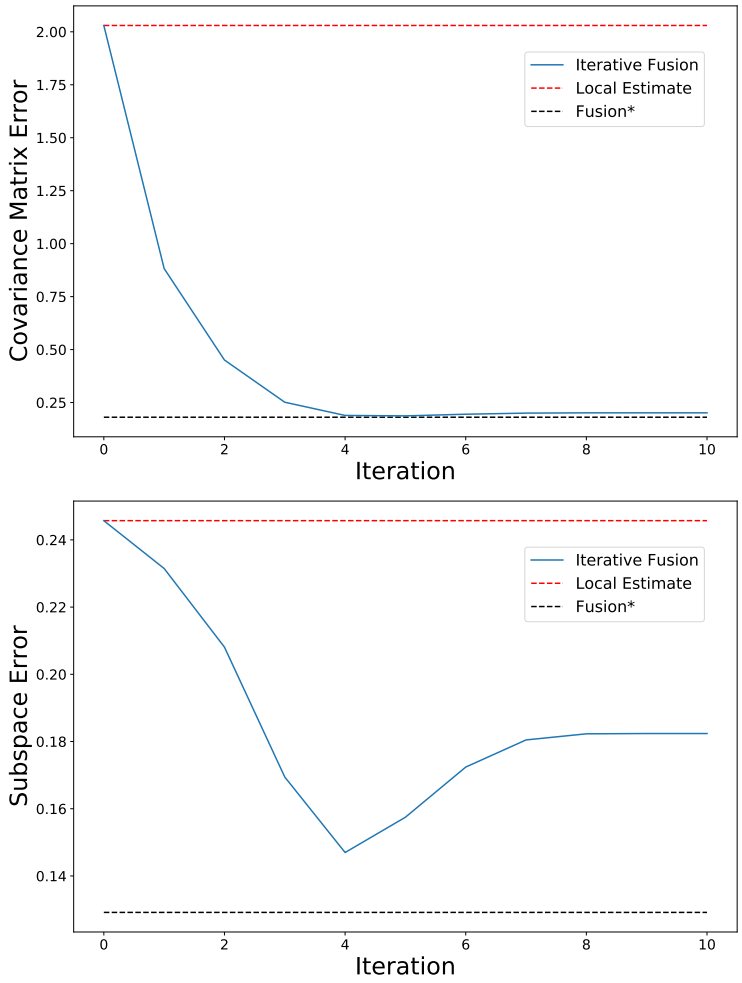


Figure 4.8: Both the covariance matrix estimation error and subspace estimation error are greatly reduced by the PCA fusion algorithms. Interestingly, under this set of simulation parameters, Iterative Fusion (Alg. 6) reaches the performance of One-Shot Fusion* (Alg. 5) (top). Averaged over 20 trials.

step (e.g. line 8 of Alg. 4) becomes a bottleneck. It will be interesting to explore potential tradeoff between accuracy and computation cost via iterative methods or matrix approximation methods. Better understanding of convergence or stopping criteria will be beneficial as well. The PCA algorithm in particular can be sped up with fast eigenvector finding methods.

Chapter 5

Conclusions and Future Works

This thesis studied several examples of how graph regularization can improve the accuracy, computational efficiency and/or sample efficiency of inference problems, specifically graph-regularized denoising, matrix factorization for remote sensing, and distributed multitask learning. **Chapter** 2: For denoising, we generalized the convex graph trend filtering (GTF) framework to certain non-convex regularizers (e.g. SCAD, MCP), which significantly reduced the bias and improved the accuracy of the denoiser on piecewise smooth graph signals. Moreover, theoretical analysis confirmed our intuition that the strength of graph regularization is bounded by the alignment of the given graph and the signal. **Chapter** 3: Next, we formulated the blind hyperspectral unmixing problem in remote sensing as a non-negative matrix factorization problem with graph total variation (gTV) regularization. The graph structure was based on the pixel similarity graph, which we approximated fast with the Nyström method. We then approximated gTV with smooth functions, i.e. the Ginzburg-Landau (GL) functional, and used the Merriman-Bence-Osher (MBO) scheme for fast optimization. Our proposed algorithm showed improved computational efficiency and comparable accuracy in experiments. **Chapter** 4: Last, we proposed a privacy-conserving and easy-to-understand approach to distributed multitask learning that requires only one round of communication to share local estimates. Fusing local estimates based on the task similarities and difficulties to improve the accuracy and sample efficiency was motivated by the results of graph regularization. We demonstrated the framework on

linear regression and PCA.

We list potential directions for future works below:

- Fill in the gaps for distributed multitask learning (Chapter 4). See Section 4.6.
- Extend matrix factorization with graph regularization (Chapter 3) to online learning setting. Online matrix factorization has applications in computational biology, specifically in single-cell RNA analysis [151]. See Appendix B.
- Some loosely connected ideas:
  - Learn the regularizer: The core of all graph regularization methods presented here is in selecting the "best" function for the specific problem. However, [152] recently explored the idea of using a *learned graph regularization* for sparse coding and denoising, instead of hand-picking them ahead of time. This is a promising area that warrants more investigation.
  - Learn the graph: The graph regularization problems presented in this thesis assumed that a graph is given, or is derived beforehand. For example, we assumed that a graph is given when theoretically analyzing denoising with graph regularization (Chapter 2), and supplied physical graphs (road networks, grid graph) for denoising experiments. And for semi-supervised classification experiment (Chapter 2) and matrix factorization in remote sensing (Chapter 3), we derived similarity graphs from the data before applying our algorithms. Can we merge the two-staged approach (derive graph $\rightarrow$ use graph regularization) into one? Can we go beyond iteratively updating the graph and the signal?
  - Plug and Play [153] and Regularization by Denoising [154] are recent frameworks that use blackbox image denoisers to solve more complicated inverse problems.

Can we extend this to a graph setting? Can we learn the graph or regularizer at the same time?

- Active learning on graph: Instead of observing the data all at once, what if we sample the graph signal one node at a time? Which sampling strategy is ideal for each task, and what role can regularization play in this setting?

Appendix A

Proofs for Denoising with Graph Regularization

A.1 Proof of Theorem 1

Proof. We denote $\boldsymbol{D} = \boldsymbol{\Delta}^{(k+1)}$. Define R as the row space of $\boldsymbol{D}$, and $R^{\perp}$ the null space. Let $\mathcal{P}_{\mathrm{R}} = \boldsymbol{D}^{\dagger}\boldsymbol{D}$, the projection onto R, and $\|\boldsymbol{y}\|_{\mathrm{R}} = \|\mathcal{P}_{\mathrm{R}}\boldsymbol{y}\|_2$. Additionally, $\mathcal{P}_{\mathrm{R}^{\perp}} = \boldsymbol{I} - \boldsymbol{D}^{\dagger}\boldsymbol{D}$, the projection onto $R^{\perp}$. Since $\hat{\boldsymbol{\beta}}$ is a stationary point of $f(\boldsymbol{\beta})$, it follows that

$$\mathbf{0} \in \nabla_{\boldsymbol{\beta}} f(\boldsymbol{\beta})|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}} = (\hat{\boldsymbol{\beta}} - \boldsymbol{x}) + \nabla_{\boldsymbol{\beta}} g(\boldsymbol{D}\boldsymbol{\beta})|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}}. \tag{A.1}$$

By the chain rule, $\nabla_{\boldsymbol{\beta}} g(\boldsymbol{D}\boldsymbol{\beta})|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}} = \{\boldsymbol{D}^{\top}\boldsymbol{z} : \boldsymbol{z} \in \nabla_{\boldsymbol{y}} g(\boldsymbol{y})|_{\boldsymbol{y}=\boldsymbol{D}\hat{\boldsymbol{\beta}}}\}$. Then by (A.1), there exists $\boldsymbol{z} \in \nabla_{\boldsymbol{y}} g(\boldsymbol{y})|_{\boldsymbol{y}=\boldsymbol{D}\hat{\boldsymbol{\beta}}}$, such that

$$\mathbf{0} = (\hat{\boldsymbol{\beta}} - \boldsymbol{x}) + \boldsymbol{D}^{\top}\boldsymbol{z}.$$

In particular, $\forall \boldsymbol{\beta} \in \mathbb{R}^n$, we have

$$\boldsymbol{\beta}^{\top}(\boldsymbol{x} - \hat{\boldsymbol{\beta}}) = (\boldsymbol{D}\boldsymbol{\beta})^{\top}\boldsymbol{z}, \tag{A.2}$$

and, specializing to $\hat{\boldsymbol{\beta}}$,

$$\hat{\boldsymbol{\beta}}^\top(\boldsymbol{x}-\hat{\boldsymbol{\beta}}) = (\boldsymbol{D}\hat{\boldsymbol{\beta}})^\top \boldsymbol{z}. \tag{A.3}$$

Subtract (A.3) from (A.2), and use the definition of subgradient to get $\forall \boldsymbol{\beta} \in \mathbb{R}^n$,

$$\begin{aligned}\boldsymbol{\beta}^\top(\boldsymbol{x}-\hat{\boldsymbol{\beta}}) - \hat{\boldsymbol{\beta}}^\top(\boldsymbol{x}-\hat{\boldsymbol{\beta}}) &= (\boldsymbol{D}\boldsymbol{\beta} - \boldsymbol{D}\hat{\boldsymbol{\beta}})^\top \boldsymbol{z} \\ &\leq g(\boldsymbol{D}\boldsymbol{\beta}) - g(\boldsymbol{D}\hat{\boldsymbol{\beta}}).\end{aligned} \tag{A.4}$$

By the measurement model $\boldsymbol{x} = \boldsymbol{\beta}^\star + \boldsymbol{\epsilon}$ and the polarization equality, i.e. $2\boldsymbol{a}^\top\boldsymbol{b} = \|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2 - \|\boldsymbol{a}-\boldsymbol{b}\|_2^2$, the left-hand side of (A.4) can be rewritten as

$$\begin{aligned}&\boldsymbol{\beta}^\top(\boldsymbol{x}-\hat{\boldsymbol{\beta}}) - \hat{\boldsymbol{\beta}}^\top(\boldsymbol{x}-\hat{\boldsymbol{\beta}}) \\ &= (\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^\top(\boldsymbol{\beta}^\star-\hat{\boldsymbol{\beta}}) + \boldsymbol{\epsilon}^\top(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}) \\ &= \frac{1}{2}\|\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}\|_2^2 + \frac{1}{2}\|\boldsymbol{\beta}^\star-\hat{\boldsymbol{\beta}}\|_2^2 - \frac{1}{2}\|\boldsymbol{\beta}-\boldsymbol{\beta}^\star\|_2^2 + \boldsymbol{\epsilon}^\top(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}).\end{aligned} \tag{A.5}$$

Combining (A.4) and (A.5) gives us $\forall \boldsymbol{\beta} \in \mathbb{R}^n$

$$\begin{aligned}&\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2 + \|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^\star\|_2^2 \\ &\leq \|\boldsymbol{\beta}-\boldsymbol{\beta}^\star\|_2^2 + 2\boldsymbol{\epsilon}^\top(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}) + 2g(\boldsymbol{D}\boldsymbol{\beta}) - 2g(\boldsymbol{D}\hat{\boldsymbol{\beta}}).\end{aligned} \tag{A.6}$$

Let us first consider $\boldsymbol{\epsilon}^\top(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})$. From Hölder's inequality,

$$\begin{aligned}\boldsymbol{\epsilon}^\top(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}) &= (\boldsymbol{D}^\dagger\boldsymbol{D}\boldsymbol{\epsilon})^\top(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}) + (\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon})^\top(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}) \\ &\leq \|(\boldsymbol{D}^\dagger)^\top\boldsymbol{\epsilon}\|_\infty\|\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\|_1 + \|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2.\end{aligned} \tag{A.7}$$

By standard tail bounds for independent Gaussian random variables, we have with probability at least $1-\delta$,

$$\|(\boldsymbol{D}^\dagger)^\top \boldsymbol{\epsilon}\|_\infty \leq \sigma\zeta_k\sqrt{2\log\left(\frac{er}{\delta}\right)}. \tag{A.8}$$

Additionally, recognize that $\|\boldsymbol{\epsilon}\|_{\mathrm{R}^\perp}^2$ is a chi-squared random variable with $C_{\mathcal{G}}$ degrees of freedom. We can then invoke the one-sided tail bound for chi-squared random variables (c.f. [155, Example 2.5]) such that for any $0 \leq t \leq 1$,

$$P\big(\|\boldsymbol{\epsilon}\|_{\mathrm{R}^\perp}^2 \geq \sigma^2 C_{\mathcal{G}}(1+t)\big) \leq \exp\left(\frac{-C_{\mathcal{G}}t^2}{8}\right).$$

Consequently, with probability at least $1-\delta$,

$$\|\boldsymbol{\epsilon}\|_{\mathrm{R}^\perp}^2 \leq \sigma^2\Big(C_{\mathcal{G}} + 2\sqrt{2C_{\mathcal{G}}\log(1/\delta)}\Big). \tag{A.9}$$

The inequalities (A.8) and (A.9) hold simultaneously with probability at least $1-2\delta$. Then, using $\lambda\|\boldsymbol{y}\|_1 \leq g(\boldsymbol{y}) + \frac{\mu}{2}\|\boldsymbol{y}\|_2^2$ and $\lambda = \sigma\zeta_k\sqrt{2\log(\frac{er}{\delta})} \geq \|(\boldsymbol{D}^\dagger)^\top\boldsymbol{\epsilon}\|_\infty$, we can bound (A.7) further as

$$\begin{aligned}\boldsymbol{\epsilon}^\top(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}) &\leq \|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2 + \lambda\|\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\|_1 \\ &\leq \|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2 + g(\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})) + \frac{\mu}{2}\|\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\|_2^2.\end{aligned}$$

Together with $\|\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\|_2^2 \leq \|\boldsymbol{D}\|^2\|(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})\|_2^2$, we can upper bound (A.6) as

$$\begin{aligned}\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2 + \|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^\star\|_2^2 \leq \|\boldsymbol{\beta}-\boldsymbol{\beta}^\star\|_2^2 + 2\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2 + \mu\|\boldsymbol{D}\|^2\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2 \\ + 2g(\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})) + 2g(\boldsymbol{D}\boldsymbol{\beta}) - 2g(\boldsymbol{D}\hat{\boldsymbol{\beta}}).\end{aligned} \tag{A.10}$$

Note that for any set T, $g(\boldsymbol{y}) = \sum_{i\in T}\rho(y_i) + \sum_{j\in T^c}\rho(y_j) = g((\boldsymbol{y})_T) + g((\boldsymbol{y})_{T^c})$. Therefore, using the triangle inequality and subadditivity and symmetry of ρ,

$$
\begin{aligned}
& g(\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})) + g(\boldsymbol{D}\boldsymbol{\beta}) - g(\boldsymbol{D}\hat{\boldsymbol{\beta}}) \\
\leq & g((\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}))_T) + g((\boldsymbol{D}\boldsymbol{\beta})_{T^c}) + g((\boldsymbol{D}\hat{\boldsymbol{\beta}})_{T^c}) + g(\boldsymbol{D}\boldsymbol{\beta}) - g((\boldsymbol{D}\hat{\boldsymbol{\beta}})_T) - g((\boldsymbol{D}\hat{\boldsymbol{\beta}})_{T^c}) \\
= & g((\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}))_T) + 2g((\boldsymbol{D}\boldsymbol{\beta})_{T^c}) + g((\boldsymbol{D}\boldsymbol{\beta})_T) - g((\boldsymbol{D}\hat{\boldsymbol{\beta}})_T) \\
\leq & 2g((\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}))_T) + 2g((\boldsymbol{D}\boldsymbol{\beta})_{T^c}). \qquad \text{(A.11)}
\end{aligned}
$$

We bound (A.11) further by the compatibility factor,

$$
\begin{aligned}
g((\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}))_T) &\leq \lambda \|(\boldsymbol{D}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}))_T\|_1 \\
&\leq \lambda\sqrt{|T|}\kappa_T^{-1}\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2. \qquad \text{(A.12)}
\end{aligned}
$$

Now combining (A.10), (A.11), and (A.12), we then have

$$
\begin{aligned}
\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2 + \|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^\star\|_2^2 \leq & \|\boldsymbol{\beta}-\boldsymbol{\beta}^\star\|_2^2 + 4g((\boldsymbol{D}\boldsymbol{\beta})_{T^c}) \\
& + 2\left(\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2 + 2\lambda\sqrt{|T|}\kappa_T^{-1}\right)\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2 + \mu\|\boldsymbol{D}\|^2\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2.
\end{aligned}
$$

Apply Young's inequality $2ab \leq a^2/\epsilon + \epsilon b^2$ with $a = \|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2 + 2\lambda\sqrt{|T|}\kappa_T^{-1}$, $b = \|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2$, and $\epsilon = 1-\mu\|\boldsymbol{D}\|^2 > 0$. We then have

$$
\begin{aligned}
& 2\left(\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2 + 2\lambda\sqrt{|T|}\kappa_T^{-1}\right)\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2 \\
\leq & \frac{1}{\epsilon}\left(\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2 + 2\lambda\sqrt{|T|}\kappa_T^{-1}\right)^2 + \epsilon\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2 \\
\leq & \frac{2}{(1-\mu\|\boldsymbol{D}\|^2)}\left(\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2^2 + 4\lambda^2|T|\kappa_T^{-2}\right) + (1-\mu\|\boldsymbol{D}\|^2)\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2.
\end{aligned}
$$

Therefore,

$$\begin{aligned}
&\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|^2+\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^\star\|_2^2 \\
\leq&\|\boldsymbol{\beta}-\boldsymbol{\beta}^\star\|_2^2+4g((\boldsymbol{D}\boldsymbol{\beta})_{T^c})+\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2+\frac{2}{(1-\mu\|\boldsymbol{D}\|^2)}\left(\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{\epsilon}\|_2^2+4\lambda^2|T|\kappa_T^{-2}\right).
\end{aligned}$$

Cancel $\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}\|_2^2$ on both sides, apply the infimum over $\boldsymbol{\beta}$, and plug in the bounds (A.9) to get

$$\begin{aligned}
\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^\star\|_2^2 \leq& \inf_{\boldsymbol{\beta}}\left\{\|\boldsymbol{\beta}-\boldsymbol{\beta}^\star\|_2^2+4g((\boldsymbol{D}\boldsymbol{\beta})_{T^c})\right\} \\
&+\frac{2\sigma^2}{(1-\mu\|\boldsymbol{D}\|^2)}\left(C_{\mathcal{G}}+2\sqrt{2C_{\mathcal{G}}\log\left(\frac{1}{\delta}\right)}+\frac{8\zeta_k^2|T|}{\kappa_T^2}\log\left(\frac{er}{\delta}\right)\right).
\end{aligned}$$

The proof extends for the vector-GTF (2.7) in a similar manner. We need to replace (A.7) by

$$\begin{aligned}
\langle\boldsymbol{E},\hat{\boldsymbol{B}}-\boldsymbol{B}\rangle&=\langle\boldsymbol{D}^\dagger\boldsymbol{D}\boldsymbol{E},\hat{\boldsymbol{B}}-\boldsymbol{B}\rangle+\langle\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{E},\hat{\boldsymbol{B}}-\boldsymbol{B}\rangle \\
&\leq\lambda\sum_{\ell=1}^{r}\left\|\boldsymbol{D}_{\ell\cdot}(\hat{\boldsymbol{B}}-\boldsymbol{B})\right\|_2+\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{E}\|_{\mathrm{F}}\|\hat{\boldsymbol{B}}-\boldsymbol{B}\|_{\mathrm{F}},
\end{aligned}$$

where $\|\mathcal{P}_{\mathrm{R}^\perp}\boldsymbol{E}\|_{\mathrm{F}}^2\leq d\sigma^2\left(C_{\mathcal{G}}+2\sqrt{2C_{\mathcal{G}}\log(d/\delta)}\right)$ with probability at least $1-\delta$. Similarly, for (A.12), we use the generalized definition of the compatibility factor κ_T, given as

$$\begin{aligned}
h((\boldsymbol{D}(\hat{\boldsymbol{B}}-\boldsymbol{B}))_T)&\leq\lambda\sum_{\ell\in T}\|(\boldsymbol{D}(\hat{\boldsymbol{B}}-\boldsymbol{B}))_{\ell\cdot}\|_2 \\
&\leq\lambda\sqrt{|T|}\kappa_T^{-1}\|\hat{\boldsymbol{B}}-\boldsymbol{B}\|_{\mathrm{F}},
\end{aligned}$$

which will lead to the claimed bound in the theorem. □

A.2 Proof of Proposition 1

Proof. By Cauchy-Schwartz inequality, we have

$$\sum_{\ell \in T} \|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{\ell\cdot}\|_2 \leq \sqrt{|T|}\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_T\|_{\mathrm{F}},$$

and note that given two matrices $\boldsymbol{U}$ and $\boldsymbol{V}$, $(\boldsymbol{UV})_T = (\boldsymbol{U})_T\boldsymbol{V}$ where T is a subset of rows indices. We also use the fact that $\|\boldsymbol{UV}\|_{\mathrm{F}} \leq \|\boldsymbol{U}\|\|\boldsymbol{V}\|_{\mathrm{F}}$. We consider two cases:

- For even k, we have

$$\begin{aligned}\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_T\|_{\mathrm{F}} &= \|(\boldsymbol{\Delta})_T\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}} \\ &\leq \|(\boldsymbol{\Delta})_T\|\|\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}} = \sqrt{\lambda_{\max}((\boldsymbol{\Delta})_T^\top(\boldsymbol{\Delta})_T)}\|\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}}.\end{aligned}$$

  Note that $(\boldsymbol{\Delta})_T$ is equivalent to the incidence matrix of a subgraph with only T edges, which allows us to bound,

$$\lambda_{\max}([(\boldsymbol{\Delta})_T]^\top(\boldsymbol{\Delta})_T)) \leq \max_{(u,v)\in T}\{d_u + d_v\} \leq 2d_{\max}$$

  where d_i is the degree of node i.

- For odd k, we have

$$\begin{aligned}\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_T\|_{\mathrm{F}} &= \|(\boldsymbol{\Delta}^\top)_T\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}} \\ &\leq \|(\boldsymbol{\Delta}^\top)_T\|\|\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}} = \sqrt{\lambda_{\max}(\boldsymbol{\Delta}^{(2)}_{T\times T})}\|\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}},\end{aligned}$$

  where $\boldsymbol{\Delta}^{(2)}_{T\times T} \in \mathbb{R}^{|T|\times|T|}$ is the principal submatrix of $\boldsymbol{\Delta}^{(2)}$ indexed by T. By Cauchy's

interlacing theorem, the maximum eigenvalue of the submatrix is upper bounded, so

$$\lambda_{\max}(\boldsymbol{\Delta}^{(2)}_{T\times T}) \leq \lambda_{\max}(\boldsymbol{\Delta}^{(2)}) \leq 2d_{\max}.$$

Therefore, for all k, $\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_T\|_{\mathrm{F}} \leq \sqrt{2d_{\max}}\|\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}}$. To conclude the proof, note

$$\begin{aligned}\sum_{\ell\in T}\|(\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B})_{\ell\cdot}\|_2 &\leq \sqrt{|T|}\sqrt{2d_{\max}}\|\boldsymbol{\Delta}^{(k)}\boldsymbol{B}\|_{\mathrm{F}} \\ &\leq \sqrt{|T|}\sqrt{2d_{\max}}\|\boldsymbol{\Delta}^{(k)}\|\|\boldsymbol{B}\|_{\mathrm{F}} \leq (2d_{\max})^{\frac{k+1}{2}}\sqrt{|T|}\|\boldsymbol{B}\|_{\mathrm{F}}.\end{aligned}$$

□

A.3 Proof of Theorem 3

We show the convergence of Alg. 1 by proving a modified version of [25, Proposition 1]. The superscript (m) denotes the values of $\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U}$ at the mth iteration of the loop inside Alg. 1.

Proposition 4 (Convergence to a feasible solution)**.** *If $\tau \geq \mu$, then the primal residual $r^{(m)} = \|\boldsymbol{\Delta}^{(k+1)}\boldsymbol{B}^{(m)} - \boldsymbol{Z}^{(m)}\|_{\mathrm{F}}$ and the dual residual $s^{(m+1)} = \|\tau(\boldsymbol{\Delta}^{(k+1)})^\top(\boldsymbol{Z}^{(m+1)} - \boldsymbol{Z}^{(m)})\|_{\mathrm{F}}$ of Alg. 1 satisfy that $\lim_{m\to\infty} r^{(m)} = 0$ and $\lim_{m\to\infty} s^{(m)} = 0$.*

Proof. Denote $\boldsymbol{D} = \boldsymbol{\Delta}^{(k+1)}$, and $\rho_\lambda(\cdot) = \rho(\cdot;\lambda,\gamma)$. Recall from Assumption 1 (c) that there exists $\mu > 0$ such that $\rho_\lambda(\|\boldsymbol{x}\|_2) + \frac{\mu}{2}\|\boldsymbol{x}\|_2^2$ is convex. Now consider the Lagrangian $\mathcal{L}(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U})$ with regard to the ℓth row $\boldsymbol{z}_\ell$ of $\boldsymbol{Z} = [\boldsymbol{z}_1^\top, ...\boldsymbol{z}_r^\top]^\top$, assuming all other variables

are fixed:

$$\begin{aligned}&\rho_\lambda(\|\boldsymbol{z}_\ell\|_2) + \frac{\tau}{2}\|\boldsymbol{z}_\ell - \boldsymbol{c}_1\|_2^2 + c_2\\ =&\rho_\lambda(\|\boldsymbol{z}_\ell\|_2) + \frac{\tau}{2}\|\boldsymbol{z}_\ell\|_2^2 - \tau\langle \boldsymbol{z}_\ell, \boldsymbol{c}_1\rangle + \frac{\tau}{2}\|\boldsymbol{c}_1\|_2^2 + c_2\end{aligned}$$

where $\boldsymbol{c}_1$ and c_2 represent terms of $\mathcal{L}(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U})$ that do not depend on $\boldsymbol{z}_\ell$. With our choice of $\tau \geq \mu$, then $\mathcal{L}(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U})$ is convex with regard to each of $\boldsymbol{B}$, $\boldsymbol{U}$, and for each row of $\boldsymbol{Z}$, allowing us to apply [156, Theorem 5.1]. Therefore, Alg. 1 converges to limit points $\boldsymbol{B}^\natural, \boldsymbol{Z}^\natural, \boldsymbol{U}^\natural$.

Then it follows that the dual residual $\lim_{m\to\infty} s^{(m)} = \|\tau \boldsymbol{D}^\top(\boldsymbol{Z}^\natural - \boldsymbol{Z}^\natural)\|_{\mathrm{F}} = 0$. For the primal residual, notice that the $\boldsymbol{U}$ update step in line 10 of Alg. 1 also shows that $\forall m, t \geq 0$,

$$\boldsymbol{U}^{(m+t)} = \boldsymbol{U}^{(m)} + \sum_{i=1}^{t}(\boldsymbol{D}\boldsymbol{B}^{(m+i)} - \boldsymbol{Z}^{(m+i)}).$$

Fixing t and setting $m \to \infty$, we have

$$\boldsymbol{U}^\natural = \boldsymbol{U}^\natural + t(\boldsymbol{D}\boldsymbol{B}^\natural - \boldsymbol{Z}^\natural)$$

holds $\forall t \geq 0$. Hence, $\boldsymbol{D}\boldsymbol{B}^\natural - \boldsymbol{Z}^\natural = \boldsymbol{0}$, and therefore $\lim_{m\to\infty} r^{(m)} = \|\boldsymbol{D}\boldsymbol{B}^\natural - \boldsymbol{Z}^\natural\|_{\mathrm{F}} = 0$. □

This proposition shows that the algorithm in the limit achieves primal and dual feasibility, and that the Augmented Lagrangian in (2.11) with $\boldsymbol{Z}^\natural$ and $\boldsymbol{U}^\natural$ becomes the original GTF formulation in (2.7). $\boldsymbol{B}$ that is produced by every iteration of Alg. 1 is a stationary point of (2.11) with fixed $\boldsymbol{Z}$ and $\boldsymbol{U}$. As a result, $\boldsymbol{B}^\natural$ is a stationary point of (2.7).

Appendix B

Preliminary Results for Online Matrix Factorization in Computational Biology

B.1 Computational Biology Motivation

Single cell RNA sequencing (scRNA-seq) is a powerful technique that can measure the gene expression levels of each cell in a population of cells. Access to gene expression levels at such high resolution allows us to study new biological questions about cell identity, e.g. uncover cell types (especially rare populations), determine changes in regulatory networks across cells, identify heterogeneity of gene expression, and track trajectories of cell lineages in development. Since these datasets are very noisy and high dimensional, i.e. thousands of genes are measured, one common approach in scRNA-seq analysis is to first project each cell onto a significantly lower dimensional latent space and perform further analysis such as clustering.

Given a scRNA-seq dataset $\boldsymbol{X}$, a matrix of number of genes $\times$ number of cells, this can be formulated into a matrix factorization (MF) problem with noise model $\boldsymbol{X} = \boldsymbol{S}^\star \boldsymbol{A}^\star +$

gaussian noise:

$$\text{MF:} \quad (\hat{\boldsymbol{S}}, \hat{\boldsymbol{A}}) = \underset{\boldsymbol{S},\boldsymbol{A}}{\operatorname{argmin}} \ \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2$$

where $\boldsymbol{S}$ is number of genes $\times$ latent space dimension and $\boldsymbol{A}$ is number of cells $\times$ latent space dimension. Roughly, $\boldsymbol{S}$ can be interpreted as a matrix that linearly projects from the high dimensional gene space to the low dimensional latent space, and $\boldsymbol{a}_i$, the ith column of $\boldsymbol{A}$, can be interpreted as the position of the ith cell in the latent space. Recent works such as [157] have suggested that adding prior knowledge on the gene co-expression network as graph regularization can help:

$$\text{netNMF-sc [157]:} \quad (\hat{\boldsymbol{S}}, \hat{\boldsymbol{A}}) = \underset{\boldsymbol{S}\geq 0,\ \boldsymbol{A}\geq 0}{\operatorname{argmin}} \ \|\Phi_\Omega\left(\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\right)\|_{\mathrm{F}}^2 + \lambda \sum_{i,j} \|\boldsymbol{s}_i - \boldsymbol{s}_j\|_2^2 W_{ij} \quad \text{(B.1)}$$

where $\boldsymbol{W}$ is the adjacency matrix derived from the gene-gene interactions, $\boldsymbol{s}_i$ is the ith row of $\boldsymbol{S}$ corresponding to the ith gene, and Φ_Ω is a binary mask such that zero entries of $\boldsymbol{X}$ are ignored in (B.1).

Thanks to new protocols developed in the past decade, we are able to acquire scRNA-seq datasets at a low cost with unprecendented throughput, measuring up to 10^6 cells at a time. Applying batch methods like (B.1) on data of this size puts enormous pressure on the computer memory, and are computationally costly. Therefore, online methods that can process a small number of cells at a time have started emerging [158]. We propose an online version of the graph-regularized problem:

$$\min_{\boldsymbol{S},\boldsymbol{A}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_{\mathrm{F}}^2 + \frac{\alpha}{2}\left(\|\boldsymbol{S}\|_{\mathrm{F}}^2 + \|\boldsymbol{A}\|_{\mathrm{F}}^2\right) + \sum_{i,j} \|\boldsymbol{s}_i - \boldsymbol{s}_j\|_2^2 W_{ij} + \sum_{i,j} \|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2^2 U_{ij}$$

where $\boldsymbol{a}_i$ is the ith column of $\boldsymbol{A}$, and $\boldsymbol{U}$ is a cell-cell graph that can potentially be derived

from prior knowledge about the cells' physical location or experimental setup, e.g. cells collected from the same organism, cells sequenced in the same batch, etc. We expect that the online algorithm will improve memory and computation efficiencies for large scRNA-seq datasets.

B.2 Online Matrix Factorization with Graph Regularization

In the online setting, we do not observe the full data right away, but instead observe $\boldsymbol{x}_t$, the tth column of $\boldsymbol{X}$, at every time step t. Let $\boldsymbol{X}_t = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t] \in \mathbb{R}^{n \times t}$ (size changes, but elements of $\boldsymbol{X}_{t-1}$ are unchanged), $\boldsymbol{S}_t \in \mathbb{R}^{n \times r}$ (size does not change, but elements are updated), and $\boldsymbol{A}_t = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_t] \in \mathbb{R}^{r \times t}$ (column $\boldsymbol{a}_t$ is added and size changes, but elements of $\boldsymbol{A}_{t-1}$ are unchanged).[1] Then, after tth observation, we iteratively update $\boldsymbol{S}_t$ and $\boldsymbol{a}_t$:

$$\boldsymbol{a}_t = \underset{\boldsymbol{a}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{x}_t - \boldsymbol{S}_{t-1}\boldsymbol{a}\|_2^2 + \frac{\alpha}{2}\|\boldsymbol{a}\|_2^2 + \sum_{i<t} \|\boldsymbol{a}_\ell - \boldsymbol{a}\|_2^2 U_{ti} \tag{B.2}$$

$$\boldsymbol{S}_t = \underset{\boldsymbol{S}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{X}_t - \boldsymbol{S}\boldsymbol{A}_t\|_{\mathrm{F}}^2 + \frac{\alpha}{2}\|\boldsymbol{S}\|_{\mathrm{F}}^2 + \sum_{i,j} \|\boldsymbol{s}_i - \boldsymbol{s}_j\|_2^2 W_{ij} \tag{B.3}$$

$\boldsymbol{a}_t$: At stationarity, (B.2) equals

$$\begin{aligned}
&0 = \boldsymbol{S}_{t-1}^\top (\boldsymbol{S}_{t-1}\boldsymbol{a}_t - \boldsymbol{x}_t) + \alpha \boldsymbol{a}_t + 2\sum_{i<t} U_{it}(\boldsymbol{a}_t - \boldsymbol{a}_i) \\
\implies &\left(\boldsymbol{S}_{t-1}^\top \boldsymbol{S}_{t-1} + \left(\alpha + 2\sum_{i \in \mathcal{N}(\boldsymbol{a}_t)} U_{it}\right)\boldsymbol{I}\right)\boldsymbol{a}_t = \boldsymbol{S}_{t-1}^\top \boldsymbol{x}_t + 2\sum_{i \in \mathcal{N}(\boldsymbol{a}_t)} U_{it}\boldsymbol{a}_i \\
\implies &\boldsymbol{a}_t = \left(\boldsymbol{S}_{t-1}^\top \boldsymbol{S}_{t-1} + \left(\alpha + 2\sum_{i \in \mathcal{N}(\boldsymbol{a}_t)} U_{it}\right)\boldsymbol{I}\right)^{-1}\left(\boldsymbol{S}_{t-1}^\top \boldsymbol{x}_t + 2\sum_{i \in \mathcal{N}(\boldsymbol{a}_t)} U_{it}\boldsymbol{a}_i\right)
\end{aligned}$$

[1] $\boldsymbol{A}_{t-1}$ can be fixed, and we just need to update $\boldsymbol{a}_t$ and $\boldsymbol{S}$. Since we go through multiple epochs of the data, $\boldsymbol{S}$ will be relatively stable by the last epoch, and columns of $\boldsymbol{A}_T$ will be comparable with each other for downstream tasks.

$\mathcal{N}(\boldsymbol{a}_t)$ are the neighbors of $\boldsymbol{a}_t$ that we have seen until t. The $r \times r$ matrix inversion should be quick.

$\boldsymbol{S}_t$: Rewrite (B.3) using graph Laplacian matrix $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ derived from the adjacency matrix $\boldsymbol{W}$, which doesn't change with time.

$$
\begin{aligned}
\boldsymbol{S}_t &= \underset{\boldsymbol{S}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{X}_t - \boldsymbol{S}\boldsymbol{A}_t\|_{\mathrm{F}}^2 + \frac{\alpha}{2}\|\boldsymbol{S}\|_{\mathrm{F}}^2 + \operatorname{tr}\left(\boldsymbol{S}^\top \boldsymbol{L}\boldsymbol{S}\right) \\
\implies 0 &= \left(\boldsymbol{S}_t\boldsymbol{A}_t - \boldsymbol{X}_t\right)\boldsymbol{A}_t^\top + \alpha\boldsymbol{S}_t + 2\boldsymbol{L}\boldsymbol{S}_t \\
\implies \boldsymbol{S}_t\boldsymbol{A}_t\boldsymbol{A}_t^\top &+ \left(\alpha\boldsymbol{I} + 2\boldsymbol{L}\right)\boldsymbol{S}_t = \boldsymbol{X}_t\boldsymbol{A}_t^\top \qquad \text{(B.4)} \\
\implies \left(\boldsymbol{I} \otimes \left(\alpha\boldsymbol{I} + 2\boldsymbol{L}\right) + \boldsymbol{A}_t\boldsymbol{A}_t^\top \otimes \boldsymbol{I}\right) &\, vec(\boldsymbol{S}_t) = vec(\boldsymbol{X}_t\boldsymbol{A}_t^\top)
\end{aligned}
$$

(B.4) is a Sylvester equation, and can be solved by many existing programs such as Matlab. The obvious problem here is that we need to keep $\boldsymbol{X}_t$ in memory. But since $\boldsymbol{X}_t = [\boldsymbol{X}_{t-1}, \boldsymbol{x}_t]$, and $\boldsymbol{A}_t = [\boldsymbol{A}_{t-1}, \boldsymbol{a}_t]$, we can write $\boldsymbol{X}_t\boldsymbol{A}_t^\top = \boldsymbol{X}_{t-1}\boldsymbol{A}_{t-1}^\top + \boldsymbol{x}_t\boldsymbol{a}_t^\top$, i.e. we can simply keep a running sum. We can also update $\boldsymbol{A}_t\boldsymbol{A}_t^\top = \boldsymbol{A}_{t-1}\boldsymbol{A}_{t-1}^\top + \boldsymbol{a}_t\boldsymbol{a}_t^\top$ in a similar fashion.

Algorithm 7 Online Matrix Factorization with Graph Regularization

Inputs: Data stream $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, graph $\boldsymbol{U}$, graph $\boldsymbol{W}$.
Initialize: $\boldsymbol{S}_1$, $\boldsymbol{A}_1$ in a batch fashion based on the first t_0 samples, $\boldsymbol{X}_{t_0}$.
Calculate $\boldsymbol{L}$ from $\boldsymbol{W}$, and $\boldsymbol{M} = \alpha \boldsymbol{I} + 2\boldsymbol{L}$.
for `epoch` $\leftarrow 1$ to `num_epoch` **do**
    Shuffle $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$.
    for $t \leftarrow 1$ to T **do**
        Observe $\boldsymbol{x}_t$.
        Query the neighbors $\mathcal{N}(\boldsymbol{a}_t)$ and the edge weights $\boldsymbol{U}_{it}$ for $i \in \mathcal{N}(\boldsymbol{a}_t)$.
        $\boldsymbol{a}_t \leftarrow \left(\boldsymbol{S}_{t-1}^\top \boldsymbol{S}_{t-1} + \left(\alpha + 2\sum_{i \in \mathcal{N}(\boldsymbol{a}_t)} U_{it}\right) \boldsymbol{I}\right)^{-1} \left(\boldsymbol{S}_{t-1}^\top \boldsymbol{x}_t + 2\sum_{i \in \mathcal{N}(\boldsymbol{a}_t)} U_{it} \boldsymbol{a}_i\right)$
        $\boldsymbol{B}_t \leftarrow \boldsymbol{B}_{t-1} + \boldsymbol{x}_t \boldsymbol{a}_t^\top$
        $\boldsymbol{C}_t \leftarrow \boldsymbol{C}_{t-1} + \boldsymbol{a}_t \boldsymbol{a}_t^\top$.
        $\boldsymbol{S}_t \leftarrow$Solve for $\boldsymbol{Y}$ in Sylvester equation $\boldsymbol{Y}\boldsymbol{C}_t + \boldsymbol{M}\boldsymbol{Y} = \boldsymbol{B}_t$
    end for
end for
Outputs: $\boldsymbol{S}_T$, $\boldsymbol{A}_T$

Appendix C

Proofs and Intermediate Results for Distributed Multitask Learning

C.1 Proof of Theorem 4

Proof. At stationarity, for each $i = 1, \ldots, n$,

$$
\begin{aligned}
0 &= \boldsymbol{A}_i^\top \left(\boldsymbol{A}_i \hat{\boldsymbol{\beta}}_i^\lambda - \boldsymbol{x}_i \right) + \lambda \sum_{j=1}^n W_{ij} \left(\hat{\boldsymbol{\beta}}_i^\lambda - \hat{\boldsymbol{\beta}}_j^\lambda \right) \\
\implies & \left(\lambda \sum_{j=1}^n W_{ij} + 1 \right) \hat{\boldsymbol{\beta}}_i^\lambda = \boldsymbol{A}_i^\top \boldsymbol{x}_i + \lambda \sum_{j=1}^n W_{ij} \hat{\boldsymbol{\beta}}_j^\lambda \\
\implies & \hat{\boldsymbol{\beta}}_i^\lambda = \frac{1}{\lambda + 1} \hat{\boldsymbol{\beta}}_i^{OLS} + \frac{\lambda}{\lambda + 1} \sum_{j=1}^n W_{ij} \hat{\boldsymbol{\beta}}_j^\lambda .
\end{aligned}
$$

We can rewrite this set of linear equations concisely in matrix form:

$$\implies \begin{bmatrix} -1 & \frac{\lambda W_{12}}{\lambda+1} & \dots & \frac{\lambda W_{1n}}{\lambda+1} \\ \frac{\lambda W_{21}}{\lambda+1} & -1 & \dots & \frac{\lambda W_{2n}}{\lambda+1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\lambda W_{n1}}{\lambda+1} & \frac{\lambda W_{n2}}{\lambda+1} & \dots & -1 \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}}_1^{\lambda} \\ \hat{\boldsymbol{\beta}}_2^{\lambda} \\ \vdots \\ \hat{\boldsymbol{\beta}}_n^{\lambda} \end{bmatrix} = -\frac{1}{\lambda+1} \begin{bmatrix} \hat{\boldsymbol{\beta}}_1^{OLS} \\ \hat{\boldsymbol{\beta}}_2^{OLS} \\ \vdots \\ \hat{\boldsymbol{\beta}}_n^{OLS} \end{bmatrix}$$

$$\implies \begin{bmatrix} \hat{\boldsymbol{\beta}}_1^{\lambda} \\ \hat{\boldsymbol{\beta}}_2^{\lambda} \\ \vdots \\ \hat{\boldsymbol{\beta}}_n^{\lambda} \end{bmatrix} = \frac{1}{\lambda+1} \left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1} \boldsymbol{W} \right)^{-1} \begin{bmatrix} \hat{\boldsymbol{\beta}}_1^{OLS} \\ \hat{\boldsymbol{\beta}}_2^{OLS} \\ \vdots \\ \hat{\boldsymbol{\beta}}_n^{OLS} \end{bmatrix}.$$

It remains to show that each row of $\boldsymbol{\mathcal{W}}^{\lambda}$ adds to 1. We assumed $\boldsymbol{W}$ is a right stochastic matrix, which implies 1) for all integer $k > 1$, $\boldsymbol{W}^k$ is also a right stochastic matrix, and 2) the maximum eigenvalue of $\frac{\lambda}{\lambda+1}\boldsymbol{W}$ must be < 1. Therefore

$$\boldsymbol{\mathcal{W}}^{\lambda} = \frac{1}{\lambda+1} \left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1} \boldsymbol{W} \right)^{-1} = \frac{1}{\lambda+1} \sum_{k=0}^{\infty} \left(\frac{\lambda}{\lambda+1} \right)^k \boldsymbol{W}^k.$$

by Neumann series expansion. Then, the sum of ith row follows

$$\begin{aligned}
\sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} &= \sum_{j=1}^{n} \left[\frac{1}{\lambda+1} \left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1} \boldsymbol{W} \right)^{-1} \right]_{ij} \\
&= \sum_{j=1}^{n} \frac{1}{\lambda+1} \sum_{k=0}^{\infty} \left(\frac{\lambda}{\lambda+1} \right)^k \left[\boldsymbol{W}^k \right]_{ij} \\
&= \frac{1}{\lambda+1} \sum_{k=0}^{\infty} \left(\frac{\lambda}{\lambda+1} \right)^k \sum_{j=1}^{n} \left[\boldsymbol{W}^k \right]_{ij} \\
&= \frac{1}{\lambda+1} \sum_{k=0}^{\infty} \left(\frac{\lambda}{\lambda+1} \right)^k \\
&= 1.
\end{aligned}$$

□

C.2 Proof of Proposition 2

Proof. The proof follows from assumptions on noise (zero-mean, independent between tasks), $\sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} = 1$, and Jensen's inequality.

$$\begin{aligned}
\mathbb{E} \left\| \hat{\boldsymbol{\beta}}_i^{\lambda} - \boldsymbol{\beta}_i^{\star} \right\|_2^2 &= \left\| \left(\sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} \boldsymbol{\beta}_j^{\star} \right) - \boldsymbol{\beta}_i^{\star} \right\|_2^2 + \mathbb{E} \left\| \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} \boldsymbol{A}_j^{\top} \boldsymbol{\epsilon}_j \right\|_2^2 \\
&= \left\| \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} \left(\boldsymbol{\beta}_j^{\star} - \boldsymbol{\beta}_i^{\star} \right) \right\|_2^2 + \sum_{j=1}^{n} (\mathcal{W}_{ij}^{\lambda})^2 \mathbb{E} \left\| \boldsymbol{A}_j^{\top} \boldsymbol{\epsilon}_j \right\|_2^2 \\
&\leq \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} \left\| \boldsymbol{\beta}_i^{\star} - \boldsymbol{\beta}_j^{\star} \right\|_2^2 + \sum_{j=1}^{n} (\mathcal{W}_{ij}^{\lambda})^2 \mathbb{E} \left\| \boldsymbol{A}_j^{\top} \boldsymbol{\epsilon}_j \right\|_2^2 \\
&= \sum_{j=1}^{n} \mathcal{W}_{ij}^{\lambda} \left\| \boldsymbol{\beta}_i^{\star} - \boldsymbol{\beta}_j^{\star} \right\|_2^2 + d \sum_{j=1}^{n} (\mathcal{W}_{ij}^{\lambda} \sigma_j)^2.
\end{aligned}$$

□

C.3 Proof of Theorem 5

Proof. Under the observation model (4.1), noise is zero-mean, additive, and independent between tasks. Then from linearity of expectation,

$$
\begin{aligned}
&\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i^{MTL}-\boldsymbol{\beta}_i^{\star}\right\|_2^2 \\
=&\mathbb{E}\left\|\sum_{j=1}^{n}\mathcal{W}_{ij}\left(\mathbb{E}\hat{\boldsymbol{\beta}}_j+\hat{\boldsymbol{\beta}}_j-\mathbb{E}\hat{\boldsymbol{\beta}}_j\right)-\boldsymbol{\beta}_i^{\star}\right\|_2^2 \\
=&\left\|\sum_{j=1}^{n}\mathcal{W}_{ij}\mathbb{E}\hat{\boldsymbol{\beta}}_j-\boldsymbol{\beta}_i^{\star}\right\|_2^2+\sum_{j=1}^{n}(\mathcal{W}_{ij})^2\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_j-\mathbb{E}\hat{\boldsymbol{\beta}}_j\right\|_2^2 \\
=&\left\|\sum_{j=1}^{n}\mathcal{W}_{ij}\mathbb{E}\hat{\boldsymbol{\beta}}_j\right\|_2^2-2\boldsymbol{\beta}_i^{\star\top}\sum_{j=1}^{n}\mathcal{W}_{ij}\mathbb{E}\hat{\boldsymbol{\beta}}_j+\sum_{j=1}^{n}(\mathcal{W}_{ij})^2\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_j-\mathbb{E}\hat{\boldsymbol{\beta}}_j\right\|_2^2+\|\boldsymbol{\beta}_i^{\star}\|_2^2. \qquad \text{(C.1)}
\end{aligned}
$$

To minimize (C.1), ignore $\|\boldsymbol{\beta}_i^{\star}\|_2^2$, and denote the ith row of $\boldsymbol{\mathcal{W}}$ as $\boldsymbol{y}^\top$ for a fixed $i=1,\ldots,n$. Then (4.8) follows from

$$
\begin{aligned}
&\min_{\boldsymbol{y}}\ \boldsymbol{y}^\top\boldsymbol{C}\boldsymbol{y}-2\boldsymbol{K}_{i\cdot}\boldsymbol{y}+\boldsymbol{y}^\top\boldsymbol{V}\boldsymbol{y} \\
\Longrightarrow\ &(\boldsymbol{C}+\boldsymbol{V})\,\boldsymbol{y}=\boldsymbol{K}_{i\cdot}^\top \\
\Longrightarrow\ &\boldsymbol{y}=(\boldsymbol{C}+\boldsymbol{V})^{-1}\,\boldsymbol{K}_{i\cdot}^\top \\
\Longrightarrow\ &\boldsymbol{\mathcal{W}}^\top=(\boldsymbol{C}+\boldsymbol{V})^{-1}\,\boldsymbol{K}^\top.
\end{aligned}
$$

□

C.4 Examples of Local Estimators for Theorem 5

Example 5 (OLS)**.** *OLS with tall* $\boldsymbol{A}_i \in \mathbb{R}^{m\times d}$*,* $m \geq d$*:*

$$\hat{\boldsymbol{\beta}}_i = (\boldsymbol{A}_i^\top \boldsymbol{A}_i)^{-1}\boldsymbol{A}_i^\top \boldsymbol{x}_i, \quad \mathbb{E}\hat{\boldsymbol{\beta}}_i = \boldsymbol{\beta}_i^\star, \quad \mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2 = \sigma_i^2 \operatorname{tr}\left(\left(\boldsymbol{A}_i^\top \boldsymbol{A}_i\right)^{-1}\right)$$

When $\boldsymbol{A}_i$ *is also orthogonal, i.e.* $\boldsymbol{A}_i^\top \boldsymbol{A}_i = \boldsymbol{I}_d$*:*

$$\hat{\boldsymbol{\beta}}_i = \boldsymbol{A}_i^\top \boldsymbol{x}_i, \quad \mathbb{E}\hat{\boldsymbol{\beta}}_i = \boldsymbol{\beta}_i^\star, \quad \mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2 = d\sigma_i^2$$

Example 6 (Ridge Regression)**.** *Ridge regression with* $\boldsymbol{A}_i$*, parameter* $\lambda_i > 0$*:*

$$\begin{aligned}
\hat{\boldsymbol{\beta}}_i &= \operatorname*{argmin}_{\boldsymbol{\beta}_i} \|\boldsymbol{x}_i - \boldsymbol{A}_i\boldsymbol{\beta}_i\|_2^2 + \lambda_i \|\boldsymbol{\beta}_i\|_2^2 \\
&= \left(\boldsymbol{A}_i^\top \boldsymbol{A}_i + 2\lambda_i \boldsymbol{I}\right)^{-1} \boldsymbol{A}_i^\top \boldsymbol{x}_i \\
\mathbb{E}\hat{\boldsymbol{\beta}}_i &= \left(\boldsymbol{A}_i^\top \boldsymbol{A}_i + 2\lambda_i \boldsymbol{I}\right)^{-1} \boldsymbol{A}_i^\top \boldsymbol{\beta}_i^\star \\
\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2 &= \sigma_i^2 tr\left(\left(\boldsymbol{A}_i^\top \boldsymbol{A}_i + 2\lambda_i \boldsymbol{I}\right)^{-1} \boldsymbol{A}_i^\top \boldsymbol{A}_i \left(\boldsymbol{A}_i^\top \boldsymbol{A}_i + 2\lambda_i \boldsymbol{I}\right)^{-1}\right)
\end{aligned}$$

When $\boldsymbol{A}_i$ *is tall and orthogonal, and* λ_i *is chosen to be optimal, i.e. minimizes MSE of* $\hat{\boldsymbol{\beta}}_i$*:*

$$\begin{aligned}
\lambda_i &= \frac{d\sigma_i^2}{2\|\boldsymbol{\beta}_i^\star\|_2^2} \\
\mathbb{E}\hat{\boldsymbol{\beta}}_i &= \frac{1}{1+2\lambda_i}\boldsymbol{\beta}_i^\star \\
\mathbb{E}\left\|\hat{\boldsymbol{\beta}}_i - \mathbb{E}\hat{\boldsymbol{\beta}}_i\right\|_2^2 &= \frac{d\sigma_i^2}{(1+2\lambda_i)^2}
\end{aligned} \tag{C.2}$$

C.5 Proof of Proposition 3

Proof. Proof is parallel to the proof of Theorem 4. Stationarity leads to the following linear equation

$$\begin{bmatrix} \hat{\boldsymbol{P}}_1^\lambda \\ \hat{\boldsymbol{P}}_2^\lambda \\ \vdots \\ \hat{\boldsymbol{P}}_n^\lambda \end{bmatrix} = \frac{1}{\lambda+1}\left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1}\boldsymbol{W}\right)^{-1} \begin{bmatrix} \hat{\boldsymbol{P}}_1 \\ \hat{\boldsymbol{P}}_2 \\ \vdots \\ \hat{\boldsymbol{P}}_n \end{bmatrix},$$

where $\frac{1}{\lambda+1}\left(\boldsymbol{I}_n - \frac{\lambda}{\lambda+1}\boldsymbol{W}\right)^{-1}$ is a known right stochastic matrix. It is easy to check that $\hat{\boldsymbol{P}}_i^\lambda$, convex combinations of $\hat{\boldsymbol{P}}_j$s, remain in the constraint set: trace is a linear operator, spectral norm follows triangle inequality, and sum of positive semi-definite matrices is positive semi-definite. □

C.6 Proof of Theorem 6

Proof. Proof is very similar to the proof of Theorem 5.

$$\begin{aligned}
&\mathbb{E}\|\hat{\boldsymbol{\Sigma}}_i^{MTL} - \boldsymbol{\Sigma}_i\|_{\mathrm{F}}^2 \\
=&\mathbb{E}\left\|\sum_{j=1}^n \mathcal{W}_{ij}\boldsymbol{\Sigma}_j - \boldsymbol{\Sigma}_i + \sum_{j=1}^n \mathcal{W}_{ij}\left(\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right)\right\|_{\mathrm{F}}^2 \\
=&\left\|\sum_{j=1}^n \mathcal{W}_{ij}\boldsymbol{\Sigma}_j - \boldsymbol{\Sigma}_i\right\|_{\mathrm{F}}^2 + \mathbb{E}\left\|\sum_{j=1}^n \mathcal{W}_{ij}\left(\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right)\right\|_{\mathrm{F}}^2 \\
=&\left\|\sum_{j=1}^n \mathcal{W}_{ij}\boldsymbol{\Sigma}_j\right\|_{\mathrm{F}}^2 - 2\sum_{j=1}^n \mathcal{W}_{ij}\langle\boldsymbol{\Sigma}_i, \boldsymbol{\Sigma}_j\rangle + \|\boldsymbol{\Sigma}_i\|_{\mathrm{F}}^2 + \sum_{j=1}^n (\mathcal{W}_{ij})^2\mathbb{E}\left\|\hat{\boldsymbol{\Sigma}}_j - \boldsymbol{\Sigma}_j\right\|_{\mathrm{F}}^2
\end{aligned}$$

□

C.7 Intermediate Results: Convex Combination of OLS Estimates

Theorem 4 motivates us to consider estimates that are convex combinations of local OLS estimates. In this section, we investigate the MSE of estimates that can be represented as

$$\hat{\boldsymbol{\beta}}_i^{cvx} = \sum_{j=1}^{n} W_{ij}^{cvx} \hat{\boldsymbol{\beta}}_j^{OLS} = \sum_{j=1}^{n} W_{ij}^{cvx} \boldsymbol{\beta}_j^{\star} + \sum_{j=1}^{n} W_{ij}^{cvx} \boldsymbol{A}_j^{\top} \boldsymbol{\epsilon}_j \tag{C.3}$$

for any right stochastic matrix $\boldsymbol{W}^{cvx} \in \mathbb{R}^{n \times n}$. Note that the graph-regularized solutions $\hat{\boldsymbol{\beta}}_i^{\lambda}$ are a special case of $\hat{\boldsymbol{\beta}}_i^{cvx}$, where $\boldsymbol{W}^{cvx} = \boldsymbol{\mathcal{W}}^{\lambda}$. In fact, the only property of $\boldsymbol{\mathcal{W}}^{\lambda}$ that was used in the proof of Proposition 2 is that $\boldsymbol{\mathcal{W}}^{\lambda}$ is a right stochastic matrix. Corollary 2 thus follows immediately.

Corollary 2. *Assume $m \geq d$, $\boldsymbol{A}_i^{\top} \boldsymbol{A}_i = \boldsymbol{I}_d$, and $\boldsymbol{W}^{cvx}$ is a right stochastic matrix, i.e. $\sum_{j=1}^{n} W_{ij}^{cvx} = 1$ for all i. Then, MSE of $\hat{\boldsymbol{\beta}}_i^{cvx}$ (C.3) is upper-bounded by*

$$\mathbb{E} \left\| \hat{\boldsymbol{\beta}}_i^{cvx} - \boldsymbol{\beta}_i^{\star} \right\|_2^2 \leq \sum_{j=1}^{n} W_{ij}^{cvx} \left\| \boldsymbol{\beta}_i^{\star} - \boldsymbol{\beta}_j^{\star} \right\|_2^2 + d \sum_{j=1}^{n} (W_{ij}^{cvx} \sigma_j)^2. \tag{C.4}$$

Given Corollary 2, one might be interested in finding a right stochastic matrix $\boldsymbol{W}^{cvx}$ such that the MSE surrogate in (C.4) is minimized, as shown in (C.5).

$$\min_{W^{cvx}} \sum_{j=1}^{n} W_{ij}^{cvx} \left\| \boldsymbol{\beta}_i^{\star} - \boldsymbol{\beta}_j^{\star} \right\|_2^2 + d \sum_{j=1}^{n} (W_{ij}^{cvx} \sigma_j)^2 \quad s.t. \quad W_{ij}^{cvx} \geq 0, \sum_{j=1}^{n} W_{ij}^{cvx} = 1. \tag{C.5}$$

Proposition 5. *For each $i = 1, \ldots, n$, (C.5) is minimized by*

$$W_{ij}^{cvx} = \left(\mu_i \frac{1}{2d\sigma_j^2} - \frac{\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2}{2d\sigma_j^2} \right)_+, \quad \sum_{j=1}^n W_{ij}^{cvx} = 1. \tag{C.6}$$

where μ_i and W_{ij}^{cvx} for a fixed i can be calculated by Alg. 8.

See Appendix C.8 for the proof of Proposition 5 and Alg. 8. Note that $\hat{\boldsymbol{\beta}}_i^{cvx}$ constructed based on Proposition 5 will achieve lower MSE than the graph regularization solutions $\hat{\boldsymbol{\beta}}_i^{\lambda}$ and the local OLS estimates $\hat{\boldsymbol{\beta}}_i^{OLS}$, but not compared to the proposed MTL estimates $\hat{\boldsymbol{\beta}}^{MTL}$. The mixing matrix described in (C.6) depends on task similarity $\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2$ and task difficulty $d\sigma_j^2$.

C.8 Proof of Proposition 5

$$\boldsymbol{W}^{cvx} = \underset{\boldsymbol{W}}{\operatorname{argmin}} \sum_{j=1}^n W_{ij} \left\|\boldsymbol{\beta}_i^\star - \boldsymbol{\beta}_j^\star\right\|_2^2 + d\sum_{j=1}^n (W_{ij}\sigma_j)^2 \quad s.t. \quad W_{ij} \geq 0, \sum_{j=1}^n W_{ij} = 1$$

is equivalent to optimizing the following for each $\boldsymbol{w}_i$, the ith row of $\boldsymbol{W}^{cvx}$:

$$\begin{aligned} &\min_{\boldsymbol{w}_i} \quad \boldsymbol{w}_i^\top \boldsymbol{b}_i + \frac{1}{2}\boldsymbol{w}_i^\top \boldsymbol{\Sigma} \boldsymbol{w}_i \quad s.t. \quad \mathbf{1}^\top \boldsymbol{w}_i = 1, \boldsymbol{w}_i \geq 0 \\ &\text{where} \quad \boldsymbol{w}_i = \left[W_{ij}^{cvx}\right]_{j=1}^n, \ \boldsymbol{b}_i = \left[\left\|\boldsymbol{\beta}_i^\star - \boldsymbol{\beta}_j^\star\right\|_2^2\right]_{j=1}^n, \ \boldsymbol{\Sigma} = diag(\left[2d\sigma_j^2\right]_{j=1}^n) \end{aligned}$$

Let's introduce the dual variables $\boldsymbol{U}_i \geq 0$, and μ_i, and temporarily drop the i subscript, i.e. overload $\boldsymbol{w} := \boldsymbol{w}_i$, $\boldsymbol{b} := \boldsymbol{b}_i$, $\boldsymbol{U} := \boldsymbol{U}_i$, $\mu := \mu_i$. The corresponding Lagrangian is

$$L(\boldsymbol{w}, \boldsymbol{U}, \mu) = \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w} + \boldsymbol{w}^\top \boldsymbol{b} - \boldsymbol{U}^\top \boldsymbol{w} + \mu(1 - \mathbf{1}^\top \boldsymbol{w}),$$

which has the following KKT conditions. Stationarity:

$$\begin{aligned}
&\boldsymbol{\Sigma w} + \boldsymbol{b} - \boldsymbol{U} - \mu\mathbf{1} = 0 \\
\implies &\boldsymbol{w} = -\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{b} - \boldsymbol{U} - \mu\mathbf{1}\right) \\
\implies &w_j = -\frac{1}{2d\sigma_j^2}\left(b_j - u_j - \mu\right)
\end{aligned}$$

complementary slackness:

$$u_j w_j = 0,$$

primal feasibility:

$$\mathbf{1}^\top \boldsymbol{w} = 1, \quad \boldsymbol{w} \geq 0,$$

and dual feasibility:

$$\boldsymbol{U} \geq 0.$$

Substitute u_j from stationarity into complementary slackness and dual feasibility:

$$\begin{aligned}
&u_j = 2d\sigma_j^2 w_j + b_j - \mu \\
&w_j\left(2d\sigma_j^2 w_j + b_j - \mu\right) = 0 \\
&2d\sigma_j^2 w_j + b_j - \mu \geq 0
\end{aligned}$$

Note that

$$\begin{aligned}
&\mu > b_j \implies w_j > 0 \implies w_j = \frac{\mu - b_j}{2d\sigma_j^2} \\
&\mu \leq b_j \implies w_j = 0
\end{aligned}$$

Therefore,

$$w_j = \max\left\{0, \frac{\mu - b_j}{2d\sigma_j^2}\right\},$$

where μ depends on the last condition, primal feasibility:

$$\sum_{j=1}^{n} w_j = 1 \implies \sum_{j=1}^{n} \left(\frac{\mu - b_j}{2d\sigma_j^2}\right)_+ = 1$$

Finally, substituting back the original variables show that

$$W_{ij}^{cvx} = \left(\mu \frac{1}{2d\sigma_j^2} - \frac{\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2}{2d\sigma_j^2}\right)_+, \quad \sum_{j=1}^{n} W_{ij}^{cvx} = 1.$$

This can be optimized by the multilevel water-filling algorithm [159, Algorithm 2]. For every ith row of $\boldsymbol{W}^{cvx}$, the setup is identical to the example in [159, Section V-A-2] with the variables

$$a_j = \frac{1}{2d\sigma_j^2}, \quad b_j = \frac{\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2}{2d\sigma_j^2}, \quad g(\mu) = \mu \sum_{j=1}^{\tilde{n}} \frac{1}{2d\sigma_j^2} - \sum_{j=1}^{\tilde{n}} \frac{\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2}{2d\sigma_j^2} - 1 = 0.$$

Alg. 8 summarizes this procedure nicely.

Algorithm 8 Multilevel Water Filling Algorithm to solve (C.5)

1: **inputs** i is fixed. $\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2$, $d\sigma_j^2$ for $j = 1, \ldots, n$.
2: Sort such that $\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2$ are in increasing order.
3: Set $\tilde{n} \leftarrow n$.
4: **if** $\|\boldsymbol{\beta}_i^\star - \boldsymbol{\beta}_{\tilde{n}}^\star\|_2^2 \geq \frac{1+\sum_{j=1}^{\tilde{n}} \frac{\left\|\boldsymbol{\beta}_i^\star - \boldsymbol{\beta}_j^\star\right\|_2^2}{2d\sigma_j^2}}{\sum_{j=1}^{\tilde{n}} \frac{1}{2d\sigma_j^2}}$ **then**
5: Set $\tilde{n} \leftarrow \tilde{n} - 1$.
6: Go back to line 4.
7: **end if**
8: Set $\mu_i = \frac{1+\sum_{j=1}^{\tilde{n}} \frac{\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2}{2d\sigma_j^2}}{\sum_{j=1}^{\tilde{n}} \frac{1}{2d\sigma_j^2}}$.
9: Undo sorting.
10: Calculate $W_{ij}^{cvx} = \left(\mu_i \frac{1}{2d\sigma_j^2} - \frac{\left\|\boldsymbol{\beta}_j^\star - \boldsymbol{\beta}_i^\star\right\|_2^2}{2d\sigma_j^2}\right)_+$
11: **outputs** W_{ij}^{cvx} for $j = 1, \ldots, n$.

Bibliography

[1] M. Newman, *Networks*. Oxford University Press, 2018. 2

[2] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, 2013. 2, 7, 11

[3] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013. 2, 11, 44, 46

[4] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018. 2, 11

[5] R. Varma*, H. Lee*, Y. Chi, and J. Kovačević, "Improving graph trend filtering with non-convex penalties," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5391–5395, IEEE, 2019. 5, 23

[6] R. Varma*, H. Lee*, J. Kovačević, and Y. Chi, "Vector-valued graph trend filtering with non-convex penalties," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 48–62, 2020. 5

[7] J. Qin, H. Lee, J. T. Chi, Y. Lou, J. Chanussot, and A. L. Bertozzi, "Fast blind hyperspectral unmixing based on graph laplacian," in *2019 10th Workshop on Hyper-*

spectral Imaging and Signal Processing: Evolution in Remote Sensing (WHISPERS), pp. 1–5, 2019. 5, 48, 59

[8] J. Qin, H. Lee, J. T. Chi, L. Drumetz, J. Chanussot, Y. Lou, and A. L. Bertozzi, "Blind hyperspectral unmixing based on graph total variation regularization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 4, pp. 3338–3351, 2021. 5

[9] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal representations on graphs: Tools and applications," *arXiv preprint arXiv:1512.05406*, 2015. 11

[10] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Transactions on Signal Processing*, vol. 63, no. 24, pp. 6510–6523, 2015. 11

[11] S. Chen, A. Sandryhaila, J. M. Moura, and J. Kovačević, "Signal recovery on graphs: Variation minimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 17, pp. 4609–4624, 2015. 11

[12] D. Romero, M. Ma, and G. B. Giannakis, "Kernel-based reconstruction of graph signals," *IEEE Transactions on Signal Processing*, vol. 65, no. 3, pp. 764–778, 2016. 11

[13] A. Elmoataz, O. Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1047–1060, 2008. 11

[14] M. Belkin, I. Matveeva, and P. Niyogi, "Regularization and semi-supervised learning on large graphs," in *International Conference on Computational Learning Theory*, pp. 624–638, Springer, 2004. 11, 17, 38

[15] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the 20th International Conference on Machine Learning*, pp. 912–919, AAAI Press, 2003. 11, 17, 38, 39

[16] Y.-X. Wang, J. Sharpnack, A. J. Smola, and R. J. Tibshirani, "Trend filtering on graphs," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 3651–3691, 2016. 11, 12, 13, 14, 16, 17, 23, 25, 26, 27, 39

[17] S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky, "ℓ_1 Trend Filtering," *SIAM Review*, vol. 51, no. 2, pp. 339–360, 2009. 11, 13, 18

[18] P. Bühlmann and S. van de Geer, *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Science & Business Media, 2011. 11

[19] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001. 11, 19

[20] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *The Annals of Statistics*, vol. 38, no. 2, pp. 894–942, 2010. 11, 19

[21] P.-L. Loh and M. J. Wainwright, "Regularized M-estimators with nonconvexity: Statistical and algorithmic theory for local optima," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 559–616, 2015. 12, 15, 19

[22] P.-L. Loh, "Statistical consistency and asymptotic normality for high-dimensional robust M-estimators," *The Annals of Statistics*, vol. 45, no. 2, pp. 866–896, 2017. 12

[23] C.-H. Zhang and T. Zhang, "A general theory of concave regularization for high-dimensional sparse estimation problems," *Statistical Science*, vol. 27, no. 4, pp. 576–593, 2012. 12, 19

[24] P. Breheny and J. Huang, "Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection," *The Annals of Applied Statistics*, vol. 5, no. 1, pp. 232–253, 2011. 12

[25] S. Ma and J. Huang, "A concave pairwise fusion approach to subgroup analysis," *Journal of the American Statistical Association*, vol. 112, no. 517, pp. 410–423, 2017. 12, 110

[26] S. Chen, F. Cerda, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovačević, "Semi-supervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring," *IEEE Transactions on Signal Processing*, vol. 62, no. 11, pp. 2879–2893, 2014. 12

[27] D. Hallac, J. Leskovec, and S. Boyd, "Network lasso: Clustering and optimization in large graphs," in *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 387–396, ACM, 2015. 12, 14

[28] A. Jung, A. O. Hero III, A. Mara, and S. Jahromi, "Semi-supervised learning via sparse label propagation," *arXiv preprint arXiv:1612.01414*, 2016. 12

[29] A. Jung, N. Tran, and A. Mara, "When is network lasso accurate?," *Frontiers in Applied Mathematics and Statistics*, vol. 3, pp. 1–11, 2018. 12, 14

[30] N. Tran, S. Basirian, and A. Jung, "When is network lasso accurate: The vector case," *arXiv preprint arXiv:1710.03942*, 2017. 12, 14

[31] E. Mammen and S. van de Geer, "Locally adaptive regression splines," *The Annals of Statistics*, vol. 25, no. 1, pp. 387–413, 1997. 13

[32] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal

algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992. 13

[33] T. F. Chan, S. Osher, and J. Shen, "The digital TV filter and nonlinear denoising," *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 231–241, 2001. 13

[34] F. Mahmood, N. Shahid, U. Skoglund, and P. Vandergheynst, "Adaptive graph-based total variation for tomographic reconstructions," *IEEE Signal Processing Letters*, vol. 25, no. 5, pp. 700–704, 2018. 13

[35] R. J. Tibshirani, "Adaptive piecewise polynomial estimation via trend filtering," *The Annals of Statistics*, vol. 42, no. 1, pp. 285–323, 2014. 13

[36] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 1, pp. 91–108, 2005. 13

[37] J. Sharpnack, A. Singh, and A. Rinaldo, "Sparsistency of the edge lasso over graphs," in *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, vol. 22, pp. 1028–1036, PMLR, 2012. 14

[38] Z. Harchaoui and C. Lévy-Leduc, "Multiple change-point estimation with a total variation penalty," *Journal of the American Statistical Association*, vol. 105, no. 492, pp. 1480–1493, 2010. 14

[39] J.-C. Hütter and P. Rigollet, "Optimal rates for total variation denoising," in *29th Annual Conference on Learning Theory*, vol. 49, pp. 1115–1146, PMLR, 2016. 14, 22, 23

[40] A. S. Dalalyan, M. Hebiri, and J. Lederer, "On the prediction performance of the lasso," *Bernoulli*, vol. 23, no. 1, pp. 552–581, 2017. 14

[41] K. Lin, J. Sharpnack, A. Rinaldo, and R. J. Tibshirani, "Approximate Recovery in Changepoint Problems, from ℓ_2 Estimation Error Rates," *arXiv preprint arXiv:1606.06746*, 2016. 14, 27

[42] A. Jung and M. Hulsebos, "The network nullspace property for compressed sensing of big data over networks," *Frontiers in Applied Mathematics and Statistics*, vol. 4, p. 9, 2018. 14

[43] S. A. van de Geer and P. Bühlmann, "On the conditions used to prove oracle results for the lasso," *Electronic Journal of Statistics*, vol. 3, pp. 1360–1392, 2009. 14, 23

[44] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, 2005. 15

[45] J. Chen and X. Huo, "Theoretical results on sparse representations of multiple-measurement vectors," *IEEE Transactions on Signal Processing*, vol. 54, no. 12, pp. 4634–4643, 2006. 15

[46] Y. C. Eldar, P. Kuppinger, and H. Bolcskei, "Block-sparse signals: Uncertainty relations and efficient recovery," *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3042–3054, 2010. 15

[47] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online Learning for Matrix Factorization and Sparse Coding," *Journal of Machine Learning Research*, vol. 11, pp. 19–60, Mar. 2010. 15

[48] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Hyperspectral Image Classification Using Dictionary-Based Sparse Representation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, pp. 3973–3985, Oct. 2011. 15

[49] Y. Li and Y. Chi, "Off-the-Grid Line Spectrum Denoising and Estimation With Multiple Measurement Vectors," *IEEE Transactions on Signal Processing*, vol. 64, pp. 1257–1269, Mar. 2016. 15

[50] Y. C. Eldar and M. Mishali, "Robust Recovery of Signals From a Structured Union of Subspaces," *IEEE Transactions on Information Theory*, vol. 55, pp. 5302–5316, Nov. 2009. 15

[51] M. E. Davies and Y. C. Eldar, "Rank Awareness in Joint Sparse Recovery," *IEEE Transactions on Information Theory*, vol. 58, pp. 1135–1146, Feb. 2012. 15

[52] P.-L. Loh and M. J. Wainwright, "Support recovery without incoherence: A case for nonconvex regularization," *The Annals of Statistics*, vol. 45, no. 6, pp. 2455–2482, 2017. 15

[53] L. Chen and Y. Gu, "The convergence guarantees of a non-convex approach for sparse recovery," *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 3754–3767, 2014. 15, 19

[54] R. Chartrand and V. Staneva, "Restricted isometry properties and nonconvex compressive sensing," *Inverse Problems*, vol. 24, no. 3, pp. 20–35, 2008. 15

[55] K. Ji, J. Tan, J. Xu, and Y. Chi, "Learning latent features with pairwise penalties in low-rank matrix completion," *IEEE Transactions on Signal Processing*, vol. 68, pp. 4210–4225, 2020. 15

[56] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, pp. 585–591, MIT Press, 2001. 17

[57] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003. 17

[58] P. P. Talukdar and K. Crammer, "New regularized algorithms for transductive learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 442–457, Springer, 2009. 17, 38, 39

[59] R. J. Tibshirani, *The Solution Path of the Generalized Lasso*. Stanford University, 2011. 18

[60] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, pp. 267–288, 1996. 18

[61] C.-H. Zhang and J. Huang, "The sparsity and bias of the lasso selection in high-dimensional linear regression," *The Annals of Statistics*, vol. 36, no. 4, pp. 1567–1594, 2008. 18

[62] F. Chung and M. Radcliffe, "On the spectra of general random graphs," *The Electronic Journal of Combinatorics*, vol. 18, no. 1, p. 215, 2011. 26, 27

[63] F. R. Chung and F. C. Graham, *Spectral Graph Theory*. American Mathematical Society, 1997. 26

[64] A. Blum, J. Hopcroft, and R. Kannan, "Foundations of data science," *Cambridge University Press*, 2020. 27

[65] A. Lubotzky, R. Phillips, and P. Sarnak, "Ramanujan graphs," *Combinatorica*, vol. 8, no. 3, pp. 261–277, 1988. 27

[66] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization

and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011. 28, 44

[67] J. Huang, P. Breheny, and S. Ma, "A selective review of group selection in high-dimensional models," *Statistical science: a review journal of the Institute of Mathematical Statistics*, vol. 27, no. 4, 2012. 29

[68] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on Machine Learning*, pp. I–115–I–123, JMLR.org, 2013. 30

[69] M. Defferrard, L. Martin, R. Pena, and N. Perraudin, "PyGSP: Graph Signal Processing in Python." 30, 32

[70] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference (SciPy 2008)*, pp. 11–16, 2008. 30

[71] G. Boeing, "OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers, Environment and Urban Systems*, vol. 65, pp. 126–139, 2017. 36

[72] Taxi and Limousine Commission (TLC), "2011 yellow taxi trip data." 36

[73] Socrata API, "2011 yellow taxi trip data." 36

[74] "2011 Pride Guide." 36

[75] D. Dua and C. Graff, "UCI machine learning repository," 2017. 39

[76] C. A. Bateson, G. P. Asner, and C. A. Wessman, "Endmember bundles: A new approach to incorporating endmember variability into spectral mixture analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 38, no. 2, pp. 1083–1094, 2000. 42

[77] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 2, pp. 354–379, 2012. 42, 48

[78] R. Heylen, M. Parente, and P. Gader, "A review of nonlinear hyperspectral unmixing methods," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 1844–1868, 2014. 42

[79] L. Drumetz, M. A. Veganzones, S. Henrot, R. Phlypo, J. Chanussot, and C. Jutten, "Blind hyperspectral unmixing using an extended linear mixing model to address spectral variability," *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3890–3905, 2016. 43

[80] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, p. 788, 1999. 43

[81] P. Sajda, S. Du, T. Brown, L. Parra, and R. Stoyanova, "Recovery of constituent spectra in 3D chemical shift imaging using nonnegative matrix factorization," in *4th International Symposium on Independent Component Analysis and Blind Signal Separation*, pp. 71–76, 2003. 43

[82] V. P. Pauca, J. Piper, and R. J. Plemmons, "Nonnegative matrix factorization for

spectral data analysis," *Linear Algebra and its Applications*, vol. 416, no. 1, pp. 29–47, 2006. 43

[83] A. Cichocki, R. Zdunek, A. H. Phan, and S. I. Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009. 43

[84] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Sparse unmixing of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 2014–2039, 2011. 43

[85] W. He, H. Zhang, and L. Zhang, "Sparsity-regularized robust non-negative matrix factorization for hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 9, pp. 4267–4279, 2016. 43

[86] D. C. Heinz and C. I. Chang, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 3, pp. 529–545, 2001. 43, 59

[87] Y. Qian, S. Jia, J. Zhou, and A. Robles-Kelly, "Hyperspectral unmixing via $L_{1/2}$ sparsity-constrained nonnegative matrix factorization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4282–4297, 2011. 43

[88] L. Drumetz, T. R. Meyer, J. Chanussot, A. L. Bertozzi, and C. Jutten, "Hyperspectral image unmixing with endmember bundles and group sparsity inducing mixed norms," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3435–3450, 2019. 43, 59, 60

[89] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3042–3054, 2014. 44, 46

[90] R. Ammanouil, A. Ferrari, and C. Richard, "Hyperspectral data unmixing with graph-based regularization," in *7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pp. 1–4, June 2015. 44

[91] K. Benzi, V. Kalofolias, X. Bresson, and P. Vandergheynst, "Song recommendation with non-negative matrix factorization and graph total variation," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2439–2443, 2016. 44, 46

[92] Y. Van-Gennip and A. L. Bertozzi, "Gamma-convergence of graph Ginzburg-Landau functionals," *Advances in Differential Equations*, vol. 17, no. 11/12, pp. 1115–1180, 2012. 44

[93] J. A. Dobrosotskaya and A. L. Bertozzi, "A wavelet-Laplace variational technique for image deconvolution and inpainting," *IEEE Transactions on Image Processing*, vol. 17, no. 5, pp. 657–663, 2008. 44, 55

[94] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the Nyström method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, 2004. 44, 47, 50

[95] R. Glowinski and A. Marrocco, "Sur l'approximation, par elements finis d'ordre un, et la resolution, par, penalisation-dualité, d'une classe de problems de Dirichlet non lineares," *Revue Française d'Automatique, Informatique, et Recherche Opéationelle*, vol. 9, pp. 41–76, 1975. 44

[96] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximations," *Computers & mathematics with applications*, vol. 2, pp. 17–40, 1976. 44

[97] B. Merriman, J. K. Bence, and S. J. Osher, "Motion of multiple junctions: A level set approach," *Journal of Computational Physics*, vol. 112, no. 2, pp. 334–363, 1994. 44, 52

[98] E. Merkurjev, J. Sunu, and A. L. Bertozzi, "Graph MBO method for multiclass segmentation of hyperspectral stand-off detection video," in *IEEE International Conference on Image Processing*, pp. 689–693, 2014. 44, 48, 51, 52

[99] Z. Meng, E. Merkurjev, A. Koniges, and A. L. Bertozzi, "Hyperspectral image classification using graph clustering methods," *Image Processing On Line*, vol. 7, pp. 218–245, 2017. 45, 48, 50, 52, 56

[100] L. I. Rudin, S. J. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992. 45

[101] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Total variation spatial regularization for sparse hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4484–4502, 2012. 45, 58, 59, 60

[102] Z. Guo, T. Wittman, and S. J. Osher, "L1 unmixing and its application to hyperspectral image enhancement," in *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XV*, vol. 7334, p. 73341M, International Society for Optics and Photonics, 2009. 46

[103] W. He, H. Zhang, and L. Zhang, "Total variation regularized reweighted sparse nonnegative matrix factorization for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3909–3921, 2017. 46

[104] F. Xiong, Y. Qian, J. Zhou, and Y. Y. Tang, "Hyperspectral unmixing via total variation regularized nonnegative tensor factorization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 4, pp. 2341–2357, 2018. 46

[105] Y. Yuan, Z. Zhang, and Q. Wang, "Improved collaborative non-negative matrix factorization and total variation for hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 998–1010, 2020. 46

[106] J. Li, J. M. Bioucas-Dias, A. Plaza, and L. Liu, "Robust collaborative nonnegative matrix factorization for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6076–6090, 2016. 46

[107] L. Zhuang, C.-H. Lin, M. A. Figueiredo, and J. M. Bioucas-Dias, "Regularization parameter selection in minimum volume hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 12, pp. 9858–9877, 2019. 46, 59, 60

[108] G. Gilboa and S. J. Osher, "Nonlocal operators with applications to image processing," *Multiscale Modeling & Simulation*, vol. 7, no. 3, pp. 1005–1028, 2008. 46

[109] Y. Lou, X. Zhang, S. J. Osher, and A. L. Bertozzi, "Image recovery via nonlocal operators," *Journal of Scientific Computing*, vol. 42, no. 2, pp. 185–197, 2010. 46

[110] W. Zhu, V. Chayes, A. Tiard, S. Sanchez, D. Dahlberg, A. L. Bertozzi, S. J. Osher, D. Zosso, and D. Kuang, "Unsupervised classification in hyperspectral imagery with

nonlocal total variation and primal-dual hybrid gradient algorithm," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 5, pp. 2786–2798, 2017. 46

[111] J. Yao, D. Meng, Q. Zhao, W. Cao, and Z. Xu, "Nonconvex-sparsity and nonlocal-smoothness-based blind hyperspectral unmixing," *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 2991–3006, 2019. 46

[112] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548–1560, 2010. 46

[113] F. Zhu, Y. Wang, S. Xiang, B. Fan, and C. Pan, "Structured sparse method for hyperspectral unmixing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 88, pp. 101–118, 2014. 46

[114] X. Lu, H. Wu, Y. Yuan, P. Yan, and X. Li, "Manifold regularized sparse NMF for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 5, pp. 2815–2826, 2012. 46, 58, 59

[115] M. Li, F. Zhu, A. J. X. Guo, and J. Chen, "A graph regularized multilinear mixing model for nonlinear hyperspectral unmixing," *Remote Sensing*, vol. 11, no. 19, p. 2188, 2019. 46

[116] X. Zhang, C. Li, J. Zhang, Q. Chen, J. Feng, L. Jiao, and H. Zhou, "Hyperspectral unmixing via low-rank representation with space consistency constraint and spectral library pruning," *Remote Sensing*, vol. 10, no. 2, p. 339, 2018. 46

[117] E. Esser, M. Moller, S. Osher, G. Sapiro, and J. Xin, "A convex model for nonnegative matrix factorization and dimensionality reduction on physical space," *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3239–3252, 2012. 47

[118] H. Hu, J. Sunu, and A. L. Bertozzi, "Multi-class graph Mumford-Shah model for plume detection using the MBO scheme," in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 209–222, Springer, 2015. 48, 51

[119] A. L. Bertozzi and A. Flenner, "Diffuse interface models on graphs for classification of high dimensional data," *SIAM Review*, vol. 58, no. 2, pp. 293–328, 2016. 51

[120] S. Esedog and Y. H. R. Tsai, "Threshold dynamics for the piecewise constant Mumford–Shah functional," *Journal of Computational Physics*, vol. 211, no. 1, pp. 367–384, 2006. 51

[121] R. V. Kohn and P. Sternberg, "Local minimisers and singular perturbations," *Proceedings of the Royal Society of Edinburgh Section A: Mathematics*, vol. 111, no. 1-2, pp. 69–84, 1989. 51

[122] C. Garcia-Cardona, E. Merkurjev, A. L. Bertozzi, A. Flenner, and A. G. Percus, "Multiclass data segmentation using diffuse interface methods on graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1600–1613, 2014. 51, 52

[123] G. Iyer, J. Chanussot, and A. L. Bertozzi, "A graph-based approach for feature extraction and segmentation of multimodal images," in *IEEE International Conference on Image Processing*, pp. 3320–3324, 2017. 51

[124] G. Iyer, J. Chanussot, and A. L. Bertozzi, "A graph-based approach for data fusion and segmentation of multimodal images," *IEEE Transactions on Geoscience and Remote Sensing*, 2020. 51

[125] W. Wang and M. A. Carreira-Perpinán, "Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application," *arXiv preprint arXiv:1309.1541*, 2013. 54

[126] J. Eckstein and D. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1-3, pp. 293–318, 1992. 56

[127] J. M. Nascimento and J. M. Dias, "Vertex component analysis: A fast algorithm to unmix hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 4, pp. 898–910, 2005. 59

[128] F. Zhu, "Hyperspectral unmixing: Ground truth labeling, datasets, benchmark performances and survey," *arXiv preprint arXiv:1708.05125*, 2017. 62

[129] S. Kumar, M. Mohri, and A. Talwalkar, "Sampling methods for the Nyström method," *Journal of Machine Learning Research*, vol. 13, no. Apr, pp. 981–1006, 2012. 68

[130] Z. Meng, A. Koniges, Y. H. He, S. Williams, T. Kurth, B. Cook, J. Deslippe, and A. L. Bertozzi, "OpenMP Parallelization and Optimization of Graph-based Machine Learning Algorithms," pp. 17–31, Springer, 2016. 68

[131] E. A. Ashley, "Towards precision medicine," *Nature Reviews Genetics*, vol. 17, no. 9, p. 507, 2016. 75

[132] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017. 75

[133] E. Dobriban and Y. Sheng, "Wonder: Weighted one-shot distributed ridge regression in high dimensions.," *Journal of Machine Learning Research*, vol. 21, no. 66, pp. 1–52, 2020. 77, 78, 86

[134] E. Dobriban and Y. Sheng, "Distributed linear regression by averaging," *The Annals of Statistics*, vol. 49, no. 2, pp. 918–943, 2021. 78

[135] Y. Zhang, J. C. Duchi, and M. J. Wainwright, "Communication-efficient algorithms for statistical optimization," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3321–3363, 2013. 78

[136] J. Fan, D. Wang, K. Wang, and Z. Zhu, "Distributed estimation of principal eigenspaces," *Annals of statistics*, vol. 47, no. 6, p. 3009, 2019. 78, 93

[137] V. Charisopoulos, A. R. Benson, and A. Damle, "Communication-efficient distributed eigenspace estimation," *arXiv preprint arXiv:2009.02436*, 2020. 78

[138] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *arXiv preprint arXiv:2002.10619*, 2020. 78

[139] S. S. Du, W. Hu, S. M. Kakade, J. D. Lee, and Q. Lei, "Few-shot learning via learning the representation, provably," *arXiv preprint arXiv:2002.09434*, 2020. 78

[140] N. Tripuraneni, C. Jin, and M. I. Jordan, "Provable meta-learning of linear representations," *arXiv preprint arXiv:2002.11684*, 2020. 78

[141] S. Wu, H. R. Zhang, and C. Ré, "Understanding and improving information transfer in multi-task learning," *arXiv preprint arXiv:2005.00944*, 2020. 78

[142] R. Nassif, S. Vlaski, C. Richard, J. Chen, and A. H. Sayed, "Multitask learning over graphs: An approach for distributed, streaming machine learning," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 14–25, 2020. 78

[143] W. Wang, J. Wang, M. Kolar, and N. Srebro, "Distributed stochastic multi-task learning with graph regularization," *arXiv preprint arXiv:1802.03830*, 2018. 78

[144] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 4427–4437, 2017. 78

[145] S. Liu, S. J. Pan, and Q. Ho, "Distributed multi-task relationship learning," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 937–946, 2017. 78

[146] D. Richards, S. N. Negahban, and P. Rebeschini, "Decentralised sparse multi-task regression," *arXiv preprint arXiv:1912.01417*, 2019. 78

[147] I. Yamane, F. Yger, M. Berar, and M. Sugiyama, "Multitask principal component analysis," in *Asian Conference on Machine Learning*, pp. 302–317, PMLR, 2016. 78, 94

[148] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," *arXiv preprint arXiv:2002.07948*, 2020. 78

[149] A. Maurer, "The rademacher complexity of linear transformation classes," in *International Conference on Computational Learning Theory*, pp. 65–78, Springer, 2006. 79

[150] L. Balzano, Y. Chi, and Y. M. Lu, "Streaming PCA and subspace tracking: The missing data case," *Proceedings of the IEEE*, vol. 106, no. 8, pp. 1293–1310, 2018. 96

[151] C. Gao and J. D. Welch, "Iterative refinement of cellular identity from single-cell data using online learning," *bioRxiv preprint bioRxiv:2020.01.16.909861*, 2020. 102

[152] S. Chen, Y. C. Eldar, and L. Zhao, "Graph unrolling networks: Interpretable neural networks for graph signal denoising," *arXiv preprint arXiv:2006.01301*, 2020. 102

[153] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *2013 IEEE Global Conference on Signal and Information Processing*, pp. 945–948, IEEE, 2013. 102

[154] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (red)," *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1804–1844, 2017. 102

[155] M. J. Wainwright, *High-dimensional statistics: A non-asymptotic viewpoint*, vol. 48. Cambridge University Press, 2019. 106

[156] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of Optimization Theory and Applications*, vol. 109, no. 3, pp. 475–494, 2001. 111

[157] R. Elyanow, B. Dumitrascu, B. E. Engelhardt, and B. J. Raphael, "netNMF-sc: Leveraging gene–gene interactions for imputation and dimensionality reduction in single-cell expression analysis," *Genome research*, vol. 30, no. 2, pp. 195–204, 2020. 113

[158] C. Gao and J. D. Welch, "Iterative refinement of cellular identity from single-cell data using online learning," *bioRxiv*, 2020. 113

[159] D. P. Palomar and J. R. Fonollosa, "Practical algorithms for a family of waterfilling

solutions," *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 686–695, 2005.

126