Provably Learning from Data: New Algorithms and Models for Matrix and Tensor Decompositions

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Sirisha Rambhatla

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Prof. Jarvis Haupt

September, 2019

Acknowledgements

It is a beautiful thing to come across an idea worth pursuing. However, after the initial euphoria, it is just hard work and assiduity that nurture it and bring it to life. I find myself to be extremely lucky to have come across such ideas, to have had the leisure to devote time to inquiry and discovery, and to have made it past the "finish line" (if there is such a thing). For the gift of this journey, I thank my advisor, Prof. Jarvis Haupt.

I have also been extremely fortunate to have Prof. Georgios Giannakis, Prof. Nikos Papanikolopoulos, and Prof. Mingyi Hong on my Ph.D. exam committee. Their feedback essentially changed what I planned to do with my doctoral degree. I am also fortunate that my time here overlapped with Prof. Nikos Sidiropoulos, who has motivated my work directly and indirectly through his teaching and encouragement.

I would also like to thank my lab-mates over the years, who kept the atmosphere in the lab collaborative and welcoming. Numerous "I have a question..." were always met with an excited "Sure!". Special mentions include Xingguo, Mojtaba, Swayambhoo, Alex, Abhinav, Gamini, and Akshay. Also special thanks to Minnesota Supercomputing Institue (MSI) for their super-computing resources which made my work possible.

I also take this opportunity to thank my family – my mom, dad, sister, and brother-in-law, my very cheerful in-laws, and the rest of the extended family. Special thanks to my sister for sending me pictures and videos of my nephew and niece.

I am grateful to my friends Aditi, Congnan, Jen, Kadambari, Pavan, Anubhav, Vikrant, Anki, and Jared, who kept me in check with reality, and provided me with much needed excuses to have fun. Special thanks to Shelley and Miles, my mentors at Robins Kaplan for their support and encouragement. I'd also like to thank the wonderful folks at Alma Cafe (especially Cameron) for their kindness and exceptional espressos, and the Hennepin County Library system for their extensive collection of books.

To the super-wise (and calm) person who listened to the daily ups and downs, my companion on this journey: Yash – Thank you!

To the not known.

Abstract

Learning and leveraging patterns in data has fueled the recent advances in data driven services. As these solutions become more ubiquitous, and get incorporated into critical applications in healthcare and transportation, there is an increasing need to understand the limits of these learning algorithms and to develop algorithms with guarantees. Moreover, with data being generated at unprecedented rates, these algorithms need to be fast, learn on-the-fly (online), handle large volumes of data (scalable), and be computationally efficient, while possessing guarantees on their behavior. Furthermore, to make the learning-based products widely applicable there is also a need to make their reasoning and decision making process transparent (interpretable).

These challenges inspire and motivate this dissertation. Specifically, we focus on analyzing various matrix/tensor demixing and factorization tasks, where we leverage the inherent interpretability endowed by the structure of problem (such as sparsity and low-rankness) to characterize the (theoretical) conditions for successful recovery, and analyze their performance in real-world settings.

To this end, we make contributions on three fronts. First, we develop algorithm-aware theoretical guarantees for sparse matrix and tensor factorization tasks. Second, we establish algorithm-agnostic theoretical results for matrix demixing models and demonstrate their applications on real-world datasets. Lastly, we develop application-specific techniques for navigation and source separation. Bringing together Algorithms, Theory, and Applications, the techniques and theoretical results developed as part of this dissertation facilitate and motivate future explorations into the inner workings of learning algorithms for their safe use in critical applications.

Contents

Acknowledgements i

Abstract iii

List of Tables x

List of Figures xiii

1 Introduction 1

1.1 Motivation 1

1.2 Doctoral Research Contributions 4

1.2.1 Early Motivations: Semi-blind source separation 4

1.2.2 Dictionary-based generalization of Robust PCA 6

1.2.3 Provable Algorithm for Dictionary learning 7

1.2.4 Provable Structured Tensor Factorization 8

1.2.5 Lidar-Based Topological Mapping and Localization 8

1.3 Organization 9

1.4 Notation 9

I Algorithm-Aware Matrix and Tensor Factorization 11

2 Provable Online Dictionary Learning and Sparse Coding 12

2.1 Overview 12

2.2 Introduction 13

2.2.1 Summary of Our Contributions 14

2.2.2 Related Works 15

2.3 Algorithm 17
2.4 Main Result 19
2.5 Neural implementation of NOODL 22
2.6 Experiments 24
2.6.1 Convergence Analysis 24
2.6.2 Phase transitions 25
2.7 Future Work 25
2.8 Conclusions 26

Appendices **27**

2.A Summary of Notation 27
2.B Proof of Theorem 2.1 27
2.C Appendix: Proof of Lemmas 35
2.D Appendix: Proofs of intermediate results 53
2.E Additional Experimental Results 71
2.E.1 Coefficient Recovery 72
2.E.2 Computational Time 72
2.F Appendix: Standard Results 74
2.F.1 Concentration results 75
2.F.2 Results from (Arora et al., 2015) 76

3 Provable Structured Tensor Factorization via Dictionary Learning **77**

3.1 Overview 77
3.2 Introduction 78
3.2.1 Overview of the algorithm 79
3.2.2 Contributions 80
3.2.3 Related works 81
3.3 Problem Formulation 83
3.4 Algorithm 84
3.5 Main Result 86
3.6 Numerical Simulations 90
3.7 Discussion and Conclusions 91

Appendices 93

3.A Summary of Notation 93

3.B Proof of Theorem 1 93

3.C Proof of Intermediate Results 101

3.D Additional Theoretical Results 106

3.E Experimental Set-up and Additional Experimental Results 107

3.E.1 Experimental Set-up 107

3.E.2 Additional Results 110

II Algorithm-Agnostic Matrix Demixing 114

4 Dictionary-based Generalization of Robust PCA 115

4.1 Overview 115

4.2 Introduction 115

4.2.1 Background 117

4.2.2 Our Contributions 118

4.3 Preliminaries 120

4.3.1 Optimality of the Solution Pair 120

4.3.2 Conditions on the Dictionary 120

4.3.3 Relevant Subspaces 121

4.3.4 Incoherence Measures and Other Parameters 122

4.4 Main Results 124

4.4.1 Exact Recovery for Entry-wise Sparsity Case 124

4.4.2 Exact Recovery for Column-wise Sparsity Case 125

4.5 Numerical Simulations 127

4.5.1 Entry-Wise Sparsity Case 128

4.5.2 Column-wise Sparsity Case 130

4.6 Conclusions and Future Work 131

Appendices 133

4.A Summary of Notation 133

4.B Proof of Main Results 133

4.B.1 Proofs for Entry-wise Case: Proof of Theorem 4.1 133

4.B.2 Proofs for Column-wise Case: Proof of Theorem 4.2 139

4.C Proofs of Intermediate Results 142
4.C.1 Proofs for Entry-wise Case 142
4.C.2 Proofs for Column-wise Case 149

5 Target Localization in Hyperspectral Images 155
5.1 Overview 155
5.2 Introduction 156
5.2.1 Model 156
5.2.2 Our Contributions 157
5.2.3 Prior Art 159
5.2.4 Related Techniques 160
5.3 Problem Formulation 162
5.3.1 Optimization problems 162
5.3.2 Conditions on the Dictionary 164
5.3.3 Relevant Subspaces 164
5.3.4 Incoherence Measures 165
5.4 Theoretical Results 167
5.4.1 Exact Recovery for Entry-wise Sparsity Case 167
5.4.2 Recovery for Column-wise Sparsity Case 168
5.5 Algorithmic Considerations 170
5.5.1 Background 170
5.5.2 Discussion of Algorithm 4 170
5.5.3 Parameter Selection 172
5.6 Experimental Evaluation 174
5.6.1 Data 174
5.6.2 Dictionary 175
5.6.3 Experimental Setup 177
5.6.4 Parameter Setup for the Algorithms 179
5.6.5 Analysis 182
5.7 Conclusions 184

III Application-Focused Techniques 185

6 Lidar-Based Topological Mapping and Localization via Tensor Decompositions 186

6.1 Overview 186

6.2 Introduction 187

6.2.1 Prior-Art 188

6.2.2 Summary of Our Technique 188

6.2.3 Our Contributions 189

6.3 Problem formulation 190

6.3.1 Modeling Lidar data as a Tensor 190

6.3.2 Building TensorMap 191

6.3.3 Localizing in TensorMap 192

6.3.4 Memory Considerations 192

6.4 Numerical Evaluations 193

6.4.1 Experimental Set-up 193

6.4.2 Selecting the Parameters 194

6.4.3 Results 194

6.4.4 Effect of Gaussian noise and Translations 195

6.4.5 Compression Ratio 196

6.4.6 Other Applications and Future Work 196

6.5 Conclusions 196

IV Tools 198

7 Software Resources 199

7.1 Reproducible Research 199

7.2 Software Packages Developed 199

7.2.1 NOODL: Neurally plausible alternating Optimization-based Online Dictionary Learning 199

7.2.2 TensorNOODL: NOODL for Structured Tensor Decomposition . 200

7.2.3 D-RPCA: Dictionary-based Robust PCA 200

7.2.4 TensorMap: Lidar-Based Topological Mapping and Localization via Tensor Decompositions 200

8 Discussion and Future Work 201
8.1 Discussion . . . 201
8.2 Future Work . . . 202

References 205

Appendix A. Acronyms 222

List of Tables

2.1 Comparison of provable algorithms for dictionary learning. 16

2.A.1 Frequently used symbols 28

2.B.1 Proof map: dependence of results. 36

2.E.1 Final error in recovery of the factors by various techniques and the computation time taken per iteration (in seconds) corresponding to Fig. 2.2 across techniques. We report the coefficient estimate after the HT step (in Arora et al. (2015)) as $\mathbf{X}_{\mathrm{HT}}$. For the techniques presented in Arora et al. (2015), we scan across 50 values of the regularization parameter for coefficient estimation using Lasso after learning the dictionary ($\mathbf{A}$), and report the optimal estimation error for the coefficients ($\mathbf{X}_{\mathrm{Lasso}}$), while for `Mairal` '09, at each step the coefficients estimate is chosen by scanning across 10 values of the regularization parameters. For $k = 100$, the algorithms of Arora et al. (2015) do not converge (shown as N/A). . 73

3.1 Comparison of provable algorithms for tensor factorization and dictionary learning. As shown here, the existing provable tensor factorization techniques do not apply to the case where $\mathbf{A}$: incoherent, $(\mathbf{B}, \mathbf{C})$: sparse. We use dictionary learning to develop a provable algorithm. 81

3.A.1 Frequently used symbols: Definitions of Probabilities 94

3.A.2 Frequently used symbols: Notation and Parameters 95

3.E.1 Choosing the step-size (η_A) for the dictionary update step. The dictionary update step-size parameter (η_A) is set to be the same for `TensorNOODL`, `Arora(b)`, and `Arora(u)` depending upon the choice of rank m, and probabilities (α, β), according to assumption **A.5**. 108

3.E.2 Tensor factorization results $\alpha, \beta = 0.005$ averaged across 3 trials. Here, $T(\text{supp}(\widehat{\mathbf{X}}))$ field shows the number of iterations T to reach the target tolerance, while the categorical field, $\text{supp}(\widehat{\mathbf{X}})$ indicates if the support of the recovered $\widehat{\mathbf{X}}$ matches that of $\mathbf{X}^*$ (Y) or not (N). 111

3.E.3 Tensor factorization results $\alpha, \beta = 0.01$ averaged across 3 trials. Here, $T(\text{supp}(\widehat{\mathbf{X}}))$ field shows the number of iterations T to reach the target tolerance, while the categorical field, $\text{supp}(\widehat{\mathbf{X}})$ indicates if the support of the recovered $\widehat{\mathbf{X}}$ matches that of $\mathbf{X}^*$ (Y) or not (N). 112

3.E.4 Tensor factorization results $\alpha, \beta = 0.05$ averaged across 3 trials. Here, $T(\text{supp}(\widehat{\mathbf{X}}))$ field shows the number of iterations T to reach the target tolerance, while the categorical field, $\text{supp}(\widehat{\mathbf{X}})$ indicates if the support of the recovered $\widehat{\mathbf{X}}$ matches that of $\mathbf{X}^*$ (Y) or not (N). 113

4.A.1 Summary of important notation and parameters 134

5.1 Entry-wise sparsity model for the Indian Pines Dataset. Simulation results are presented for our proposed approach (D-RPCA(E)), robust-PCA based approach on transformed data $\mathbf{D}^{\dagger}\mathbf{M}$ (RPCA†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^{\dagger}\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning procedure ρ; see Algorithm 5 . Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, "$*$" denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance). 179

5.2 Entry-wise sparsity model and Pavia University Dataset. Simulation results are presented for the proposed approach (D-RPCA(E)), robust-PCA based approach on transformed data (RPCA†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^{\dagger}\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning step ρ. Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, "$*$" denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance). 180

5.3 Column-wise sparsity model and Indian Pines Dataset. Simulation results are presented for the proposed approach (D-RPCA(C)), Outlier Pursuit (OP) based approach on transformed data (OP†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^\dagger\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning step ρ. Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, " $*$ " denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance). . . 181

5.4 Column-wise sparsity model and Pavia University Dataset. Simulation results for the proposed approach (D-RPCA(C)), Outlier Pursuit (OP) based approach (OP†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^\dagger\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning step ρ. Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, "$*$" denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance). 182

A.1 Acronyms 222

List of Figures

1.1 Need for interpretable models and algorithms with guarantees [1]. 2

1.2 Overview of research efforts in relation to the three pillars of contemporary learning problems – Algorithms, Theoretical Guarantees, and Applications. The edges indicate specific problems at the intersection of these pillars. My research contributions are highlighted in yellow (hyperlinked to specific chapters of this dissertation). 4

2.1 A neural implementation of NOODL. Panel (a) shows the neural architecture, which consists of three layers: an input layer, a weighted residual evaluation layer (evaluates $\eta_x\big(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\mathbf{x}^{(r)}_{(j)}\big)$), and an output layer. Panel (b) shows the operation of the neural architecture in panel (a). The update of $\mathbf{x}^{(r+1)}_{(j)}$ is given by (2.4). 23

2.2 Comparative analysis of convergence properties. Panels (a-i), (b-i), (c-i), and (d-i) show the convergence of NOODL, `Arora15(‘‘biased’’)`, `Arora15(‘‘unbiased’’)` and `Mairal` ‘09, for different sparsity levels for $n = 1000$, $m = 1500$ and $p = 5000$. Since NOODL also recovers the coefficients, we show the corresponding recovery of the dictionary, coefficients, and overall fit in panels (a-ii), (b-ii), (c-ii), and (d-ii), respectively. Further, panels (e-i) and (e-ii) show the phase transition in samples p (per iteration) with the size of the dictionary m averaged across 10 Monte Carlo simulations for the two factors. Here, $n = 100$, $k = 3$, $\eta_x = 0.2$, $\tau = 0.1$, $\epsilon_0 = 2/\log(n)$, η_A is chosen as per **A.5**. A trial is considered successful if the relative Frobenius error incurred by $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{X}}$ is below 5×10^{-7} after 50 iterations. 25

3.1 The structured tensor $\underline{\mathbf{Z}}\in \mathbb{R}^{n\times J\times K}$ considered in this work. The tensor has a few mode-1 fibers which are dense. 78

3.2 Problem Formulation: The dense columns of the structured tensor $\underline{\mathbf{Z}} \in \mathbb{R}^{n \times J \times K}$ are collected in a matrix ($\mathbf{Y}$). The matrix $\mathbf{Y}$ is viewed as arising from a dictionary learning model. 83

3.1 Dependence induced by the transposed Khatri-Rao structure. 87

3.1 Data samples required by `TensorNOODL` using the number of iterations for convergence (see footnote 6). Panels (a), (b), and (c) show the number of iterations taken by `TensorNOODL` to achieve a target tolerance of 10^{-10} for $\mathbf{A}$ for $J = K = 100$, 300, and 500, respectively across the choices of rank $m = \{50, 150, 300, 450, 600\}$ and $\alpha = \beta = \{0.005, 0.01, 0.05\}$, averaged across three Monte Carlo runs. 91

4.1 Recovery for varying rank of $\mathbf{L}$, sparsity of $\mathbf{S}$ and number of dictionary elements in $\mathbf{D}$ as per Theorem 4.1. Each plot shows average recovery across 10 trials for varying ranks and sparsity up to $s_e^{\max} = m$, where $n = m = 100$ and the white region represents correct recovery. We decide success if $\|\mathbf{L} - \widehat{\mathbf{L}}\|_F / \|\mathbf{L}\|_F \leq 0.02$ and $\|\mathbf{S} - \widehat{\mathbf{S}}\|_F / \|\mathbf{S}\|_F \leq 0.02$, where $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{S}}$ are the recovered $\mathbf{L}$ and $\mathbf{S}$, respectively. Panels (a)-(b) show the recovery of the low-rank part $\mathbf{L}$ and (c)-(d) show the recovery of the sparse part with varying dictionary sizes $d = 5$ and 150, respectively. Also, the predicted trend between rank r and sparsity s_e as per Theorem 4.1, eq.(4.9) is shown in red in panels (a-b). 127

4.2 Recovery for varying rank of $\mathbf{L}$, sparsity of $\mathbf{S}$ and number of dictionary elements in $\mathbf{R}$. Panels (a)-(b) show the recovery of the low-rank part $\mathbf{L}$ and (c)-(d) show the recovery of the sparse part with varying dictionary sizes $d = 5$ and 150, respectively. The experimental set-up and the success metric remains the same as in Fig. 4.1. 128

4.3 Comparison of phase transitions in rank and sparsity between D-RPCA(E) and RPCA† for recovery of $\mathbf{S}$ for different dictionary sizes. Panels (a) and (b) correspond to $d = 5$ and $d = 50$, respectively. Experimental set-up and the success metric remains same as Fig. 4.2. The area in green corresponds to recovery by RPCA† where at least 1 out of the 10 Monte-Carlo trials succeeds. 129

4.4 Phase transitions in rank r and column sparsity s_c across 10 Monte-Carlo simulations. Panels (a), (b), (c) show the precision i.e., (True Positives /(True Positives + False Positives)) in identifying the outlier columns of $\mathbf{S}$ for $d = 50$ using (a) OP and (b) OP†, and D-RPCA(C) for $d = 150$, respectively. In addition, panel (b) also shows the performance by OP† for $d = 50$ in green, marking the region where precision is greater than 0, super imposed over D-RPCA(C). Here, we threshold the column norms of the recovered matrix $\mathbf{S}$ at 2×10^{-3} before computing the precision, and a trial is declared successful if it achieves a precision of 0.99 or higher. . 131

5.1 The HS image data-cube corresponding to the Indian Pines dataset. . . 157

5.2 Correlated spectral signatures. The spectral signatures of even different materials are highly correlated. Shown here are spectral signatures of classes from the Indian Pines dataset (Baumgardner et al., 2015). Here, the shaded region shows the lower and upper ranges of reflectance values the signatures take. 158

5.1 Ground-truth classes in the datasets. Panels (a) and (b) show the ground truth classes for the Indian Pines dataset (Baumgardner et al., 2015) and Pavia University dataset (Gamba, 2002), respectively. 174

5.2 Recovery of the low-rank component $\mathbf{L}$ and the dictionary sparse component $\mathbf{DS}$ for different values of λ for the proposed technique at f = 50-th channel of the (Baumgardner et al., 2015) (shown in panel (a)) corresponding to the results shown in Table 5.1(c). Panel (b) corresponds to the ground truth for class-16. Panel (c) and (d) show the recovery of the low-rank part and dictionary sparse part for a λ at the best operating point. While, panels (e) and (f) show the recovery of these components at $\lambda_e = 85\%$ of $\lambda_e^{\max}$. Here, $\lambda_e^{\max}$ denotes the maximum value λ_e can take; see Section 5.5.3. 183

6.1 The Ford Dataset (Pandey et al., 2011). Panels (a) and (b) show the trajectory traced by the vehicle, and nodes of a representative topological map (in red), respectively. 187

6.2 Learning the topological map. We represent each 3-D point cloud corresponding to each Lidar scan (a), as a matrix (b) after conversion to polar coordinates. We aggregate the matricized scans to form length-k segment tensors $\underline{\mathbf{X}}_{\ell}$, and learn the orthogonal Tucker3 models on each of these (shown in panels (c) and (d)). 189

6.3 Localizing based on a scan. Each test scan, after matricization (as described in Section 6.3.3), is processed by each $\mathbf{U}_{\ell}$ and $\mathbf{V}_{\ell}$ to form "signatures" $\widetilde{\mathbf{G}}_{\ell}$, which are then compared (in Frobenius norm sense) to the core tensors $\underline{\mathbf{G}}_{\ell}$ of TensorMap for best match. 190

6.1 Effect of choice of $\{r_1, r_2, k\}$ on the performance accuracy. Panels (a-d) show the effect of choice of segment lengths k and varying r_1 for fixed $r_2 = 5, 10, 15$, and 25, respectively. Similarly, panels (e-h) show the effect of choice of segment lengths $k$and r_2 for fixed $r_1 = 5, 10, 15$, and 25, respectively. Here, segment lengths k considered are $50, 100, 200, 475$, and 760. Panels (i)-(m) show the nodes for each segment corresponding to choice of k (in red), with the start/end point of the path denoted in green. . . 191

6.1 Performance of TensorMap on the Ford Dataset with $\{r_1, r_2, k\}$ chosen as $\{5, 5, 760\}$, respectively. Panel (a) shows the classification of test scans into segments. The corresponding surrogate for velocity (blue), the decision of vehicle movement (green), and the errors made by TensorMap (red) are shown in panel (b). Notice how majority of the errors occur when the vehicle is stationary. Panels (c) and (f) show the relative error between the original segment tensor and the model learnt by TensorMap. Panel (d) shows the scan classification performance of the technique, actual test set (blue) the closest (Frobenius norm) train set scan found by TensorMap. The corresponding decision of vehicle movement (green) and the errors made (red) are shown in (e). Panel (g) shows the confusion matrix corresponding to the classification of test scans to segments shown in (a), and (h) shows the nodes of TensorMap (red) superimposed on the actual map (blue). 193

6.2 Effect of two types of noise on accuracy. (a) Effect of zero-mean Gaussian noise of variance σ^2, added to each point, and (b) effect of translations (in meters) to the right (simulated). 195

8.1 Future Work: Developing analysis for a one layer Sparse Autoencoder. Figure shows how each sample of the data is processed by first forming a latent space (sparse) representation. Next, the discrepancy between the input $\mathbf{y}$ and the output $\widehat{\mathbf{y}}$ is used to update the weights (dictionary elements). 203

Chapter 1

Introduction

1.1 Motivation

Machine learning and artificial intelligence (AI) related products are emerging as the drivers of the technological and economic development of the next few decades[1]. As these algorithm become the core technologies from being ancillary tools, researchers, lawmakers, and businesses are grappling with how this trend will shape our future.

On one hand where e-commerce services have embraced the advancements in the area, fields like transportation, security, medicine, finance, legal, military, and other critical industries have been cautiously optimistic. The reservations to incorporate latest techniques can be attributed, in part, to the opaque decision making process employed by some of these techniques and the lack of associated theoretical guarantees.

We are only beginning to understand what blackbox learning solutions (e.g. convolutional neural networks) learn (Geirhos et al., 2019); the cautionary tales of using such tools for critical tasks such as melanoma recognition illustrate the challenges (Winkler et al., 2019). Specifically, in such applications the quality of a prediction/decision is also dependent on 1) how well the underlying process is understood, 2) knowing why a solution was reached, 3) when these techniques succeed/fail, and 4) if the algorithm has seen sufficient examples to make a decision. In essence, we need interpretatbility and guarantees on performance.

[1] Louis Columbus "10 Charts That Will Change Your Perspective On Artificial Intelligence's Growth" Forbes(Jan. 12, 2018).

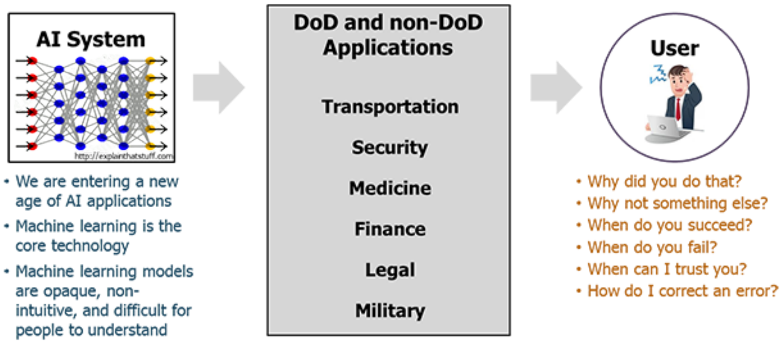

AI System
http://explainthatstuff.com
• We are entering a new age of AI applications
• Machine learning is the core technology
• Machine learning models are opaque, non-intuitive, and difficult for people to understand
DoD and non-DoD Applications
Transportation
Security
Medicine
Finance
Legal
Military
User
• Why did you do that?
• Why not something else?
• When do you succeed?
• When do you fail?
• When can I trust you?
• How do I correct an error?

Figure 1.1: Need for interpretable models and algorithms with guarantees [8].

As a result, notwithstanding the empirical success, understanding learning algorithms and their limitations has emerged as the primary challenge to expand the success of these techniques to a wide array of fields[2]. Recent calls for explainable AI, such as DARPA's "Explainable Artificial Intelligence (XAI)" program (Fig. 1.1) resonate with these goals [3]. Arguably, deploying such techniques in even "non-critical" applications, such as search engines, recommender systems, social-media applications, and chat-bots pose significant risk to civil liberties [4] [5] [6] [7].

Given the potential impact on applications requiring interpretability as well as guarantees for their safe operation, there is an urgent need to develop alternative algorithms and architectures that achieve these goals. To address this gap, we analyze matrix and tensor decomposition models to develop practical algorithms with guarantees for various learning tasks.

We present a case of *Safe AI*, wherein interpretability (built via priors such as low-rankness and sparsity) and guaranteed algorithms are leveraged to bring transparency

[2] Will Knight, "The Dark Secret at the Heart of AI", MIT Tech Review (April 11, 2017).

[3] Dr. Matt Turek, "Explainable Artificial Intelligence (XAI)" Defence Advanced Research Projects Agency (DARPA).

[4] Conor Dougherty, "Google Photos Mistakenly Labels Black People Gorillas", New York Times (July 1, 2015).

[5] Max Fisher and Amanda Taub, "On YouTube's Digital Playground, an Open Gate for Pedophiles", New York Times (June 3, 2019).

[6] Niraj Chokshi, "Facial Recognitions Many Controversies, From Stadium Surveillance to Racist Software", New York Times (May 15, 2019).

[7] Daniel Victor, "Microsoft Created a Twitter Bot to Learn From Users. It Quickly Became a Racist Jerk.", New York Times (March 24, 2016).

[8] DARPA's Explainable Artificial Intelligence (XAI).

and reliability to the decision making process. Our specific contributions aim at establishing guarantees for algorithm-aware and algorithm-agnostic techniques to characterize the conditions under which the quantities of interest can be recovered.

For instance, we consider the inherently non-convex optimization task of matrix factorization (*dictionary learning*) in Section 1.2.3 and Chapter 2, wherein we present a guaranteed scalable online algorithm for this factorization task. Dictionary learning is extensively used in healthcare applications in electroencephalogram (EEG) (Barthélemy et al., 2013), electrocardiogram (ECG) (Mailhé et al., 2009), Magnetic Resonance Imaging (MRI) (Huang et al., 2014), functional MRI (f-MRI) (Lee et al., 2010), and Ultrasound Tomography (UST) (Tosic et al., 2010) for denoising, classification, and clustering tasks. While convex relaxation-based alternating minimization techniques (such as Mairal et al. (2009)) have been a staple for these applications, they only offers limited convergence guarantees.

In addition to explaining the success of popular alternating minimization-based heuristics, our analysis exposes the gaps in existing provable techniques that only focus on recovering one of the factors (the dictionary); these techniques assume that the other factor (the sparse factor) using a separate estimation algorithm *after* dictionary recovery (Arora et al., 2014, 2015; Agarwal et al., 2014). However, since sparse factor estimation is heavily dependent on the dictionary estimate, any non-negligible error in the dictionary can preclude us from recovering the sparse factor (even recovering its *support*). Further since sparse factor recovery can be crucial for a downstream classification and clustering task in critical applications, relying on heuristics with limited guarantees, or provable algorithm which do not provide guarantees for sparse factor recovery may prove to be unrealiable.

To this end, our algorithm recovers both unknown factors (under conditions on the initialization, sparsity and incoherence) by considering the hard-constrained (ℓ_0) task. In addition to providing exact recovery guarantees, it is also scalable and can be implemented in neural architectures; see Section 1.2.3 and Chapter 2 for details. In the next section, we summarize our specific contributions towards our motivating goal of Safe AI.

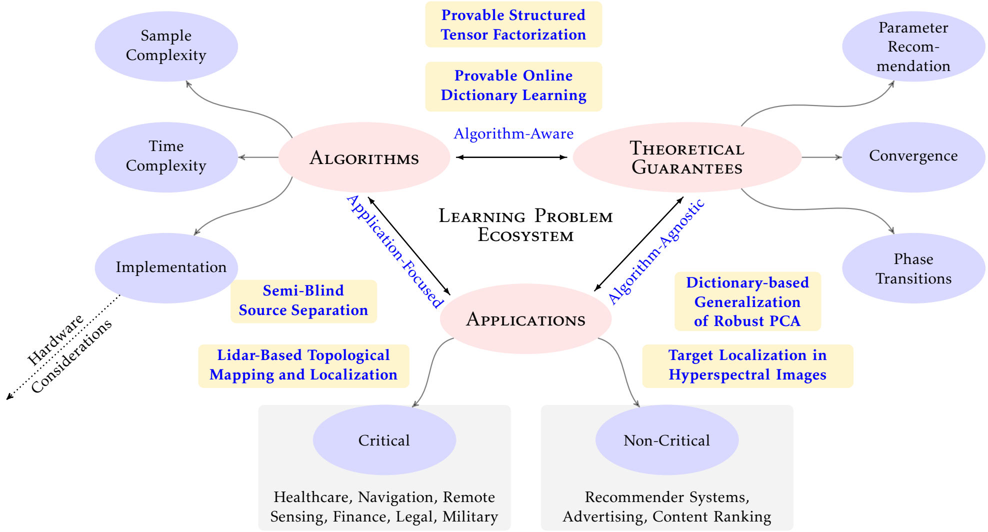

Provable Structured Tensor Factorization
Provable Online Dictionary Learning
Sample Complexity
Parameter Recommendation
Time Complexity
Algorithms
Algorithm-Aware
Theoretical Guarantees
Convergence
Application-Focused
Learning Problem Ecosystem
Algorithm-Agnostic
Implementation
Phase Transitions
Semi-Blind Source Separation
Dictionary-based Generalization of Robust PCA
Applications
Hardware Considerations
Lidar-Based Topological Mapping and Localization
Target Localization in Hyperspectral Images
Critical
Non-Critical
Healthcare, Navigation, Remote Sensing, Finance, Legal, Military
Recommender Systems, Advertising, Content Ranking

Figure 1.2: Overview of research efforts in relation to the three pillars of contemporary learning problems – Algorithms, Theoretical Guarantees, and Applications. The edges indicate specific problems at the intersection of these pillars. My research contributions are highlighted in yellow (hyperlinked to specific chapters of this dissertation).

1.2 Doctoral Research Contributions

Motivated from the learning-algorithm exigency outlined above, my doctoral work touches upon what I identify as the three pillars of the contemporary learning problem ecosystem, namely, 1) Algorithms, 2) Applications, and 3) Theoretical Guarantees, shown in Fig. 1.2. The confluence of each of these gives rise to a different problem paradigm and underscores a particular research focus. For instance, developing algorithms for a particular application (Algorithms + Applications), theoretical guarantees for algorithm agnostic techniques (Applications + Theoretical Guarantees), and finally, provable algorithms (Theoretical Guarantees + Algorithms).

1.2.1 Early Motivations: Semi-blind source separation

The motivation to analyze the properties of learning problems stemmed from my Master's thesis work on a single channel semi-blind source separation task encountered in analysis of the audio signals generated by electro-shock law enforcement devices; see Rambhatla and Haupt (2013a); Rambhatla (2012). The task here was to identify whether the device is delivering current to a subject or not, from a single audio recording captured by a on-board microphone. The state of the device (delivering current or not) is critical in acertaining liability in an incident where the device used. The existing techniques relied on an expert to listen for subtle changes in the charateristic quasi-periodic audio signal (generated by the device's RLC circuit) caused by the change in resistance when the probes are properly attached.

Since these characteristic responses could be simulated in a controlled environment, when deployed in real-world, the microphone also records other background activity involving the altercation, which makes identifying the state of the device challenging. Nevertheless, motivated from the dictionary learning setting and using our knowledge about the characteristic responses under different resistive loads, we posed this problem as a *semi-blind* dictionary learning problem. To this end, we set the known part of the dictionary $\mathbf{D}$ as these characteristic responses (semi-blind), and modeled the given data matrix $\mathbf{M} \in \mathbb{R}^{n \times p}$ as

$$\mathbf{M} = \mathbf{A}\mathbf{X} + \mathbf{D}\mathbf{S}, \tag{1.1}$$

where the the dictionary $\mathbf{A}$ captures the features in the unknown background activity. Specifically, the aim here is to recover the unknown dictionary matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ and the sparse coefficients $\mathbf{X} \in \mathbb{R}^{m \times p}$ and $\mathbf{S} \in \mathbb{R}^{d \times p}$, given an *a priori* known dictionary $\mathbf{D} \in \mathbb{R}^{n \times d}$ by solving the following optimization problem

$$\min_{\mathbf{A}_u, \mathbf{X}} \tfrac{1}{2} \left\| \mathbf{M} - [\mathbf{A} \mid \mathbf{D}] \begin{bmatrix} \mathbf{X} \\ \mathbf{S} \end{bmatrix} \right\|_F^2 + \lambda \left\| \begin{bmatrix} \mathbf{X} \\ \mathbf{S} \end{bmatrix} \right\|_1 .$$

As in the case of the dictionary learning, the optimization formulation shown above is inherently non-convex since both $\mathbf{A}$ and $\mathbf{X}$ are unknown. To this end, we developed an alternating minimization based approach – semi-blind morphological component analysis (SBMCA) (Rambhatla and Haupt, 2013a; Rambhatla, 2012) – which alternates between sparse coefficient recovery and dictionary update to recover the factors.

Notwithstanding the promising empirical results, the alternating minimization-based approach offered no limited theoretical guarantees. As a result, *developing and establishing guarantees on the performance of algorithm–agnostic and –aware learning techniques for their safe use in critical tasks served as the primary motivation and the focus of this dissertation.*

1.2.2 Dictionary-based generalization of Robust PCA

As another variation of the problem described above – motivated from a hyperspectral demixing task – we first considered a closely related convex demixing task, where a data matrix $\mathbf{M}$ is generated via a superposition of a low-rank component $\mathbf{L}$, and a dictionary sparse component $\mathbf{DS}$, wherein the dictionary $\mathbf{D}$ is known *a priori*, i.e.,

$$\mathbf{M} = \mathbf{L} + \mathbf{DS}.$$

The aim here is to recover the rank r low-rank component and the sparse coefficient matrix $\mathbf{S}$. To this end, we studied the conditions on rank and sparsity for which solving the following convex optimization problem recovers the components exactly.

$$\min_{\mathbf{L},\mathbf{S}} \tfrac{1}{2}\|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1 \quad \text{subject to} \quad \mathbf{M} = \mathbf{L} + \mathbf{DS},$$

where the nuclear norm $\|.\|_*$ stands-in as a convex relaxation of the rank constraint, and the ℓ_1-norm for the ℓ_0-"norm".

Originally, studied by Mardani et al. (2013) where the known dictionary $\mathbf{D}$ is *over-complete* or *fat*, i.e. $n \leq d$ with rows of dictionary $\mathbf{D}$ being orthogonal, we extended the results to a case where $\mathbf{D}$ can be both *thin* or *fat*, while removing the orthogonality requirement; see Rambhatla et al. (2016a). As a result, the model became amenable for localizing a target in a hyperspectral image based on its spectral signature Rambhatla et al. (2017a). Further, we also study two sparsity structures, 1) where $\mathbf{S}$ contains a few non-zeros *globally*, and 2) where only a few columns of $\mathbf{S}$ have non-zero elements, i.e., $\mathbf{S}$ is column sparse; see Li et al. (2018a); Rambhatla et al. (2018a,b).

This work and its application in target localization in hyperspectral imaging – discussed in Part II Chapter 4 and Chapter 5, respectively – also revealed a surprising result. *Contrary to the belief that when the known dictionary* $\mathbf{D}$ *is thin one can pre-multiply by the pseudo-inverse* $\mathbf{D}^\dagger$ *of* $\mathbf{D}$ *to transform the problem to that of Robust PCA (Candès et al., 2011), our analysis shows that such a multiplication may make the updated low-rank component* $\mathbf{D}^\dagger\mathbf{L}$ *full-rank (or near full-rank), and hence may no longer follow the model specifications. In other words, we found that for these problems, the concept of "low-rank" is relative to the maximum allowable rank of the matrix and any pre-multiplication/processing may destroy this structure. Our experimental evaluation corroborates this finding, for both sparsity models and we present these in Chapter 4 Section 4.2.2 and 4.5.*

1.2.3 Provable Algorithm for Dictionary learning

In an effort to develop theoretical guarantees for the semi-blind demixing task described above, we began exclusively focusing on investigating the applicability of the recent provable algorithms for dictionary learning (Agarwal et al., 2014; Arora et al., 2014, 2015).

In dictionary learning, the aim is to express given data as a linear combination of a few columns of a matrix (referred to as a *dictionary*), wherein the dictionary ($\mathbf{A}^* \in \mathbb{R}^{n\times m}$) and the weights characterizing the linear combination (referred to as *sparse coefficients*, $\mathbf{x}^*_{(i)} \in \mathbb{R}^m$) are *a priori* unknown, i.e., columns $\mathbf{y}_{(i)} \in \mathbb{R}^n$ of given data matrix $\mathbf{Y} \in \mathbb{R}^{n\times p}$ are generated as

$$\mathbf{y}_{(i)} = \mathbf{A}^*\mathbf{x}^*_{(i)} + \mathbf{w}_{(i)},$$

where $\mathbf{w}_{(i)}$ is the i.i.d. Gaussian noise, and each sparse coefficient vector $\mathbf{x}^*_{(i)}$ contains at most k non-zeros.

In our analysis, we found that the provable techniques mentioned above provide guarantees on the dictionary recovery only Rambhatla et al. (2019). The underlying assumption being that the sparse coefficients can be recovered after dictionary recovery by using a sparse recovery algorithm (e.g. Lasso (Tibshirani, 1996)). Contrary to this belief, any non-negligible error in the dictionary precludes the use of existing sparse recovery results for exact recovery (or even support recovery). As a result, these existing techniques are not viable for recovery of both the dictionary and coefficients.

In the quest for guaranteed coefficient recovery, our analysis revealed the symbiotic relationship between the dictionary and coefficient recovery. Specifically, we show that as opposed to the existing techniques, making progress on the coefficient recovery also helps us to improve our dictionary estimate. With this, we develop an online dictionary learning algorithm – Neurally plausible alternating Optimization-based Online Dictionary Learning (NOODL) – with exact recovery guarantees for both the dictionary and the coefficients. In addition to being suitable for large-scale distributed and *neural* implementations, the algorithm removes an inherent performance-limiting, dimension-dependent bias incurred by the prior-art and has linear convergence guarantees; see Chapter 2 for details.

1.2.4 Provable Structured Tensor Factorization

Our exact recovery results for the dictionary and the coefficients under the dictionary learning model are useful in a number of applications where recovery of coefficients is critical. For instance, we use these for a 3-way tensor factorization task, where the aim is recover the Canonical polyadic (CP) factors of a tensor $\underline{\mathbf{Z}}$, with the CP decomposition of $\underline{\mathbf{Z}}$ defined as

$$\underline{\mathbf{Z}} = \textstyle\sum_{m=1}^{M} \mathbf{A}_m \circ \mathbf{B}_m \circ \mathbf{C}_m.$$

Here, $\mathbf{A} \in \mathbb{R}^{n \times M}$, $\mathbf{B} \in \mathbb{R}^{J \times M}$, and $\mathbf{C} \in \mathbb{R}^{K \times M}$ are the constituent CP factors, where the columns of factor $\mathbf{A}$ are unit norm and obey some incoherence assumptions, and the factors $\mathbf{B}$ and $\mathbf{C}$ are sparse. As a result, the mode-1 unfolding of the tensor $\underline{\mathbf{Z}}$, given by

$$\mathbf{Z}_1^\top = \mathbf{A}\underbrace{(\mathbf{C} \odot \mathbf{B})^\top}_{\mathbf{X}},$$

falls into the dictionary learning setting. Here, "$\odot$" denotes the Khatri-Rao product (column-wise Kronecker product of $\mathbf{C}$ and $\mathbf{B}$).

Leveraging our provable dictionary learning results (Section 1.2.3) we develop a structured tensor factorization algorithm – TensorNOODL – to recover the CP factors of the tensor of interest. Here, the exact coefficient recovery result allows us to untangle the CP factors $\mathbf{C}$ and $\mathbf{B}$ from $\mathbf{X}$ (upto sign and scaling ambiguity) from $\mathbf{X}$. However, our previous dictionary learning results (described in Section 1.2.3) are not directly applicable, and we dedicate a significant portion of the analysis to reconcile the additional dependence that arises due to the Khatri-Rao structure. The details of the analysis are presented in Chapter 3.

1.2.5 Lidar-Based Topological Mapping and Localization

We also develop an algorithm – TensorMap – for building tensor decomposition-based topological maps using Lidar data and to localize in them (Rambhatla et al., 2018c). This rounds-up the third aspect of the learning problem ecosystem, shown in Fig. 1.2, where we develop an algorithm for an application of interest.

In addition to the application in vehicle navigation, our technique provides an efficient way to store a series of Lidar scans (that constitute a map) and leverage the tensor decomposition properties to localize effectively even in feature deficient or slow-changing surroundings. The details of this effort are presented in Chapter 6.

1.3 Organization

We organize our discussion based on the three aspects of the learning problem ecosystems identified in Fig. 1.2, namely – I. Algorithm-Aware, II. Algorithm-Agnostic, and III. Application-Focused approaches. We first detail our algorithm-aware techniques for provable matrix and tensor factorization (introduced in Section 1.2.3 and 1.2.4) in Chapter 2 and Chapter 3 of Part I. Next in Part II of this dissertation, we describe the algorithm-agnostic techniques for a matrix demixing task (Chapter 4)– introduced in Section 1.2.2 – with its application to a target localization task (Chapter 5). Further in Part III (Chapter 6), we present an application-focused technique for building and localizing in Lidar-based topological maps corresponding to our discussion in Section 1.2.5. Finally, we give an overview of the software packages developed as part of this dissertation in Part IV, and conclude this discussion by synthesizing the main takeaways in Chapter 8. We present detailed proofs of our theoretical results in the appendices after each of the individual chapters.

1.4 Notation

We now introduce some common notation used in our discussion. Additional specialized notations are defined where used, and symbols are also summarized in Table 4.A.1, 2.A.1, 3.A.1, and 3.A.2 the Appendices.

Given an integer n, we let $[n] = \{1, 2, \ldots, n\}$. The bold upper-case underlined, bold upper-case, and lower-case letters are used to denote tensors $\underline{\mathbf{M}}$, matrices $\mathbf{M}$ and vectors $\mathbf{v}$, respectively. We denote the i-th column, i-th row, (i,j) element of a matrix, and i-th element of a vector by $\mathbf{M}_i$, $\mathbf{M}_{(i,:)}$, $\mathbf{M}_{ij}$, and $\mathbf{v}_i$ (and $\mathbf{v}(i)$), respectively. The superscript $(\cdot)^{(n)}$ denotes the n-th iterate, while the subscript $(\cdot)_{(n)}$ is reserved for the n-th data sample.

For a matrix $\mathbf{M}$, we use $\|\mathbf{M}\| := \sigma_{\max}(\mathbf{M})$ and $\|\mathbf{M}\|_{\mathrm{F}}$ for the spectral norm and Frobenius norm, respectively, where $\sigma_{\max}(\mathbf{M})$ denotes the maximum singular value of the matrix. Further, we use $\|\mathbf{M}\|_\infty := \max_{i,j}|\mathbf{M}_{ij}|$, $\|\mathbf{M}\|_{\infty,\infty} := \max_i \|\mathbf{e}_i^\top \mathbf{M}\|_1$, and $\|\mathbf{M}\|_{\infty,2} := \max_i \|\mathbf{M}\mathbf{e}_i\|$, where $\mathbf{M}_{i,j}$ denotes the (i,j) element of $\mathbf{M}$ and $\mathbf{e}_i$ denotes the canonical basis vector with 1 at the i-th location and 0 elsewhere. In addition, $\|.\|_*$, $\|.\|_1$, and $\|.\|_{1,2}$ refer to the nuclear norm, entry-wise ℓ_1- norm, and $\ell_{1,2}$ norm (sum of the ℓ_2 norms of the columns) of a matrix, respectively, which serve as convex relaxations of rank, sparsity, and column-wise sparsity, respectively.

Next, given a vector $\mathbf{v}$, we use $\|\mathbf{v}\|$, $\|\mathbf{v}\|_0$, and $\|\mathbf{v}\|_1$ to denote the ℓ_2 norm, ℓ_0 (number of non-zero entries), and ℓ_1 norm, respectively. We also use standard Landau notations $\mathcal{O}(\cdot), \Omega(\cdot)$ $(\widetilde{\mathcal{O}}(\cdot), \widetilde{\Omega}(\cdot))$ to indicate the asymptotic behavior (ignoring logarithmic factors). Further, we use $g(n) = \mathcal{O}^*(f(n))$ to indicate that $g(n) \leq Lf(n)$ for a small enough constant L, which is independent of n. We use $c(\cdot)$ for constants parameterized by the quantities in $(\cdot)$.

We denote the hard-thresholding operator by $\mathcal{T}_\tau(z) := z \cdot \mathbb{1}_{|z|\geq\tau}$, where "$\mathbb{1}$" is the indicator function and τ is the threshold. We use supp$(\cdot)$ for the support (the set of non-zero elements) and sign$(\cdot)$ for the element-wise signum function. Finally, we use $\mathbf{D}_{(\mathbf{v})}$ as a diagonal matrix with elements of a vector $\mathbf{v}$ on the diagonal. Given a matrix $\mathbf{M}$, we use $\mathbf{M}_{-i}$ to denote a resulting matrix without i-th column.

Part I

Algorithm-Aware Matrix and Tensor Factorization

Chapter 2

Provable Online Dictionary Learning and Sparse Coding

2.1 Overview

We consider the dictionary learning problem, where the aim is to model the given data as a linear combination of a few columns of a matrix known as a *dictionary*, where the sparse weights forming the linear combination are known as *coefficients*. Since the dictionary and coefficients, parameterizing the linear model are unknown, the corresponding optimization is inherently non-convex. This was a major challenge until recently, when provable algorithms for dictionary learning were proposed. Yet, these provide guarantees only on the recovery of the dictionary, without explicit recovery guarantees on the coefficients. Moreover, any estimation error in the dictionary adversely impacts the ability to successfully localize and estimate the coefficients. This potentially limits the utility of existing provable dictionary learning methods in applications where coefficient recovery is of interest. To this end, we develop NOODL: a simple Neurally plausible alternating Optimization-based Online Dictionary Learning algorithm, which recovers *both* the dictionary and coefficients *exactly* at a geometric rate, when initialized appropriately. Our algorithm, NOODL, is also scalable and amenable for large scale distributed implementations in neural architectures, by which we mean that it only involves simple linear and non-linear operations. Finally, we corroborate these theoretical results via experimental evaluation of the proposed algorithm with the current state-of-the-art techniques.

2.2 Introduction

Sparse models avoid overfitting by favoring simple yet highly expressive representations. Since signals of interest may not be inherently sparse, expressing them as a sparse linear combination of a few columns of a dictionary is used to exploit the sparsity properties. Of specific interest are overcomplete dictionaries, since they provide a flexible way of capturing the richness of a dataset, while yielding sparse representations that are robust to noise; see Mallat and Zhang (1993); Chen et al. (1998); Donoho et al. (2006). In practice however, these dictionaries may not be known, warranting a need to learn such representations – known as *dictionary learning* (DL) or *sparse coding* (Olshausen and Field, 1997). Formally, this entails learning an *a priori* unknown dictionary $\mathbf{A} \in \mathbb{R}^{n\times m}$ and sparse coefficients $\mathbf{x}^*_{(j)} \in \mathbb{R}^m$ from data samples $\mathbf{y}_{(j)} \in \mathbb{R}^n$ generated as

$$\mathbf{y}_{(j)} = \mathbf{A}^*\mathbf{x}^*_{(j)}, \ \|\mathbf{x}^*_{(j)}\|_0 \leq k \ \text{ for all } \ j = 1, 2, \ldots \tag{2.1}$$

This particular model can also be viewed as an extension of the low-rank model (Pearson, 1901). Here, instead of sharing a low-dimensional structure, each data vector can now reside in a separate low-dimensional subspace. Therefore, together the data matrix admits a *union-of-subspace* model. As a result of this additional flexibility, DL finds applications in a wide range of signal processing and machine learning tasks, such as denoising (Elad and Aharon, 2006), image inpainting (Mairal et al., 2009), clustering and classification (Ramirez et al., 2010; Rambhatla and Haupt, 2013a; Rambhatla et al., 2016a, 2017a, 2018b,a), and analysis of deep learning primitives (Ranzato et al., 2008; Gregor and LeCun, 2010); see also Elad (2010), and references therein.

Notwithstanding the non-convexity of the associated optimization problems (since both factors are unknown), alternating minimization-based dictionary learning techniques have enjoyed significant success in practice. Popular heuristics include regularized least squares-based (Olshausen and Field, 1997; Lee et al., 2007; Mairal et al., 2009; Lewicki and Sejnowski, 2000; Kreutz-Delgado et al., 2003), and greedy approaches such as the method of optimal directions (MOD) (Engan et al., 1999) and k-SVD (Aharon et al., 2006). However, dictionary learning, and matrix factorization models in general, are difficult to analyze in theory; see also Li et al. (2016b).

To this end, motivated from a string of recent theoretical works (Gribonval and

Schnass, 2010; Jenatton et al., 2012; Geng and Wright, 2014), provable algorithms for DL have been proposed recently to explain the success of aforementioned alternating minimization-based algorithms (Agarwal et al., 2014; Arora et al., 2014, 2015). However, these works exclusively focus on guarantees for dictionary recovery. On the other hand, for applications of DL in tasks such as classification and clustering – which rely on coefficient recovery – it is crucial to have guarantees on coefficients recovery as well.

Contrary to conventional prescription, a sparse approximation step after recovery of the dictionary does not help; since any error in the dictionary – which leads to an error-in-variables (EIV) (Fuller, 2009) model for the dictionary – degrades our ability to even recover the support of the coefficients (Wainwright, 2009). Further, when this error is non-negligible, the existing results guarantee recovery of the sparse coefficients only in ℓ_2-norm sense (Donoho et al., 2006). As a result, there is a need for scalable dictionary learning techniques with guaranteed recovery of both factors.

2.2.1 Summary of Our Contributions

In this work, we present a simple online DL algorithm motivated from the following regularized least squares-based problem, where $S(\cdot)$ is a nonlinear function that promotes sparsity.

$$\min_{\mathbf{A},\{\mathbf{x}_{(j)}\}_{j=1}^{p}} \sum_{j=1}^{p} \|\mathbf{y}_{(j)} - \mathbf{A}\mathbf{x}_{(j)}\|_2^2 + \sum_{j=1}^{p} S(\mathbf{x}_{(j)}). \tag{P1}$$

Although our algorithm does not optimize this objective, it leverages the fact that the problem (P1) is convex w.r.t $\mathbf{A}$, given the sparse coefficients $\{\mathbf{x}_{(j)}\}$. Following this, we recover the dictionary by choosing an appropriate gradient descent-based strategy (Arora et al., 2015; Engan et al., 1999). To recover the coefficients, we develop an iterative hard thresholding (IHT)-based update step (Haupt and Nowak, 2006; Blumensath and Davies, 2009), and show that – given an appropriate initial estimate of the dictionary and a mini-batch of p data samples at each iteration t of the online algorithm – alternating between this IHT-based update for coefficients, and a gradient descent-based step for the dictionary leads to geometric convergence to the true factors, i.e., $\mathbf{x}_{(j)} \rightarrow \mathbf{x}^*_{(j)}$ and $\mathbf{A}_i^{(t)} \rightarrow \mathbf{A}_i^*$ as $t \rightarrow \infty$.

In addition to achieving exact recovery of both factors, our algorithm – Neurally plausible alternating Optimization-based Online Dictionary Learning (NOODL) – has linear convergence properties. Furthermore, it is scalable, and involves simple operations, making it an attractive choice for practical DL applications. Our major contributions are summarized as follows:

- **Provable coefficient recovery:** To the best of our knowledge, this is the first result on *exact* recovery of the sparse coefficients $\{\mathbf{x}^*_{(j)}\}$, including their support recovery, for the DL problem. The proposed IHT-based strategy to update coefficient under the EIV model, is of independent interest for recovery of the sparse coefficients via IHT, which is challenging even when the dictionary is known; see also Yuan et al. (2016) and Li et al. (2016c).
- **Unbiased estimation of factors and linear convergence:** The recovery guarantees on the coefficients also helps us to get rid of the bias incurred by the prior-art in dictionary estimation. Furthermore, our technique geometrically converges to the true factors.
- **Online nature and neural implementation:** The online nature of algorithm, makes it suitable for machine learning applications with streaming data. In addition, the separability of the coefficient update allows for distributed implementations in neural architectures (only involves simple linear and non-linear operations) to solve large-scale problems. To showcase this, we also present a prototype neural implementation of NOODL.

In addition, we also verify these theoretical properties of NOODL through experimental evaluations on synthetic data, and compare its performance with state-of-the-art provable DL techniques.

2.2.2 Related Works

With the success of the alternating minimization-based techniques in practice, a push to study the DL problem began when Gribonval and Schnass (2010) showed that for $m = n$, the solution pair $(\mathbf{A}^*, \mathbf{X}^*)$ lies at a local minima of the following non-convex optimization program, where $\mathbf{X} = [\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \ldots, \mathbf{x}_{(p)}]$ and $\mathbf{Y} = [\mathbf{y}_{(1)}, \mathbf{y}_{(2)}, \ldots, \mathbf{y}_{(p)}]$, with high probability over the randomness of the coefficients,

$$\min_{\mathbf{A},\mathbf{X}} \|\mathbf{X}\|_1 \quad \text{s.t. } \mathbf{Y} = \mathbf{A}\mathbf{X}, \quad \|\mathbf{A}_i\| = 1, \forall\ i \in [m]. \tag{2.2}$$

Table 2.1: Comparison of provable algorithms for dictionary learning.

Method	Conditions			Recovery Guarantees	
	Initial Gap of Dictionary	Maximum Sparsity	Sample Complexity	Dictionary	Coefficients
NOODL (this work)	$\mathcal{O}^*\left(\frac{1}{\log(n)}\right)$	$\mathcal{O}^*\left(\frac{\sqrt{n}}{\mu\log(n)}\right)$	$\widetilde{\Omega}\left(mk^2\right)$	No bias	No bias
`Arora15(‘‘biased’’)`†			$\widetilde{\Omega}(mk)$	$\mathcal{O}(\sqrt{k/n})$	N/A
`Arora15(‘‘unbiased’’)`†			poly(m)	*Negligible* bias §	N/A
Barak et al. (2015)¶	N/A	$\mathcal{O}(m^{(1-\delta)})$ for $\delta > 0$	$n^{O(d)}/\text{poly}(k/m)$	ϵ	N/A
Agarwal et al. (2014)‡	$\mathcal{O}^*(1/\text{poly(m)})$	$\mathcal{O}\left(\sqrt[6]{n}/\mu\right)$	$\Omega(m^2)$	No bias	N/A
Spielman et al. (2012) (for $n \leq m$)	N/A	$\mathcal{O}(\sqrt{n})$	$\widetilde{\Omega}(n^2)$	No bias	N/A

Dictionary recovery reported in terms of column-wise error. † See Section 2.6 for description. ‡ This procedure is not *online*. § The bias is not explicitly quantified. The authors claim it will be *negligible*. ¶ Here, $d = \Omega(\frac{1}{\epsilon}\log(m/n))$ for column-wise error of ϵ.

Following this, Geng and Wright (2014) and Jenatton et al. (2012) extended these results to the overcomplete case ($n < m$), and the noisy case, respectively. Concurrently, Jung et al. (2014, 2016) studied the nature of the DL problem for $S(\cdot) = \|\cdot\|_1$ (in (P1)), and derived a lower-bound on the minimax risk of the DL problem. However, these works do not provide any algorithms for DL.

Motivated from these theoretical advances, Spielman et al. (2012) proposed an algorithm for the under-complete case $n \geq m$ that works up-to a sparsity of $k = O(\sqrt{n})$. Later, Agarwal et al. (2014) and Arora et al. (2014) proposed clustering-based provable algorithms for the overcomplete setting, motivated from MOD (Engan et al., 1999) and k-SVD (Aharon et al., 2006), respectively. Here, in addition to requiring stringent conditions on dictionary initialization, Agarwal et al. (2014) alternates between solving a quadratic program for coefficients and an MOD-like (Engan et al., 1999) update for the dictionary, which is too expensive in practice. Recently, a DL algorithm that works for almost linear sparsity was proposed by Barak et al. (2015); however, as shown in Table 2.1, this algorithm may result in exponential running time. Finally, Arora et al. (2015) proposed a provable online DL algorithm, which provided improvements on initialization, sparsity, and sample complexity, and is closely related to our work. A follow-up work by Chatterji and Bartlett (2017) extends this to random initializations while recovering the dictionary exactly, however the effect described therein kicks-in only in very high dimensions. We summarize the relevant provable DL techniques in Table 2.1.

The algorithms discussed above implicitly assume that the coefficients can be recovered, after dictionary recovery, via some sparse approximation technique. However, as alluded to earlier, the guarantees for coefficient recovery – when the dictionary is known approximately – may be limited to some ℓ_2 norm bounds (Donoho et al., 2006). This means that, the resulting coefficient estimates may not even be sparse. Therefore, for practical applications, there is a need for efficient online algorithms with guarantees, which serves as the primary motivation for our work.

2.3 Algorithm

We now detail the specifics of our algorithm – NOODL, outlined in Algorithm 1. NOODL recovers both the dictionary and the coefficients exactly given an appropriate initial estimate $\mathbf{A}^{(0)}$ of the dictionary. Specifically, it requires $\mathbf{A}^{(0)}$ to be $(\epsilon_0, 2)$-close to $\mathbf{A}^*$ for $\epsilon_0 = \mathcal{O}^*(1/\log(n))$, where (ϵ, κ)-closeness is defined as follows. This implies that, the initial dictionary estimate needs to be column-wise, and in spectral norm sense, close to $\mathbf{A}^*$, which can be achieved via certain initialization algorithms, such as those presented in Arora et al. (2015).

Definition 2.1 ((ϵ, κ)-closeness)**.** *A dictionary* $\mathbf{A}$ *is* (ϵ, κ)*-close to* $\mathbf{A}^*$ *if* $\|\mathbf{A} - \mathbf{A}^*\| \leq \kappa\|\mathbf{A}^*\|$*, and if there is a permutation* $\pi : [m] \rightarrow [m]$ *and a collection of signs* $\sigma : [m] \rightarrow \{\pm 1\}$ *such that* $\|\sigma(i)\mathbf{A}_{\pi(i)} - \mathbf{A}^*_i\| \leq \epsilon,\ \forall\ i \in [m]$.

Due to the streaming nature of the incoming data, NOODL takes a mini-batch of p data samples at the t-th iteration of the algorithm, as shown in Algorithm 1. It then proceeds by alternating between two update stages: coefficient estimation ("Predict") and dictionary update ("Learn") as follows.

Predict Stage: For a general data sample $\mathbf{y} = \mathbf{A}^*\mathbf{x}^*$, the algorithm begins by forming an initial coefficient estimate $\mathbf{x}^{(0)}$ based on a hard thresholding (HT) step as shown in (2.3), where $\mathcal{T}_\tau(z) := z \cdot \mathbb{1}_{|z| \geq \tau}$ for a vector $\mathbf{z}$. Given this initial estimate $\mathbf{x}^{(0)}$, the algorithm iterates over $R = \Omega(\log(1/\delta_R))$ IHT-based steps (2.4) to achieve a target tolerance of δ_R, such that $(1 - \eta_x)^R \leq \delta_R$. Here, $\eta_x^{(r)}$ is the learning rate, and $\tau^{(r)}$ is the threshold at the r-th iterate of the IHT. In practice, these can be fixed to some constants for all iterations; see **A.6** for details. Finally at the end of this stage, we have estimate $\widehat{\mathbf{x}}^{(t)} := \mathbf{x}^{(R)}$ of $\mathbf{x}^*$.

Learn Stage: Using this estimate of the coefficients, we update the dictionary at t-th iteration $\mathbf{A}^{(t)}$ by an approximate gradient descent step (2.6), using the empirical gradient estimate (2.5) and the learning rate $\eta_A = \Theta(m/k)$; see also **A.5**. Finally, we normalize the columns of the dictionary and continue to the next batch. The running time of each step t of NOODL is therefore $\mathcal{O}(mnp\log(1/\delta_R))$. For a target tolerance

Algorithm 1: NOODL: Neurally plausible alternating Optimization-based Online Dictionary Learning.

```
Input: Fresh data samples $\mathbf{y}_{(j)} \in \mathbb{R}^n$ for $j \in [p]$ at each iteration $t$ generated as per
        (2.1), where $|\mathbf{x}^*_i| \geq C$ for $i \in \text{supp}(\mathbf{x}^*)$. Parameters $\eta_A$, $\eta_x^{(r)}$ and $\tau^{(r)}$ chosen as
        per A.5 and A.6. No. of iterations $T = \Omega(\log(1/\epsilon_T))$ and $R = \Omega(\log(1/\delta_R))$,
        for target tolerances $\epsilon_T$ and $\delta_R$.
Output: The dictionary $\mathbf{A}^{(t)}$ and coefficient estimates $\widehat{\mathbf{x}}_{(j)}^{(t)}$ for $j \in [p]$ at each
        iterate $t$.
Initialize: Estimate $\mathbf{A}^{(0)}$, which is $(\epsilon_0, 2)$-near to $\mathbf{A}^*$ for $\epsilon_0 = \mathcal{O}^*(1/\log(n))$
for $t = 0$ to $T - 1$ do
    Predict: (Estimate Coefficients)
    for $j = 1$ to $p$ do
        Initialize: $\mathbf{x}_{(j)}^{(0)} = \mathcal{T}_{C/2}(\mathbf{A}^{(t)\top}\mathbf{y}_{(j)})$    (2.3)
        for $r = 0$ to $R - 1$ do
            Update: $\mathbf{x}_{(j)}^{(r+1)} = \mathcal{T}_{\tau^{(r)}}(\mathbf{x}_{(j)}^{(r)} - \eta_x^{(r)}\ \mathbf{A}^{(t)\top}(\mathbf{A}^{(t)}\mathbf{x}_{(j)}^{(r)} - \mathbf{y}_{(j)}))$    (2.4)
        end
    end
    $\widehat{\mathbf{x}}_{(j)}^{(t)} := \mathbf{x}_{(j)}^{(R)}$ for $j \in [p]$
    Learn: (Update Dictionary)
        Form empirical gradient estimate: $\widehat{\mathbf{g}}^{(t)} = \frac{1}{p}\sum_{j=1}^{p}(\mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)}^{(t)} - \mathbf{y}_{(j)})\text{sign}(\widehat{\mathbf{x}}_{(j)}^{(t)})^\top$
                                                                                        (2.5)
        Take a gradient descent step: $\mathbf{A}^{(t+1)} = \mathbf{A}^{(t)} - \eta_A\ \widehat{\mathbf{g}}^{(t)}$    (2.6)
        Normalize: $\mathbf{A}_i^{(t+1)} = \mathbf{A}_i^{(t+1)}/\|\mathbf{A}_i^{(t+1)}\|\ \forall\ i \in [m]$
end
```

of ϵ_T and δ_T, such that $\|\mathbf{A}_i^{(T)} - \mathbf{A}_i^*\| \leq \epsilon_T, \forall i \in [m]$ and $|\widehat{\mathbf{x}}_i^{(T)} - \mathbf{x}_i^*| \leq \delta_T$ we choose $T = \max(\Omega(\log(1/\epsilon_T)), \Omega(\log(\sqrt{k}/\delta_T)))$.

NOODL uses an initial HT step and an approximate gradient descent-based strategy as in Arora et al. (2015). Following which, our IHT-based coefficient update step yields an estimate of the coefficients at each iteration of the online algorithm. Coupled with the guaranteed progress made on the dictionary, this also removes the bias in dictionary estimation. Further, the simultaneous recovery of both factors also avoids an often expensive post-processing step for recovery of the coefficients.

2.4 Main Result

We start by introducing a few important definitions. First, as discussed in the previous section we require that the initial estimate $\mathbf{A}^{(0)}$ of the dictionary is $(\epsilon_0, 2)$-close to $\mathbf{A}^*$. In fact, we require this closeness property to hold at each subsequent iteration t, which is a key ingredient in our analysis. This initialization achieves two goals. First, the $\|\sigma(i)\mathbf{A}_{\pi(i)} - \mathbf{A}_i^*\| \leq \epsilon_0$ condition ensures that the signed-support of the coefficients are recovered correctly (with high probability) by the hard thresholding-based coefficient initialization step, where signed-support is defined as follows.

Definition 2.2. *The signed-support of a vector* $\mathbf{x}$ *is defined as* $\mathrm{sign}(\mathbf{x}) \cdot \mathrm{supp}(\mathbf{x})$.

Next, the $\|\mathbf{A} - \mathbf{A}^*\| \leq 2\|\mathbf{A}^*\|$ condition keeps the dictionary estimates close to $\mathbf{A}^*$ and is used in our analysis to ensure that the gradient direction (2.5) makes progress. Further, in our analysis, we ensure ϵ_t (defined as $\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \leq \epsilon_t$) contracts at every iteration, and assume $\epsilon_0, \epsilon_t = \mathcal{O}^*(1/\log(n))$. Also, we assume that the dictionary $\mathbf{A}$ is fixed (deterministic) and μ-incoherent, defined as follows.

Definition 2.3. *A matrix* $\mathbf{A} \in \mathbb{R}^{n\times m}$ *with unit-norm columns is* μ*-incoherent if for all* $i \neq j$ *the inner-product between the columns of the matrix follow* $|\langle \mathbf{A}_i, \mathbf{A}_j\rangle| \leq \mu/\sqrt{n}$.

The incoherence parameter measures the degree of closeness of the dictionary elements. Smaller values (i.e., close to 0) of μ are preferred, since they indicate that the dictionary elements do not resemble each other. This helps us to effectively tell dictionary elements apart (Donoho and Huo, 2001a; Candes and Romberg, 2007). We assume that $\mu = \mathcal{O}(\log(n))$ (Donoho and Huo, 2001a). Next, we assume that the coefficients are drawn from a distribution class $\mathcal{D}$ defined as follows.

Definition 2.4 (Distribution class $\mathcal{D}$)**.** *The coefficient vector* $\mathbf{x}^*$ *belongs to an unknown distribution* $\mathcal{D}$*, where the support* $S = \mathrm{supp}(\mathbf{x}^*)$ *is at most of size* k, $\mathbf{Pr}[i \in S] = \Theta(k/m)$ *and* $\mathbf{Pr}[i, j \in S] = \Theta(k^2/m^2)$. *Moreover, the distribution is normalized such that* $\mathbf{E}[\mathbf{x}_i^*|i \in S] = 0$ *and* $\mathbf{E}[\mathbf{x}_i^{*^2}|i \in S] = 1$, *and when* $i \in S$, $|\mathbf{x}_i^*| \geq C$ *for some constant* $C \leq 1$. *In addition, the non-zero entries are sub-Gaussian and pairwise independent conditioned on the support.*

The randomness of the coefficient is necessary for our finite sample analysis of the convergence. Here, there are two sources of randomness. The first is the randomness of the support, where the non-zero elements are assumed to pair-wise independent. The second is the value an element in the support takes, which is assumed to be zero mean with variance one, and bounded in magnitude. Similar conditions are also required for support recovery of sparse coefficients, even when the dictionary is known (Wainwright, 2009; Yuan et al., 2016). Note that, although we only consider the case $|\mathbf{x}_i^*| \geq C$ for ease of discussion, analogous results may hold more generally for $\mathbf{x}_i^*$s drawn from a distribution with sufficiently (exponentially) small probability of taking values in $[-C, C]$.

Recall that, given the coefficients, we recover the dictionary by making progress on the least squares objective (P1) (ignoring the term penalizing $S(\cdot)$). Note that, our algorithm is based on finding an appropriate direction to ensure descent based on the geometry of the objective. To this end, we adopt a gradient descent-based strategy for dictionary update. However, since the coefficients are not exactly known, this results in an approximate gradient descent-based approach, where the empirical gradient estimate is formed as (2.5). In our analysis, we establish the conditions under which both the empirical gradient vector (corresponding to each dictionary element) and the gradient matrix concentrate around their means. To ensure progress at each iterate t, we show that the expected gradient vector is $(\Omega(k/m), \Omega(m/k), 0)$-correlated with the descent direction, defined as follows.

Definition 2.5. *A vector $\mathbf{g}^{(t)}$ is $(\rho_-, \rho_+, \zeta_t)$-correlated with a vector $\mathbf{z}^*$ if*

$$\langle \mathbf{g}^{(t)}, \mathbf{z}^{(t)} - \mathbf{z}^* \rangle \geq \rho_- \|\mathbf{z}^{(t)} - \mathbf{z}^*\|^2 + \rho_+ \|\mathbf{g}^{(t)}\|^2 - \zeta_t.$$

This can be viewed as a local descent condition which leads to the true dictionary columns; see also Candès et al. (2015), Chen and Wainwright (2015b) and Arora et al. (2015). In convex optimization literature, this condition is implied by the $2\rho_-$-strong convexity, and $1/2\rho_+$-smoothness of the objective. We show that for NOODL, $\zeta_t = 0$, which facilitates linear convergence to $\mathbf{A}^*$ without incurring any bias. Overall our specific model assumptions for the analysis can be formalized as:

A.1 $\mathbf{A}^*$ is μ-incoherent (Def. 2.3), where $\mu = \mathcal{O}(\log(n))$, $\|\mathbf{A}^*\| = \mathcal{O}(\sqrt{m/n})$ and $m = \mathcal{O}(n)$;

A.2 The coefficients are drawn from the distribution class $\mathcal{D}$, as per Def. 2.4;

A.3 The sparsity k satisfies $k = \mathcal{O}^*(\sqrt{n}/\mu \log(n))$;

A.4 $\mathbf{A}^{(0)}$ is $(\epsilon_0, 2)$-close to $\mathbf{A}^*$ as per Def. 2.1, and $\epsilon_0 = \mathcal{O}^*(1/\log(n))$;

A.5 The step-size for dictionary update satisfies $\eta_A = \Theta(m/k)$;

A.6 The step-size and threshold for coefficient estimation satisfies $\eta_x^{(r)} < c_1(\epsilon_t, \mu, n, k) = \widetilde{\Omega}(k/\sqrt{n}) < 1$ and $\tau^{(r)} = c_2(\epsilon_t, \mu, k, n) = \widetilde{\Omega}(k^2/n)$ for small constants c_1 and c_2.

We are now ready to state our main result. A summary of the notation followed by a details of the analysis is provided in Appendix 2.A and Appendix 2.B, respectively.

Theorem 2.1 (Main Result)**.** *Suppose that assumptions **A.1**-**A.6** hold, and Algorithm 1 is provided with $p = \widetilde{\Omega}(mk^2)$ new samples generated according to model (2.1) at each iteration t. Then, with probability at least $(1-\delta_{alg}^{(t)})$ for some small constant $\delta_{alg}^{(t)}$, given $R = \Omega(\log(n))$, the coefficient estimate $\widehat{\mathbf{x}}_i^{(t)}$ at t-th iteration has the correct signed-support and satisfies*

$$(\widehat{\mathbf{x}}_i^{(t)} - \mathbf{x}_i^*)^2 = \mathcal{O}(k(1-\omega)^{t/2}\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|), \text{ for all } i \in \operatorname{supp}(\mathbf{x}^*).$$

Furthermore, for some $0 < \omega < 1/2$, the estimate $\mathbf{A}^{(t)}$ at (t)-th iteration satisfies

$$\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 \leq (1-\omega)^t\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|^2, \text{ for all } t = 1, 2, \ldots..$$

Our main result establishes that when the model satisfies **A.1**~**A.3**, the errors corresponding to the dictionary and coefficients geometrically decrease to the true model parameters, given appropriate dictionary initialization and learning parameters (step sizes and threshold); see **A.4**~**A.6**. In other words, to attain a target tolerance of ϵ_T and δ_T, where $\|\mathbf{A}_i^{(T)} - \mathbf{A}_i^*\| \leq \epsilon_T$, $|\widehat{\mathbf{x}}_i^{(T)} - \mathbf{x}_i^*| \leq \delta_T$, we require $T = \max(\Omega(\log(1/\epsilon_T)), \Omega(\log(\sqrt{k}/\delta_T)))$ outer iterations and $R = \Omega(\log(1/\delta_R))$ IHT steps per outer iteration. Here, $\delta_R \geq (1-\eta_x)^R$ is the target decay tolerance for the IHT steps. An appropriate number of IHT steps, R, remove the dependence of final coefficient error (per outer iteration) on the initial $\mathbf{x}^{(0)}$. In Arora et al. (2015), this dependence in fact results in an irreducible error, which is the source of bias in dictionary estimation. As a result, since (for NOODL) the error in the coefficients only depends on the error in the dictionary, it can be made arbitrarily small, at a geometric rate, by the choice of ϵ_T, δ_T, and δ_R. Also, note that, NOODL can tolerate i.i.d. noise, as long as the noise variance is controlled to enable the concentration results to hold; we consider the noiseless case here for ease of discussion, which is already highly involved.

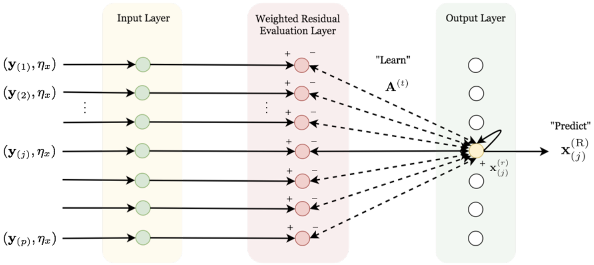
Intuitively, Theorem 2.1 highlights the symbiotic relationship between the two factors. It shows that, to make progress on one, it is imperative to make progress on the other. The primary condition that allows us to make progress on both factors is the signed-support recovery (Def. 2.2). However, the introduction of IHT step adds complexity in the analysis of both the dictionary and coefficients. To analyze the coefficients, in addition to deriving conditions on the parameters to preserve the correct signed-support, we analyze the recursive IHT update step, and decompose the noise term into a component that depends on the error in the dictionary, and the other that depends on the initial coefficient estimate. For the dictionary update, we analyze the interactions between elements of the coefficient vector (introduces by the IHT-based update step) and show that the gradient vector for the dictionary update is $(\Omega(k/m), \Omega(m/k), 0)$-correlated with the descent direction. In the end, this leads to exact recovery of the coefficients and removal of bias in the dictionary estimation. Note that our analysis pipeline is standard for the convergence analysis for iterative algorithms. However, the introduction of the IHT-based strategy for coefficient update makes the analysis highly involved as compared to existing results, e.g., the simple HT-based coefficient estimate in Arora et al. (2015).

NOODL has an overall running time of $\mathcal{O}(mnp\log(1/\delta_R)\max(\log(1/\epsilon_T), \log(\sqrt{k}/\delta_T))$ to achieve target tolerances ϵ_T and δ_T, with a total sample complexity of $p \cdot T = \widetilde{\Omega}(mk^2)$. Thus to remove bias, the IHT-based coefficient update introduces a factor of $\log(1/\delta_R)$ in the computational complexity as compared to Arora et al. (2015) (has a total sample complexity of $p \cdot T = \widetilde{\Omega}(mk)$), and also does not have the exponential running time and sample complexity as Barak et al. (2015); see Table 2.1.

2.5 Neural implementation of NOODL

The neural plausibility of our algorithm implies that it can be implemented as a neural network. This is because, NOODL employs simple linear and non-linear operations (such as inner-product and hard-thresholding) and the coefficient updates are separable across data samples, as shown in (2.4) of Algorithm 1. To this end, we present a neural implementation of our algorithm in Fig. 2.1, which showcases the applicability of NOODL in large-scale distributed learning tasks, motivated from the implementations described in (Olshausen and Field, 1997) and (Arora et al., 2015).

The neural architecture shown in Fig. 2.1(a) has three layers – input layer, weighted residual evaluation layer, and the output layer. The input to the network is a data and step-size pair $(\mathbf{y}_{(j)}, \eta_x)$ to each input node. Given an input, the second layer evaluates the weighted residuals as shown in Fig. 2.1. Finally, the output layer neurons evaluate the IHT iterates $\mathbf{x}_{(j)}^{(r+1)}$ (2.4). We illustrate the operation of this architecture using the


Input Layer
Weighted Residual Evaluation Layer
Output Layer
$(\mathbf{y}_{(1)}, \eta_x)$
$(\mathbf{y}_{(2)}, \eta_x)$
$(\mathbf{y}_{(j)}, \eta_x)$
$(\mathbf{y}_{(p)}, \eta_x)$
"Learn"
$\mathbf{A}^{(t)}$
"Predict"
$\mathbf{x}_{(j)}^{(\mathrm{R})}$
$\mathbf{x}_{(j)}^{(r)}$
+
−

(a) Neural implementation of NOODL

Figure 2.1: A neural implementation of NOODL. Panel (a) shows the neural architecture, which consists of three layers: an input layer, a weighted residual evaluation layer (evaluates $\eta_x\big(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\mathbf{x}_{(j)}^{(r)}\big)$), and an output layer. Panel (b) shows the operation of the neural architecture in panel (a). The update of $\mathbf{x}_{(j)}^{(r+1)}$ is given by (2.4).

$\ell = 0$	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$	$\ell = 5$	$\dots$	$\ell = 2R+1$	**Hebbian Learning**: Residual sharing and dictionary update.
Output: $\mathbf{x} \leftarrow \mathbf{0}$	$\mathbf{0}$	$\mathbf{x}_{(j)}^{(0)} = \mathcal{T}_\tau(\mathbf{A}^{(t)^\top}\mathbf{y}_{(j)})$	$\mathbf{x}_{(j)}^{(0)}$	$\mathbf{x}_{(j)}^{(1)}$	$\mathbf{x}_{(j)}^{(1)}$	$\dots$	$\mathbf{x}_{(j)}^{(R)}$	
Residual: $\mathbf{0}$	$\mathbf{y}_{(j)}$	$\mathbf{y}_{(j)}$	$\eta_x(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\mathbf{x}_{(j)}^{(0)})$	$\eta_x(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\mathbf{x}_{(j)}^{(0)})$	$\eta_x(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\mathbf{x}_{(j)}^{(1)})$	$\dots$	$\eta_x(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\mathbf{x}_{(j)}^{(R-1)})$	
Input: $(\mathbf{y}_{(j)}, 1)$	.	$(\mathbf{y}_{(j)}, \eta_x)$	.	.	.	$\dots$	$(\mathbf{y}_{(j)}, 1)$	

(b) The timing sequence of the neural implementation.

timing diagram in Fig. 2.1(b). The main stages of operation are as follows.

Initial Hard Thresholding Phase: The coefficients initialized to zero, and an input $(\mathbf{y}_{(j)}, 1)$ is provided to the input layer at a time instant $\ell = 0$, which communicates these to the second layer. Therefore, the residual at the output of the weighted residual evaluation layer evaluates to $\mathbf{y}_{(j)}$ at $\ell = 1$. Next, at $\ell = 2$, this residual is communicated to the output layer, which results in evaluation of the initialization $\mathbf{x}_{(j)}^{(0)}$ as per (2.3). This iterate is communicated to the second layer for the next residual evaluation. Also, at this time, the input layer is injected with $(\mathbf{y}_{(j)}, \eta_x)$ to set the step size parameter η_x for the IHT phase, as shown in Fig. 2.1(b).

Iterative Hard Thresholding (IHT) Phase: Beginning $\ell = 3$, the timing sequence enters the IHT phase. Here, the output layer neurons communicate the iterates $\mathbf{x}_{(j)}^{(r+1)}$ to the second layer for evaluation of subsequent iterates as shown in Fig. 2.1(b). The process then continues till the time instance $\ell = 2R+1$, for $R = \Omega(\log(1/\delta_R))$ to generate the final coefficient estimate $\widehat{\mathbf{x}}_{(j)}^{(t)} := \mathbf{x}_{(j)}^{(R)}$ for the current batch of data. At this time, the input layer is again injected with $(\mathbf{y}_{(j)}, 1)$ to prepare the network for residual sharing and gradient evaluation for dictionary update.

Dictionary Update Phase: The procedure now enters the dictionary update phase, denoted as "Hebbian Learning" in the timing sequence. In this phase, each output layer neuron communicates the final coefficient estimate $\widehat{\mathbf{x}}_{(j)}^{(t)} = \mathbf{x}_{(j)}^{(R)}$ to the second layer, which evaluates the residual for one last time (with $\eta_x = 1$), and shares it across all second layer neurons ("Hebbian learning"). This allows each second layer neuron to evaluate the empirical gradient estimate (2.5), which is used to update the current dictionary estimate (stored as weights) via an approximate gradient descent step. This completes one outer iteration of Algorithm 1, and the process continues for T iterations to achieve target tolerances ϵ_T and δ_T, with each step receiving a new mini-batch of data.

2.6 Experiments

We now analyze the convergence properties and sample complexity of NOODL via experimental evaluations [1]. The experimental data generation set-up, additional results, including analysis of computational time, are shown in Appendix 2.E.

2.6.1 Convergence Analysis

We compare the performance of our algorithm NOODL with the current state-of-the-art alternating optimization-based online algorithms presented in Arora et al. (2015), and the popular algorithm presented in Mairal et al. (2009) (denoted as `Mairal '09`). First of these, `Arora15(''biased'')`, is a simple neurally plausible method which incurs a bias and has a sample complexity of $\Omega(mk)$. The other, referred to as `Arora15('' unbiased'')`, incurs no bias as per Arora et al. (2015), but the sample complexity results were not established.

Discussion: Fig. 2.2 panels (a-i), (b-i), (c-i), and (d-i) show the performance of the aforementioned methods for $k = 10,\ 20,\ 50,$ and 100, respectively. Here, for all experiments we set $\eta_x = 0.2$ and $\tau = 0.1$. We terminate NOODL when the error in dictionary is less than 10^{-10}. Also, for coefficient update, we terminate when change in the iterates is below 10^{-12}. For $k = 10,\ 20$ and $k = 50$, we note that `Arora15(''biased'')` and `Arora15(''unbiased'')` incur significant bias, while NOODL converges to $\mathbf{A}^*$ *linearly*. NOODL also converges for significantly higher choices of sparsity k, i.e., for $k = 100$ as shown in panel (d), beyond $k = \mathcal{O}(\sqrt{n})$, indicating a potential for improving this bound. Further, we observe that `Mairal '09` exhibits significantly slow convergence as compared to NOODL. Also, in panels (a-ii), (b-ii), (c-ii) and (d-ii) we show the corresponding performance of NOODL in terms of the error in the overall fit ($\|\mathbf{Y} - \mathbf{AX}\|_\mathrm{F}/\|\mathbf{Y}\|_\mathrm{F}$), and the error in the coefficients and the dictionary, in terms of relative Frobenius error metric discussed above. We observe that the error in dictionary and coefficients drops linearly as indicated by our main result.

[1] The associated code is made available at https://github.com/srambhatla/NOODL; see Chapter 7 for details.

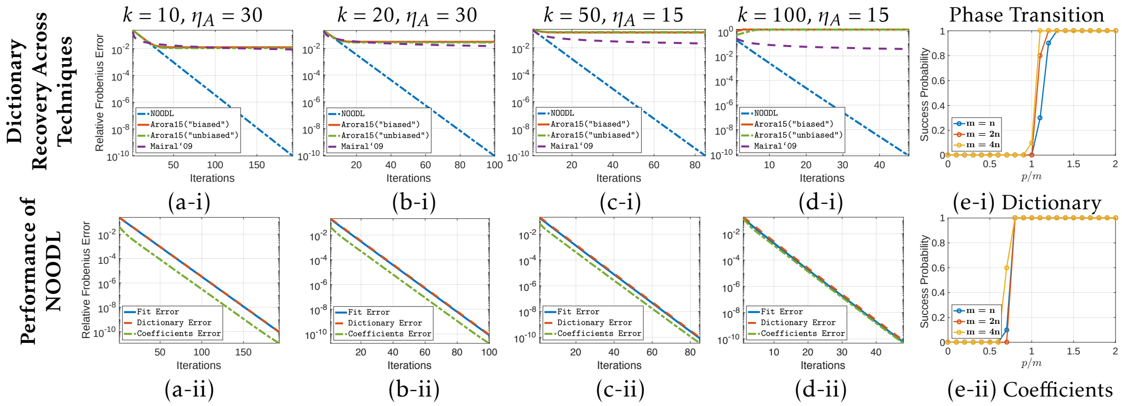


Figure 2.2: Comparative analysis of convergence properties. Panels (a-i), (b-i), (c-i), and (d-i) show the convergence of NOODL, `Arora15(‘‘biased’’)`, `Arora15(‘‘unbiased’’)` and `Mairal ‘09`, for different sparsity levels for $n = 1000$, $m = 1500$ and $p = 5000$. Since NOODL also recovers the coefficients, we show the corresponding recovery of the dictionary, coefficients, and overall fit in panels (a-ii), (b-ii), (c-ii), and (d-ii), respectively. Further, panels (e-i) and (e-ii) show the phase transition in samples p (per iteration) with the size of the dictionary m averaged across 10 Monte Carlo simulations for the two factors. Here, $n = 100$, $k = 3$, $\eta_x = 0.2$, $\tau = 0.1$, $\epsilon_0 = 2/\log(n)$, η_A is chosen as per **A.5**. A trial is considered successful if the relative Frobenius error incurred by $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{X}}$ is below 5×10^{-7} after 50 iterations.

2.6.2 Phase transitions

Fig. 2.2 panels (e-i) and (e-ii), shows the phase transition in number of samples with respect to the size of the dictionary m. We observe a sharp phase transition at $\frac{p}{m} = 1$ for the dictionary, and at $\frac{p}{m} = 0.75$ for the coefficients. This phenomenon is similar to that observed by Agarwal et al. (2014) (however, theoretically they required $p = \mathcal{O}(m^2)$). Here, we confirm number of samples required by NOODL are linearly dependent on the dictionary elements m.

2.7 Future Work

We consider the online DL setting in this work. We note that, empirically NOODL works for the batch setting also. However, analysis for this case will require more sophisticated concentration results, which can address the resulting dependence between iterations of the algorithm. In addition, our experiments indicate that NOODL works beyond the sparsity ranges prescribed by our theoretical results. Arguably, the bounds on sparsity can potentially be improved by moving away from the incoherence-based analysis. We also note that in our experiments, NOODL converges even when initialized outside the prescribed initialization region, albeit it achieves the linear rate once it satisfies the closeness condition A.4. These potential directions may significantly impact the analysis and development of provable algorithms for other factorization problems as well. We leave these research directions, and a precise analysis under the noisy setting, for future explorations.

2.8 Conclusions

We present NOODL, to the best of our knowledge, the first neurally plausible provable online algorithm for exact recovery of both factors of the dictionary learning (DL) model. NOODL alternates between: (a) an iterative hard thresholding (IHT)-based step for coefficient recovery, and (b) a gradient descent-based update for the dictionary, resulting in a simple and scalable algorithm, suitable for large-scale distributed implementations. We show that once initialized appropriately, the sequence of estimates produced by NOODL converge *linearly* to the true dictionary and coefficients without incurring any bias in the estimation. Complementary to our theoretical and numerical results, we also design an implementation of NOODL in a neural architecture for use in practical applications. In essence, the analysis of this inherently nonconvex problem impacts other matrix and tensor factorization tasks arising in signal processing, collaborative filtering, and machine learning.

Appendices: Provable Online Dictionary Learning and Sparse Coding

2.A Summary of Notation

We summarizes the definitions of some frequently used symbols in our analysis in Table 2.A.1. Also note that, since we show that $\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \le \epsilon_t$ contracts in every step, therefore we fix $\epsilon_t, \epsilon_0 = \mathcal{O}^*(1/\log(n))$ in our analysis.

2.B Proof of Theorem 2.1

We now prove our main result. The detailed proofs of intermediate lemmas and claims are organized in Appendix 2.C and Appendix 2.D, respectively. Furthermore, the standard concentration results are stated in Appendix 2.F for completeness. Also, see Table 2.B.1 for a map of dependence between the results.

Overview

Given an $(\epsilon_0, 2)$-close estimate of the dictionary, the main property that allows us to make progress on the dictionary is the recovery of the correct sign and support of the coefficients. Therefore, we first show that the initial coefficient estimate (2.3) recovers the correct signed-support in Step I.A. Now, the IHT-based coefficient update step also needs to preserve the correct signed-support. This is to ensure that the approximate gradient descent-based update for the dictionary makes progress. Therefore, in Step I.B, we derive the conditions under which the signed-support recovery condition is

Table 2.A.1: Frequently used symbols

Dictionary Related

Symbol	Definition	
$\mathbf{A}_i^{(t)}$	i-th column of the dictionary estimate at the t-th iterate.	
ϵ_t	$\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \leq \epsilon_t = \mathcal{O}^*(\frac{1}{\log(n)})$	Upper-bound on column-wise error at the t-th iterate.
μ_t	$\frac{\mu_t}{\sqrt{n}} = \frac{\mu}{\sqrt{n}} + 2\epsilon_t$	Incoherence between the columns of $\mathbf{A}^{(t)}$; See Claim 1.
$\lambda_j^{(t)}$	$\lambda_j^{(t)} := \lvert\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^* \rangle\rvert \leq \frac{\epsilon_t^2}{2}$	Inner-product between the error and the dictionary element.
$\Lambda_S^{(t)}(i,j)$	$\Lambda_S^{(t)}(i,j) = \begin{cases} \lambda_j^{(t)}, & \text{for } j = i, i \in S \\ 0, & \text{otherwise.} \end{cases}$	A diagonal matrix of size $\lvert S\rvert \times \lvert S\rvert$ with $\lambda_j^{(t)}$ on the diagonal for $j \in S$.

Coefficient Related

Symbol	Definition	
$\mathbf{x}_i^{(r)}$	i-th element the coefficient estimate at the r-th IHT iterate.	
C	$\lvert\mathbf{x}_i^*\rvert \geq C$ for $i \in \text{supp}(\mathbf{x}^*)$ and $C \leq 1$	Lower-bound on $\mathbf{x}_i^*$s.
S	$S := \text{supp}(\mathbf{x}^*)$ where $\lvert S\rvert \leq k$	Support of $\mathbf{x}^*$
δ_R	$\delta_R := (1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}})^R \geq (1 - \eta_x)^R$	Decay parameter for coefficients.
δ_T	$\lvert\widehat{\mathbf{x}}_i^{(T)} - \mathbf{x}_i^*\rvert \leq \delta_T \forall i \in \text{supp}(\mathbf{x}^*)$	Target coefficient element error tolerance.
$C_i^{(\ell)}$	$C_i^{(\ell)} := \lvert\mathbf{x}_i^* - \mathbf{x}_i^{(\ell)}\rvert$ for $i \in \text{supp}(\mathbf{x}^*)$	Error in non-zero elements of the coefficient vector.

Probabilities

Symbol	Definition	Symbol	Definition
q_i	$q_i = \mathbf{Pr}[i \in S] = \Theta(\frac{k}{m})$	$q_{i,j}$	$q_{i,j} = \mathbf{Pr}[i,j \in S] = \Theta(\frac{k^2}{m^2})$
p_i	$p_i = \mathbf{E}[\mathbf{x}_i^* \text{sign}(\mathbf{x}_i^*) \lvert \mathbf{x}_i^* \neq 0]$	$\delta_{\mathcal{T}}^{(t)}$	$\delta_{\mathcal{T}}^{(t)} = 2m \exp(-C^2/\mathcal{O}^*(\epsilon_t^2))$
$\delta_\beta^{(t)}$	$\delta_\beta^{(t)} = 2k \exp(-1/\mathcal{O}(\epsilon_t))$	$\delta_{\text{HW}}^{(t)}$	$\delta_{\text{HW}}^{(t)} = \exp(-1/\mathcal{O}(\epsilon_t))$
$\delta_{\mathbf{g}_i}^{(t)}$	$\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(k))$	$\delta_{\mathbf{g}}^{(t)}$	$\delta_{\mathbf{g}}^{(t)} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$

Other terms

Symbol	Definition
$\xi_j^{(r+1)}$	$\xi_j^{(r+1)} := \sum\limits_{i \neq j} (\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^* \rangle + \langle \mathbf{A}_j^*, \mathbf{A}_i^* \rangle)\mathbf{x}_i^* - \sum\limits_{i \neq j} \langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)} \rangle \mathbf{x}_i^{(r)}$
$\beta_j^{(t)}$	$\beta_j^{(t)} := \sum\limits_{i \neq j} (\langle \mathbf{A}_j^*, \mathbf{A}_i^* - \mathbf{A}_i^{(t)} \rangle + \langle \mathbf{A}_j^* - \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)} \rangle + \langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^* \rangle)\mathbf{x}_i^*$
t_β	$t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$ is an upper-bound on $\beta_j^{(t)}$ with probability at least $(1 - \delta_\beta^{(t)})$
$\widetilde{\xi}_j^{(r+1)}$	$\widetilde{\xi}_j^{(r+1)} := \beta_j^{(t)} + \sum\limits_{i \neq j} \lvert\langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)} \rangle\rvert \, \lvert\mathbf{x}_i^* - \mathbf{x}_i^{(r)}\rvert$
$\Delta_j^{(t)}$	$\Delta_j^{(t)} := \mathbf{E}[\mathbf{A}_S^{(t)} \vartheta_S^{(R)} \text{sign}(\mathbf{x}_j^*)]$
$\vartheta_i^{(R)}$	$\vartheta_i^{(R)} := \sum\limits_{r=1}^{R} \eta_x \xi_i^{(r)} (1 - \eta_x)^{R-r} + \gamma_i^{(R)}$
$\gamma_i^{(R)}$	$\gamma_i^{(R)} := (1 - \eta_x)^R (\mathbf{x}_i^{(0)} - \mathbf{x}_i^*(1 - \lambda_i^{(t)}))$
γ	$\gamma := \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\text{sign}(\mathbf{x}_j^*)\mathbb{1}_{\overline{\mathcal{F}}_{\mathbf{x}^*}}]$; See † below.
$\widehat{\mathbf{x}}_i$	$\widehat{\mathbf{x}}_i := \mathbf{x}_i^{(R)} = \mathbf{x}_i^*(1 - \lambda_i^{(t)}) + \vartheta_i^{(R)}$

† $\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}$ is the indicator function corresponding to the event that $\text{sign}(\mathbf{x}^*) = \text{sign}(\widehat{\mathbf{x}})$, denoted by $\mathcal{F}_{\mathbf{x}^*}$, and similarly for the complement $\overline{\mathcal{F}_{\mathbf{x}^*}}$

preserved by the IHT update.

To get a handle on the coefficients, in Step II.A, we derive an upper-bound on the error incurred by each non-zero element of the estimated coefficient vector, i.e., $|\widehat{\mathbf{x}}_i - \mathbf{x}^*_i|$ for $i \in S$ for a general coefficient vector $\mathbf{x}^*$, and show that this error only depends on ϵ_t (the column-wise error in the dictionary) given enough IHT iterations R as per the chosen decay parameter δ_R. In addition, for analysis of the dictionary update, we develop an expression for the estimated coefficient vector in Step II.B.

We then use the coefficient estimate to show that the gradient vector satisfies the local descent condition (Def. 2.5). This ensures that the gradient makes progress after taking the gradient descent-based step (2.6). To begin, we first develop an expression for the expected gradient vector (corresponding to each dictionary element) in Step III.A. Here, we use the closeness property Def 2.1 of the dictionary estimate. Further, since we use an empirical estimate, we show that the empirical gradient vector concentrates around its mean in Step III.B. Now using Lemma 2.15, we have that descent along this direction makes progress.

Next in Step IV.A and Step IV.B, we show that the updated dictionary estimate maintains the closeness property Def 2.1. This sets the stage for the next dictionary update iteration. As a result, our main result establishes the conditions under which any t-th iteration succeeds.

Our main result is as follows.

Theorem 2.1 (Main Result) *Suppose that assumptions **A.1**-**A.6** hold, and Algorithm 1 is provided with* $p = \widetilde{\Omega}(mk^2)$ *new samples generated according to model* (2.1) *at each iteration* t. *Then, with probability at least* $(1 - \delta^{(t)}_{alg})$, *given* $R = \Omega(\log(n))$, *the coefficient estimate* $\widehat{\mathbf{x}}^{(t)}_i$ *at t-th iteration has the correct signed-support and satisfies*

$$(\widehat{\mathbf{x}}^{(t)}_i - \mathbf{x}^*_i)^2 = \mathcal{O}(k(1-\omega)^{t/2}\|\mathbf{A}^{(0)}_i - \mathbf{A}^*_i\|), \textit{ for all } i \in \mathrm{supp}(\mathbf{x}^*).$$

Furthermore, for some $0 < \omega < 1/2$, *the estimate* $\mathbf{A}^{(t)}$ *at* (t)*-th iteration satisfies*

$$\|\mathbf{A}^{(t)}_i - \mathbf{A}^*_i\|^2 \leq (1-\omega)^t\|\mathbf{A}^{(0)}_i - \mathbf{A}^*_i\|^2, \textit{ for all } t = 1, 2, \ldots..$$

Here, $\delta^{(t)}_{alg}$ *is some small constant, where* $\delta^{(t)}_{alg} = \delta^{(t)}_{\mathcal{T}} + \delta^{(t)}_{\beta} + \delta_{\mathrm{HW}} + \delta^{(t)}_{\mathbf{g}_i} + \delta^{(t)}_{\mathbf{g}}$, $\delta^{(t)}_{\mathcal{T}} = 2m \exp(-C^2/\mathcal{O}^*(\epsilon_t^2))$, $\delta^{(t)}_{\beta} = 2k\exp(-1/\mathcal{O}(\epsilon_t))$, $\delta^{(t)}_{\mathrm{HW}} = \exp(-1/\mathcal{O}(\epsilon_t))$, $\delta^{(t)}_{\mathbf{g}_i} = \exp(-\Omega(k))$, $\delta^{(t)}_{\mathbf{g}} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$, *and* $\|\mathbf{A}^{(t)}_i - \mathbf{A}^*_i\| \leq \epsilon_t$.

Step I: Coefficients have the correct signed-support

As a first step, we ensure that our coefficient estimate has the correct signed-support (Def. 2.2). To this end, we first show that the initialization has the correct signed-support, and then show that the iterative hard-thresholding (IHT)-based update step preserves the correct signed-support for a suitable choice of parameters.

- **Step I.A: Showing that the initial coefficient estimate has the correct signed-support–** Given an $(\epsilon_0, 2)$-close estimate $\mathbf{A}^{(0)}$ of $\mathbf{A}^*$, we first show that for a general sample $\mathbf{y}$ the initialization step (2.3) recovers the correct signed-support with probability at least $(1-\delta_{\mathcal{T}}^{(t)})$, where $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$. This is encapsulated by the following lemma.

  Lemma 2.1 (**Signed-support recovery by coefficient initialization step**)**.** Suppose $\mathbf{A}^{(t)}$ is ϵ_t-close to $\mathbf{A}^*$. Then, if $\mu = \mathcal{O}(\log(n))$, $k = \mathcal{O}^*(\sqrt{n}/\mu\log(n))$, and $\epsilon_t = \mathcal{O}^*(1/\sqrt{\log(m)})$, with probability at least $(1-\delta_{\mathcal{T}}^{(t)})$ for each random sample $\mathbf{y} = \mathbf{A}^*\mathbf{x}^*$:

  $$\mathrm{sign}(\mathcal{T}_{C/2}((\mathbf{A}^{(t)})^\top\mathbf{y}) = \mathrm{sign}(\mathbf{x}^*),$$

  where $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$.

  Note that this result only requires the dictionary to be column-wise close to the true dictionary, and works for less stringent conditions on the initial dictionary estimate, i.e., requires $\epsilon_t = \mathcal{O}^*(1/\sqrt{\log(m)})$ instead of $\epsilon_t = \mathcal{O}^*(1/\log(m))$; see also (Arora et al., 2015).

- **Step I.B: The iterative IHT-type updates preserve the correct signed support–** Next, we show that the IHT-type coefficient update step (2.4) preserves the correct signed-support for an appropriate choice of step-size parameter $\eta_x^{(r)}$ and threshold $\tau^{(r)}$. The choice of these parameters arises from the analysis of the IHT-based update step. Specifically, we show that at each iterate r, the step-size $\eta_x^{(r)}$ should be chosen to ensure that the component corresponding to the true coefficient value is greater than the "interference" introduced by other non-zero coefficient elements. Then, if the threshold is chosen to reject this "noise", each iteration of the IHT-based update step preserves the correct signed-support.

  Lemma 2.2 (**IHT update step preserves the correct signed-support**)**.** Suppose $\mathbf{A}^{(t)}$ is ϵ_t-close to $\mathbf{A}^*$, $\mu = \mathcal{O}(\log(n))$, $k = \mathcal{O}^*(\sqrt{n}/\mu\log(n))$, and $\epsilon_t = \mathcal{O}^*(1/\log(m))$

Then, with probability at least $(1-\delta_\beta^{(t)}-\delta_\mathcal{T}^{(t)})$, each iterate of the IHT-based coefficient update step shown in (2.4) has the correct signed-support, if for a constant $c_1^{(r)}(\epsilon_t,\mu,k,n) = \widetilde{\Omega}(k^2/n)$, the step size is chosen as $\eta_x^{(r)} \le c_1^{(r)}$, and the threshold $\tau^{(r)}$ is chosen as

$$\tau^{(r)} = \eta_x^{(r)}(t_\beta + \tfrac{\mu_t}{\sqrt{n}}\|\mathbf{x}^{(r-1)} - \mathbf{x}^*\|_1) := c_2^{(r)}(\epsilon_t,\mu,k,n) = \widetilde{\Omega}(k^2/n),$$

for some constants c_1 and c_2. Here, $t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$, $\delta_\mathcal{T}^{(t)} = 2m\ \exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$,and $\delta_\beta^{(t)} = 2k\ \exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.

Note that, although we have a dependence on the iterate r in choice of $\eta_x^{(r)}$ and $\tau^{(r)}$, these can be set to some constants independent of r. In practice, this dependence allows for greater flexibility in the choice of these parameters.

Step II: Analyzing the coefficient estimate

We now derive an upper-bound on the error incurred by each non-zero coefficient element. Further, we derive an expression for the coefficient estimate at the t-th round of the online algorithm $\widehat{\mathbf{x}}^{(t)} := \mathbf{x}^{(R)}$; we use $\widehat{\mathbf{x}}$ instead of $\widehat{\mathbf{x}}^{(t)}$ for simplicity.

- **Step II.A: Derive a bound on the error incurred by the coefficient estimate–** Since Lemma 2.2 ensures that $\widehat{\mathbf{x}}$ has the correct signed-support, we now focus on the error incurred by each coefficient element on the support by analyzing $\widehat{\mathbf{x}}$. To this end, we carefully analyze the effect of the recursive update (2.4), to decompose the error incurred by each element on the support into two components – one that depends on the initial coefficient estimate $\mathbf{x}^{(0)}$ and other that depends on the error in the dictionary.

  We show that the effect of the component that depends on the initial coefficient estimate diminishes by a factor of $(1-\eta_x+\eta_x\frac{\mu_t}{\sqrt{n}})$ at each iteration r. Therefore, for a decay parameter δ_R, we can choose the number of IHT iterations R, to make this component arbitrarily small. Therefore, the error in the coefficients only depends on the per column error in the dictionary, formalized by the following result.

  Lemma 2.3 (**Upper-bound on the error in coefficient estimation**)**.** With probability at least $(1-\delta_\beta^{(t)}-\delta_\mathcal{T}^{(t)})$ the error incurred by each element $i_1 \in \mathrm{supp}(\mathbf{x}^*)$ of the coefficient estimate is upper-bounded as

$$|\widehat{\mathbf{x}}_{i_1} - \mathbf{x}^*_{i_1}| \le \mathcal{O}(t_\beta) + \Big((R+1)k\eta_x \tfrac{\mu_t}{\sqrt{n}} \max_i |\mathbf{x}^{(0)}_i - \mathbf{x}^*_i| + |\mathbf{x}^{(0)}_{i_1} - \mathbf{x}^*_{i_1}|\Big)\delta_R, = \mathcal{O}(t_\beta)$$

where $t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$, $\delta_R := (1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}})^R$, $\delta_{\mathcal{T}}^{(t)} = 2m\ \exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$, $\delta_\beta^{(t)} = 2k\ \exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$, and μ_t is the incoherence between the columns of $\mathbf{A}^{(t)}$; see Claim 1.

This result allows us to show that if the column-wise error in the dictionary decreases at each iteration t, then the corresponding estimates of the coefficients also improve.

- *Step II.B: Developing an expression for the coefficient estimate*– Next, we derive the expression for the coefficient estimate in the following lemma. This expression is used to analyze the dictionary update.

  Lemma 2.4 (**Expression for the coefficient estimate at the end of R-th IHT iteration**)**.** With probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_\beta^{(t)})$ the i_1-th element of the coefficient estimate, for each $i_1 \in \text{supp}(\mathbf{x}^*)$, is given by

  $$\widehat{\mathbf{x}}_{i_1} := \mathbf{x}^{(R)}_{i_1} = \mathbf{x}^*_{i_1}(1 - \lambda^{(t)}_{i_1}) + \vartheta^{(R)}_{i_1}.$$

  Here, $\vartheta^{(R)}_{i_1}$ is $|\vartheta^{(R)}_{i_1}| = \mathcal{O}(t_\beta)$, where $t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$. Further, $\lambda^{(t)}_{i_1} = |\langle \mathbf{A}^{(t)}_{i_1} - \mathbf{A}^*_{i_1}, \mathbf{A}^*_{i_1}\rangle| \le \frac{\epsilon_t^2}{2}$, $\delta_{\mathcal{T}}^{(t)} = 2m\ \exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$ and $\delta_\beta^{(t)} = 2k\ \exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.

  We again observe that the error in the coefficient estimate depends on the error in the dictionary via $\lambda^{(t)}_{i_1}$ and $\vartheta^{(R)}_{i_1}$.

Step III: Analyzing the gradient for dictionary update

Given the coefficient estimate we now show that the choice of the gradient as shown in (2.5) makes progress at each step. To this end, we analyze the gradient vector corresponding to each dictionary element to see if it satisfies the local descent condition of Def. 2.5. Our analysis of the gradient is motivated from Arora et al. (2015). However, as opposed to the simple HT-based coefficient update step used by Arora et al. (2015), our IHT-based coefficient estimate adds to significant overhead in terms of analysis. Notwithstanding the complexity of the analysis, we show that this allows us to remove the bias in the gradient estimate.

To this end, we first develop an expression for each expected gradient vector, show that the empirical gradient estimate concentrates around its mean, and finally show that the empirical gradient vector is $(\Omega(k/m), \Omega(m/k), 0)$-correlated with the descent direction, i.e. has no bias.

- **Step III.A: Develop an expression for the expected gradient vector corresponding to each dictionary element–** The expression for the expected gradient vector $\mathbf{g}_j^{(t)}$ corresponding to j-th dictionary element is given by the following lemma.

  Lemma 2.5 (**Expression for the expected gradient vector**)**.** Suppose that $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$-near to $\mathbf{A}^*$. Then, the dictionary update step in Algorithm 1 amounts to the following for the j-th dictionary element

  $$\mathbf{E}[\mathbf{A}_j^{(t+1)}] = \mathbf{A}_j^{(t)} + \eta_A \mathbf{g}_j^{(t)},$$

  where $\mathbf{g}_j^{(t)}$ is given by

  $$\mathbf{g}_j^{(t)} = q_j p_j \Big((1 - \lambda_j^{(t)}) \mathbf{A}_j^{(t)} - \mathbf{A}_j^* + \tfrac{1}{q_j p_j} \Delta_j^{(t)} \pm \gamma \Big),$$

  $\lambda_j^{(t)} = |\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^* \rangle|$, and $\Delta_j^{(t)} := \mathbf{E}[\mathbf{A}_S^{(t)} \vartheta_S^{(R)} \mathrm{sign}(\mathbf{x}_j^*)]$, where $\|\Delta_j^{(t)}\| = \mathcal{O}(\sqrt{m} q_{i,j} p_j \epsilon_t$ $\|\mathbf{A}^{(t)}\|)$.

- **Step III.B: Show that the empirical gradient vector concentrates around its expectation–** Since we only have access to the empirical gradient vectors, we show that these concentrate around their expected value via the following lemma.

  Lemma 2.6 (**Concentration of the empirical gradient vector**)**.** Given $p = \widetilde{\Omega}(mk^2)$ samples, the empirical gradient vector estimate corresponding to the i-th dictionary element, $\widehat{\mathbf{g}}_i^{(t)}$ concentrates around its expectation, i.e.,

  $$\|\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}\| \leq o(\tfrac{k}{m} \epsilon_t).$$

  with probability at least $(1 - \delta_{\mathbf{g}_i}^{(t)} - \delta_\beta^{(t)} - \delta_\mathcal{T}^{(t)} - \delta_{\mathrm{HW}}^{(t)})$, where $\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(k))$.

- *Step III.C: Show that the empirical gradient vector is correlated with the descent direction*– Next, in the following lemma we show that the empirical gradient vector $\widehat{\mathbf{g}}_j^{(t)}$ is correlated with the descent direction. This is the main result which enables the progress in the dictionary (and coefficients) at each iteration t.

Lemma 2.7 (**Empirical gradient vector is correlated with the descent direction**)**.** Suppose $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$-near to $\mathbf{A}^*$, $k = \mathcal{O}(\sqrt{n})$ and $\eta_A = \mathcal{O}(m/k)$. Then, with probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_{\beta}^{(t)} - \delta_{\text{HW}}^{(t)} - \delta_{\mathbf{g}_i}^{(t)})$ the empirical gradient vector $\widehat{\mathbf{g}}_j^{(t)}$ is $(\Omega(k/m), \Omega(m/k), 0)$-correlated with $(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*)$, and for any $t \in [T]$,

$$\|\mathbf{A}_j^{(t+1)} - \mathbf{A}_j^*\|^2 \leq (1 - \rho\_\eta_A)\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2.$$

This result ensures for at any $t \in [T]$, the gradient descent-based updates made via (2.5) gets the columns of the dictionary estimate closer to the true dictionary, i.e., $\epsilon_{t+1} \leq \epsilon_t$. Moreover, this step requires closeness between the dictionary estimate $\mathbf{A}^{(t)}$ and $\mathbf{A}^*$, in the spectral norm-sense, as per Def 2.1.

Step IV: Show that the dictionary maintains the closeness property

As discussed above, the closeness property (Def 2.1) is crucial to show that the gradient vector is correlated with the descent direction. Therefore, we now ensure that the updated dictionary $\mathbf{A}^{(t+1)}$ maintains this closeness property. Lemma 2.7 already ensures that $\epsilon_{t+1} \leq \epsilon_t$. As a result, we show that $\mathbf{A}^{(t+1)}$ maintains closeness in the spectral norm-sense as required by our algorithm, i.e., that it is still $(\epsilon_{t+1}, 2)$-close to the true dictionary. Also, since we use the gradient matrix in this analysis, we show that the empirical gradient matrix concentrates around its mean.

- *Step IV.A: The empirical gradient matrix concentrates around its expectation*: We first show that the empirical gradient matrix concentrates as formalized by the following lemma.

  Lemma 2.8 (**Concentration of the empirical gradient matrix**)**.** With probability at least $(1-\delta_{\beta}^{(t)}-\delta_{\mathcal{T}}^{(t)}-\delta_{\text{HW}}^{(t)}-\delta_{\mathbf{g}}^{(t)})$, $\|\widehat{\mathbf{g}}^{(t)}-\mathbf{g}^{(t)}\|$ is upper-bounded by $\mathcal{O}^*(\frac{k}{m}\|\mathbf{A}^*\|)$, where $\delta_{\mathbf{g}}^{(t)} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$.

- *Step IV.B: The "closeness" property is maintained after the updates made using the empirical gradient estimate*: Next, the following lemma shows that the updated dictionary $\mathbf{A}^{(t+1)}$ maintains the closeness property.

  Lemma 2.9 ($\mathbf{A}^{(t+1)}$ **maintains closeness**)**.** Suppose $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$ near to $\mathbf{A}^*$ with $\epsilon_t = \mathcal{O}^*(1/\log(n))$, and number of samples used in step t is $p = \widetilde{\Omega}(mk^2)$, then with probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_{\beta}^{(t)} - \delta_{\text{HW}}^{(t)} - \delta_{\mathbf{g}}^{(t)})$, $\mathbf{A}^{(t+1)}$ satisfies $\|\mathbf{A}^{(t+1)} - \mathbf{A}^*\| \leq 2\|\mathbf{A}^*\|$.

Step V: Combine results to show the main result

Proof of Theorem 2.1. From Lemma 2.7 we have that with probability at least $(1-\delta_T^{(t)} - \delta_\beta^{(t)} - \delta_{\text{HW}}^{(t)} - \delta_{\mathbf{g}_i}^{(t)})$, $\mathbf{g}_j^{(t)}$ is $(\Omega(k/m), \Omega(m/k), 0)$-correlated with $\mathbf{A}_j^*$. Further, Lemma 2.9 ensures that each iterate maintains the closeness property. Now, applying Lemma 2.15 we have that, for $\eta_A \leq \Theta(m/k)$, with probability at least $(1-\delta_{\text{alg}}^{(t)})$ any $t \in [T]$ satisfies

$$\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2 \leq (1-\omega)^t \|\mathbf{A}_j^{(0)} - \mathbf{A}_j^*\|^2 \leq (1-\omega)^t \epsilon_0^2.$$

where for $0 < \omega < 1/2$ with $\omega = \Omega(k/m)\eta_A$. That is, the updates converge geometrically to $\mathbf{A}^*$. Further, from Lemma 2.3, we have that the result on the error incurred by the coefficients. Here, $\delta_{\text{alg}}^{(t)} = \delta_T^{(t)} + \delta_\beta^{(t)} + \delta_{\text{HW}}^{(t)} + \delta_{\mathbf{g}_i}^{(t)} + \delta_{\mathbf{g}}^{(t)})$. That is, the updates converge geometrically to $\mathbf{A}^*$. Further, from Lemma 2.3, we have that the error in the coefficients only depends on the error in the dictionary, which leads us to our result on the error incurred by the coefficients. This completes the proof of our main result. □

2.C Appendix: Proof of Lemmas

We present the proofs of the Lemmas used to establish our main result. Also, see Table 2.B.1 for a map of dependence between the results, and Appendix 2.D for proofs of intermediate results.

Proof of Lemma 2.1. Let $\mathbf{y} \in \mathbb{R}^n$ be general sample generated as $\mathbf{y} = \mathbf{A}^*\mathbf{x}^*$, where $\mathbf{x}^* \in \mathbb{R}^m$ is a sparse random vector with support $S = \text{supp}(\mathbf{x}^*)$ distributed according to **D.2.4.**

The initial decoding step at the t-th iteration (shown in Algorithm 1) involves evaluating the inner-product between the estimate of the dictionary $\mathbf{A}^{(t)}$, and $\mathbf{y}$. The i-th element of the resulting vector can be written as

$$\langle \mathbf{A}_i^{(t)}, \mathbf{y} \rangle = \langle \mathbf{A}_i^{(t)}, \mathbf{A}_i^* \rangle \mathbf{x}_i^* + \mathbf{w}_i,$$

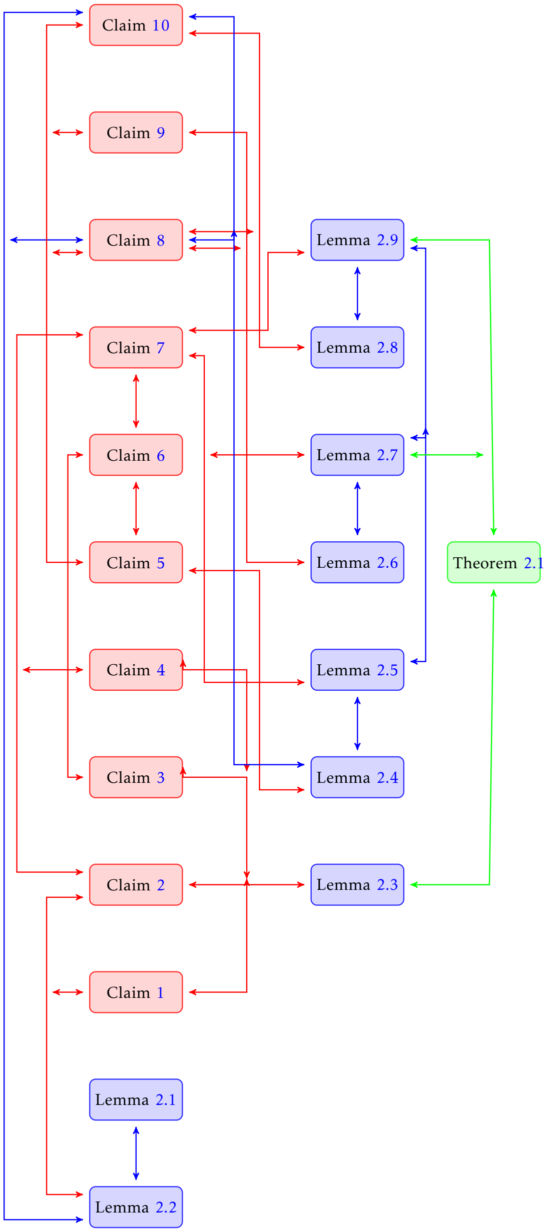
where $\mathbf{w}_i = \langle \mathbf{A}_i^{(t)}, \mathbf{A}_{-i}^* \mathbf{x}_{-i}^* \rangle$. Now, since $\|\mathbf{A}_i^* - \mathbf{A}_i^{(t)}\|_2 \leq \epsilon_t$ and

$$\|\mathbf{A}_i^* - \mathbf{A}_i^{(t)}\|_2^2 = \|\mathbf{A}_i^*\|^2 + \|\mathbf{A}_i^{(t)}\|^2 - 2\langle \mathbf{A}_i^{(t)}, \mathbf{A}_i^* \rangle = 2 - 2\langle \mathbf{A}_i^{(t)}, \mathbf{A}_i^* \rangle,$$

Table 2.B.1: Proof map: dependence of results.

Lemmas	Result	Dependence
Lemma 2.1	Signed-support recovery by coefficient initialization step	–
Lemma 2.2	IHT update step preserves the correct signed-support	Claim 1, Lemma 2.1, and Claim 2
Lemma 2.3	Upper-bound on the error in coefficient estimation	Claim 1, Claim 2, Claim 3, and Claim 4
Lemma 2.4	Expression for the coefficient estimate at the end of R-th IHT iteration	Claim 5
Lemma 2.5	Expression for the expected gradient vector	Lemma 2.4 and Claim 7
Lemma 2.6	Concentration of the empirical gradient vector	Claim 8 and Claim 9
Lemma 2.7	Empirical gradient vector is correlated with the descent direction	Lemma 2.5, Claim 7 and Lemma 2.6
Lemma 2.8	Concentration of the empirical gradient matrix	Claim 8 and Claim 10
Lemma 2.9	$\mathbf{A}^{(t+1)}$ maintains closeness	Lemma 2.5, Claim 7 and Lemma 2.8

Claims	Result	Dependence
Claim 1	Incoherence of $\mathbf{A}^{(t)}$	–
Claim 2	Bound on $\beta_j^{(t)}$: the noise component in coefficient estimate that depends on ϵ_t	–
Claim 3	Error in coefficient estimation for a general iterate $(r+1)$	–
Claim 4	An intermediate result for bounding the error in coefficient calculations	Claim 2
Claim 5	Bound on the noise term in the estimation of a coefficient element in the support	Claim 6
Claim 6	An intermediate result for $\vartheta_{i_1}^{(R)}$ calculations	Claim 3
Claim 7	Bound on the noise term in expected gradient vector estimate	Claim 6 and Claim 2
Claim 8	An intermediate result for concentration results	Lemma 2.2 ,Lemma 2.4 and Claim 5
Claim 9	Bound on variance parameter for concentration of gradient vector	Claim 5
Claim 10	Bound on variance parameter for concentration of gradient matrix	Lemma 2.2 , Lemma 2.4 and Claim 5



we have

$$|\langle \mathbf{A}_i^{(t)}, \mathbf{A}_i^* \rangle| \geq 1 - \epsilon_t^2/2.$$

Therefore, the term

$$|\langle \mathbf{A}_i^{(t)}, \mathbf{A}_i^* \rangle \mathbf{x}_i^*| \begin{cases} \geq (1 - \frac{\epsilon_t^2}{2})C & \text{,if } i \in S, \\ = 0 & \text{,otherwise.} \end{cases}$$

Now, we focus on the $\mathbf{w}_i$ and show that it is small. By the definition of $\mathbf{w}_i$ we have

$$\mathbf{w}_i = \langle \mathbf{A}_i^{(t)}, \mathbf{A}_{-i}^* \mathbf{x}_{-i}^* \rangle = \sum_{\ell \neq i} \langle \mathbf{A}_i^{(t)}, \mathbf{A}_\ell^* \rangle \mathbf{x}_\ell^* = \sum_{\ell \in S \backslash \{i\}} \langle \mathbf{A}_i^{(t)}, \mathbf{A}_\ell^* \rangle \mathbf{x}_\ell^*.$$

Here, since $var(\mathbf{x}_\ell^*) = 1$, $\mathbf{w}_i$ is a zero-mean random variable with variance

$$var(\mathbf{w}_i) = \sum_{\ell \in S \backslash \{i\}} \langle \mathbf{A}_i^{(t)}, \mathbf{A}_\ell^* \rangle^2.$$

Now, each term in this sum can be bounded as,

$$\begin{aligned} \langle \mathbf{A}_i^{(t)}, \mathbf{A}_\ell^* \rangle^2 &= (\langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle + \langle \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle)^2 \\ &\leq 2(\langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle^2 + \langle \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle^2) \\ &\leq 2(\langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle^2 + \frac{\mu^2}{n}). \end{aligned}$$

Next, $\sum_{\ell \neq i} \langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle^2$ can be upper-bounded as

$$\sum_{\ell \in S \backslash \{i\}} \langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle^2 \leq \|\mathbf{A}_{S \backslash \{i\}}^*\|^2 \epsilon_t^2.$$

Therefore, we have the following as per our assumptions on μ and k,

$$\|\mathbf{A}_{S \backslash \{i\}}^*\|^2 \leq (1 + k \frac{\mu}{\sqrt{n}}) \leq 2,$$

using Gershgorin Circle Theorem (Gershgorin, 1931). Therefore, we have

$$\sum_{\ell \in S \backslash \{i\}} \langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_\ell^* \rangle^2 \leq 2\epsilon_t^2.$$

Finally, we have that

$$\sum_{\ell \in S\backslash\{i\}} \langle \mathbf{A}_i^{(t)}, \mathbf{A}_\ell^* \rangle^2 \leq 2(2\epsilon_t^2 + k\tfrac{\mu^2}{n}) = \mathcal{O}^*(\epsilon_t^2).$$

Now, we apply the Chernoff bound for sub-Gaussian random variables $\mathbf{w}_i$ (shown in Lemma 2.12) to conclude that

$$\mathbf{Pr}[|\mathbf{w}_i| \geq C/4] \leq 2\exp(-\tfrac{C^2}{\mathcal{O}^*(\epsilon_t^2)}).$$

Further, $\mathbf{w}_i$ corresponding to each m should follow this bound, applying union bound we conclude that

$$\mathbf{Pr}[\max_i |\mathbf{w}_i| \geq C/4] \leq 2m\exp(-\tfrac{C^2}{\mathcal{O}^*(\epsilon_t^2)}) := \delta_{\mathcal{T}}^{(t)}.$$

□

Proof of Lemma 2.2. Consider the $(r+1)$-th iterate $\mathbf{x}^{(r+1)}$ for the t-th dictionary iterate, where $\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \leq \epsilon_t$ for all $i \in [1, m]$ evaluated as the following by the update step described in Algorithm 1,

$$\begin{aligned}\mathbf{x}^{(r+1)} &= \mathbf{x}^{(r)} - \eta_x^{(r+1)} \mathbf{A}^{(t)^\top}(\mathbf{A}^{(t)}\mathbf{x}^{(r)} - \mathbf{y}) \\ &= (\mathbf{I} - \eta_x^{(r+1)}\mathbf{A}^{(t)^\top}\mathbf{A}^{(t)})\mathbf{x}^{(r)} - \eta_x^{(r+1)}\mathbf{A}^{(t)^\top}\mathbf{A}^*\mathbf{x}^*, \qquad (2.7)\end{aligned}$$

where $\eta_x^{(1)} < 1$ is the learning rate or the step-size parameter. Now, using Lemma 2.1 we know that $\mathbf{x}^{(0)}$ (2.3) has the correct signed-support with probability at least $(1-\delta_{\mathcal{T}}^{(t)})$. Further, since $\mathbf{A}^{(t)^\top}\mathbf{A}^*$ can be written as

$$\mathbf{A}^{(t)^\top}\mathbf{A}^* = (\mathbf{A}^{(t)} - \mathbf{A}^*)^\top\mathbf{A}^* + \mathbf{A}^{*\top}\mathbf{A}^*,$$

we can write the $(r+1)$-th iterate of the coefficient update step using (2.7) as

$$\mathbf{x}^{(r+1)} = (\mathbf{I} - \eta_x^{(r+1)}\mathbf{A}^{(t)^\top}\mathbf{A}^{(t)})\mathbf{x}^{(r)} - \eta_x^{(r+1)}(\mathbf{A}^{(t)} - \mathbf{A}^*)^\top\mathbf{A}^*\mathbf{x}^* + \eta_x^{(r+1)}\mathbf{A}^{*\top}\mathbf{A}^*\mathbf{x}^*.$$

Further, the j-th entry of this vector is given by

$$\mathbf{x}_j^{(r+1)} = (\mathbf{I} - \eta_x^{(r+1)}\mathbf{A}^{(t)^\top}\mathbf{A}^{(t)})_{(j,:)}\mathbf{x}^{(r)} - \eta_x^{(r+1)}((\mathbf{A}^{(t)} - \mathbf{A}^*)^\top\mathbf{A}^*)_{(j,:)}\mathbf{x}^* + \eta_x^{(r+1)}(\mathbf{A}^{*\top}\mathbf{A}^*)_{(j,:)}\mathbf{x}^*. \qquad (2.8)$$

We now develop an expression for the j-th element of each of the term in (2.8) as follows. First, we can write the first term as

$$(\mathbf{I} - \eta_x^{(r+1)}\mathbf{A}^{(t)\top}\mathbf{A}^{(t)})_{(j,:)}\mathbf{x}^{(r)} = (1-\eta_x^{(r+1)})\mathbf{x}_j^{(r)} - \eta_x^{(r+1)}\sum_{i\neq j}\langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)}\rangle \mathbf{x}_i^{(r)}.$$

Next, the second term in (2.8) can be expressed as

$$\begin{aligned}\eta_x^{(r+1)}((\mathbf{A}^{(t)} - \mathbf{A}^*)^\top \mathbf{A}^*)_{(j,:)}\mathbf{x}^* &= \eta_x^{(r+1)}\sum_i \langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^*\rangle \mathbf{x}_i^* \\ &= \eta_x^{(r+1)}\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^*\rangle \mathbf{x}_j^* + \eta_x^{(r+1)}\sum_{i\neq j}\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^*\rangle \mathbf{x}_i^*.\end{aligned}$$

Finally, we have the following expression for the third term,

$$\eta_x^{(r+1)}(\mathbf{A}^{*\top}\mathbf{A}^*)_{(j,:)}\mathbf{x}^* = \eta_x^{(r+1)}\mathbf{x}_j^* + \eta_x^{(r+1)}\sum_{i\neq j}\langle \mathbf{A}_j^*, \mathbf{A}_i^*\rangle \mathbf{x}_i^*.$$

Now using our definition of $\lambda_j^{(t)} = |\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^*\rangle| \leq \frac{\epsilon_t^2}{2}$, combining all the results for (2.8), and using the fact that since $\mathbf{A}^{(t)}$ is close to $\mathbf{A}^*$, vectors $\mathbf{A}_j^{(t)} - \mathbf{A}_j^*$ and $\mathbf{A}_j^*$ enclose an obtuse angle, we have the following for the j-th entry of the $(r+1)$-th iterate, $\mathbf{x}^{(r+1)}$ is given by

$$\mathbf{x}_j^{(r+1)} = (1-\eta_x^{(r+1)})\mathbf{x}_j^{(r)} + \eta_x^{(r+1)}(1-\lambda_j^{(t)})\mathbf{x}_j^* + \eta_x^{(r+1)}\xi_j^{(r+1)}. \tag{2.9}$$

Here $\xi_j^{(r+1)}$ is defined as

$$\xi_j^{(r+1)} := \sum_{i\neq j}(\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^*\rangle + \langle \mathbf{A}_j^*, \mathbf{A}_i^*\rangle)\mathbf{x}_i^* - \sum_{i\neq j}\langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)}\rangle \mathbf{x}_i^{(r)}.$$

Since, $\langle \mathbf{A}_j^*, \mathbf{A}_i^*\rangle - \langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)}\rangle = \langle \mathbf{A}_j^*, \mathbf{A}_i^* - \mathbf{A}_i^{(t)}\rangle + \langle \mathbf{A}_j^* - \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)}\rangle$, we can write $\xi_j^{(r+1)}$ as

$$\xi_j^{(r+1)} = \beta_j^{(t)} + \sum_{i\neq j}\langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)}\rangle(\mathbf{x}_i^* - \mathbf{x}_i^{(r)}), \tag{2.10}$$

where $\beta_j^{(t)}$ is defined as

$$\beta_j^{(t)} := \sum_{i\neq j}(\langle \mathbf{A}_j^*, \mathbf{A}_i^* - \mathbf{A}_i^{(t)}\rangle + \langle \mathbf{A}_j^* - \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)}\rangle + \langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^*\rangle)\mathbf{x}_i^*. \tag{2.11}$$

Note that $\beta_j^{(t)}$ does not change for each iteration r of the coefficient update step. Further, by Claim 2 we show that $|\beta_j^{(t)}| \leq t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$ with probability at least $(1-\delta_\beta^{(t)})$. Next, we define $\widetilde{\xi}_j^{(r+1)}$ as

$$\widetilde{\xi}_j^{(r+1)} := \beta_j^{(t)} + \sum_{i \neq j} |\langle \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)} \rangle| |\mathbf{x}_i^* - \mathbf{x}_i^{(r)}|. \tag{2.12}$$

where $\xi_j^{(r+1)} \leq \widetilde{\xi}_j^{(r+1)}$. Further, using Claim 1,

$$\widetilde{\xi}_j^{(r+1)} \leq t_\beta + \tfrac{\mu_t}{\sqrt{n}} \|\mathbf{x}_j^* - \mathbf{x}_j^{(r)}\|_1 := \widetilde{\xi}_{\max}^{(r+1)} = \widetilde{\mathcal{O}}(\tfrac{k}{\sqrt{n}}), \tag{2.13}$$

since $\|\mathbf{x}^{(r-1)} - \mathbf{x}^*\|_1 = \mathcal{O}(k)$. Therefore, for the $(r+1)$-th iteration, we choose the threshold to be

$$\tau^{(r+1)} := \eta_x^{(r+1)} \widetilde{\xi}_{\max}^{(r+1)}, \tag{2.14}$$

and the step-size by setting the "noise" component of (2.9) to be smaller than the "signal" part, specifically, half the signal component, i.e.,

$$\eta_x^{(r+1)} \widetilde{\xi}_{\max}^{(r+1)} \leq \tfrac{(1-\eta_x^{(r+1)})}{2} \mathbf{x}_{\min}^{(r)} + \tfrac{\eta_x^{(r+1)}}{2}(1 - \tfrac{\epsilon_t^2}{2})C,$$

Also, since we choose the threshold as $\tau^{(r)} := \eta_x^{(r)} \widetilde{\xi}_{\max}^{(r)}$, $\mathbf{x}_{\min}^{(r)} = \eta_x^{(r)} \widetilde{\xi}_{\max}^{(r)}$, where $\mathbf{x}_{\min}^{(0)} = C/2$, we have the following for the $(r+1)$-th iteration,

$$\eta_x^{(r+1)} \widetilde{\xi}_{\max}^{(r+1)} \leq \tfrac{(1-\eta_x^{(r+1)})}{2} \eta_x^{(r)} \widetilde{\xi}_{\max}^{(r)} + \tfrac{\eta_x^{(r+1)}}{2}(1 - \tfrac{\epsilon_t^2}{2})C.$$

Therefore, for this step we choose $\eta_x^{(r+1)}$ as

$$\eta_x^{(r+1)} \leq \frac{\frac{\eta_x^{(r)}}{2} \widetilde{\xi}_{\max}^{(r)}}{\widetilde{\xi}_{\max}^{(r+1)} + \frac{\eta_x^{(r)}}{2} \widetilde{\xi}_{\max}^{(r)} - \frac{1}{2}(1 - \frac{\epsilon_t^2}{2})C}, \tag{2.15}$$

Therefore, $\eta_x^{(r+1)}$ can be chosen as

$$\eta_x^{(r+1)} \leq c^{(r+1)}(\epsilon_t, \mu, k, n),$$

for a small constant $c^{(r+1)}(\epsilon_t, \mu, k, n)$, $\eta_x^{(r+1)}$. In addition, if we set all $\eta_x^{(r)} = \eta_x$, we have that $\eta_x = \widetilde{\Omega}(\frac{k}{\sqrt{n}})$ and therefore $\tau^{(r)} = \tau = \widetilde{\Omega}(\frac{k^2}{n})$. Further, since we initialize with the hard-thresholding step, the entries in $|\mathbf{x}^{(0)}| \geq C/2$. Here, we define $\widetilde{\xi}^{(0)}_{\max} = C$ and $\eta_x^{(0)} = 1/2$, and set the threshold for initial step as $\eta_x^{(0)}\widetilde{\xi}^{(0)}_{\max}$.

□

Proof of Lemma 2.3. Using the definition of $\widetilde{\xi}^{(\ell)}_{i_1}$ as in (2.12), we have

$$\widetilde{\xi}^{(\ell)}_{i_1} = \beta^{(t)}_{i_1} + \sum_{i_2 \neq i_1} |\langle \mathbf{A}^{(t)}_{i_1}, \mathbf{A}^{(t)}_{i_2} \rangle| |\mathbf{x}^*_{i_2} - \mathbf{x}^{(\ell-1)}_{i_2}|.$$

From Claim 2 we have that $|\beta^{(t)}_{i_1}| \leq t_\beta$ with probability at least $(1 - \delta^{(t)}_\beta)$. Further, using Claim 1 , and letting $C^{(\ell)}_i := |\mathbf{x}^*_i - \mathbf{x}^{(\ell)}_i| = |\mathbf{x}^{(\ell)}_i - \mathbf{x}^*_i|$, $\widetilde{\xi}^{(\ell)}_{i_1}$ can be upper-bounded as

$$\widetilde{\xi}^{(\ell)}_{i_1} \leq \beta^{(t)}_{i_1} + \tfrac{\mu_t}{\sqrt{n}} \sum_{i_2 \neq i_1} C^{(\ell-1)}_{i_2}. \tag{2.16}$$

Rearranging the expression for $(r+1)$-th update (2.9), and using (2.16) we have the following upper-bound

$$C^{(r+1)}_{i_1} \leq (1 - \eta_x^{(r+1)}) C^{(r)}_{i_1} + \eta_x^{(r+1)} \lambda^{(t)}_{i_1} |\mathbf{x}^*_{i_1}| + \eta_x^{(r+1)} \widetilde{\xi}^{(r+1)}_{i_1}.$$

Next, recursively substituting in for $C^{(r)}_{i_1}$, where we define $\prod_{q=\ell}^{\ell}(1 - \eta_x^{(q+1)}) = 1$,

$$C^{(r+1)}_{i_1} \leq C^{(0)}_{i_1} \prod_{q=0}^{r}(1 - \eta_x^{(q+1)}) + \lambda^{(t)}_{i_1} |\mathbf{x}^*_{i_1}| \sum_{\ell=1}^{r+1} \eta_x^{(\ell)} \prod_{q=\ell}^{r+1}(1 - \eta_x^{(q+1)}) + \sum_{\ell=1}^{r+1} \eta_x^{(\ell)} \widetilde{\xi}^{(\ell)}_{i_1} \prod_{q=\ell}^{r+1}(1 - \eta_x^{(q+1)}).$$

Substituting for the upper-bound of $\widetilde{\xi}^{(\ell)}_{i_1}$ from (2.16),

$$C^{(r+1)}_{i_1} \leq \alpha^{(r+1)}_{i_1} + \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r+1} \eta_x^{(\ell)} \sum_{i_2 \neq i_1} C^{(\ell-1)}_{i_2} \prod_{q=\ell}^{r+1}(1 - \eta_x^{(q+1)}). \tag{2.17}$$

Here, $\alpha^{(r+1)}_{i_1}$ is defined as

$$\alpha^{(r+1)}_{i_1} = C^{(0)}_{i_1} \prod_{q=0}^{r}(1 - \eta_x^{(q+1)}) + (\lambda^{(t)}_{i_1} |\mathbf{x}^*_{i_1}| + \beta^{(t)}_{i_1}) \sum_{\ell=1}^{r+1} \eta_x^{(\ell)} \prod_{q=\ell}^{r+1}(1 - \eta_x^{(q+1)}). \tag{2.18}$$

Our aim now will be to express $C^{(\ell)}_{i_1}$ for $\ell > 0$ in terms of $C^{(0)}_{i_2}$. Let each $\alpha^{(\ell)}_j \leq \alpha^{(\ell)}_i$ where $j = i_1, i_2, \ldots, i_k$. Similarly, let $C^{(0)}_j \leq C^{(0)}_i$ for $j = i_1, i_2, \ldots, i_k$, and all $\eta_x^{(\ell)} = \eta_x$. Then, using

Claim 3 we have the following expression for $C_{i_1}^{(R+1)}$,

$$C_{i_1}^{(R+1)} \leq \alpha_{i_1}^{(R+1)} + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{R} \alpha_{\max}^{(\ell)} \Big(1-\eta_x+\eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell} \\ + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \Big(1-\eta_x+\eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R}.$$

Here, $(1-\eta_x)^R \leq (1-\eta_x+\eta_x \frac{\mu_t}{\sqrt{n}})^R \leq \delta_R$. Next from Claim 4 we have that with probability at least $(1-\delta_\beta^{(t)})$,

$$\sum_{\ell=1}^{R} \alpha_{\max}^{(\ell)} \Big(1-\eta_x+\eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell} \leq C_{\max}^{(0)} R \delta_R + \tfrac{1}{\eta_x(1-\frac{\mu_t}{\sqrt{n}})}(\tfrac{\epsilon_t^2}{2}|\mathbf{x}_{\max}^*| + t_\beta).$$

Therefore, for $c_x = \frac{\mu_t}{\sqrt{n}}/(1-\frac{\mu_t}{\sqrt{n}})$

$$C_{i_1}^{(R+1)} \leq \alpha_{i_1}^{(R+1)} + (k-1)c_x(\tfrac{\epsilon_t^2}{2}|\mathbf{x}_{\max}^*| + t_\beta) + (R+1)(k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \delta_R.$$

Now, using the definition of $\alpha_{i_1}^{(R+1)}$, and using the result on sum of geometric series, we have

$$\alpha_{i_1}^{(R+1)} = C_{i_1}^{(0)}(1-\eta_x)^{R+1} + (\lambda_{i_1}^{(t)}|\mathbf{x}_{i_1}^*| + \beta_{i_1}^{(t)}) \sum_{s=1}^{R+1} \eta_x(1-\eta_x)^{R-s+1}, \\ = C_{i_1}^{(0)}\delta_R + \lambda_{i_1}^{(t)}|\mathbf{x}_{i_1}^*| + \beta_{i_1}^{(t)} \leq C_{i_1}^{(0)}\delta_{R+1} + \tfrac{\epsilon_t^2}{2}|\mathbf{x}_{\max}^*| + t_\beta.$$

Therefore, $C_{i_1}^{(R)}$ is upper-bounded as

$$C_{i_1}^{(R)} \leq (c_x k+1)(\tfrac{\epsilon_t^2}{2}|\mathbf{x}_{\max}^*| + t_\beta) + (R+1)k\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)}\delta_R + C_{i_1}^{(0)}\delta_R.$$

Further, since $k = \mathcal{O}(\sqrt{n}/\mu\log(n))$, $kc_x < 1$, therefore, we have

$$C_{i_1}^{(R)} \leq \mathcal{O}(t_\beta) + (R+1)k\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)}\delta_R + C_{i_1}^{(0)}\delta_R,$$

with probability at least $(1-\delta_\beta^{(t)})$. Here, $(R+1)k\eta_x \frac{\mu_t}{\sqrt{n}} C_{\max}^{(0)}\delta_R + C_{i_1}^{(0)}\delta_R \approxeq 0$ for an appropriately large R. Therefore, the error in each non-zero coefficient is

$$C_{i_1}^{(R)} = \mathcal{O}(t_\beta).$$

with probability at least $(1-\delta_\beta^{(t)})$. □

Proof of Lemma 2.4. Using the expression for $\mathbf{x}_{i_1}^{(R)}$ as defined in (2.9), and recursively substituting for $\mathbf{x}_{i_1}^{(r)}$ we have

$$\mathbf{x}_{i_1}^{(R)} = (1-\eta_x)^R \mathbf{x}_j^{(0)} + \mathbf{x}_{i_1}^* \sum_{r=1}^{R} \eta_x (1-\lambda_{i_1}^{(t)})(1-\eta_x)^{R-r} + \sum_{r=1}^{R} \eta_x \xi_{i_1}^{(r)} (1-\eta_x)^{R-r},$$

where we set all η_x^r to be η_x. Further, on defining

$$\vartheta_{i_1}^{(R)} := \sum_{r=1}^{R} \eta_x \xi_{i_1}^{(r)} (1-\eta_x)^{R-r} + \gamma_{i_1}^{(R)}, \tag{2.19}$$

where $\gamma_{i_1}^{(R)} := (1-\eta_x)^R (\mathbf{x}_{i_1}^{(0)} - \mathbf{x}_{i_1}^* (1-\lambda_{i_1}^{(t)}))$, we have

$$\begin{aligned} \mathbf{x}_{i_1}^{(R)} &= (1-\eta_x)^R \mathbf{x}_{i_1}^{(0)} + \mathbf{x}_{i_1}^* (1-\lambda_{i_1}^{(t)})(1-(1-\eta_x)^R) + \sum_{r=1}^{R} \eta_x \xi_{i_1}^{(r)} (1-\eta_x)^{R-r}, \\ &= \mathbf{x}_{i_1}^* (1-\lambda_{i_1}^{(t)}) + \vartheta_{i_1}^{(R)}. \end{aligned} \tag{2.20}$$

Note that $\gamma_{i_1}^{(R)}$ can be made appropriately small by choice of R. Further, by Claim 5 we have

$$|\vartheta_{i_1}^{(R)}| \le \mathcal{O}(t_\beta).$$

with probability at least $(1-\delta_\beta^{(t)})$, where $t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$. □

Proof of Lemma 2.5. From Lemma 2.4 we have that for each $j \in S$,

$$\widehat{\mathbf{x}}_S := \mathbf{x}_S^{(R)} = (\mathbf{I} - \Lambda_S^{(t)})\mathbf{x}_S^* + \vartheta_S^{(R)},$$

with probability at least $(1-\delta_\mathcal{T}^{(t)} - \delta_\beta^{(t)})$. Further, let $\mathcal{F}_{\mathbf{x}^*}$ be the event that $\mathrm{sign}(\mathbf{x}^*) = \mathrm{sign}(\widehat{\mathbf{x}})$, and let $\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}$ denote the indicator function corresponding to this event. As we show in Lemma 2.2, this event occurs with probability at least $(1-\delta_\beta^{(t)} - \delta_\mathcal{T}^{(t)})$. Using this, we can write the expected gradient vector corresponding to the j-th sample as $\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}$

$$\begin{aligned} \mathbf{g}_j^{(t)} &= \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\mathrm{sign}(\mathbf{x}_j^*)\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}] + \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\mathrm{sign}(\mathbf{x}_j^*)\mathbb{1}_{\overline{\mathcal{F}}_{\mathbf{x}^*}}], \\ &= \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\mathrm{sign}(\mathbf{x}_j^*)\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}] \pm \gamma. \end{aligned}$$

Here, $\gamma := \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\text{sign}(\mathbf{x}_j^*)\mathbb{1}_{\overline{\mathcal{F}}_{\mathbf{x}^*}}]$ is small and depends on $\delta_{\mathcal{T}}^{(t)}$ and $\delta_{\beta}^{(t)}$, which in turn drops with ϵ_t. Therefore, γ diminishes with ϵ_t. Further, since $\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}} + \mathbb{1}_{\overline{\mathcal{F}}_{\mathbf{x}^*}} = 1$, and $\mathbf{Pr}[\mathcal{F}_{\mathbf{x}^*}] = (1 - \delta_{\beta}^{(t)} - \delta_{\mathcal{T}}^{(t)})$, is very large,

$$\begin{aligned}\mathbf{g}_j^{(t)} &= \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\text{sign}(\mathbf{x}_j^*)(1 - \mathbb{1}_{\overline{\mathcal{F}}_{\mathbf{x}^*}})] \pm \gamma,\\ &= \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\text{sign}(\mathbf{x}_j^*)] \pm \gamma.\end{aligned}$$

Therefore, we can write $\mathbf{g}_j^{(t)}$ as

$$\begin{aligned}\mathbf{g}_j^{(t)} &= \mathbf{E}[(\mathbf{A}^{(t)}\widehat{\mathbf{x}} - \mathbf{y})\text{sign}(\mathbf{x}_j^*)] \pm \gamma,\\ &= \mathbf{E}[(1-\eta_x)^R \mathbf{A}_S^{(t)}\mathbf{x}_S^{(0)} + \mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})\mathbf{x}_S^* + \mathbf{A}_S^{(t)}\vartheta_S^{(R)} - \mathbf{A}_S^* x_S^*)\text{sign}(\mathbf{x}_j^*)] \pm \gamma.\end{aligned}$$

Since $\mathbf{E}[(1-\eta_x)^R \mathbf{A}_S^{(t)}\mathbf{x}_S^{(0)}]$ can be made very small by choice of R, we absorb this term in γ. Therefore,

$$\mathbf{g}_j^{(t)} = \mathbf{E}[\mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})\mathbf{x}_S^* + \mathbf{A}_S^{(t)}\vartheta_S^{(R)} - \mathbf{A}_S^* x_S^*)\text{sign}(\mathbf{x}_j^*)] \pm \gamma.$$

Writing the expectation by sub-conditioning on the support,

$$\begin{aligned}\mathbf{g}_j^{(t)} &= \mathbf{E}_S[\mathbf{E}_{x_S^*}[\mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})\mathbf{x}_S^*\text{sign}(\mathbf{x}_j^*) - \mathbf{A}_S^*\mathbf{x}_S^*\text{sign}(\mathbf{x}_j^*) + \mathbf{A}_S^{(t)}\vartheta_S^{(R)}\text{sign}(\mathbf{x}_j^*)|S]] \pm \gamma,\\ &= \mathbf{E}_S[\mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})\mathbf{E}_{x_S^*}[\mathbf{x}_S^*\text{sign}(\mathbf{x}_j^*)|S] - \mathbf{A}_S^*\mathbf{E}_{x_S^*}[\mathbf{x}_S^*\text{sign}(\mathbf{x}_j^*)|S]] + \mathbf{E}[\mathbf{A}_S^{(t)}\vartheta_S^{(R)}\text{sign}(\mathbf{x}_j^*)] \pm \gamma,\\ &= \mathbf{E}_S[p_j(1 - \lambda_j^{(t)})\mathbf{A}_j^{(t)} - p_j\mathbf{A}_j^*] + \Delta_j^{(t)} \pm \gamma,\end{aligned}$$

where we have used the fact that $\mathbf{E}_{x_S^*}[\text{sign}(\mathbf{x}_j^*)] = 0$ and introduced

$$\Delta_j^{(t)} = \mathbf{E}[\mathbf{A}_S^{(t)}\vartheta_S^{(R)}\text{sign}(\mathbf{x}_j^*)].$$

Next, since $p_j = \mathbf{E}_{x_S^*}[\mathbf{x}_j^*\text{sign}(\mathbf{x}_j^*)|j \in S]$, therefore,

$$\mathbf{g}_j^{(t)} = \mathbf{E}_S[p_j(1 - \lambda_j^{(t)})\mathbf{A}_j^{(t)} - p_j\mathbf{A}_j^*] + \Delta_j^{(t)} \pm \gamma.$$

Further, since $q_j = \mathbf{Pr}[j \in S] = \mathcal{O}(k/m)$,

$$\mathbf{g}_j^{(t)} = q_j p_j\Big((1 - \lambda_j^{(t)})\mathbf{A}_j^{(t)} - \mathbf{A}_j^* + \tfrac{1}{q_j p_j}\Delta_j^{(t)} \pm \gamma\Big).$$

Further, by Claim 7 we have that

$$\|\Delta_j^{(t)}\| = \mathcal{O}(\sqrt{m} q_{i,j} p_j \epsilon_t \|\mathbf{A}^{(t)}\|)].$$

This completes the proof. □

Proof of Lemma 2.6. Let $W = \{j : i \in \mathrm{supp}(\mathbf{x}^*_{(j)})\}$ and then we have that

$$\widehat{\mathbf{g}}_i^{(t)} = \tfrac{|W|}{p} \tfrac{1}{|W|} \textstyle\sum_j (\mathbf{y}_{(j)} - \mathbf{A}^{(t)} \widehat{\mathbf{x}}_{(j)}) \mathrm{sign}(\widehat{\mathbf{x}}_{(j)}(i)),$$

where $\widehat{\mathbf{x}}_{(j)}(i)$ denotes the i-th element of the coefficient estimate corresponding to the (j)-th sample. Here, for $\ell = |W|$ the summation

$$\textstyle\sum_j \tfrac{1}{\ell} (\mathbf{y}_{(j)} - \mathbf{A}^{(t)} \widehat{\mathbf{x}}_{(j)}) \mathrm{sign}(\widehat{\mathbf{x}}_{(j)}(i)),$$

has the same distribution as $\Sigma_{j=1}^{\ell} \mathbf{z}_j$, where each $\mathbf{z}_j$ belongs to a distribution as

$$\mathbf{z} := \tfrac{1}{\ell} (\mathbf{y} - \mathbf{A}^{(t)} \widehat{\mathbf{x}}) \mathrm{sign}(\widehat{\mathbf{x}}_i) | i \in S.$$

Also, $\mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)} \widehat{\mathbf{x}}) \mathrm{sign}(\widehat{\mathbf{x}}_i)] = q_i \mathbf{E}[\mathbf{z}]$, where $q_i = \mathbf{Pr}[\mathbf{x}^*_i \neq 0] = \Theta(\frac{k}{m})$. Therefore, since $p = \widetilde{\Omega}(mk^2)$, we have $\ell = pq_i = \widetilde{\Omega}(k^3)$ non-zero vectors,

$$\|\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}\| = \mathcal{O}(\tfrac{k}{m}) \|\Sigma_{j=1}^{\ell} (\mathbf{z}_j - \mathbf{E}[\mathbf{z}])\|. \tag{2.21}$$

Let $\mathbf{w}_j = \mathbf{z}_j - \mathbf{E}[\mathbf{z}]$, we will now apply the vector Bernstein result shown in Lemma 2.11. For this, we require bounds on two parameters for these – $L := \|\mathbf{w}_j\|$ and $\sigma^2 := \|\Sigma_j \mathbf{E}[\|\mathbf{w}_j\|^2]\|$. Note that, since the quantity of interest is a function of $\mathbf{x}^*_i$, which are sub-Gaussian, they are only bounded *almost surely*. To this end, we will employ Lemma 2.14 (Lemma 45 in (Arora et al., 2015)) to get a handle on the concentration.

Bound on the norm $\|\mathbf{w}\|$: This bound is evaluated in Claim 8, which states that with probability at least $(1 - \delta_\beta^{(t)} - \delta_\mathcal{T}^{(t)} - \delta_{\mathrm{HW}}^{(t)})$,

$$L := \|\mathbf{w}\| = \|\mathbf{z} - \mathbf{E}[\mathbf{z}]\| = \tfrac{2}{\ell} \|(\mathbf{y} - \mathbf{A}^{(t)} \widehat{\mathbf{x}}) \mathrm{sign}(\widehat{\mathbf{x}}_i) | i \in S\| \le \tfrac{2}{\ell} \|(\mathbf{y} - \mathbf{A}^{(t)} \widehat{\mathbf{x}})\| = \widetilde{\mathcal{O}}(\tfrac{kt_\beta}{\ell}).$$

Bound on variance parameter $\mathbf{E}[\|\mathbf{w}\|^2]$: Using Claim 9, we have $\mathbf{E}[\|\mathbf{z}\|^2] = \mathcal{O}(k\epsilon_t^2) +$

$\mathcal{O}(kt_\beta^2)$. Therefore, the bound on the variance parameter σ^2 is given by

$$\sigma^2 := \|\Sigma_j \mathbf{E}[\|\mathbf{w}_j\|^2]\| \leq \|\Sigma_j \mathbf{E}[\|\mathbf{z}_j\|^2]\| \leq \mathcal{O}(\tfrac{k}{\ell}\epsilon_t^2) + \mathcal{O}(\tfrac{kt_\beta^2}{\ell}).$$

From Claim 2 we have that with probability at least $(1-\delta_\beta^{(t)})$, $t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$. Applying vector Bernstein inequality shown in Lemma 2.11 and using Lemma 2.14 (Lemma 45 in (Arora et al., 2015)), choosing $\ell = \widetilde{\Omega}(k^3)$, we conclude

$$\|\textstyle\sum_{j=1}^{\ell} \mathbf{z}_j - \mathbf{E}[\mathbf{z}]\| = \mathcal{O}(L) + \mathcal{O}(\sigma) = o(\epsilon_t),$$

with probability at least $(1-\delta_{\mathbf{g}_i}^{(t)})$, where $\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(k))$. Finally, substituting in (2.21) we have

$$\|\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}\| = \mathcal{O}(\tfrac{k}{m})o(\epsilon_t).$$

with probability at least $(1-\delta_{\mathbf{g}_i}^{(t)} - \delta_\beta^{(t)} - \delta_{\mathcal{T}}^{(t)} - \delta_{\text{HW}}^{(t)})$. □

Proof of Lemma 2.7. Since we only have access to the empirical estimate of the gradient $\widehat{\mathbf{g}}_i^{(t)}$, we will show that this estimate is correlated with $(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*)$. To this end, first from Lemma 2.6 we have that the empirical gradient vector concentrates around its mean, specifically,

$$\|\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}\| \leq o(\tfrac{k}{m}\epsilon_t),$$

with probability at least $(1-\delta_{\mathbf{g}_i}^{(t)} - \delta_\beta^{(t)} - \delta_{\mathcal{T}}^{(t)} - \delta_{\text{HW}}^{(t)})$. From Lemma 2.5, we have the following expression for the expected gradient vector

$$\mathbf{g}_j^{(t)} = p_j q_j(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*) + p_j q_j(-\lambda_j^{(t)}\mathbf{A}_j^{(t)} + \tfrac{1}{p_j q_j}\Delta_j^{(t)} \pm \gamma).$$

Let $\mathbf{g}_j^{(t)} = 4\rho\_(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*) + v$, where $4\rho\_ = p_j q_j$ and v is defined as

$$v = p_j q_j(-\lambda_j^{(t)}\mathbf{A}_j^{(t)} + \tfrac{1}{p_j q_j}\Delta_j^{(t)} \pm \gamma). \tag{2.22}$$

Then, $\widehat{\mathbf{g}}_i^{(t)}$ can be written as

$$\widehat{\mathbf{g}}_i^{(t)} = \widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)} + \mathbf{g}_i^{(t)},$$

$$= (\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}) + 4\rho\_(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*) + v,$$
$$= 4\rho\_(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*) + \widetilde{v}, \tag{2.23}$$

where $\widetilde{v} = v + (\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)})$. Let $\|\widetilde{v}\| \leq \rho\_\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|$. Using the definition of v as shown in (2.22) we have

$$\|\widetilde{v}\| \leq q_j p_j \lambda_j^{(t)} \|\mathbf{A}_j^{(t)}\| + \|\Delta_j^{(t)}\| + o(\tfrac{k}{m}\epsilon_t) \pm \gamma.$$

Now for the first term, since $\|\mathbf{A}_j^{(t)}\| = 1$, we have $\lambda_j^{(t)} = |\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^* \rangle| = \frac{1}{2}\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2$, therefore

$$q_j p_j \lambda_j^{(t)} \|\mathbf{A}_j^{(t)}\| = q_j p_j \tfrac{1}{2}\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2,$$

Further, using Claim 7

$$\|\Delta_j^{(t)}\| = \mathcal{O}(\sqrt{m} q_{i,j} p_{i_1} \epsilon_t \|\mathbf{A}^{(t)}\|).$$

Now, since $\|\mathbf{A}^{(t)} - \mathbf{A}^*\| \leq 2\|\mathbf{A}^*\|$ (the closeness property (Def.2.1) is maintained at every step using Lemma 2.9), and further since $\|\mathbf{A}^*\| = \mathcal{O}(\sqrt{m/n})$, we have that

$$\|\mathbf{A}^{(t)}\| \leq \|\mathbf{A}^{(t)} - \mathbf{A}^*\| + \|\mathbf{A}^*\| = \mathcal{O}(\sqrt{\tfrac{m}{n}}).$$

Therefore, we have

$$\|\Delta_j^{(t)}\| + o(\tfrac{k}{m}\epsilon_t) \pm \gamma = \mathcal{O}(\sqrt{m} q_{i,j} p_{i_1} \epsilon_t \|\mathbf{A}^{(t)}\|).$$

Here, we use the fact that γ drops with decreasing ϵ_t as argued in Lemma 2.5. Next, using (2.23), we have

$$\|\widehat{\mathbf{g}}_j^{(t)}\| \leq 4\rho\_\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\| + \|\widetilde{v}\|.$$

Now, letting

$$\|\Delta_j^{(t)}\| + o(\tfrac{k}{m}\epsilon_t) \pm \gamma = \mathcal{O}(\sqrt{m} q_{i,j} p_{i_1} \epsilon_t \|\mathbf{A}^{(t)}\|) \leq \tfrac{q_i p_i}{2}\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|, \tag{2.24}$$

we have that, for $k = \mathcal{O}(\sqrt{n})$

$$\|\widetilde{v}\| \leq q_i p_i \|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|.$$

Substituting for $\|\widetilde{v}\|$, this implies that $\|\widehat{\mathbf{g}^{(t)}}_j\|^2 \leq 25\rho_-^2\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2$. Further, we also have the following lower-bound

$$\langle \widehat{\mathbf{g}}_j^{(t)}, \mathbf{A}_j^{(t)} - \mathbf{A}_j^* \rangle \geq 4\rho_- \|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2 - \|\widetilde{v}\|\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|.$$

Here, we use the fact that R.H.S. can be minimized only if $\widetilde{v}$ is directed opposite to the direction of $\mathbf{A}_j^{(t)} - \mathbf{A}_j^*$. Now, we show that this gradient is $(\rho_-, 1/100\rho_-, 0)$ correlated,

$$\begin{aligned}
&\langle \widehat{\mathbf{g}}_i^{(t)}, \mathbf{A}_i^{(t)} {-} \mathbf{A}_i^* \rangle - \rho_- \|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 - \tfrac{1}{100\rho_-}\|\widehat{\mathbf{g}}_i^{(t)}\|^2, \\
&\qquad \geq 4\rho_- \|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 - \|\widetilde{v}\|\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| - \rho_- \|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 - \tfrac{1}{100\rho_-}\|\widehat{\mathbf{g}}_i^{(t)}\|^2, \\
&\qquad \geq 4\rho_- \|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 - 2\rho_- \|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 - \tfrac{25\rho_-^2\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2}{100\rho_-}, \\
&\qquad \geq \rho_- \|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 \geq 0.
\end{aligned}$$

Therefore, for this choice of k, i.e. $k = \mathcal{O}(\sqrt{n})$, there is no bias in dictionary estimation in comparison to Arora et al. (2015). This gain can be attributed to estimating the coefficients simultaneously with the dictionary. Further, since we choose $4\rho_- = p_j q_j$, we have that $\rho_- = \Theta(k/m)$, as a result $\rho_+ = 1/100\rho_- = \Omega(m/k)$. Applying Lemma 2.15 we have

$$\|\mathbf{A}_j^{(t+1)} - \mathbf{A}_j^*\|^2 \leq (1 - \rho_- \eta_A)\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2,$$

for $\eta_A = \mathcal{O}(m/k)$ with probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_{\beta}^{(t)} - \delta_{\mathbf{g}_i}^{(t)})$. □

Proof of Lemma 2.8. Here, we will prove that $\widehat{\mathbf{g}}^{(t)}$ defined as

$$\widehat{\mathbf{g}}^{(t)} = \textstyle\sum_j (\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top,$$

concentrates around its mean. Notice that each summand $(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top$ is a random matrix of the form $(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top$. Also, we have $\mathbf{g}^{(t)}$ defined as

$$\mathbf{g}^{(t)} = \mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top].$$

To bound $\|\widehat{\mathbf{g}}^{(t)} - \mathbf{g}^{(t)}\|$, we are interested in $\|\sum_{j=1}^{p} \mathbf{W}_j\|$, where each matrix $\mathbf{W}_j$ is given by

$$\mathbf{W}_j = \tfrac{1}{p}(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top - \tfrac{1}{p}\mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top].$$

Noting that $\mathbf{E}[\mathbf{W}_j] = 0$, we will employ the matrix Bernstein result (Lemma 2.10) to bound $\|\widehat{\mathbf{g}}^{(t)} - \mathbf{g}^{(t)}\|$. To this end, we will bound $\|\mathbf{W}_j\|$ and the variance proxy

$$v(\mathbf{W}_j) = \max\{\|\textstyle\sum_{j=1}^{p} \mathbf{E}[\mathbf{W}_j\mathbf{W}_j^\top]\|, \|\textstyle\sum_{j=1}^{p} \mathbf{E}[\mathbf{W}_j^\top\mathbf{W}_j]\|\}.$$

Bound on $\|\mathbf{W}_j\|$– First, we can bound both terms in the expression for $\mathbf{W}_j$ by triangle inequality as

$$\begin{aligned}\|\mathbf{W}_j\| &\le \tfrac{1}{p}\|(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top\| + \tfrac{1}{p}\|\mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top\|,\\ &\le \tfrac{2}{p}\|(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top\|.\end{aligned}$$

Here, we use Jensen's inequality for the second term, followed by upper-bounding the expected value of the argument by $\|(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top\|$.

Next, using Claim 8 we have that with probability at least $(1 - \delta_\beta^{(t)} - \delta_{\mathcal{T}}^{(t)} - \delta_{\mathrm{HW}}^{(t)})$, $\|\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}\|$ is $\widetilde{\mathcal{O}}(kt_\beta)$, and the fact that $\|\mathrm{sign}(x)^T\| = \sqrt{k}$,

$$\|\mathbf{W}_j\| \le \tfrac{2}{p}\sqrt{k}\|(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\| = \mathcal{O}(\tfrac{k\sqrt{k}}{p}t_\beta).$$

Bound on the variance statistic $v(\mathbf{W}_j)$– For the variance statistic, we first look at $\|\sum \mathbf{E}[\mathbf{W}_j\mathbf{W}_j^\top]\|$,

$$\begin{aligned}\mathbf{E}[\mathbf{W}_j\mathbf{W}_j^\top] = \tfrac{1}{p^2}\mathbf{E}[(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top - \mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top]\\ \times [\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})^\top - (\mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top)^\top].\end{aligned}$$

Since $\mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top]\mathbf{E}[(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}})^\top]^\top$ is positive semidefinite,

$$\mathbf{E}[\mathbf{W}_j\mathbf{W}_j^\top] \le \tfrac{1}{p^2}\mathbf{E}[(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})(\mathbf{y}_{(j)} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})^\top].$$

Now, since each $\widehat{\mathbf{x}}_{(j)}$ has k non-zeros, $\mathrm{sign}(\widehat{\mathbf{x}}_{(j)})^\top\mathrm{sign}(\widehat{\mathbf{x}}_{(j)}) = k$, and using Claim 10, with probability at least $(1-\delta_{\mathcal{T}}^{(t)}-\delta_{\beta}^{(t)})$

$$\begin{aligned}\|\textstyle\sum \mathbf{E}[\mathbf{W}_j\mathbf{W}_j^\top]\| &\le \tfrac{k}{p}\|\mathbf{E}[(\mathbf{y}_{(j)}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})(\mathbf{y}_{(j)}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})^\top]\|,\\ &= \mathcal{O}(\tfrac{k^3t_\beta^2}{pm})\|\mathbf{A}^*\|^2.\end{aligned}$$

Similarly, expanding $\mathbf{E}[\mathbf{W}_j^\top\mathbf{W}_j]$, and using the fact that $\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})\text{sign}(\widehat{\mathbf{x}})^\top]^\top\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})\text{sign}(\widehat{\mathbf{x}})^\top]$ is positive semi-definite. Now, using Claim 8 and the fact that entries of $\mathbf{E}[(\text{sign}(\widehat{\mathbf{x}}_{(j)})\text{sign}(\widehat{\mathbf{x}}_{(j)})^\top]$ are q_i on the diagonal and zero elsewhere, where $q_i=\mathcal{O}(k/m)$,

$$\begin{aligned}\|\textstyle\sum \mathbf{E}[\mathbf{W}_j^\top\mathbf{W}_j]\| &\le \tfrac{1}{p}\|\mathbf{E}[(\text{sign}(\widehat{\mathbf{x}}_{(j)})(\mathbf{y}_{(j)}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})^\top(\mathbf{y}_{(j)}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)})\text{sign}(x_{(j)}^{(R)})^\top]\|,\\ &\le \tfrac{1}{p}\|\mathbf{E}[(\text{sign}(\widehat{\mathbf{x}}_{(j)})\text{sign}(\widehat{\mathbf{x}}_{(j)})^\top]\|\|\mathbf{y}_{(j)}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}_{(j)}\|^2,\\ &\le \mathcal{O}(\tfrac{k}{mp})\widetilde{\mathcal{O}}(k^2t_\beta^2)=\widetilde{\mathcal{O}}(\tfrac{k^3t_\beta^2}{mp}).\end{aligned}$$

Now, we are ready to apply the matrix Bernstein result. Since, $m=O(n)$ the variance statistic comes out to be $\mathcal{O}(\frac{k^3t_\beta^2}{pm})\|\mathbf{A}^*\|^2$, then as long as we choose $p=\widetilde{\Omega}(mk^2)$ (using the bound on t_β), with probability at least $(1-\delta_\beta^{(t)}-\delta_{\mathcal{T}}^{(t)}-\delta_{\text{HW}}^{(t)}-\delta_{\mathbf{g}}^{(t)})$

$$\begin{aligned}\|\widehat{\mathbf{g}}^{(t)}-\mathbf{g}^{(t)}\| &\le \mathcal{O}(\tfrac{k\sqrt{k}}{p}t_\beta)+\|\mathbf{A}^*\|\sqrt{\mathcal{O}(\tfrac{k^3t_\beta^2}{pm})},\\ &= \mathcal{O}^*(\tfrac{k}{m}\|\mathbf{A}^*\|).\end{aligned}$$

where $\delta_{\mathbf{g}}^{(t)}=(n+m)\exp(-\Omega(m\sqrt{\log(n)})$. □

Proof of Lemma 2.9. This lemma ensures that the dictionary iterates maintain the closeness property (Def.2.1) and satisfies the prerequisites for Lemma 2.7.

The update step for the i-th dictionary element at the $s+1$ iteration can be written as

$$\begin{aligned}\mathbf{A}_i^{(t+1)}-\mathbf{A}_i^* &= \mathbf{A}_i^{(t)}-\mathbf{A}_i^*-\eta_A\widehat{\mathbf{g}}_i^{(t)},\\ &= \mathbf{A}_i^{(t)}-\mathbf{A}_i^*-\eta_A\mathbf{g}_i^{(t)}-\eta_A(\widehat{\mathbf{g}}_i^{(t)}-\mathbf{g}_i^{(t)}).\end{aligned}$$

Here, $\mathbf{g}_i^{(t)}$ is given by the following as per Lemma 2.5 with probability at least $(1-\delta_{\mathcal{T}}^{(t)}-$

$\delta_\beta^{(t)})$

$$\mathbf{g}_i^{(t)} = q_i p_i(\mathbf{A}_i^{(t)} - \mathbf{A}_i^*) + q_i p_i(-\lambda_i^{(t)}\mathbf{A}_i^{(t)} + \tfrac{1}{q_i p_i}\Delta_i^{(t)} \pm \gamma).$$

Substituting the expression for $\mathbf{g}_i^{(t)}$ in the dictionary update step,

$$\mathbf{A}_i^{(t+1)} - \mathbf{A}_i^* = (1 - \eta_A p_i q_i)(\mathbf{A}_i^{(t)} - \mathbf{A}_i^*) - \eta_A p_i q_i \lambda_i^{(t)}\mathbf{A}_i^{(t)} - \eta_A \Delta_i^{(t)} - \eta_A(\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}) \pm \gamma,$$

where $\Delta_j^{(t)} = \mathbf{E}[\mathbf{A}^{(t)}\vartheta^{(R)}\mathrm{sign}(\mathbf{x}_j^*)]_j$. Therefore, the update step for the dictionary (matrix) can be written as

$$\mathbf{A}^{(t+1)} - \mathbf{A}^* = (\mathbf{A}^{(t)} - \mathbf{A}^*)\mathrm{diag}((1 - \eta_A p_i q_i)) + \eta_A \mathbf{U} - \eta_A \mathbf{V} \pm \gamma - \eta_A(\widehat{\mathbf{g}}^{(t)} - \mathbf{g}^{(t)}), \qquad (2.25)$$

where, $\mathbf{U} = \mathbf{A}^{(t)}\mathrm{diag}(p_i q_i \lambda_i^{(t)})$ and $\mathbf{V} = \mathbf{A}^{(t)}\mathbf{Q}$, with the matrix $\mathbf{Q}$ given by,

$$\mathbf{Q}_{i,j} = q_{i,j}\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_i^{(R)}\mathrm{sign}(\mathbf{x}_j^*)|S],$$

and using the following intermediate result shown in Claim 7,

$$\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_i^{(R)}\mathrm{sign}(\mathbf{x}_j^*)|S]\begin{cases} \le \gamma_i^{(R)}, & \text{for } i = j, \\ = \mathcal{O}(p_j \epsilon_t), & \text{for } i \ne j, \end{cases}$$

we have $\|\mathbf{Q}_i\| = \mathcal{O}(\sqrt{m} q_{i,j} p_i \epsilon_t)$. Hence, we have

$$\|\mathbf{Q}\|_F \le \mathcal{O}(m q_{i,j} p_i \epsilon_t).$$

Therefore,

$$\|\mathbf{V}\| \le \|\mathbf{A}^{(t)}\mathbf{Q}\| \le \|\mathbf{A}^{(t)}\|\|\mathbf{Q}\|_F = \mathcal{O}(m q_{i,j} p_i \epsilon_t \|\mathbf{A}^*\|) = \mathcal{O}(\tfrac{k^2}{m\log(n)})\|\mathbf{A}^*\|.$$

We will now proceed to bound each term in (2.25). Starting with $(\mathbf{A}^{(t)}{-}\mathbf{A}^*)\mathrm{diag}(1 - \eta_A p_i q_i)$, and using the fact that $p_i = O(1)$, $q_i = O(k/m)$, and $\|\mathbf{A}^{(t)} - \mathbf{A}^*\| \le 2\|\mathbf{A}^*\|$, we have

$$\|(\mathbf{A}^{(t)} - \mathbf{A}^*)\mathrm{diag}(1 - \eta_A p_i q_i)\| \le (1 - \min_i \eta_A p_i q_i)\|(\mathbf{A}^{(t)} - \mathbf{A}^*)\| \le 2(1 - \Omega(\eta_A k/m))\|\mathbf{A}^*\|.$$

Next, since $\|\mathbf{A}_j^{(t)}\| = 1$, we have $\lambda_j^{(t)} = |\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^* \rangle| = \frac{1}{2}\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2$, and $\lambda_i^{(t)} \le \epsilon_t^2/2$, therefore

$$\|\mathbf{U}\| = \|\mathbf{A}^{(t)}\mathrm{diag}(p_iq_i\lambda_i^{(t)})\| \leq \max_i\ p_iq_i\frac{\epsilon_t^2}{2}\|\mathbf{A}^{(t)} - \mathbf{A}^* + \mathbf{A}^*\| \leq o(k/m)\|\mathbf{A}^*\|.$$

Using the results derived above, and the the result derived in Lemma 2.8 which states that with probability at least $(1-\delta_{\beta}^{(t)} - \delta_{\mathcal{T}}^{(t)} - \delta_{\mathrm{HW}}^{(t)} - \delta_{\mathbf{g}}^{(t)})$, $\|\widehat{\mathbf{g}}^{(t)} - \mathbf{g}^{(t)}\| = \mathcal{O}^*(\frac{k}{m}\|\mathbf{A}^*\|))$ we have

$$\begin{aligned}\|\mathbf{A}^{(t+1)} - \mathbf{A}^*\| &= \|(\mathbf{A}^{(t)} - \mathbf{A}^*)\mathbf{D}_{(1-\eta_A p_i q_i)}\| + \eta_A\|\mathbf{U}\| + \eta_A\|\mathbf{V}\| + \eta_A\|(\widehat{\mathbf{g}}^{(t)} - \mathbf{g}^{(t)})\| \pm \gamma, \\ &\leq 2(1 - \Omega(\eta_A\tfrac{k}{m})\|\mathbf{A}^*\| + o(\eta_A\tfrac{k}{m})\|\mathbf{A}^*\| + \mathcal{O}(\eta_A\tfrac{k^2}{m\log(n)})\|\mathbf{A}^*\| + o(\eta_A\tfrac{k}{m}\|\mathbf{A}^*\|) \pm \gamma, \\ &\leq 2\|\mathbf{A}^*\|.\end{aligned}$$

□

2.D Appendix: Proofs of intermediate results

Claim 1 (**Incoherence of $\mathbf{A}^{(t)}$**). *If $\mathbf{A}^* \in \mathbb{R}^{n\times m}$ is μ-incoherent and $\|\mathbf{A}_i^* - \mathbf{A}_i^{(t)}\| \leq \epsilon_t$ holds for each $i \in [1 \ldots m]$, then $\mathbf{A}^{(t)} \in \mathbb{R}^{n\times m}$ is μ_t-incoherent, where $\mu_t = \mu + 2\sqrt{n}\epsilon_t$.*

Proof of Claim 1. We start by looking at the incoherence between the columns of $\mathbf{A}^*$, for $j \neq i$,

$$\begin{aligned}\langle \mathbf{A}_i^*, \mathbf{A}_j^* \rangle &= \langle \mathbf{A}_i^* - \mathbf{A}_i^{(t)}, \mathbf{A}_j^* \rangle + \langle \mathbf{A}_i^{(t)}, \mathbf{A}_j^* \rangle, \\ &= \langle \mathbf{A}_i^* - \mathbf{A}_i^{(t)}, \mathbf{A}_j^* \rangle + \langle \mathbf{A}_i^{(t)}, \mathbf{A}_j^* - \mathbf{A}_j^{(t)} \rangle + \langle \mathbf{A}_i^{(t)}, \mathbf{A}_j^{(t)} \rangle.\end{aligned}$$

Since $\langle \mathbf{A}_i^*, \mathbf{A}_j^* \rangle \leq \frac{\mu}{\sqrt{n}}$,

$$\begin{aligned}|\langle \mathbf{A}_i^{(t)}, \mathbf{A}_j^{(t)} \rangle| &\leq \langle \mathbf{A}_i^*, \mathbf{A}_j^* \rangle - \langle \mathbf{A}_i^* - \mathbf{A}_i^{(t)}, \mathbf{A}_j^* \rangle - \langle \mathbf{A}_i^{(t)}, \mathbf{A}_j^* - \mathbf{A}_j^{(t)} \rangle, \\ &\leq \frac{\mu}{\sqrt{n}} + 2\epsilon_t.\end{aligned}$$

□

Claim 2 (**Bound on $\beta_j^{(t)}$: the noise component in coefficient estimate that depends on ϵ_t**). *With probability $(1 - \delta_\beta^{(t)})$, $|\beta_j^{(t)}|$ is upper-bounded by $t_\beta = \mathcal{O}(\sqrt{k}\epsilon_t)$, where $\delta_\beta^{(t)} = 2k\ \exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.*

Proof of Claim 2. We have the following definition for $\beta_j^{(t)}$ from (2.11),

$$\beta_j^{(t)} = \sum_{i\neq j} (\langle \mathbf{A}_j^*, \mathbf{A}_i^* - \mathbf{A}_i^{(t)} \rangle + \langle \mathbf{A}_j^* - \mathbf{A}_j^{(t)}, \mathbf{A}_i^{(t)} \rangle + \langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^* \rangle)\mathbf{x}_i^*.$$

Here, since $\mathbf{x}_i^*$ are independent sub-Gaussian random variables, $\beta_j^{(t)}$ is a sub-Gaussian random variable with the variance parameter evaluated as shown below

$$var[\beta_j^{(t)}] = \textstyle\sum_{i\neq j} (\langle \mathbf{A}_j^*, \mathbf{A}_i^{(t)} - \mathbf{A}_i^* \rangle + \langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^{(t)} \rangle + \langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_i^* \rangle)^2 \leq 9k\epsilon_t^2.$$

Therefore, by Lemma 2.12

$$\mathbf{Pr}[|\beta_j^{(t)}| > t_\beta] \leq 2\exp(-\frac{t_\beta^2}{18k\epsilon_t^2}).$$

Now, we need this for each $\beta_j^{(t)}$ for $j \in \text{supp}(\mathbf{x}^*)$, union bounding over k coefficients

$$\mathbf{Pr}[\max|\beta_j^{(t)}| > t_\beta] \leq \delta_\beta^{(t)},$$

where $\delta_\beta^{(t)} = 2k\,\exp(-\frac{t_\beta^2}{18k\epsilon_t^2})$. Choosing $t_\beta = \mathcal{O}(\sqrt{k\epsilon_t})$, we have that $\delta_\beta^{(t)} = 2k\,\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$. □

Claim 3 (**Error in coefficient estimation for a general iterate** $(r+1)$)**.** *The error in a general iterate r of the coefficient estimation is upper-bounded as*

$$C_{i_1}^{(r+1)} \leq \alpha_{i_1}^{(r+1)} + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r} \alpha_{\max}^{(\ell)}\Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{r-\ell}$$
$$+ (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)}\Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{r}.$$

Proof of Claim 3 . From (2.17) we have the following expression for $C_{i_1}^{(r+1)}$

$$C_{i_1}^{(r+1)} \leq \alpha_{i_1}^{(r+1)} + \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r+1} \eta_x^{(\ell)} \sum_{i_2 \neq i_1} C_{i_2}^{(\ell-1)} \prod_{q=\ell}^{r+1} (1 - \eta_x^{(q+1)}).$$

Our aim will be to recursively substitute for $C_{i_1}^{(\ell-1)}$ to develop an expression for $C_{i_1}^{(r+1)}$ as a function of $C_{\max}^{0}$. To this end, we start by analyzing the iterates $C_{i_1}^{(1)}$, $C_{i_1}^{(2)}$, and so on to develop an expression for $C_{i_1}^{(r+1)}$ as follows.

Expression for $C_{i_1}^{(1)}$ – Consider $C_{i_1}^{(1)}$

$$\begin{aligned} C_{i_1}^{(1)} &\leq \alpha_{i_1}^{(1)} + \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{1} \eta_x \sum_{i_2 \neq i_1} C_{i_2}^{(\ell-1)} \prod_{q=\ell}^{1} (1 - \eta_x), \\ &= \alpha_{i_1}^{(1)} + \eta_x\Big(\tfrac{\mu_t}{\sqrt{n}} \sum_{i_1 \neq i_2} C_{i_2}^{(0)}\Big). \end{aligned} \tag{2.26}$$

Expression for $C_{i_1}^{(2)}$– Next, $C_{i_1}^{(2)}$ is given by

$$\begin{aligned} C_{i_1}^{(2)} &\leq \alpha_{i_1}^{(2)} + \eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{2} \sum_{i_2 \neq i_1} C_{i_2}^{(\ell-1)} \prod_{q=\ell}^{2} (1 - \eta_x), \\ &\leq \alpha_{i_1}^{(2)} + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big(\sum_{i_2 \neq i_1} C_{i_2}^{(1)} + \sum_{i_2 \neq i_1} C_{i_2}^{(0)}(1 - \eta_x)\Big). \end{aligned}$$

Further, we know from (2.26) we have

$$C^{(1)}_{i_2} = \alpha^{(1)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_3 \neq i_2} C^{(0)}_{i_3}.$$

Therefore, since $\sum\limits_{i_2 \neq i_1} \sum\limits_{i_3 \neq i_2} = \sum\limits_{i_3 \neq i_2, i_1}$,

$$\begin{aligned} C^{(2)}_{i_1} &\leq \alpha^{(2)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \Big(\sum_{i_2 \neq i_1} \Big(\alpha^{(1)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_3 \neq i_2} C^{(0)}_{i_3} \Big) + \sum_{i_2 \neq i_1} C^{(0)}_{i_2} (1 - \eta_x) \Big), \\ &= \alpha^{(2)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_2 \neq i_1} \alpha^{(1)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \Big(\eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_3 \neq i_2, i_1} C^{(0)}_{i_3} + \sum_{i_2 \neq i_1} C^{(0)}_{i_2} (1 - \eta_x) \Big). \end{aligned} \tag{2.27}$$

Expression for $C^{(3)}_{i_1}$– Next, we writing $C^{(3)}_{i_1}$,

$$\begin{aligned} C^{(3)}_{i_1} &\leq \alpha^{(3)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \textstyle\sum_{\ell=1}^{3} \displaystyle\sum_{i_2 \neq i_1} C^{(\ell-1)}_{i_2} (1 - \eta_x)^{3-\ell}, \\ &= \alpha^{(3)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_2 \neq i_1} \Big(C^{(0)}_{i_2} (1 - \eta_x)^2 + C^{(1)}_{i_2} (1 - \eta_x) + C^{(2)}_{i_2} \Big), \\ &\leq \alpha^{(3)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_2 \neq i_1} \Big(C^{(0)}_{i_2} (1 - \eta_x)^2 + \Big(\alpha^{(1)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_3 \neq i_2} C^{(0)}_{i_3} \Big) (1 - \eta_x) + C^{(2)}_{i_2} \Big). \end{aligned}$$

Here, using (2.27) we have the following expression for $C^{(2)}_{i_2}$

$$C^{(2)}_{i_2} \leq \alpha^{(2)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_3 \neq i_2} \alpha^{(1)}_{i_3} + \eta_x \frac{\mu_t}{\sqrt{n}} \Big(\eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_4 \neq i_3, i_2} C^{(0)}_{i_4} + \sum_{i_3 \neq i_2} C^{(0)}_{i_3} (1 - \eta_x) \Big).$$

Substituting for $C^{(2)}_{i_2}$ in the expression for $C^{(3)}_{i_1}$, and rearranging the terms in the expression for $C^{(3)}_{i_1}$, we have

$$\begin{aligned} C^{(3)}_{i_1} \leq \alpha^{(3)}_{i_1} &+ \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_2 \neq i_1} \alpha^{(2)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \Big((1 - \eta_x) \sum_{i_2 \neq i_1} \alpha^{(1)}_{i_2} + \eta_x \frac{\mu_t}{\sqrt{n}} \sum_{i_3 \neq i_2, i_1} \alpha^{(1)}_{i_3} \Big) \\ &+ \eta_x \frac{\mu_t}{\sqrt{n}} \Big((1 - \eta_x)^2 \sum_{i_2 \neq i_1} C^{(0)}_{i_2} + 2(1 - \eta_x)(\eta_x \frac{\mu_t}{\sqrt{n}}) \sum_{i_3 \neq i_2, i_1} C^{(0)}_{i_3} + (\eta_x \frac{\mu_t}{\sqrt{n}})^2 \sum_{i_4 \neq i_3, i_2, i_1} C^{(0)}_{i_4} \Big). \end{aligned} \tag{2.28}$$

Expression for $C^{(4)}_{i_1}$– Now, consider $C^{(4)}_{i_1}$

$$\begin{aligned} &C^{(4)}_{i_1} \leq \alpha^{(4)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \textstyle\sum_{\ell=1}^{4} \sum_{i_2 \neq i_1} C^{(\ell-1)}_{i_2} (1 - \eta_x)^{4-\ell}, \\ &\leq \alpha^{(4)}_{i_1} + \eta_x \frac{\mu_t}{\sqrt{n}} \Big(\textstyle\sum_{i_2 \neq i_1} C^{(0)}_{i_2} (1 - \eta_x)^3 + \sum_{i_2 \neq i_1} C^{(1)}_{i_2} (1 - \eta_x)^2 + \sum_{i_2 \neq i_1} C^{(2)}_{i_2} (1 - \eta_x)^1 \end{aligned}$$

$$+ \textstyle\sum_{i_2 \neq i_1} C_{i_2}^{(3)} (1-\eta_x)^0 \Big).$$

Substituting for $C_{i_2}^{(3)}$ from (2.28), $C_{i_2}^{(2)}$ from (2.27), $C_{i_2}^{(1)}$ using (2.26), and rearranging,

$$\begin{aligned}
C_{i_1}^{(4)} \leq \alpha_{i_1}^{(4)} + \eta_x \tfrac{\mu_t}{\sqrt{n}} \Bigg[& \sum_{i_2 \neq i_1} \alpha_{i_2}^{(3)} + \Bigg((1-\eta_x)^1 \textstyle\sum_{i_2 \neq i_1} \alpha_{i_2}^{(2)} + \eta_x \tfrac{\mu_t}{\sqrt{n}} \displaystyle\sum_{i_3 \neq i_2, i_1} \alpha_{i_3}^{(2)} \Bigg) \\
& + \Big(\sum_{i_2 \neq i_1} \alpha_{i_2}^{(1)} (1-\eta_x)^2 + 2\eta_x \tfrac{\mu_t}{\sqrt{n}} (1-\eta_x) \sum_{i_3 \neq i_2, i_1} \alpha_{i_3}^{(1)} + (\eta_x \tfrac{\mu_t}{\sqrt{n}})^2 \sum_{i_4 \neq i_3, i_2, i_1} \alpha_{i_4}^{(1)} \Big) \Bigg] \\
+ \eta_x \tfrac{\mu_t}{\sqrt{n}} \Bigg[& \sum_{i_2 \neq i_1} C_{i_2}^{(0)} (1-\eta_x)^3 + 3\eta_x \tfrac{\mu_t}{\sqrt{n}} (1-\eta_x)^2 \sum_{i_3 \neq i_2, i_1} C_{i_3}^{(0)} \\
& + 3(\eta_x \tfrac{\mu_t}{\sqrt{n}})^2 (1-\eta_x)^1 \sum_{i_4 \neq i_3, i_2, i_1} C_{i_4}^{(0)} + (\eta_x \tfrac{\mu_t}{\sqrt{n}})^3 \sum_{i_5 \neq i_4, i_3, i_2, i_1} C_{i_5}^{(0)} \Bigg].
\end{aligned}$$

Notice that the terms have a binomial series like form. To reveal this structure, let each $\alpha_j^{(\ell)} \leq \alpha_{\max}^{(\ell)}$ where $j = i_1, i_2, \ldots, i_k$. Similarly, let $C_j^{(0)} \leq C_{\max}^{(0)}$ for $j = i_1, i_2, \ldots, i_k$. Therefore, we have

$$\begin{aligned}
C_{i_1}^{(4)} \leq \alpha_{i_1}^{(4)} + \eta_x \tfrac{\mu_t}{\sqrt{n}} \Bigg[& (k-1)\alpha_i^{(3)} + \alpha_i^{(2)} \Bigg((1-\eta_x)^1 (k-1) + \eta_x \tfrac{\mu_t}{\sqrt{n}} (k-2) \Bigg) \\
& + \alpha_i^{(1)} \Bigg((k-1)(1-\eta_x)^2 + 2(k-2)\eta_x \tfrac{\mu_t}{\sqrt{n}} (1-\eta_x) + (k-3)(\eta_x \tfrac{\mu_t}{\sqrt{n}})^2 \Bigg) \Bigg] \\
& + \eta_x \tfrac{\mu_t}{\sqrt{n}} C_i^{(0)} \Bigg[(k-1)(1-\eta_x)^3 + 3(k-2)\eta_x \tfrac{\mu_t}{\sqrt{n}} (1-\eta_x)^2 \\
& \qquad + 3(k-3)(\eta_x \tfrac{\mu_t}{\sqrt{n}})^2 (1-\eta_x)^1 + (k-4)(\eta_x \tfrac{\mu_t}{\sqrt{n}})^3 \Bigg].
\end{aligned}$$

Further upper-bounding the expression, we have

$$\begin{aligned}
C_{i_1}^{(4)} \leq \alpha_{i_1}^{(4)} + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} \Bigg[& \alpha_i^{(3)} + \alpha_i^{(2)} \Bigg((1-\eta_x) + \eta_x \tfrac{\mu_t}{\sqrt{n}} \Bigg) \\
& + \alpha_i^{(1)} \Bigg((1-\eta_x)^2 + 2\eta_x \tfrac{\mu_t}{\sqrt{n}} (1-\eta_x) + (\eta_x \tfrac{\mu_t}{\sqrt{n}})^2 \Bigg) \Bigg] \\
+ (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} C_i^{(0)} \Bigg[& (1-\eta_x)^3 + 3\eta_x \tfrac{\mu_t}{\sqrt{n}} (1-\eta_x)^2 + 3(\eta_x \tfrac{\mu_t}{\sqrt{n}})^2 (1-\eta_x) + (\eta_x \tfrac{\mu_t}{\sqrt{n}})^3 \Bigg].
\end{aligned}$$

Therefore,

$$C_{i_1}^{(4)} \leq \alpha_{i_1}^{(4)} + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} \Bigg[\alpha_i^{(3)} + \alpha_i^{(2)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}} \Big)^1 + \alpha_i^{(1)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}} \Big)^2 \Bigg]$$

$$+ (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} C_i^{(0)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^3. \qquad (2.29)$$

Expression for $C_{i_1}^{(r+1)}$– With this, we are ready to write the general term,

$$C_{i_1}^{(r+1)} \leq \alpha_{i_1}^{(r+1)} + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r} \alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{r-\ell}$$
$$+ (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^r.$$

□

Claim 4 (**An intermediate result for bounding the error in coefficient calculations**). *With probability* $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_\beta^{(t)})$,

$$\sum_{\ell=1}^{R} \alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell} \leq C_i^{(0)} R \delta_R + \tfrac{1}{\eta_x (1 - \frac{\mu_t}{\sqrt{n}})} (\tfrac{\epsilon_t^2}{2} |\mathbf{x}_{\max}^*| + t_\beta).$$

Proof of Claim 4 . Using (2.18), the quantity $\alpha_i^{(\ell)}$ is defined as

$$\alpha_i^{(\ell)} = C_i^{(0)} (1 - \eta_x)^\ell + (\lambda_i^{(t)} |\mathbf{x}_i^*| + \beta_i^{(t)}) \sum_{s=1}^{\ell} \eta_x (1 - \eta_x)^{\ell - s + 1}.$$

Therefore, we are interested in

$$\sum_{\ell=1}^{R} C_i^{(0)} (1 - \eta_x)^\ell \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell}$$
$$+ (\lambda_i^{(t)} |\mathbf{x}_i^*| + \beta_i^{(t)}) \sum_{\ell=1}^{R} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell} \sum_{s=1}^{\ell} \eta_x (1 - \eta_x)^{\ell - s + 1}.$$

Consider the first term which depends on $C_i^{(0)}$. Since $(1 - \eta_x) \leq (1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}})$, we have

$$C_i^{(0)} \sum_{\ell=1}^{R} (1 - \eta_x)^\ell \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell} \leq C_i^{(0)} R \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^R \leq C_i^{(0)} R \delta_R,$$

where δ_R is a small constant, and a parameter which determines the number of iterations R required for the coefficient update step. Now, coming back to the quantity of interest

$$\sum_{\ell=1}^{R} \alpha_i^{(\ell)} \Big(1 - \eta_x + \eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell} \leq C_i^{(0)} R \delta_R$$

$$+\,(\lambda_i^{(t)}|\mathbf{x}_i^*|+\beta_i^{(t)})\sum_{\ell=1}^{R}\Big(1-\eta_x+\eta_x\tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell}\sum_{s=1}^{\ell}\eta_x(1-\eta_x)^{\ell-s+1}.$$

Now, using sum of geometric series result, we have that $\sum\limits_{s=1}^{\ell}\eta_x(1-\eta_x)^{\ell-s+1}$, and

$$\sum_{\ell=1}^{R}\Big(1-\eta_x+\eta_x\tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell}=\frac{1-\Big(1-\eta_x+\eta_x\frac{\mu_t}{\sqrt{n}}\Big)^{R}}{\eta_x-\eta_x\frac{\mu_t}{\sqrt{n}}}\le\frac{1}{\eta_x(1-\frac{\mu_t}{\sqrt{n}})}.$$

Therefore, with probability at least $(1-\delta_\beta^{(t)})$,

$$\sum_{\ell=1}^{R}\alpha_{\max}^{(\ell)}\Big(1-\eta_x+\eta_x\tfrac{\mu_t}{\sqrt{n}}\Big)^{R-\ell}\le C_i^{(0)}R\delta_R+\frac{1}{\eta_x(1-\frac{\mu_t}{\sqrt{n}})}\big(\tfrac{\epsilon_t^2}{2}|\mathbf{x}_{\max}^*|+t_\beta\big),$$

where $\lambda_i^{(t)}\le\frac{\epsilon_t^2}{2}$ and $|\beta_i^{(t)}|=t_\beta$ with probability at least $(1-\delta_\beta^{(t)})$ using Claim 2. □

Claim 5 (**Bound on the noise term in the estimation of a coefficient element in the support**). *With probability* $(1-\delta_\beta^{(t)})$, *each entry* $\vartheta_{i_1}^{(R)}$ *of* $\vartheta^{(R)}$ *is upper-bounded as*

$$|\vartheta_{i_1}^{(R)}|\le\mathcal{O}(t_\beta).$$

Proof of Claim 5. From (2.19) $\vartheta_{i_1}^{(R)}$ is defined as

$$\vartheta_{i_1}^{(R)}:=\sum_{r=1}^{R}\eta_x\xi_{i_1}^{(r)}(1-\eta_x)^{R-r}+\gamma_{i_1}^{(R)},$$

where $\gamma_{i_1}^{(R)}:=(1-\eta_x)^R(\mathbf{x}_{i_1}^{(0)}-\mathbf{x}_{i_1}^*(1-\lambda_{i_1}^{(t)}))$. Further, $\xi_{i_1}^{(r)}$ is as defined in (2.10),

$$\xi_{i_1}^{(r)}=\beta_{i_1}^{(t)}+\sum_{i_2\neq i_1}|\langle\mathbf{A}_{i_1}^{(t)},\mathbf{A}_{i_2}^{(t)}\rangle|\text{sign}(\langle\mathbf{A}_{i_1}^{(t)},\mathbf{A}_{i_2}^{(t)}\rangle)C_{i_2}^{(r-1)}\text{sign}(\mathbf{x}_{i_2}^*-\mathbf{x}_{i_2}^{(r)}).$$

Therefore, we have the following expression for $\vartheta_{i_1}^{(R)}$

$$\begin{aligned}\vartheta_{i_1}^{(R)}=&\beta_{i_1}^{(t)}\sum_{r=1}^{R}\eta_x(1-\eta_x)^{R-r}\\&+\sum_{r=1}^{R}\eta_x\sum_{i_2\neq i_1}|\langle\mathbf{A}_{i_1}^{(t)},\mathbf{A}_{i_2}^{(t)}\rangle|\text{sign}(\langle\mathbf{A}_{i_1}^{(t)},\mathbf{A}_{i_2}^{(t)}\rangle)C_{i_2}^{(r-1)}\text{sign}(\mathbf{x}_{i_2}^*-\mathbf{x}_{i_2}^{(r)})(1-\eta_x)^{R-r}+\gamma_{i_1}^{(R)}.\end{aligned}\tag{2.30}$$

Now $\vartheta_{i_1}^{(R)}$ can be upper-bounded as

$$\begin{aligned}\vartheta_{i_1}^{(R)} &\leq \beta_{i_1}^{(t)} \sum_{r=1}^{R} \eta_x (1-\eta_x)^{R-r} + \eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{r=1}^{R} \textstyle\sum_{i_2 \neq i_1} C_{i_2}^{(r-1)} (1-\eta_x)^{R-r} + \gamma_{i_1}^{(R)}, \\ &\leq \beta_{i_1}^{(t)} + (k-1)\eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{r=1}^{R} C_{i_2}^{(r-1)} (1-\eta_x)^{R-r} + \gamma_{i_1}^{(R)}.\end{aligned}$$

Since from Claim 6 we have

$$\begin{aligned}C_{i_2}^{(r-1)}(1-\eta_x)^{R-r} \leq (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^{*}| + \beta_{\max}^{(t)}) \Big[\sum_{s=1}^{r-1} \eta_x (1-\eta_x)^{R-s} + k c_x (1-\eta_x)^{R-r} \Big] \\ + k\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \delta_{R-2}.\end{aligned}$$

Further, since $1-(1-\eta_x)^{r-1} \leq 1$, we have that

$$\sum_{r=1}^{R} \sum_{s=1}^{r-1} \eta_x (1-\eta_x)^{R-s} = \sum_{r=1}^{R} \eta_x (1-\eta_x)^{R-r+1} \tfrac{1-(1-\eta_x)^{r-1}}{\eta_x} \leq \sum_{r=1}^{R} (1-\eta_x)^{R-r+1} \leq \tfrac{1}{\eta_x}.$$

Therefore,

$$|\vartheta_{i_1}^{(R)}| \leq |\beta_{i_1}^{(t)}| + (k-1)\tfrac{\mu_t}{\sqrt{n}} (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^{*}| + |\beta_{\max}^{(t)}|)(1+kc_x) + \Big(k\eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^2 R C_{\max}^{(0)} \delta_{R-2} + \gamma_{i_1}^{(R)}.$$

Now, since each $|\beta_i^{(t)}| = t_\beta$ with probability at least $(1-\delta_\beta^{(t)})$ for the t-th iterate, and $k = \mathcal{O}^*(\frac{\sqrt{n}}{\mu \log(n)})$, therefore $kc_x < 1$, we have that

$$|\vartheta_{i_1}^{(R)}| \leq \mathcal{O}(t_\beta).$$

with probability at least $(1-\delta_\beta^{(t)})$. □

Claim 6 (An intermediate result for $\vartheta_{i_1}^{(R)}$ calculations). For $c_x = \frac{\mu_t}{\sqrt{n}}/(1-\frac{\mu_t}{\sqrt{n}})$, we have

$$\begin{aligned}C_{i_2}^{(r-1)}(1-\eta_x)^{R-r} \leq (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^{*}| + \beta_{\max}^{(t)}) \left[\sum_{s=1}^{r-1} \eta_x (1-\eta_x)^{R-s} + k c_x (1-\eta_x)^{R-r} \right] \\ + k\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \delta_{R-2}.\end{aligned}$$

Proof of Claim 6. Here, from Claim 3 we have that for any i_1,

$$C_{i_1}^{(r+1)} \leq \alpha_{i_1}^{(r+1)} + k\eta_x \tfrac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r} \alpha_{\max}^{(\ell)} \Big(1-\eta_x+\eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{r-\ell} + k\eta_x \tfrac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \Big(1-\eta_x+\eta_x \tfrac{\mu_t}{\sqrt{n}}\Big)^{r}.$$

therefore $C_{i_2}^{(r-1)}$ is given by

$$C_{i_2}^{(r-1)} \leq \alpha_{i_2}^{(r-1)} + k\eta_x \frac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r-2} \alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell-2} + k\eta_x \frac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-2}.$$

Further, the term of interest $C_{i_2}^{(r-1)}(1-\eta_x)^{R-r}$ can be upper-bounded by

$$\begin{aligned} C_{i_2}^{(r-1)}(1-\eta_x)^{R-r} \leq \alpha_{i_2}^{(r-1)}(1-\eta_x)^{R-r} + (1-\eta_x)^{R-r} k\eta_x \frac{\mu_t}{\sqrt{n}} \sum_{\ell=1}^{r-2} \alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell-2} \\ + k\eta_x \frac{\mu_t}{\sqrt{n}} C_{\max}^{(0)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-2} (1-\eta_x)^{R-r}. \end{aligned}$$

From the definition of $\alpha_i^{(\ell)}$ from (2.18), $\alpha_{i_2}^{(r-1)}$ can be written as

$$\alpha_{i_2}^{(r-1)} = C_{\max}^{(0)}(1-\eta_x)^{r-1} + (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^*| + \beta_{\max}^{(t)}) \sum_{s=1}^{r-1} \eta_x (1-\eta_x)^{r-s}.$$

Therefore, we have

$$\alpha_{i_2}^{(r-1)}(1-\eta_x)^{R-r} = C_{\max}^{(0)}(1-\eta_x)^{R-1} + (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^*| + \beta_{\max}^{(t)}) \sum_{s=1}^{r-1} \eta_x (1-\eta_x)^{R-s}.$$

Next, to get a handle on $\alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell-2}$, consider the following using the definition of $\alpha_i^{(\ell)}$ from (2.18), where $\eta_x^{(i)} = \eta_x$ for all i,

$$\begin{aligned} \sum_{\ell=1}^{r} \alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell} &= \sum_{\ell=1}^{r} C_{\max}^{(0)}(1-\eta_x)^{\ell} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell} \\ &\quad + (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^*| + \beta_{\max}^{(t)}) \sum_{\ell=1}^{r} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell} \sum_{s=1}^{\ell} \eta_x (1-\eta_x)^{\ell-s+1}, \\ &\leq \sum_{\ell=1}^{r} C_{\max}^{(0)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r} + (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^*| + \beta_{\max}^{(t)}) \sum_{\ell=1}^{r} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell}. \end{aligned}$$

Therefore,

$$\begin{aligned} (1-\eta_x)^{R-r} \sum_{\ell=1}^{r-2} \alpha_{\max}^{(\ell)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell-2} &\leq \sum_{\ell=1}^{r-2} C_{\max}^{(0)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-2} (1-\eta_x)^{R-r} \\ &\quad + (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^*| + \beta_{\max}^{(t)})(1-\eta_x)^{R-r} \sum_{\ell=1}^{r-2} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{r-\ell-2}, \\ &\leq (R-2) C_{\max}^{(0)} \Big(1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}}\Big)^{R-2} + (\lambda_{\max}^{(t)} |\mathbf{x}_{\max}^*| + \beta_{\max}^{(t)}) \frac{(1-\eta_x)^{R-r}}{\eta_x (1 - \frac{\mu_t}{\sqrt{n}})}. \end{aligned}$$

Therefore,

$$\begin{aligned}(1-\eta_x)^{R-r}\sum_{\ell=1}^{r-2}\alpha_{\max}^{(\ell)}\Big(1-\eta_x+\eta_x\tfrac{\mu_t}{\sqrt{n}}\Big)^{r-\ell-2}&\\ \leq (r-2)C_{\max}^{(0)}\Big(1-\eta_x+\eta_x\tfrac{\mu_t}{\sqrt{n}}\Big)^{R-2}&+(\lambda_{\max}^{(t)}|\mathbf{x}_{\max}^*|+\beta_{\max}^{(t)})\tfrac{(1-\eta_x)^{R-r}}{\eta_x(1-\frac{\mu_t}{\sqrt{n}})}.\end{aligned}$$

Further, since $(1-\eta_x)\leq(1-\eta_x+\eta_x\frac{\mu_t}{\sqrt{n}})$,

$$k\eta_x\tfrac{\mu_t}{\sqrt{n}}C_{\max}^{(0)}\Big(1-\eta_x+\eta_x\tfrac{\mu_t}{\sqrt{n}}\Big)^{r-2}(1-\eta_x)^{R-r}\leq k\eta_x\tfrac{\mu_t}{\sqrt{n}}C_{\max}^{(0)}\delta_{R-2}.$$

Therefore, combining all the results we have that, for a constant $c_x=\frac{\mu_t}{\sqrt{n}}/(1-\frac{\mu_t}{\sqrt{n}})$,

$$\begin{aligned}&C_{i_2}^{(r-1)}(1-\eta_x)^{R-r}\\ &\quad\leq(\lambda_{\max}^{(t)}|\mathbf{x}_{\max}^*|+\beta_{\max}^{(t)})\Bigg[\sum_{s=1}^{r-1}\eta_x(1-\eta_x)^{R-s}+kc_x(1-\eta_x)^{R-r}\Bigg]+k\eta_x\tfrac{\mu_t}{\sqrt{n}}C_{\max}^{(0)}\delta_{R-2}.\end{aligned}$$

□

Claim 7 (Bound on the noise term in expected gradient vector estimate). $\|\Delta_j^{(t)}\|$ *where* $\Delta_j^{(t)}:=\mathbf{E}[\mathbf{A}^{(t)}\vartheta^{(R)}\text{sign}(\mathbf{x}_j^*)]$ *is upper-bounded as,*

$$\|\Delta_j^{(t)}\|=\mathcal{O}(\sqrt{m}q_{i,j}p_j\epsilon_t\|\mathbf{A}^{(t)}\|)].$$

Proof of Claim 7.

$$\Delta_j^{(t)}=\mathbf{E}[\mathbf{A}^{(t)}\vartheta^{(R)}\text{sign}(\mathbf{x}_j^*)]=\mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_S^{(R)}\text{sign}(\mathbf{x}_j^*)|S]]$$

From (2.30) we have the following definition for $\vartheta_j^{(R)}$

$$\vartheta_j^{(R)}=\beta_j^{(t)}+\sum_{r=1}^{R}\eta_x\sum_{i\neq j}|\langle\mathbf{A}_j^{(t)},\mathbf{A}_i^{(t)}\rangle|\text{sign}(\langle\mathbf{A}_j^{(t)},\mathbf{A}_i^{(t)}\rangle)C_i^{(r-1)}\text{sign}(\mathbf{x}_i^*-\mathbf{x}_i^{(r)})(1-\eta_x)^{R-r}+\gamma_j^{(R)},$$

where $\beta_j^{(t)}$ is defined as the following (2.11)

$$\beta_j^{(t)}=\sum_{i\neq j}(\langle\mathbf{A}_j^*,\mathbf{A}_i^*-\mathbf{A}_i^{(t)}\rangle+\langle\mathbf{A}_j^*-\mathbf{A}_j^{(t)},\mathbf{A}_i^{(t)}\rangle)\mathbf{x}_i^*+\textstyle\sum_{i\neq j}\langle\mathbf{A}_j^{(t)}-\mathbf{A}_j^*,\mathbf{A}_i^*\rangle\mathbf{x}_i^*.$$

Consider $\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_S^{(R)}\text{sign}(\mathbf{x}_j^*)|S]$, where $\vartheta_S^{(R)}$ is a vector with each element as defined in (2.30). Therefore, the elements of the vector $\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_S^{(R)}\text{sign}(\mathbf{x}_j^*)|S]]$ are given by

$$\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_i^{(R)}\text{sign}(\mathbf{x}_j^*)|S] = \begin{cases} \mathbf{E}_{\mathbf{x}_S^*}[\vartheta_i^{(R)}\text{sign}(\mathbf{x}_j^*)|S], & \text{for } i \neq j, \\ \mathbf{E}_{\mathbf{x}_S^*}[\vartheta_j^{(R)}\text{sign}(\mathbf{x}_j^*)|S], & \text{for } i = j. \end{cases}$$

Consider the general term of interest

$$\begin{aligned} &\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_i^{(R)}\text{sign}(\mathbf{x}_j^*)|S] \\ &\le \sum_{r=1}^{R} \eta_x(1-\eta_x)^{R-r}\underbrace{\mathbf{E}_{\mathbf{x}_S^*}[\beta_i\text{sign}(\mathbf{x}_j^*)|S]}_{\clubsuit} \\ &\qquad\qquad + \tfrac{\mu_t}{\sqrt{n}}\sum_{r=1}^{R}\eta_x(1-\eta_x)^{R-r}\textstyle\sum_{s\neq i}\underbrace{\mathbf{E}_{\mathbf{x}_S^*}[C_s^{(r-1)}\text{sign}(\mathbf{x}_s^* - \mathbf{x}_s^{(r)})\text{sign}(\mathbf{x}_j^*)|S]}_{\spadesuit} + \gamma_i^{(R)}. \end{aligned}$$

Further, since

$$\mathbf{E}_{\mathbf{x}_S^*}[\mathbf{x}_i^*\text{sign}(\mathbf{x}_j^*)|S] = \begin{cases} 0, & \text{for } i \neq j, \\ p_j, & \text{for } i = j, \end{cases}$$

we have that

$$\clubsuit := \mathbf{E}_{\mathbf{x}_S^*}[\beta_i^{(t)}\text{sign}(\mathbf{x}_j^*)|S] \le \begin{cases} 3p_j\epsilon_t & ,\text{for } i \neq j, \\ 0 & ,\text{for } i = j. \end{cases} \tag{2.31}$$

Further, for $\spadesuit_s := \mathbf{E}_{\mathbf{x}_S^*}[C_s^{(r-1)}\text{sign}(\mathbf{x}_s^* - \mathbf{x}_s^{(r)})\text{sign}(\mathbf{x}_j^*)|S]$ we have that

$$\spadesuit_s = \begin{cases} \mathbf{E}_{\mathbf{x}_S^*}[C_j^{(r-1)}(\mathbf{x}_j^* - \mathbf{x}_j^{(r-1)})\text{sign}(\mathbf{x}_j^*)|S] \le C_j^{(r-1)}, & \text{for } s = j \\ 0, & \text{for } s \neq j. \end{cases}$$

In addition, for $\sum_{s\neq i}\spadesuit_s$ we have that

$$\textstyle\sum_{s\neq i}\spadesuit_s = \begin{cases} C_j^{(r-1)}, & \text{for } i \neq j \\ 0, & \text{for } i = j. \end{cases} \tag{2.32}$$

Therefore, using the results for ♣ and $\sum\limits_{s\neq i} \spadesuit_s$, we have that $\mathbf{E}_{\mathbf{x}^*_S}[\vartheta_j^{(R)}\text{sign}(\mathbf{x}^*_j)|S] = \gamma_i^{(R)}$ for $i = j$, and for $i \neq j$ we have

$$
\begin{aligned}
&\mathbf{E}_{\mathbf{x}^*_S}[\vartheta_i^{(R)}\text{sign}(\mathbf{x}^*_j)|S] \\
&\quad \leq 3p_j\epsilon_t + \tfrac{\mu_t}{\sqrt{n}}\sum_{r=1}^{R}\mathbf{E}_{\mathbf{x}^*_S}[C_j^{(r-1)}\text{sign}(\mathbf{x}^*_j - \mathbf{x}_j^{(r)})\text{sign}(\mathbf{x}^*_j)|S]\eta_x(1-\eta_x)^{R-r} + \gamma_i^{(R)}, \\
&\quad \leq 3p_j\epsilon_t + \tfrac{\mu_t}{\sqrt{n}}\sum_{r=1}^{R}C_j^{(r-1)}\eta_x(1-\eta_x)^{R-r} + \gamma_i^{(R)}. \qquad (2.33)
\end{aligned}
$$

Here, from Claim 6, for $c_x = \frac{\mu_t}{\sqrt{n}}/(1-\frac{\mu_t}{\sqrt{n}})$ we have

$$
\begin{aligned}
&C_j^{(r-1)}(1-\eta_x)^{R-r} \\
&\quad \leq (\lambda_{\max}^{(t)}|\mathbf{x}^*_{\max}| + \beta_{\max}^{(t)})\left[\sum_{s=1}^{r-1}\eta_x(1-\eta_x)^{R-s} + kc_x(1-\eta_x)^{R-r}\right] + k\eta_x\tfrac{\mu_t}{\sqrt{n}}C_{\max}^{(0)}\delta_{R-2}.
\end{aligned}
$$

Further, due to our assumptions on sparsity, $kc_x \leq 1$; in addition by Claim 2, and with probability at least $(1-\delta_\beta^{(t)})$ we have $|\beta_{\max}^{(t)}| \leq t_\beta$, substituting,

$$
\begin{aligned}
&\sum_{r=1}^{R}C_j^{(r-1)}\eta_x(1-\eta_x)^{R-r} \\
&\quad \leq (\lambda_{\max}^{(t)}|\mathbf{x}^*_{\max}| + \beta_{\max}^{(t)})\left[\sum_{r=1}^{R}\eta_x\sum_{s=1}^{r-1}\eta_x(1-\eta_x)^{R-s} + kc_x\sum_{r=1}^{R}\eta_x(1-\eta_x)^{R-r}\right], \\
&\quad \leq (\lambda_{\max}^{(t)}|\mathbf{x}^*_{\max}| + t_\beta)(1+kc_x), \\
&\quad = \mathcal{O}(t_\beta),
\end{aligned}
$$

with probability at least $(1-\delta_\beta^{(t)})$. Combining results from (2.31), (2.32) and substituting for the terms in (2.33) using the analysis above,

$$
\mathbf{E}_{\mathbf{x}^*_S}[\vartheta_i^{(R)}\text{sign}(\mathbf{x}^*_j)|S]\begin{cases}\leq \gamma_i^{(R)}, & \text{for } i = j, \\ \leq 3p_j\epsilon_t + \frac{\mu}{\sqrt{n}}t_\beta + \gamma_i^{(R)} = \mathcal{O}(p_j\epsilon_t), & \text{for } i \neq j.\end{cases}
$$

Note that since $\gamma_i^{(R)} := (1-\eta_x)^R(\mathbf{x}_i^{(0)} - \mathbf{x}^*_i(1-\lambda_i^{(t)}))$ can be made small by choice of R. Also, since $\mathbf{Pr}[i,j \in S] = q_{i,j}$, we have

$$
\|\Delta_j^{(t)}\| = \|\mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{E}_{\mathbf{x}^*_S}[\vartheta_S^{(R)}\text{sign}(\mathbf{x}^*_j)|S]]\|,
$$

$$\leq \mathcal{O}(\sqrt{m}q_{i,j}p_j\epsilon_t\|\mathbf{A}^{(t)}\|).$$

□

Claim 8 (An intermediate result for concentration results). *With probability* $(1-\delta_\beta^{(t)}-\delta_\mathcal{T}^{(t)}-\delta_{\text{HW}}^{(t)})$ $\|\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}\|$ *is upper-bounded by* $\widetilde{\mathcal{O}}(kt_\beta)$.

Proof of Claim 8. First, using Lemma 2.4 we have

$$\widehat{\mathbf{x}}_{i_1} := \mathbf{x}_{i_1}^{(R)} = \mathbf{x}_{i_1}^*(1-\lambda_{i_1}^{(t)}) + \vartheta_{i_1}^{(R)}.$$

Therefore, the vector $\widehat{\mathbf{x}}_S$, for $S \in \text{supp}(\mathbf{x}^*)$ can be written as

$$\widehat{\mathbf{x}}_S := \mathbf{x}_S^{(R)} = (\mathbf{I}-\Lambda_S^{(t)})\mathbf{x}_S^* + \vartheta_S^{(R)}, \tag{2.34}$$

where $\widehat{\mathbf{x}}$ has the correct signed-support with probability at least $(1-\delta_\mathcal{T})$ using Lemma 2.2. Using this result, we can write $\|\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}\|$ as

$$\|\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}\| = \|\mathbf{A}_S^*\mathbf{x}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I}-\Lambda_S^{(t)})\mathbf{x}_S^* - \mathbf{A}_S^{(t)}\vartheta_S^{(R)}\|.$$

Now, since $\Lambda_{ii}^{(t)} \leq \frac{\epsilon_t^2}{2}$ we have

$$\begin{aligned}\|\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}}\| &\leq \|\mathbf{A}_S^*\mathbf{x}_S^* - (1-\tfrac{\epsilon_t^2}{2})\mathbf{A}_S^{(t)}\mathbf{x}_S^* - \mathbf{A}_S^{(t)}\vartheta_S^{(R)}\|,\\ &= \|\underbrace{((1-\tfrac{\epsilon_t^2}{2})(\mathbf{A}_S^*-\mathbf{A}_S^{(t)}) + \tfrac{\epsilon_t^2}{2}\mathbf{A}_S^*)}_{\clubsuit}\mathbf{x}_S^* - \underbrace{\mathbf{A}_S^{(t)}\vartheta_S^{(R)}}_{\spadesuit}\|.\end{aligned}$$

With $\mathbf{x}_S^*$ being independent and sub-Gaussian, using Lemma 2.13, which is a result based on the Hanson-Wright result (Hanson and Wright, 1971) for sub-Gaussian random variables, and since $\|\mathbf{A}_S^{(t)}-\mathbf{A}_S^*\| \leq \|\mathbf{A}_S^{(t)}-\mathbf{A}_S^*\|_F \leq \sqrt{k}\epsilon_t$, we have that with probability at least $(1-\delta_{\text{HW}}^{(t)})$

$$\|\clubsuit\mathbf{x}_S^*\| = \|((1-\tfrac{\epsilon_t^2}{2})(\mathbf{A}_S^*-\mathbf{A}_S^{(t)}) + \tfrac{\epsilon_t^2}{2}\mathbf{A}_S^*)\mathbf{x}_S^*\| \leq \widetilde{\mathcal{O}}(\|(1-\tfrac{\epsilon_t^2}{2})(\mathbf{A}_S^*-\mathbf{A}_S^{(t)}) + \tfrac{\epsilon_t^2}{2}\mathbf{A}_S^*\|_F),$$

where $\delta_{\text{HW}}^{(t)} = \exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.

Now, consider the $\|\clubsuit\|_F$, since $\|\mathbf{A}_S^{(t)}-\mathbf{A}_S^*\|_F \leq \sqrt{k}\epsilon_t$

$$\|\clubsuit\|_F := \|(1-\tfrac{\epsilon_t^2}{2})(\mathbf{A}_S^*-\mathbf{A}_S^{(t)}) + \tfrac{\epsilon_t^2}{2}\mathbf{A}_S^*\|_F \leq (1-\tfrac{\epsilon_t^2}{2})\|(\mathbf{A}_S^*-\mathbf{A}_S^{(t)})\|_F + \tfrac{\epsilon_t^2}{2}\|\mathbf{A}_S^*\|_F,$$

$$\leq \sqrt{k}(1-\tfrac{\epsilon_t^2}{2})\epsilon_t + \tfrac{\epsilon_t^2}{2}\|\mathbf{A}^*_S\|_F.$$

Consider the $\|\spadesuit\|$ term. Using Claim 5, each $\vartheta^{(R)}_j$ is bounded by $\mathcal{O}(t_\beta)$. with probability at least $(1-\delta^{(t)}_\beta)$ Therefore,

$$\|\spadesuit\| = \|\mathbf{A}^{(t)}_S\vartheta^{(R)}_S\| \leq \|\mathbf{A}^{(t)}_S\|\|\vartheta^{(R)}_S\| = \|\mathbf{A}^{(t)}_S\|\sqrt{k}\mathcal{O}(t_\beta).$$

Again, since $\|\mathbf{A}^{(t)}_S - \mathbf{A}^*_S\| \leq \|\mathbf{A}^{(t)}_S - \mathbf{A}^*_S\|_F \leq \sqrt{k}\epsilon_t$,

$$\|\mathbf{A}^{(t)}_S\| \leq \|\mathbf{A}^{(t)}_S - \mathbf{A}^*_S + \mathbf{A}^*_S\| \leq \|\mathbf{A}^{(t)}_S - \mathbf{A}^*_S\| + \|\mathbf{A}^*_S\| \leq \sqrt{k}\epsilon_t + 2.$$

Finally, combining all the results and using the fact that $\|\mathbf{A}^*_S\|_F \leq \sqrt{k}\|\mathbf{A}^*_S\| \leq 2\sqrt{k}$, ,

$$\begin{aligned}\|\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}}\| &= \widetilde{\mathcal{O}}(\sqrt{k}(1-\tfrac{\epsilon_t^2}{2})\epsilon_t + \epsilon_t^2\sqrt{k}) + \|\mathbf{A}^{(t)}_S\|\sqrt{k}\mathcal{O}(t_\beta),\\ &= \widetilde{\mathcal{O}}(kt_\beta).\end{aligned}$$

□

Claim 9 (**Bound on variance parameter for concentration of gradient vector**). *For* $\mathbf{z} := (\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}}_i)|i \in S$ *the variance parameter* $\mathbf{E}[\|\mathbf{z}\|^2]$ *is bounded as* $\mathbf{E}[\|\mathbf{z}\|^2] = \mathcal{O}(k\epsilon_t^2) + \mathcal{O}(kt_\beta^2)$ *with probability at least* $(1-\delta^{(t)}_\beta - \delta^{(t)}_{\mathcal{T}})$.

Proof of Claim 9. For the variance $\mathbf{E}[\|\mathbf{z}\|^2]$, we focus on the following,

$$\mathbf{E}[\|\mathbf{z}\|^2] = \mathbf{E}[\|(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}}_i)\|^2|i \in S].$$

Here, $\widehat{\mathbf{x}}_S$ is given by

$$\widehat{\mathbf{x}}_S = (\mathbf{I} - \Lambda^{(t)}_S)\mathbf{x}^*_S + \vartheta^{(R)}_S.$$

Therefore, $\mathbf{E}[\|\mathbf{z}\|^2]$ can we written as

$$\begin{aligned}&\mathbf{E}[\|(\mathbf{y} - \mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}}_i)\|^2|i \in S]\\ &\qquad = \mathbf{E}[\|(\mathbf{y} - \mathbf{A}^{(t)}_S(\mathbf{I} - \Lambda^{(t)}_S)\mathbf{x}^*_S - \mathbf{A}^{(t)}_S\vartheta^{(R)}_S)\mathrm{sign}(\widehat{\mathbf{x}}_i)\|^2|i \in S],\\ &\qquad \leq \underbrace{\mathbf{E}[\|(\mathbf{A}^*_S - \mathbf{A}^{(t)}_S(\mathbf{I} - \Lambda^{(t)}_S))\mathbf{x}^*_S\|^2|i \in S]}_{\heartsuit} + \underbrace{\mathbf{E}[\|\mathbf{A}^{(t)}_S\vartheta^{(R)}_S\mathrm{sign}(\widehat{\mathbf{x}}_i)\|^2|i \in S]}_{\diamondsuit}. \qquad (2.35)\end{aligned}$$

We will now consider each term in (2.35) separately. We start with $\heartsuit$. Since $\mathbf{x}_S^*$s are conditionally independent of S, $\mathbf{E}[\mathbf{x}_S^*\mathbf{x}_S^{*\top}] = \mathbf{I}$. Therefore, we can simplify this expression as

$$\heartsuit := \mathbf{E}[\|(\mathbf{A}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)}))\mathbf{x}_S^*\|^2 | i \in S] = \mathbf{E}[\|\mathbf{A}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})\|_F^2 | i \in S].$$

Rearranging the terms we have the following for $\heartsuit$,

$$\heartsuit = \mathbf{E}[\|\mathbf{A}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})\|_F^2 | i \in S] = \mathbf{E}[\|\mathbf{A}_S^*\Lambda_S^{(t)} + (\mathbf{A}_S^* - \mathbf{A}_S^{(t)})(\mathbf{I} - \Lambda_S^{(t)})\|_F^2 | i \in S].$$

Therefore, $\heartsuit$ can be upper-bounded as

$$\heartsuit \leq \underbrace{\mathbf{E}[\|\mathbf{A}_S^*\Lambda_S^{(t)}\|_F^2 | i \in S]}_{\heartsuit_1} + \underbrace{\mathbf{E}[\|(\mathbf{A}_S^* - \mathbf{A}_S^{(t)})(\mathbf{I} - \Lambda_S^{(t)})\|_F^2 | i \in S]}_{\heartsuit_2} + \underbrace{2\mathbf{E}[\|\mathbf{A}_S^*\Lambda_S^{(t)}\|_F\|(\mathbf{A}_S^* - \mathbf{A}_S^{(t)})(\mathbf{I} - \Lambda_S^{(t)})\|_F | i \in S]}_{\heartsuit_3}. \quad (2.36)$$

For $\heartsuit_1$, since $\|\mathbf{A}_S^{(t)}\| \leq \sqrt{k}\epsilon_t + 2$, we have

$$\heartsuit_1 := \mathbf{E}[\|\mathbf{A}_S^*\Lambda_S^{(t)}\|_F^2 | i \in S] \leq \mathbf{E}[\|\mathbf{A}_S^*\|\|\Lambda_S^{(t)}\|_F^2 | i \in S] \leq \|\mathbf{A}_S^*\| \textstyle\sum_{j\in S}(\lambda_j^{(t)})^2 \leq k(\sqrt{k}\epsilon_t + 2)\frac{\epsilon_t^4}{4}.$$

Next, since $(1 - \lambda_j^{(t)}) \leq 1$, we have the following bound for $\heartsuit_2$

$$\heartsuit_2 := \mathbf{E}[\|(\mathbf{A}_S^* - \mathbf{A}_S^{(t)})(\mathbf{I} - \Lambda_S^{(t)})\|_F^2 | i \in S] \leq \mathbf{E}[\|\mathbf{A}_S^* - \mathbf{A}_S^{(t)}\|_F^2 | i \in S] \leq \|\mathbf{A}_S^* - \mathbf{A}_S^{(t)}\|_F^2 \leq k\epsilon_t^2.$$

Further, $\heartsuit_3$ can be upper-bounded by using bounds for $\heartsuit_1$ and $\heartsuit_2$. Combining the results of upper-bounding $\heartsuit_1$, $\heartsuit_2$, and $\heartsuit_3$ we have the following for (2.36)

$$\heartsuit \leq \mathbf{E}[\|(\mathbf{A}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)}))\mathbf{x}_S^*\|^2 | i \in S] = \mathcal{O}(k\epsilon_t^2).$$

Next, by Claim 5, $\vartheta_j^{(R)}$ is upper-bounded as $|\vartheta_j^{(R)}| \leq \mathcal{O}(t_\beta)$. with probability $(1 - \delta_\beta^{(t)})$. Therefore, the term $\diamondsuit$, the second term of (2.35), can be bounded as

$$\diamondsuit \leq \|\mathbf{A}_S^{(t)}\vartheta_S^{(R)}\text{sign}(\widehat{\mathbf{x}}_i)\|^2 \leq (\sqrt{k}\epsilon_t + 2)^2 k\mathcal{O}(t_\beta)^2 = \mathcal{O}(kt_\beta^2).$$

Finally, combining all the results, the term of interest in (2.35) has the following form

$$\mathbf{E}[\|(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})\mathrm{sign}(\widehat{\mathbf{x}}_i)\|^2 | i \in S] = \mathcal{O}(k\epsilon_t^2) + \mathcal{O}(kt_\beta^2).$$

□

Claim 10 (**Bound on variance parameter for concentration of gradient matrix**). *With probability* $(1-\delta_{\mathcal{T}}^{(t)}-\delta_{\beta}^{(t)})$*, the variance parameter* $\|\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}\widehat{\mathbf{x}})^\top]\|$ *is upper-bounded by* $\mathcal{O}(\frac{k^2t_\beta^2}{m})\|\mathbf{A}^*\|^2$.

Proof of Claim 10. Let $\mathcal{F}_{\mathbf{x}^*}$ be the event that $\mathrm{sign}(\mathbf{x}^*) = \mathrm{sign}(\widehat{\mathbf{x}})$, and let $\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}$ denote the indicator function corresponding to this event. As we show in Lemma 2.2, this event occurs with probability at least $(1-\delta_\beta^{(t)}-\delta_{\mathcal{T}}^{(t)})$, therefore,

$$\begin{aligned}
&\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top] \\
&\qquad = \mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top \mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}}] + \mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top \mathbb{1}_{\bar{\mathcal{F}}_{x^*}}], \\
&\qquad = \mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top \mathbb{1}_{\mathcal{F}_{x^*}}] \pm \gamma.
\end{aligned}$$

Here, γ is small. Under the event $\mathcal{F}_{x^*}$, $\widehat{\mathbf{x}}$ has the correct signed-support. Again, since $\mathbb{1}_{\mathcal{F}_{\mathbf{x}^*}} = \mathbf{1} - \mathbb{1}_{\bar{\mathcal{F}}_{\mathbf{x}^*}}$,

$$\begin{aligned}
\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top] &= \mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top(\mathbf{1}-\mathbb{1}_{\bar{\mathcal{F}}_{x^*}})] \pm \gamma, \\
&= \mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top] \pm \gamma.
\end{aligned}$$

Now, using Lemma 2.4 with probability at least $(1-\delta_{\mathcal{T}}^{(t)}-\delta_\beta^{(t)})$, $\widehat{\mathbf{x}}_S$ admits the following expression

$$\widehat{\mathbf{x}}_S := \mathbf{x}_S^{(R)} = (\mathbf{I}-\Lambda_S^{(t)})\mathbf{x}_S^* + \vartheta_S^{(R)}.$$

Therefore we have

$$\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}} = (\mathbf{A}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I}-\Lambda_S^{(t)}))\mathbf{x}_S^* - \mathbf{A}_S^{(t)}\vartheta_S^{(R)}.$$

Using the expression above $\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top]$ can be written as

$$\begin{aligned}
&\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top] \\
&\quad = \mathbf{E}[((\mathbf{A}^*_S-\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S))\mathbf{x}^*_S-\mathbf{A}^{(t)}_S\vartheta^{(R)}_S)((\mathbf{A}^*_S-\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S))\mathbf{x}^*_S-\mathbf{A}^{(t)}_S\vartheta^{(R)}_S)^\top].
\end{aligned}$$

Sub-conditioning, we have

$$\begin{aligned}
\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})&(\mathbf{y}\text{–}\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top] \\
&=\mathbf{E}_S[(\mathbf{A}^*_S-\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S))\mathbf{E}_{\mathbf{x}^*_S}[\mathbf{x}^*_S\mathbf{x}^{*\top}_S|S](\mathbf{A}^{*\top}_S-(\mathbf{I}-\Lambda^{(t)}_S)\mathbf{A}^{(t)^\top}_S)] \\
&\quad-\mathbf{E}_S[\mathbf{A}^{(t)}_S\mathbf{E}_{\mathbf{x}^*_S}[\vartheta^{(R)}_S\mathbf{x}^{*\top}_S|S](\mathbf{A}^{*\top}_S-(\mathbf{I}-\Lambda^{(t)}_S)\mathbf{A}^{(t)^\top}_S)] \\
&\quad-\mathbf{E}_S[(\mathbf{A}^*_S-\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S))\mathbf{E}_{\mathbf{x}^*_S}[\mathbf{x}^*_S(\vartheta^{(R)}_S)^\top|S]\mathbf{A}^{(t)^\top}_S] \\
&\quad+\mathbf{E}_S[\mathbf{A}^{(t)}_S\mathbf{E}_{\mathbf{x}^*_S}[\vartheta^{(R)}_S(\vartheta^{(R)}_S)^\top|S]\mathbf{A}^{(t)^\top}_S].
\end{aligned}$$

Now, since $\mathbf{E}_{\mathbf{x}^*_S}[\mathbf{x}^*_S\mathbf{x}^{*\top}_S|S]=\mathbf{I}$,

$$\begin{aligned}
\|\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top]\| \le\, &\underbrace{\|\mathbf{E}_S[(\mathbf{A}^*_S-\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S))(\mathbf{A}^{*\top}_S-(\mathbf{I}-\Lambda^{(t)}_S)\mathbf{A}^{(t)^\top}_S)]\|}_{\clubsuit} \\
&+\underbrace{\|\mathbf{E}_S[\mathbf{A}^{(t)}_S\mathbf{E}_{\mathbf{x}^*_S}[\vartheta^{(R)}_S\mathbf{x}^{*\top}_S|S](\mathbf{A}^{*\top}_S-(\mathbf{I}-\Lambda^{(t)}_S)\mathbf{A}^{(t)^\top}_S)]\|}_{\spadesuit} \\
&+\underbrace{\|\mathbf{E}_S[(\mathbf{A}^*_S-\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S))\mathbf{E}_{\mathbf{x}^*_S}[\mathbf{x}^*_S(\vartheta^{(R)}_S)^\top|S]\mathbf{A}^{(t)^\top}_S]\|}_{\heartsuit} \\
&+\underbrace{\|\mathbf{E}_S[\mathbf{A}^{(t)}_S\mathbf{E}_{\mathbf{x}^*_S}[\vartheta^{(R)}_S(\vartheta^{(R)}_S)^\top|S]\mathbf{A}^{(t)^\top}_S]\|}_{\diamondsuit}. \qquad (2.37)
\end{aligned}$$

Let's start with the first term ($\clubsuit$) of (2.37), which can be written as

$$\begin{aligned}
\clubsuit :\le \underbrace{\|\mathbf{E}_S[\mathbf{A}^*_S\mathbf{A}^{*\top}_S]\|}_{\clubsuit_1}+\underbrace{\|\mathbf{E}_S[\mathbf{A}^*_S(\mathbf{I}-\Lambda^{(t)}_S)\mathbf{A}^{(t)^\top}_S]\|}_{\clubsuit_2}+\underbrace{\|\mathbf{E}_S[\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S)\mathbf{A}^{*\top}_S]\|}_{\clubsuit_3} \\
+\underbrace{\|\mathbf{E}_S[\mathbf{A}^{(t)}_S(\mathbf{I}-\Lambda^{(t)}_S)^2\mathbf{A}^{(t)^\top}_S]\|}_{\clubsuit_4}. \qquad (2.38)
\end{aligned}$$

Now consider each term of equation (2.38). First, since

$$\mathbf{E}_S[\mathbf{A}_S^*\mathbf{A}_S^{*\top}] = \mathbf{E}_S[\sum_{i,j\in S} \mathbf{A}_i^*\mathbf{A}_j^{*\top}\mathbb{1}_{i,j\in S}] = \sum_{i,j=1}^{m} \mathbf{A}_i^*\mathbf{A}_i^{*\top}\mathbf{E}_S[\mathbb{1}_{i,j\in S}],$$

and $\mathbf{E}_S[\mathbb{1}_{i,j\in S}] = \mathcal{O}(\frac{k^2}{m^2})$, we can upper-bound $\clubsuit_1 := \|\mathbf{E}_S[\mathbf{A}_S^*\mathbf{A}_S^{*\top}]\|$ as

$$\clubsuit_1 := \|\mathbf{E}_S[\mathbf{A}_S^*\mathbf{A}_S^{*\top}]\| = \mathcal{O}(\tfrac{k^2}{m^2})\|\mathbf{A}^*\mathbf{1}^{m\times m}\mathbf{A}^{*\top}\| = \mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|^2,$$

where $\mathbf{1}^{m\times m}$ denotes an $m \times m$ matrix of ones. Now, we turn to $\clubsuit_2 := \|\mathbf{E}_S[\mathbf{A}_S^*(\mathbf{I} - \Lambda_S^{(t)})\mathbf{A}_S^{(t)\top}]\|$ in (2.38), which can be simplified as

$$\clubsuit_2 \le \|\sum_{i,j=1}^{m} \mathbf{A}_i^*\mathbf{A}_j^{(t)\top}\mathbf{E}_S[\mathbb{1}_{i,j\in S}]\| \le \mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|\|\mathbf{A}^{(t)}\|.$$

Further, since $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$-close to $\mathbf{A}^*$, we have that $\|\mathbf{A}^{(t)}\| \le \|\mathbf{A}^{(t)} - \mathbf{A}^*\| + \|\mathbf{A}^*\| \le 3\|\mathbf{A}^*\|$, therefore

$$\clubsuit_2 := \|\mathbf{E}_S[\mathbf{A}_S^*(\mathbf{I} - \Lambda_S^{(t)})\mathbf{A}_S^{(t)\top}]\| = \mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|^2.$$

Similarly, $\clubsuit_3$ (2.38) is also $\mathcal{O}(\frac{k^2}{m})\|\mathbf{A}^*\|^2$. Next, we consider $\clubsuit_4 := \|\mathbf{E}_S[\mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)})^2\mathbf{A}_S^{(t)\top}]\|$ in (2.38) which can also be bounded similarly as

$$\clubsuit_4 = \mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|^2.$$

Therefore, we have the following for $\clubsuit$ in (2.37)

$$\clubsuit := \mathbf{E}_S[(\mathbf{A}_S^* - \mathbf{A}_S^{(t)}(\mathbf{I} - \Lambda_S^{(t)}))(\mathbf{A}_S^{*\top} - (\mathbf{I} - \Lambda_S^{(t)})\mathbf{A}_S^{(t)\top})] = \mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|^2. \tag{2.39}$$

Consider $\spadesuit$ in (2.37). Letting $\mathbf{M} = \mathbf{E}_{\mathbf{x}_S^*}[\vartheta_S^{(R)}\mathbf{x}_S^{*\top}|S]$, and using the analysis similar to that shown in 7, we have that elements of $\mathbf{M} \in \mathbb{R}^{k\times k}$ are given by

$$\mathbf{M}_{i,j} = \mathbf{E}_{\mathbf{x}_S^*}[\vartheta_i^{(R)}\mathbf{x}_j^*|S]\begin{cases} \le \mathcal{O}(\gamma_i^{(R)}), & \text{for } i = j, \\ = \mathcal{O}(\epsilon_t), & \text{for } i \ne j. \end{cases}$$

We have the following,

$$\spadesuit := \mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_S^{(R)}\mathbf{x}_S^{*^\top}|S](\mathbf{A}_S^{*^\top} - (\mathbf{I}-\Lambda_S^{(t)})\mathbf{A}_S^{(t)^\top})] = \mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{M}(\mathbf{A}_S^{*^\top} - (\mathbf{I}-\Lambda_S^{(t)})\mathbf{A}_S^{(t)^\top})].$$

Therefore, since $\mathbf{E}_S[\mathbb{1}_{i,j\in S}|S] = \mathcal{O}(\frac{k^2}{m^2})$, and $\|\mathbf{1}^{m\times m}\| = m$,

$$\begin{aligned}\spadesuit := \|\mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{M}(\mathbf{A}_S^{*^\top} - (\mathbf{I}-\Lambda_S^{(t)})\mathbf{A}_S^{(t)^\top})]\| \\ &= \|\sum_{i,j=1}^{m}\mathbf{M}_{i,j}\mathbf{A}_i^{(t)}(\mathbf{A}_j^{*^\top} - (1-\lambda_j^{(t)})\mathbf{A}_j^{(t)^\top})\mathbf{E}_S[\mathbb{1}_{i,j\in S}|S]\|, \\ &= \mathcal{O}(\epsilon_t)\|\sum_{i,j=1}^{m}\mathbf{A}_i^{(t)}(\mathbf{A}_j^{*^\top} - (1-\lambda_j^{(t)})\mathbf{A}_j^{(t)^\top})\mathbf{E}_S[\mathbb{1}_{i,j\in S}|S]\|, \\ &= \mathcal{O}(\epsilon_t)\mathcal{O}(\tfrac{k^2}{m^2})(\|\mathbf{A}^{(t)}\mathbf{1}^{m\times m}\mathbf{A}^{*^\top}\| + \|\mathbf{A}^{(t)}\mathbf{1}^{m\times m}\mathbf{A}^{(t)^\top}\|), \\ &= \mathcal{O}(\epsilon_t)\mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|^2.\end{aligned}$$

Therefore,

$$\spadesuit := \|\mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{M}(\mathbf{A}_S^{*^\top} - (\mathbf{I}-\Lambda_S^{(t)})\mathbf{A}_S^{(t)^\top})]\| = \mathcal{O}(\tfrac{k^2}{m})\epsilon_t\|\mathbf{A}^*\|^2.$$

Similarly, $\heartsuit$ in (2.37) is also bounded as $\spadesuit$. Next, we consider $\diamondsuit$ in (2.37). In this case, letting $\mathbf{E}_{\mathbf{x}_S^*}[\vartheta_S^{(R)}(\vartheta_S^{(R)})^\top|S] = \mathbf{N}$, where $\mathbf{N} \in \mathbb{R}^{k\times k}$ is a matrix whose each entry $\mathbf{N}_{i,j} \leq |\vartheta_i^{(R)}||\vartheta_j^{(R)}|$. Further, by Claim 5, each element $\vartheta_j^{(R)}$ is upper-bounded as

$$|\vartheta_j^{(R)}| \leq \mathcal{O}(t_\beta).$$

with probability at least $(1-\delta_\beta^{(t)})$. Therefore,

$$\diamondsuit = \|\sum_{i,j=1}^{m}\mathbf{N}_{i,j}\mathbf{A}_i^{(t)}\mathbf{A}_j^{(t)^\top}\mathbf{E}_S[\mathbb{1}_{i,j\in S}|S]\| = \max_{i,j}|\vartheta_i^{(R)}||\vartheta_j^{(R)}|\mathcal{O}(\tfrac{k^2}{m^2})\|\sum_{i,j=1}^{m}\mathbf{A}_i^{(t)}\mathbf{A}_j^{(t)^\top}\|.$$

Again, using the result on $|\vartheta_{i_1}^{(R)}|$, we have

$$\diamondsuit := \|\mathbf{E}_S[\mathbf{A}_S^{(t)}\mathbf{N}\mathbf{A}_S^{(t)^\top}]\| = m\max_{i,j}|\vartheta_i^{(R)}||\vartheta_j^{(R)}|\mathcal{O}(\tfrac{k^2}{m^2})\|\mathbf{A}^{(t)}\|\|\mathbf{A}^{(t)}\| = \mathcal{O}(\tfrac{k^2t_\beta^2}{m})\|\mathbf{A}^*\|^2.$$

Combining all the results for $\clubsuit$, $\spadesuit$, $\heartsuit$ and $\diamondsuit$, we have,

$$\|\mathbf{E}[(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})(\mathbf{y}-\mathbf{A}^{(t)}\widehat{\mathbf{x}})^\top]\|$$

$$= \mathcal{O}(\tfrac{k^2}{m})\|\mathbf{A}^*\|^2 + \mathcal{O}(\tfrac{k^2}{m})\epsilon_t\|\mathbf{A}^*\|^2 + \mathcal{O}(\tfrac{k^2}{m})\epsilon_t\|\mathbf{A}^*\|^2 + \mathcal{O}(\tfrac{k^2 t_\beta^2}{m})\|\mathbf{A}^*\|^2,$$
$$= \mathcal{O}(\tfrac{k^2 t_\beta^2}{m})\|\mathbf{A}^*\|^2.$$

□

2.E Additional Experimental Results

We now present some additional results to highlight the features of NOODL. Specifically, we compare the performance of NOODL (for both dictionary and coefficient recovery) with the state-of-the-art provable techniques for DL presented in Arora et al. (2015) (when the coefficients are recovered via a sparse approximation step after DL)[2]. We also compare the performance of NOODL with the popular online DL algorithm in Mairal et al. (2009), denoted by `Mairal '09`. Here, the authors show that alternating between a ℓ_1-based sparse approximation and dictionary update based on block co-ordinate descent converges to a stationary point, as compared to the true factors in case of NOODL.

Data Generation: We generate a $(n = 1000) \times (m = 1500)$ matrix, with entries drawn from $\mathcal{N}(0,1)$, and normalize its columns to form the ground-truth dictionary $\mathbf{A}^*$. Next, we perturb $\mathbf{A}^*$ with random Gaussian noise, such that the unit-norm columns of the resulting matrix, $\mathbf{A}^{(0)}$ are $2/\log(n)$ away from $\mathbf{A}^*$, in ℓ_2-norm sense, i.e., $\epsilon_0 = 2/\log(n)$; this satisfies the initialization assumptions in **A.4**. At each iteration, we generate $p = 5000$ samples $\mathbf{Y} \in \mathbb{R}^{1000\times5000}$ as $\mathbf{Y} = \mathbf{A}^*\mathbf{X}^*$, where $\mathbf{X}^* \in \mathbb{R}^{m\times p}$ has at most $k = 10$, 20, 50, and 100, entries per column, drawn from the Radamacher distribution. We report the results in terms of relative Frobenius error for all the experiments, i.e., for a recovered matrix $\widehat{\mathbf{M}}$, we report $\|\widehat{\mathbf{M}} - \mathbf{M}^*\|_{\mathrm{F}}/\|\mathbf{M}^*\|_{\mathrm{F}}$. To form the coefficient estimate for `Mairal '09` via Lasso (Tibshirani, 1996) we use the FISTA (Beck and Teboulle, 2009) algorithm by searching across 10 values of the regularization parameter at each iteration. Note that, although our phase transition analysis for NOODL shows that $p = m$ suffices, we use $p = 5000$ in our convergence analysis for a fair comparison with related techniques.

[2] The associated code is made available at https://github.com/srambhatla/NOODL; see Chapter 2 for details.

2.E.1 Coefficient Recovery

Table 2.E.1 summarizes the results of the convergence analysis shown in Fig. 2.2. Here, we compare the dictionary and coefficient recovery performance of NOODL with other techniques. For `Arora15(‘‘biased’’)` and `Arora15(‘‘unbiased’’)`, we report the error in recovered coefficients after the HT step ($\mathbf{X}_{\text{HT}}$) and the best error via sparse approximation using Lasso[3] Tibshirani (1996), denoted as $\mathbf{X}_{\text{Lasso}}$, by scanning over 50 values of regularization parameter. For `Mairal` ‘09 at each iteration of the algorithm we scan across 10 values[4] of the regularization parameter, to recover the best coefficient estimate using Lasso (via FISTA), denoted as $\mathbf{X}_{\text{Lasso}}$.

We observe that NOODL exhibits significantly superior performance across the board. Also, we observe that using sparse approximation after dictionary recovery, when the dictionary suffers from a bias, leads to poor coefficient recovery[5], as is the case with `Arora15(‘‘biased’’)`, `Arora15(‘‘unbiased’’)`, and `Mairal` ‘09. This highlights the applicability of our approach in real-world machine learning tasks where coefficient recovery is of interest. In fact, it is a testament to the fact that, even in cases where dictionary recovery is the primary goal, making progress on the coefficients is also important for dictionary recovery.

In addition, the coefficient estimation step is also online in case of NOODL, while for the state-of-the-art provable techniques (which only recover the dictionary and incur bias in estimation) need additional sparse approximation step for coefficient recovery. Moreover, these sparse approximation techniques (such as Lasso) are expensive to use in practice, and need significant tuning.

2.E.2 Computational Time

In addition to these convergence results, we also report the computational time taken by each of these algorithms in Table 2.E.1. The results shown here were compiled using

[3] We use the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) (Beck and Teboulle, 2009), which is among the most efficient algorithms for solving the ℓ_1-regularized problems. Note that, in our experiments we fix the step-size for FISTA as $1/L$, where L is the estimate of the Lipschitz constant (since $\mathbf{A}$ is not known exactly).

[4] Note that, although scanning across 50 values of the regularization parameter for this case would have led to better coefficient estimates and dictionary recovery, we choose 10 values for this case since it is very expensive to scan across 50 of regularization parameter at each step. This also highlights why `Mairal` ‘09 may be prohibitive for large scale applications.

[5] When the dictionary is not known exactly, the guarantees may exist on coefficient recovery only in terms of closeness in ℓ_2-norm sense, due to the error-in-variables (EIV) model for the dictionary (Fuller, 2009; Wainwright, 2009).

Table 2.E.1: Final error in recovery of the factors by various techniques and the computation time taken per iteration (in seconds) corresponding to Fig. 2.2 across techniques. We report the coefficient estimate after the HT step (in Arora et al. (2015)) as $\mathbf{X}_{\text{HT}}$. For the techniques presented in Arora et al. (2015), we scan across 50 values of the regularization parameter for coefficient estimation using Lasso after learning the dictionary ($\mathbf{A}$), and report the optimal estimation error for the coefficients ($\mathbf{X}_{\text{Lasso}}$), while for `Mairal` ‘09, at each step the coefficients estimate is chosen by scanning across 10 values of the regularization parameters. For $k = 100$, the algorithms of Arora et al. (2015) do not converge (shown as N/A).

Technique	**Recovered Factor and Timing**	**k = 10**	**k = 20**	**k = 50**	**k = 100**
`NOODL`	$\mathbf{A}$	9.44×10^{-11}	8.82×10^{-11}	9.70×10^{-11}	7.33×10^{-11}
	$\mathbf{X}$	1.14×10^{-11}	1.76×10^{-11}	3.58×10^{-11}	4.74×10^{-11}
	Avg. Time/iteration	46.500 sec	53.303 sec	64.800 sec	96.195 sec
`Arora15` (‘‘`biased`’’)	$\mathbf{A}$	0.013	0.031	0.137	N/A
	$\mathbf{X}_{\text{HT}}$	0.077	0.120	0.308	N/A
	$\mathbf{X}_{\text{Lasso}}$	0.006	0.018	0.097	N/A
	Avg. Time/iteration (*Accounting for one Lasso update*)	39.390 sec	39.371 sec	39.434 sec	40.063 sec
	Avg. Time/iteration (*Overall Lasso search*)	389.368 sec	388.886 sec	389.566 sec	395.137 sec
`Arora15` (‘‘`unbiased`’’)	$\mathbf{A}$	0.011	0.027	0.148	N/A
	$\mathbf{X}_{\text{HT}}$	0.078	0.122	0.371	N/A
	$\mathbf{X}_{\text{Lasso}}$	0.005	0.015	0.0921	N/A
	Avg. Time/iteration (*Accounting for one Lasso update*)	567.830 sec	597.543 sec	592.081 sec	686.694 sec
	Avg. Time/iteration (*Overall Lasso search*)	917.809 sec	947.059 sec	942.212 sec	1041.767 sec
`Mairal` ‘09	$\mathbf{A}$	0.009	0.015	0.021	0.037
	$\mathbf{X}_{\text{Lasso}}$	0.183	0.209	0.275	0.353
	Avg. Time/iteration (*Accounting for one Lasso update*)	39.110 sec	39.077 sec	39.163 sec	39.672 sec
	Avg. Time/iteration (*Overall Lasso search*)	388.978 sec	388.614 sec	389.512 sec	394.566 sec

5 cores and 200GB RAM of Intel Xeon E5 – 2670 Sandy Bridge and Haswell E5-2680v3 processors.

The primary takeaway is that although NOODL takes marginally more time per iteration as compared to other methods when accounting for just one Lasso update step for the coefficients, it (a) is in fact faster per iteration since it does not involve any computationally expensive tuning procedure to scan across regularization parameters; owing to its geometric convergence property (b) achieves orders of magnitude superior error at convergence, and as a result, (c) overall takes significantly less time to reach such a solution. Further, NOODL’s computation time can be further reduced via implementations using the neural architecture illustrated in Section 2.5.

Note that since the coefficient estimates using just the HT step at every step may not yield a usable result for `Arora15(‘‘unbiased’’)` and `Arora15(‘‘biased’’)` as shown in Table 2.E.1, in practice, one has to employ an additional ℓ_1-based sparse recovery step. Therefore, for a fair comparison, we account for running sparse recovery step(s) using Lasso (via the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) (Beck and Teboulle, 2009)) at every iteration of the algorithms `Arora15(‘‘biased’’)` and `Arora15(‘‘unbiased’’)`.

For our technique, we report the average computation time taken per iteration. However, for the rest of the techniques, the coefficient recovery using Lasso (via FISTA) involves a search over various values of the regularization parameters (10 values for this current exposition). As a result, we analyze the computation time per iteration via two metrics. First of these is the average computation time taken per iteration by accounting for the average time take per Lasso update (denoted as “*Accounting for one Lasso update*”), and the second is the average time taken per iteration to scan over all (10) values of the regularization parameter (denoted as “*Overall Lasso search*”) .

As shown in Table 2.E.1, in comparison to NOODL the techniques described in Arora et al. (2015) still incur a large error at convergence, while the popular online DL algorithm of Mairal et al. (2009) exhibits very slow convergence rate. Combined with the convergence results shown in Fig. 2.2, we observe that due to NOODL’s superior convergence properties, it is overall faster and also geometrically converges to the true factors. This again highlights the applicability of NOODL in practical applications, while guaranteeing convergence to the true factors.

2.F Appendix: Standard Results

Definition 2.6 (sub-Gaussian Random variable)**.** Let $x \sim \text{subGaussian}(\sigma^2)$. Then, for any $t > 0$, it holds that

$$\mathbf{Pr}[|x| > t] \leq 2\exp\left(\tfrac{t^2}{2\sigma^2}\right).$$

2.F.1 Concentration results

Lemma 2.10 (Matrix Bernstein (Tropp, 2015))**.** Consider a finite sequence $\mathbf{W}_k \in \mathbb{R}^{n\times m}$ of independent, random, centered matrices with dimension n. Assume that each random matrix satisfies $\mathbf{E}[\mathbf{W}_k] = 0$ and $\|\mathbf{W}_k\| \le R$ almost surely. Then, for all $t \ge 0$,

$$\mathbf{Pr}\Big\{\|\sum_k \mathbf{W}_k\| \ge t\Big\} \le (n+m)\exp\Big(\tfrac{-t^2/2}{\sigma^2+Rt/3}\Big),$$

where $\sigma^2 := \max\{\|\sum_k \mathbf{E}[\mathbf{W}_k\mathbf{W}_k^\top]\|, \|\sum_k \mathbf{E}[\mathbf{W}_k^\top \mathbf{W}_k]\|\}$.

Furthermore,

$$\mathbf{E}[\|\sum_k \mathbf{W}_k\|] \le \sqrt{2\sigma^2\log(n+m)} + \tfrac{1}{3}R\log(n+m).$$

Lemma 2.11 (Vector Bernstein (Tropp, 2015))**.** Consider a finite sequence $\mathbf{w}_k \in \mathbb{R}^n$ of independent, random, zero mean vectors with dimension n. Assume that each random vector satisfies $\mathbf{E}[\mathbf{w}_k] = 0$ and $\|\mathbf{w}_k\| \le R$ almost surely. Then, for all $t \ge 0$,

$$\mathbf{Pr}\Big\{\|\sum_k \mathbf{w}_k\| \ge t\Big\} \le 2n\exp\Big(\tfrac{-t^2/2}{\sigma^2+Rt/3}\Big),$$

where $\sigma^2 := \|\sum_k \mathbf{E}[\|\mathbf{w}_k\|^2]\|$. Furthermore,

$$\mathbf{E}[\|\sum_k \mathbf{w}_k\|] \le \sqrt{2\sigma^2\log(2n)} + \tfrac{1}{3}R\log(2n).$$

Lemma 2.12. Chernoff Bound for sub-Gaussian Random Variables Let w be an independent sub-Gaussian random variables with variance parameter σ^2, then for any $t > 0$ it holds that

$$\mathbf{Pr}[|w| > t] \le 2\exp(-\tfrac{t^2}{2\sigma^2}).$$

Lemma 2.13 (**Sub-Gaussian concentration** (**Rudelson and Vershynin, 2013**))**.** Let $\mathbf{M} \in \mathbb{R}^{n\times m}$ be a fixed matrix. Let $\mathbf{w}$ be a vector of independent, sub-Gaussian random variables with mean zero and variance one. Then, for an absolute constant c,

$$\mathbf{Pr}[\|\mathbf{Mx}\|_2 - \|\mathbf{M}\|_F > t] \le \exp(-\tfrac{ct^2}{\|\mathbf{M}\|^2}).$$

2.F.2 Results from (Arora et al., 2015)

Lemma 2.14 ((Arora et al., 2015) Lemma 45)**.** Suppose that the distribution of $\mathbf{Z}$ satisfies $\mathbf{Pr}[\|\mathbf{Z}\| \geq L(\log(1/\rho))^C] \leq \rho]$ for some constant $C > 0$, then

1. If $p = n^{\mathcal{O}(1)}$ then $\|\mathbf{Z}^{(j)}\| \leq \widetilde{\mathcal{O}}(L)$ holds for each j with probability at least$(1-\rho)$ and,

2. $\|\mathbf{E}[\mathbf{Z}\mathbb{1}_{\|\mathbf{Z}\|\geq\widetilde{\Omega}(L)}]\| = n^{-\omega(1)}$.

In particular, if $\frac{1}{p}\sum_{j=1}^{p}\mathbf{Z}^{(j)}(1 - \mathbb{1}_{\|\mathbf{Z}\|\geq\widetilde{\Omega}(L)})$ is concentrated with probability $(1-\rho)$, then so is $\frac{1}{p}\sum_{j=1}^{p}\mathbf{Z}^{(j)}$.

Lemma 2.15 (Theorem 40 (Arora et al., 2015))**.** Suppose random vector $\mathbf{g}^{(t)}$ is a $(\rho_{\_}, \rho_{+}, \zeta_t)$-correlated with high probability with $\mathbf{z}^*$ for $t \in [T]$ where $T \leq poly(n)$, and η_A satisfies $0 < \eta_A \leq 2\rho_{+}$, then for any $t \in [T]$,

$$\mathbf{E}[\|\mathbf{z}^{(t+1)} - \mathbf{z}^*\|^2] \leq (1 - 2\rho_{\_}\eta_A)\|\mathbf{z}^{(t)} - \mathbf{z}^*\| + 2\eta_A\zeta_t.$$

In particular, if $\|\mathbf{z}^{(0)} - \mathbf{z}^*\| \leq \epsilon_0$ and $\zeta_t \leq (\rho_{\_})o((1-2\rho_{\_}\eta)^t)\epsilon_0^2 + \zeta$, where $\zeta = \max_{t\in[T]}\zeta_t$, then the updates converge to $\mathbf{z}^*$ geometrically with systematic error $\zeta/\rho_{\_}$ in the sense that

$$\mathbf{E}[\|\mathbf{z}^{(t+1)} - \mathbf{z}^*\|^2] \leq (1 - 2\rho_{\_}\eta_A)^t\epsilon_0^2 + \zeta/\rho_{\_}.$$

Chapter 3

Provable Structured Tensor Factorization via Dictionary Learning

3.1 Overview

We consider the problem of factorizing a structured tensor into its constituent Canonical Polyadic (CP) factors. This decomposition, which can be viewed as a generalization of singular value decomposition (SVD) for tensors, reveals how the tensor dimensions (features) interact with each other. However, since these factors are *a priori* unknown, the corresponding optimization problems are inherently non-convex. The existing guaranteed algorithms which handle this non-convexity only apply to cases where all factors have the same structure, and also incur an irreducible error. To address this gap, we develop a provable algorithm for structured tensor factorization, wherein one of the factors obeys some incoherence conditions, and the others are sparse. Motivated by recent dictionary learning results, we show that, under some relatively mild conditions on initialization, rank, and sparsity, our algorithm recovers the factors *exactly* (up to scaling and permutation) at a linear rate. Moreover, its scalability and ability to learn on-the-fly makes it suitable for real-world applications.

3.2 Introduction

Canonical Polyadic (CP) /PARAFAC decomposition aims to express a tensor as a sum of rank-1 tensors, each of which is formed by the outer-product of columns of constituent factors. Specifically, for decomposing a 3-way tensor, the task entails factorizing a given tensor $\underline{\mathbf{Z}} \in \mathbb{R}^{n \times J \times K}$, as

$$\underline{\mathbf{Z}} = \textstyle\sum_{i=1}^{m} \mathbf{A}_i^* \circ \mathbf{B}_i^* \circ \mathbf{C}_i^* = [\![\mathbf{A}^*, \mathbf{B}^*, \mathbf{C}^*]\!], \tag{3.1}$$

where $\mathbf{A}_i^*$, $\mathbf{B}_i^*$ and $\mathbf{C}_i^*$ are columns of factors $\mathbf{A}^*$, $\mathbf{B}^*$, and $\mathbf{C}^*$, respectively, and are *a priori* unknown; See Kolda and Bader (2009) and references therein for details.

In this work, we develop a provable algorithm for factorization of tensor(s) $\underline{\mathbf{Z}}^{(t)}$ (arriving, or made available for sequential processing, at an instance t), assumed to be generated as (3.1), wherein the factor $\mathbf{A}^*$ is *incoherent* and the factors $\mathbf{B}^{*(t)}$ and $\mathbf{C}^{*(t)}$ are sparse.

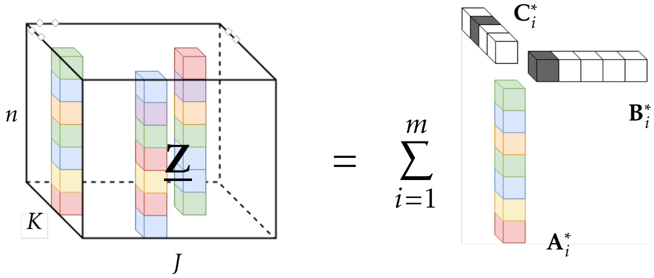


Figure 3.1: The structured tensor $\underline{\mathbf{Z}} \in \mathbb{R}^{n \times J \times K}$ considered in this work. The tensor has a few mode-1 fibers which are dense.

These structural assumptions result in a tensor that only has a few non-zero *fibers*, as in Fig.3.1, and may arise in applications where users only interact with specific items, i.e., the user – item interactions are *sparse*. For instance, in web analytics for scrolling pattern analysis across users and sites (Mueller and Lockerd, 2001), analysis of patient responses to different probe locations (Deburchgraeve et al., 2009; Becker et al., 2015), electro-dermal responses of users to different audio-visual stimuli or activities ((Grundlehner et al., 2009; Silveira et al., 2013), and so on. Overall, the factorization in (3.1) reveals similarity (clustering) between different users, items, their corresponding temporal *signatures*, and insights into how these dimensions interact with each other.

A popular choice for the factorization task shown in (3.1) is via the alternating least squares (ALS) algorithm; see Kolda and Bader (2009). Here, to steer the algorithm towards specific solutions (such as ℓ_1 loss for sparsity) one can add appropriate regularization terms to the least-square objective (Martínez-Montes et al., 2008; Allen, 2012; Papalexakis et al., 2013). However, this approach suffers from three major issues – a) it is difficult to characterize the nature of the solution, i.e., there may not be any recovery or (limited) convergence guarantees, b) one may need to solve an implicit model selection problem (e.g., choose the *tensor rank* m, which may not be known *a priori*), and c) adding regularization may be computationally expensive, which in addition, may not be scalable.

With an aim to develop guaranteed algorithms for tensor factorization, recent works – based on tensor power method (Anandkumar et al., 2015), convex relaxations (Tang and Shah, 2015), sum-of-squares formulations (Barak et al., 2015; Ma et al., 2016; Schramm and Steurer, 2017), and variants of ALS algorithm (Sharan and Valiant, 2017) – have focused on recovery of tensor factors based on some notion of incoherence of individual factor matrices, but cannot tackle the case when the constituent factors do not have a shared structure, i.e. they need that all factors are sparse, incoherent, or both; see also Sun et al. (2017). Furthermore, these algorithms may be computationally expensive in practice, not amenable for online (streaming) tensor data factorization, in addition to incurring *bias* in estimation. Consequently, there is a need to develop fast, scalable provable algorithms for (*exact*) factorization of structured tensors arriving (or processed) in a streaming fashion (online), generated by interactions of heterogeneously structured factors, for emerging applications in neurophysiology, text mining, community detection, and other signal/image processing tasks (see Sidiropoulos et al. (2017) and references therein).

3.2.1 Overview of the algorithm

The structured tensor of interest (Fig. 3.1) contains only a few non-zero fibers. With incoherence conditions on the factor $\mathbf{A}^*$ and sparsity assumptions on $\mathbf{B}^{*(t)}$ and $\mathbf{C}^{*(t)}$, we model these fibers as being generated by the dictionary learning model, where the task is to learn an *a priori* unknown dictionary $\mathbf{A}^* \in \mathbb{R}^{n\times m}$ and sparse coefficients $\mathbf{x}^*_{(j)} \in \mathbb{R}^m$ from data samples $\mathbf{y}_{(j)} \in \mathbb{R}^n$ as

$$\mathbf{y}_{(j)} = \mathbf{A}^*\mathbf{x}^*_{(j)}, \ \|\mathbf{x}^*_{(j)}\|_0 \leq s \ \text{ for all } \ j = 1, 2, \ldots \tag{3.2}$$

Analyzing (and developing an algorithm for untangling) the specific Kronecker dependence structure (of the sparse coefficients) that arises due to this matrix factorization view of the tensor decomposition task, we develop a provable tensor factorization algorithm motivated by recent exact recovery results for online dictionary learning.

Our analysis leverages results from Chapter 2 (Rambhatla et al. (2019)), however, the Kronecker dependence structure precludes us from applying this result directly. As a result, a careful characterization of this structure forms the bulk of our analysis. This structure also leads us consider more involved distributional assumptions on the sparse factors. Interestingly, in this case, a lower-bound on sparsity arises, to ensure that there are adequate data samples to support the learning algorithm. Since we adopt a matricized view of the tensor factorization task, we also develop (and establish the correctness of) a separate SVD-based algorithm to untangle a Kronecker-structured sparse matrix. This matricized view of the tensor decomposition task can be of independent interest.

3.2.2 Contributions

We develop an algorithm to recover the CP factors of tensor(s) $\underline{\mathbf{Z}}^{(t)}\in\mathbb{R}^{n\times J\times K}$, arriving (or made available) at the t^{th} instance, generated as per (3.1) from constituent factors $\mathbf{A}^* \in \mathbb{R}^{n\times m}$, $\mathbf{B}^{*(t)} \in \mathbb{R}^{J\times m}$, and $\mathbf{C}^{*(t)} \in \mathbb{R}^{K\times m}$, where the unit-norm columns of $\mathbf{A}^*$ obey some incoherence assumptions, and $\mathbf{B}^{*(t)}$ and $\mathbf{C}^{*(t)}$ are sparse[1]. Our specific contributions are summarized below.

- **Provable algorithm for heterogeneously-structured tensor factorization**: To the best of our knowledge, our algorithm is the first to accomplish *exact* provable tensor factorization – an inherently non-convex task – when the factors do not obey the same structural assumptions.
- **Exact recovery and linear convergence**: Our algorithm – `TensorNOODL` – recovers the true CP factors of the structured tensor(s) $\underline{\mathbf{Z}}$ *exactly* (up to scaling and permutations) at a linear rate, starting with an appropriate initialization $\mathbf{A}^{(0)}$ of $\mathbf{A}^*$, i.e., we have $\mathbf{A}_i^{(t)}\rightarrow\mathbf{A}_i^*$, $\widehat{\mathbf{B}}_i^{(t)}\rightarrow\pi_{B_i}\mathbf{B}_i^{*(t)}$, and $\widehat{\mathbf{C}}_i^{(t)}\rightarrow\pi_{C_i}\mathbf{C}_i^{*(t)}$, as iterations $t\rightarrow\infty$, where π_{B_i} and π_{C_i} are constants.
- **Rank revealing decomposition and uniqueness**: Although estimating the rank of a given tensor is NP hard, the incoherence assumption on $\mathbf{A}^*$, and distributional

[1] Henceforth, we denote $\underline{\mathbf{Z}}^{(t)}$ as $\underline{\mathbf{Z}}$, and drop $(\cdot)^{(t)}$ from $\mathbf{B}^{*(t)}$ and $\mathbf{C}^{*(t)}$ as well, for simplicity.

Table 3.1: Comparison of provable algorithms for tensor factorization and dictionary learning. As shown here, the existing provable tensor factorization techniques do not apply to the case where **A**: incoherent, (**B**, **C**): sparse. We use dictionary learning to develop a provable algorithm.

Method	Conditions			Recovery Guarantees	
	Model Considered	Rank	Initialization Constraints	Estimation Bias	Convergence
TensorNOODL (this work)	**A**: incoherent, (**B**, **C**): sparse	$m = \mathcal{O}(n)$	$\mathcal{O}^*\left(\frac{1}{\log(n)}\right)$	No Bias	Linear
Sun et al. (2017)‡	(**A**, **B**, **C**): all incoherent and sparse	$m = o(n^{1.5})$	$o(1)$	$\lVert\mathbf{A}_{ij} - \widehat{\mathbf{A}}_{ij}\rVert_\infty = \mathcal{O}(\frac{1}{n^{0.25}})^\dagger$	Not established
Sharan and Valiant (2017)‡	(**A**, **B**, **C**): all incoherent	$m = o(n^{0.25})$	Random	$\lVert\mathbf{A}_i - \widehat{\mathbf{A}}_i\rVert_2 = \mathcal{O}(\sqrt{\frac{m}{n}})^\dagger$	Quadratic
Anandkumar et al. (2015)‡	(**A**, **B**, **C**): all incoherent	$m = \mathcal{O}(n)$	$\mathcal{O}^*\left(\frac{1}{\sqrt{n}}\right)^\P$	$\lVert\mathbf{A}_i - \widehat{\mathbf{A}}_i\rVert_2 = \widetilde{\mathcal{O}}(\frac{1}{\sqrt{n}})^\dagger$	Linear§
		$m = o(n^{1.5})$	$\mathcal{O}(1)$	$\lVert\mathbf{A}_i - \widehat{\mathbf{A}}_i\rVert_2 = \widetilde{\mathcal{O}}(\frac{\sqrt{m}}{n})^\dagger$	Linear
Arora et al. (2015)	Dictionary Learning (3.2)	$m = \mathcal{O}(n)$	$\mathcal{O}^*\left(\frac{1}{\log(n)}\right)$	$\mathcal{O}(\sqrt{s/n})$	Linear
		$m = \mathcal{O}(n)$	$\mathcal{O}^*\left(\frac{1}{\log(n)}\right)$	*Negligible* bias §	Linear
Mairal et al. (2009)	Dictionary Learning (3.2)	Convergence to stationary point; similar guarantees by Huang et al. (2016).			

‡ This procedure is not *online*. † Result applies for each $i \in [1, m]$. ¶ Polynomial number of initializations m^{β^2} are required, for $\beta \geq m/n$. § The procedure has an *almost* Quadratic rate initially. ♮ Requires poly(m) samples; not *neurally plausible*.

assumptions on $\mathbf{B}^{*(t)}$ and $\mathbf{C}^{*(t)}$, ensure that our matrix factorization view of the tensor is *rank revealing* (Sidiropoulos et al., 2017). In other words, our model assumptions ensure that the dictionary initialization algorithms (such as Arora et al. (2015)) can recover the rank of the tensor. Following which, our algorithm can recover the true factors (up to scaling and permutation) with high probability.

- **Online, fast, and scalable**: The online nature of our algorithm, with specific guidelines on choosing the parameters, and its *neural plausibility* (involving simple linear and non-linear operations), make it suitable for large-scale distributed implementations in real-world applications. Our numerical simulations demonstrate superior performance both in terms of accuracy and timing.

3.2.3 Related works

Tensor Factorization – Canonical polyadic (CP)/PARAFAC decomposition (3.1) captures relationships between the latent factors of the tensor. Here, the number of rank-1 tensors in the sum (3.1) defines the notion of rank for a tensor. One fascinating feature about tensor decompositions is that, unlike matrices decompositions, these can be unique under relatively mild conditions (Kruskal, 1977; Sidiropoulos and Bro, 2000). However, determining the rank of a given tensor is NP-hard (Håstad, 1990), and so are most tensor problems, for instance tensor decompositions and rank determination

(Hillar and Lim, 2013). Nevertheless, ALS-based approaches have emerged as a popular choice for various tensor factorization tasks. However, establishing convergence to even a stationary point is difficult (Mohlenkamp, 2013). Variants of ALS which do come with some convergence guarantees do so at the expense of complexity (Li et al., 2015; Razaviyayn et al., 2013), and convergence rate (Uschmajew, 2012); See Kolda and Bader (2009) and Sidiropoulos et al. (2017).

On the other hand, guaranteed methods for tensor factorization initially relied on computationally expensive orthogonalizing step (*whitening*), and therefore, did not extend to the overcomplete setting ($m > n$) (Comon, 1994; Kolda and Mayo, 2011; Zhang and Golub, 2001; Le et al., 2011; Huang and Anandkumar, 2015; Anandkumar et al., 2014, 2016). As a result, works such as Tang and Shah (2015) and Anandkumar et al. (2015), developed algorithms based on convex relaxations and tensor power iterations, respectively, by relaxing the orthogonality requirement of previous works to an *incoherence* condition to extend results to overcomplete settings. Here, in addition to incuring bias, the algorithm proposed in Anandkumar et al. (2015) may be impractical in practice since it uses multiple random initializations, and for each initialization it further relies on repeated power iterations coupled with clustering and co-ordinate descent steps to recover incoherent factors. As a result, to leverage the simplicity of the ALS algorithm, Sharan and Valiant (2017) developed a provable variant – orth-ALS, by including an orthogonalization step. However, this step precludes the use of this algorithm in overcomplete settings. Further, works such as Sun and Luo (2016), consider the case wherein all factors are *sparse* as well as incoherent.

Overall, the existing provable techniques (summarized in Table 3.1) in addition to being computationally expensive, apply to cases where all factors obey some incoherence (and in some cases sparsity) conditions, i.e., *they assume that the factors are similarly structured*. That is, these techniques do not extend to the case where the factors are heterogeneously structured. As a result, there is a need for fast and scalable provable tensor factorization techniques which can recover heterogeneously structured factors; our work provides one such result.

Dictionary Learning – Popularized by the rich sparse inference literature, *overcomplete* ($m \geq n$) representations lead to sparse(r) representations which are robust to noise; see Mallat and Zhang (1993); Chen et al. (1998); Donoho et al. (2006). Learning such sparsifying overcomplete representations is known as *dictionary learning* (Olshausen and Field, 1997; Lewicki and Sejnowski, 2000; Mairal et al., 2009). Analogous to the ALS

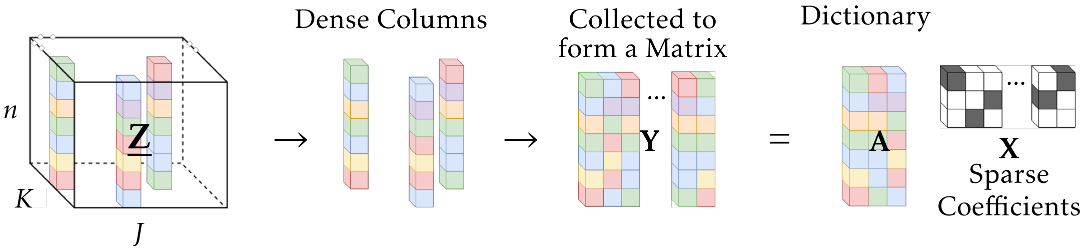


Figure 3.2: Problem Formulation: The dense columns of the structured tensor $\underline{\mathbf{Z}} \in \mathbb{R}^{n \times J \times K}$ are collected in a matrix ($\mathbf{Y}$). The matrix $\mathbf{Y}$ is viewed as arising from a dictionary learning model.

algorithm, the alternating minimization-based techniques became widely popular in practice, however lacked theoretical guarantees. On the provable algorithms front, starting with Spielman et al. (2012), who propose an algorithm for the undercomplete setting, Agarwal et al. (2014); Arora et al. (2014); Barak et al. (2015) developed provable algorithms for the overcomplete case, however their computational complexity and stringent initialization requirements limited their use. This motivated Arora et al. (2015) to develop a scalable online dictionary learning algorithm with a more relaxed initialization requirement. Following these, instead of only focusing on dictionary recovery (like the techniques described above), in Chapter 2 we proposed a simple gradient descent-based algorithm for joint estimation of the dictionary and the coefficients, which resulted in *exact* recovery of both factors at a linear rate.

Furthermore, tensor factorization algorithms have also been used to learn orthogonal (Barak et al. (2015) and Ma et al. (2016)), and convolutional (Huang and Anandkumar, 2015) dictionaries. In this work, we complete the circle by accomplishing tensor factorization via dictionary learning for exact recovery of the constituent factors (up to scaling and permutation) of a structured tensor.

Notation. See Appendix 3.A for details.

3.3 Problem Formulation

Our aim is to recover the CP factors of a tensor $\underline{\mathbf{Z}}$ assumed to be generated via the model shown in (3.1). Without loss of generality, let the factor $\mathbf{A}^*$ follow some incoherence assumptions, while the factors $\mathbf{B}^*$ and $\mathbf{C}^*$ are sparse. Then, the *mode*-1 *unfolding* $\mathbf{Z}_1 \in \mathbb{R}^{JK \times n}$ of $\underline{\mathbf{Z}}$ is given by

$$\mathbf{Z}_1^\top = \mathbf{A}^*(\mathbf{C}^* \odot \mathbf{B}^*)^\top = \mathbf{A}^*\mathbf{S}^*, \tag{3.3}$$

where $\mathbf{S}^* \in \mathbb{R}^{m\times JK}$ is defined as $\mathbf{S}^* := (\mathbf{C}^*\odot\mathbf{B}^*)^\top$. As a result, the matrix $\mathbf{S}^*$ has a *transposed Khatri-Rao* structure. In other words, the i-th row of $\mathbf{S}^*$ is given by $(\mathbf{C}_i^* \otimes \mathbf{B}_i^*)^\top$. Further, since $\mathbf{B}^*$ and $\mathbf{C}^*$ are sparse, only a few columns of the matrix $\mathbf{S}^*$ (say p) have non-zero elements. Now, letting $\mathbf{Y} \in \mathbb{R}^{n\times p}$ be a matrix formed by collecting the non-zero columns of $\mathbf{Z}_1^\top$, then we have

$$\mathbf{Y} = \mathbf{A}^*\mathbf{X}^*, \tag{3.4}$$

where $\mathbf{X}^* \in \mathbb{R}^{m\times p}$ denotes the sparse matrix corresponding to the non-zero columns of $\mathbf{S}^*$. This now resembles the dictionary learning task shown in (3.4). As a result, we can employ any provable dictionary learning algorithm (such as the proposed in Chapter 2) to recover the dictionary factor $\mathbf{A}^*$ and the sparse coefficients $\mathbf{X}^*$ at each time step t of the (online) algorithm. Now, we also develop an algorithm to untangle the estimates of sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$ from this $\mathbf{X}^*$ estimate. The overall modelling procedure is summarized in Fig. 3.2.

3.4 Algorithm

Our algorithm, `TensorNOODL` (shown in Algorithm 2) is motivated from the dictionary learning algorithm presented in Chapter 2, and operates by posing the tensor decomposition problem as a matrix factorization task. The input to the algorithm is an $(\epsilon_0, 2)$-close estimate $\mathbf{A}^{(0)}$ of $\mathbf{A}^*$ for $\epsilon_0 = \mathcal{O}^*(1/\log(n))$, where (ϵ,κ)-closeness is defined as follows.

Definition 3.1 ((ϵ,κ)-closeness)**.** *A matrix* $\mathbf{A}$ *is* (ϵ,κ)*-close to* $\mathbf{A}^*$ *if* $\|\mathbf{A}-\mathbf{A}^*\| \leq \kappa\|\mathbf{A}^*\|$, *and if there is a permutation* $\pi : [m] \rightarrow [m]$ *and a collection of signs* $\sigma : [m] \rightarrow \{\pm 1\}$ *such that* $\|\sigma(i)\mathbf{A}_{\pi(i)} - \mathbf{A}_i^*\| \leq \epsilon,\ \forall\ i \in [m]$.

This initialization (which can be achieved by algorithms such as Arora et al. (2015)) ensures that the estimate $\mathbf{A}^{(0)}$ is both, column-wise and in spectral norm sense, close to $\mathbf{A}^*$. Then a *fresh* tensor $\underline{\mathbf{Z}}$ (generated independently as per (3.1)), arrives at the t-th iteration of the algorithm, which then proceeds in three phases as described below.

Coefficient estimation – This stage uses R iterative hard thresholding (IHT) steps (3.6) – with step-size $\eta_x^{(r)}$ and threshold $\tau^{(r)}$ [2] chosen according to assumption **A.6** – to arrive at an estimate $\widehat{\mathbf{X}}^{(t)}$ (or $\mathbf{X}^{(R)}$). Here, the number of iterations R are determined based

[2] $(.)^{(r)}$ denotes that the parameters can change with iterations.

Algorithm 2: TensorNOODL: Neurally plausible alternating Optimization-based Online Dictionary Learning for Tensor decompositions.

```
Input: Structured tensor $\underline{\mathbf{Z}} \in \mathbb{R}^{n \times J \times K}$ at each iteration $t$ generated as per (3.1).
       Parameters $\eta_A$, $\eta_x$, $\tau$, $T$, and $R$ chosen as per A.5 and A.6 for the
       dictionary learning step.
Output: The dictionary $\mathbf{A}^{(t)}$ and the factor estimates $\mathbf{B}^{(t)}$ and $\mathbf{C}^{(t)}$ (corresponding
        to the current tensor $\underline{\mathbf{Z}}$) at each iterate $t$.
Initialize: Estimate $\mathbf{A}^{(0)}$, which is $(\epsilon_0, 2)$-near to $\mathbf{A}^*$ for $\epsilon_0 = \mathcal{O}^*(1/\log(n))$
for $t = 0$ to $T-1$ do
    Predict: (Estimate Coefficients)
    Initialize: $\mathbf{X}^{(0)} = \mathcal{T}_{C/2}(\mathbf{A}^{(t)^\top}\mathbf{Y})$                          (3.5)
    for $r = 0$ to $R-1$ do
        Update: $\mathbf{X}^{(r+1)} = \mathcal{T}_{\tau^{(r)}}(\mathbf{X}^{(r)} - \eta_x^{(r)}\,\mathbf{A}^{(t)^\top}(\mathbf{A}^{(t)}\mathbf{X}^{(r)} - \mathbf{Y}))$     (3.6)
    end
    $\widehat{\mathbf{X}}(\widehat{\mathbf{X}}^{(t)}) := \mathbf{X}^{(R)}$ (We drop $(.)^{(t)}$ in our discussion for simplicity.)
    Learn: (Update Dictionary)
      Form empirical gradient estimate: $\widehat{\mathbf{g}}^{(t)} = \frac{1}{p}(\mathbf{A}^{(t)}\widehat{\mathbf{X}}^{(t)}_{\text{indep}} - \mathbf{Y})\text{sign}(\widehat{\mathbf{X}}^{(t)}_{\text{indep}})^\top$
                                                                                  (3.7)
      Take a gradient descent step: $\mathbf{A}^{(t+1)} = \mathbf{A}^{(t)} - \eta_A\,\widehat{\mathbf{g}}^{(t)}$                    (3.8)
      Normalize: $\mathbf{A}_i^{(t+1)} = \mathbf{A}_i^{(t+1)}/\|\mathbf{A}_i^{(t+1)}\|\ \forall\ i \in [m]$
    Recover Sparse Factors
      Form $\widehat{\mathbf{S}}$ by putting back columns of $\widehat{\mathbf{X}}^{(t)}$ at the non-zero column locations
    of $\mathbf{Z}_1^\top$.    $[\widehat{\mathbf{B}}, \widehat{\mathbf{C}}] = $ UNTANGLE-KRP$(\widehat{\mathbf{S}})$
end
```

Algorithm 3: Untangle Khatri-Rao Product (KRP): Recovering the Sparse factors

```
Input: Estimated KRP, $\widehat{\mathbf{S}}$
Output: Estimates $\widehat{\mathbf{B}}$ and $\widehat{\mathbf{C}}$ of matrices $\mathbf{B}^*$ and $\mathbf{C}^*$ up to scaling.
for $i = 1 \ldots m$ do
    Reshape: the $i$-th row of $\widehat{\mathbf{S}}$ into $\mathbf{M}^{(i)} \in \mathbb{R}^{J \times K}$.
    Set: $\widehat{\mathbf{B}}_i \leftarrow \sqrt{\sigma_1}\mathbf{u}_1$, and $\widehat{\mathbf{C}}_i \leftarrow \sqrt{\sigma_1}\mathbf{v}_1$, where $\sigma_1$, $\mathbf{u}_1$, and $\mathbf{v}_1$ denote the largest
    singular value and the corresponding left and right singular vectors of $\mathbf{M}^{(i)}$,
    respectively.
end
```

on the target tolerance of the desired coefficient estimate, specifically we choose $R = \Omega(\log(1/\delta_R))$, where $(1-\eta_x^{(r)})^R \leq \delta_R$.

Dictionary update – The coefficient estimates are used to update the dictionary using an approximate gradient descent strategy (3.8) with step size η_A chosen according to assumption **A.5**. The algorithm requires $T = \max(\Omega(\log(1/\epsilon_T)), \Omega(\log(\sqrt{s}/\delta_T)))$ iterations to achieve $\|\mathbf{A}_i^{(T)} - \mathbf{A}_i^*\| \leq \epsilon_T, \forall i \in [m]$ and $|\widehat{\mathbf{X}}_{ij}^{(T)} - \mathbf{X}_{ij}^*| \leq \delta_T$; see Chapter 2 and Rambhatla et al. (2019).

Untangling the Khatri-Rao Structure – As shown in (3.4), since we only operate on the non-zero fibers of $\mathbf{Z}_1^\top$, i.e., `TensorNOODL` is agnostic to the tensor structure of the data. Therefore, we need a separate procedure to estimates the sparse factors ($\mathbf{B}^*$ and $\mathbf{C}^*$) using $\widehat{\mathbf{X}}$. To this end, we first form the estimate $\widehat{\mathbf{S}}$ of $\mathbf{S}^*$ by placing columns of $\widehat{\mathbf{X}}$ at their corresponding locations of $\mathbf{Z}_1^\top$. Next, we use a SVD-based algorithm (Algorithm 3) to recover the sparse factors (up to scaling and permutation) using the element-wise ζ-close estimate of $\mathbf{S}^*$, i.e., $|\widehat{\mathbf{S}}_{ij} - \mathbf{S}_{ij}^*| \leq \zeta$ predicted by Algorithm 2.

3.5 Main Result

In this section we formalize our model assumptions and state our main result; detailed analysis is in Appendix 3.B. We begin with the notion of incoherence required for $\mathbf{A}^*$ (henceforth refered to as *dictionary*) columns. Specifically, we require that $\mathbf{A}^*$ is μ-incoherent, defined as follows.

Definition 3.2. *A matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ with unit-norm columns is μ-incoherent if for all $i \neq j$ the inner-product between the columns of the matrix follow $|\langle \mathbf{A}_i, \mathbf{A}_j \rangle| \leq \mu/\sqrt{n}$.*

This ensures that the atoms of the dictionary can be distinguished from each other, and can be viewed as a relaxed orthogonality constraint. Next, we assume that the sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$ are drawn from distribution classes $\Gamma_{\alpha,C}^{\mathrm{sG}}$ and $\Gamma_{\beta}^{\mathrm{Rad}}$, respectively, where $\Gamma_{\gamma,C}^{\mathrm{sG}}$ and $\Gamma_{\gamma}^{\mathrm{Rad}}$ are defined as follows.

Definition 3.3 (Distribution Class $\Gamma_{\gamma,C}^{\mathrm{sG}}$ and $\Gamma_{\gamma}^{\mathrm{Rad}}$)**.** A matrix $\mathbf{M}$ belongs to a

- Distribution class $\Gamma_{\gamma,C}^{\mathrm{sG}}$: if each entry of $\mathbf{M}$ is independently non-zero with probability γ, and the values at the non-zero locations are sub-Gaussian, zero-mean with unit variance and bounded away from C for some positive constant $C \leq 1$, i.e., $|\mathbf{M}_{ij}| \geq C$ for $(i,j) \in \mathrm{supp}(\mathbf{M})$.
- Distribution class $\Gamma_{\gamma}^{\mathrm{Rad}}$: if each entry of $\mathbf{M}$ is independently non-zero with probability γ, and the values at the non-zero locations are drawn from the Rademacher distribution.

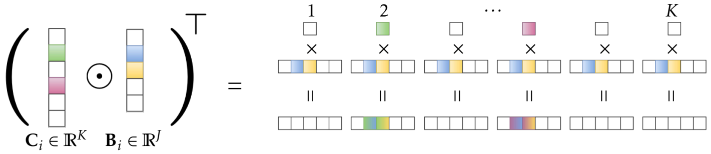

1
2
···
K
$\mathbf{C}_i \in \mathbb{R}^K$
$\mathbf{B}_i \in \mathbb{R}^J$

Figure 3.1: Dependence induced by the transposed Khatri-Rao structure.

In essence, we assume that elements of $\mathbf{B}^*$ ($\mathbf{C}^*$) are non-zero with probability α (β), and that for $\mathbf{B}^*$ the values at the non-zero locations are drawn from a zero-mean unit-variance sub-Gaussian disribution, bounded away from zero and the non-zero values of $\mathbf{C}^*$ are drawn from the Rademacher distribution. This assumption ensures that the resulting sparse coefficient matrix $\mathbf{X}^*$ obeys the distributional assumptions required for the success of the dictionary learning task[3].

We now turn our attention to the implications of the dependence structure of $\mathbf{S}^*$. Fig. 3.1 shows a row of the matrix $\mathbf{S}^*$, each entry of which is formed by multiplication of an element of $\mathbf{C}^*_i$ with each element of columns of $\mathbf{B}^*_i$. As a result, each row of the resulting matrix $\mathbf{S}^*$ has K *blocks* (of size J), where the k-th block is controlled by $\mathbf{C}^*_{k,i}$. Therefore, the (i,j)-th entry of transposed Khatri-Rao structured matrix $\mathbf{S}^*$ can be written as

$$\mathbf{S}^*_{ij} = \mathbf{C}^*(\left\lfloor \tfrac{j}{J} \right\rfloor + 1, i)\mathbf{B}^*(j - J\left\lfloor \tfrac{j}{J} \right\rfloor, i).$$

As a result, depending upon the sparsity of $\mathbf{B}^*$ and $\mathbf{C}^*$, $\mathbf{S}^*$ may have all-zero (degenerate) columns, resulting in degenerate columns in $\mathbf{Z}_1^\top$. Therefore, we only use $\mathbf{Y}$, the non-zero columns of $\mathbf{Z}_1^\top$.

We also observe that although elements in a column of $\mathbf{S}^*$ are independent of each other, the Khatri-Rao structure induces a dependence structure across a row when the elements depend on the same value of $\mathbf{B}^*$ or $\mathbf{C}^*$. This dependence may not be an issue in practice, where we can use all non-zero columns of $\mathbf{Z}_1^\top$. However for the analysis, it is necessary to select a group of independent data samples from $\mathbf{Z}_1^\top$. One way to choose an independent set of samples is by collecting the first element from the first block, second element from the second block and so on, to ensure that the samples are

[3] The non-zero entries of $\mathbf{C}^*$ can also be assumed to be drawn from a sub-Gaussian distribution (like those of $\mathbf{B}^*$) at the expense of sparsity, incoherence, dimension(s), and sample complexity. Specifically when non-zero entries of $\mathbf{B}^*$ and $\mathbf{C}^*$ are drawn from sub-Gaussian distribution (as per $\Gamma^{\text{sG}}_{\gamma,C}$), we will need the dictionary learning algorithm to work with the coefficient matrix $\mathbf{X}^*$ (formed by product of entries of $\mathbf{B}^*$ and $\mathbf{C}^*$) which will now have sub-Exponential distributed non-zero entries. Since sub-Exponential tails decay slower than those of sub-Gaussians, we will need additional restrictions on other parameters.

not formed from the same element from either of the sparse factors. This results in a size $L = \min(J, K)$ independent samples for a given $\mathbf{Z}_1^\top$. Overall, these assumptions on factors $\mathbf{B}^*$ and $\mathbf{C}^*$ mean that the L independent columns of $\mathbf{X}^*$ ($\mathbf{X}^*_{\text{indep}}$) belong to the distribution class $\mathcal{D}$ defined as follows; see also Chapter 2 and Rambhatla et al. (2019).

Definition 3.4 (Distribution class $\mathcal{D}$)**.** *The coefficient vector* $\mathbf{x}^*$ *belongs to an unknown distribution* $\mathcal{D}$*, where the support* $S = \text{supp}(\mathbf{x}^*)$ *is at most of size* s*,* $\mathbf{Pr}[i \in S] = \Theta(s/m)$ *and* $\mathbf{Pr}[i, j \in S] = \Theta(s^2/m^2)$*. Moreover, the distribution is normalized such that* $\mathbf{E}[\mathbf{x}_i^*|i \in S] = 0$ *and* $\mathbf{E}[\mathbf{x}_i^{*^2}|i \in S] = 1$*, and when* $i \in S$*,* $|\mathbf{x}_i^*| \geq C$ *for some constant* $C \leq 1$*. In addition, the non-zero entries are sub-Gaussian and pairwise independent conditioned on the support.*

Further, the $(\epsilon_0, 2)$-closeness (Def. 3.1) ensures that the signed-support of the coefficients are recovered correctly (with high probability), where signed-support is defined as follows.

Definition 3.5. *The signed-support of a vector* $\mathbf{x}$ *is defined as* $\text{sign}(\mathbf{x}) \cdot \text{supp}(\mathbf{x})$*.*

Moreover, the unit-norm constraint on $\mathbf{A}^*$ implies that the scaling (including the sign) ambiguity only exists in the recovery of the factors $\mathbf{B}^*$ and $\mathbf{C}^*$. To this end, we will regard our algorithm to be successful as long as it recovers factors in the following sense.

Definition 3.6 (Sparse factor scaling indeterminacy)**.** *Factorizations* $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ *are considered equivalent up to scaling, i.e,* $[\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\mathbf{A}^*, \mathbf{B}^*\mathbf{D}_{\sigma_b}, \mathbf{C}^*\mathbf{D}_{\sigma_c}]\!]$ *where* $\sigma_b(\sigma_c)$ *is a vector of scalings (including signs) corresponding to each column of the factors* $\mathbf{B}$ *and* $\mathbf{C}$*, respectively.*

We employ an *approximate* (since $\mathbf{X}^*$ is not known exactly) gradient descent-based strategy (3.7) to update $\mathbf{A}^{(t)}$ by finding an appropriate direction $\mathbf{g}_i^{(t)}$ to ensure descent. We show that $(\Omega(s/m), \Omega(m/s), 0)$-correlatedness (defined below) of the expected gradient vector allows us to make progress at every iteration, where "0" indicates that we do not incurr any bias. This can be viewed as a local descent condition which leads to the true dictionary columns; see also Candès et al. (2015); Chen and Wainwright (2015b); Arora et al. (2015); Rambhatla et al. (2019).

Definition 3.7. *A vector* $\mathbf{g}_i^{(t)}$ *is* $(\rho_-, \rho_+, \zeta_t)$*-correlated with a vector* $\mathbf{z}^*$ *if*

$$\langle \mathbf{g}_i^{(t)}, \mathbf{z}^{(t)} - \mathbf{z}^* \rangle \geq \rho_- \|\mathbf{z}^{(t)} - \mathbf{z}^*\|^2 + \rho_+ \|\mathbf{g}_i^{(t)}\|^2 - \zeta_t.$$

Overall, our model assumptions can be formalized as follows, with which we state our main result.

A.1 $\mathbf{A}^*$ is μ-incoherent (Def. 3.2), where $\mu = \mathcal{O}(\log(n))$, $\|\mathbf{A}^*\| = \mathcal{O}(\sqrt{m/n})$ and $m = \mathcal{O}(n)$;

A.2 $\mathbf{A}^{(0)}$ is $(\epsilon_0, 2)$-close to $\mathbf{A}^*$ as per Def. 3.1, and $\epsilon_0 = \mathcal{O}^*(1/\log(n))$;

A.3 Sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$ are drawn from distribution class $\Gamma^{\text{sG}}_{\alpha,C}$ and $\Gamma^{\text{Rad}}_{\beta}$, respectively (Def. 3.3);

A.4 The sparsity controlling parameters α and β obey $\alpha\beta = \mathcal{O}(\sqrt{n}/m\mu\ \log(n))$ for $m = \Omega(\log(\min(J,K))/\alpha\beta)$, and the resulting column sparsity s of $\mathbf{S}^*$ is $s = \mathcal{O}(\alpha\beta m)$;

A.5 The step-size for dictionary update satisfies $\eta_A = \Theta(m/s)$;

A.6 The step-size and threshold for coefficient estimation satisfies $\eta_x^{(r)} < c_1(\epsilon_t, \mu, n, s) = \widetilde{\Omega}(s/\sqrt{n}) < 1$ and $\tau^{(r)} = c_2(\epsilon_t, \mu, s, n) = \widetilde{\Omega}(s^2/n)$ for small constants c_1 and c_2.

Theorem 3.1 (Main Result). *Suppose a tensor* $\underline{\mathbf{Z}} \in \mathbb{R}^{n\times J\times K}$ *provided to Algorithm 2 at each iteration t admits a decomposition of the form (3.1) with factors* $\mathbf{A}^* \in \mathbb{R}^{n\times m}$, $\mathbf{B}^* \in \mathbb{R}^{J\times m}$ *and* $\mathbf{C}^* \in \mathbb{R}^{K\times m}$ *and* $\min(J,K) = \Omega(ms^2)$. *Further, suppose that the assumptions **A.1-A.6** hold. Then, given* $R = \Omega(\log(n))$, *with probability at least* $(1-\delta_{alg})$ *for some small constant* δ_{alg}, *the coefficient estimate* $\widehat{\mathbf{X}}^{(t)}$ *at t-th iteration has the correct signed-support and satisfies*

$$(\widehat{\mathbf{X}}^{(t)}_{i,j} - \mathbf{X}^*_{i,j})^2 \le \zeta^2 := \mathcal{O}(s(1-\omega)^{t/2}\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|), \textit{ for all } (i,j) \in \operatorname{supp}(\mathbf{X}^*).$$

Furthermore, for some $0 < \omega < 1/2$, *the estimate* $\mathbf{A}^{(t)}$ *at t-th iteration satisfies*

$$\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 \le (1-\omega)^t\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|^2, \textit{ for all } t = 1, 2, \ldots.$$

Consequently, Algorithm 3 recovers the supports of the sparse factors $\mathbf{B}^*$ *and* $\mathbf{C}^*$ *correctly, and* $\|\widehat{\mathbf{B}}_i - \mathbf{B}_i^*\|_2 \le \epsilon_B$ *and* $\|\widehat{\mathbf{C}}_i - \mathbf{C}_i^*\|_2 \le \epsilon_C$, *where* $\epsilon_B = \epsilon_C = \mathcal{O}(\frac{\zeta^2}{\alpha\beta})$.

In other words, Theorem 3.1 states that for Algorithm 2 to succeed, the columns of the incoherent factor $\mathbf{A}^*$ are sufficiently spread out ensuring identifiability (**A.1**), and that the sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$ are sparse (**A.3**), but not *too sparse* (**A.4**). The lower-bound on rank/sparsity ($m = \Omega(\min(J,K)/\alpha\beta)$) ensures that there are adequate number of samples available for the learning procedure for a given rank m. As a result, higher the rank, more the data samples required at every iteration (**A.4**). This effect is captured by the $\min(J,K) = \Omega(ms^2)$ requirement, wherein $s = \mathcal{O}(\alpha\beta m)$ with high probability. This lower bound is due to the samples complexity of (Rambhatla et al., 2019); the proposed algorithm also works for smaller (J,K) in practice. The algorithm also relies on an appropriate initialization of the dictionary factor ($\mathbf{A}^{(0)}$), which can be achieved by techniques such as Arora et al. (2015) (**A.2**). Such algorithms can also be used for model selection, i.e., determining m. Furthermore, the our main result also characterizes the conditions on learning parameters (step sizes and threshold **A.5**∼**A.6**) for the success of the algorithm.

Leveraging the dictionary learning view for this tensor factorization task, `Tensor NOODL` is, to the best of our knowledge, the first algorithm to provably factorize such a 3-way structured tensor [4]. The analysis presented here can be potentially extended to higher order tensors with similar structure, however, will entail further careful analysis to entangle the sparse factors effectively.

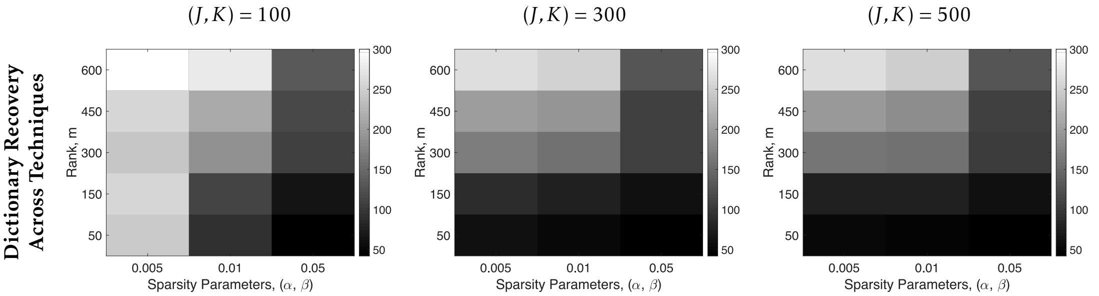
3.6 Numerical Simulations

We now evaluate the performance of our algorithm for a structured tensor factorization task[5]. Note that, the provable tensor factorization algorithms shown in Table 3.1, i.e., are not suitable for handling the incoherence along with the structured sparsity of the factors, and are suitable only for cases wherein all the factors obey the same structural assumptions (incoherence). In addition, as discussed in Section 3.2.3, Anandkumar et al. (2015) entails a computationally expensive algorithm, Sharan and Valiant (2017) works for undercomplete settings, and Sun et al. (2017) requires sparsity as well as incoherence. Therefore, we compare the performance of `TensorNOODL` with online dictionary learning algorithms presented in Arora et al. (2015) (`Arora(b)` and `Arora(u)`), and the algorithm proposed in Mairal et al. (2009), which can be viewed as a variant of ALS (with matricized view of the task) with convergence results (to a stationary point). In the experiments, we focus on the recovery of $\mathbf{X}^*$ (including support recovery) since the performance of Algorithm 3 (to recover $\mathbf{B}^*$ and $\mathbf{C}^*$) solely depends on exact recovery of $\mathbf{X}^*$. We fix the random seeds in trials for reproducibility. The setting, comparisons with techniques, and detailed results are in Appendix 3.E.

Discussion – `TensorNOODL` achieves orders of magnitude superior recovery as compared to competing techniques both for the recovery of $\mathbf{A}^*$, and $\mathbf{X}^*$. Furthermore, only `TensorNOODL` recovers the correct support of the sparse matrix $\mathbf{X}^*$, crucial for recovery of the sparse factors (Appendix 3.E). In Fig. 3.1, we analyze the number of samples

[4] Note that the existing provable techniques shown in Table 3.1 (such as Sharan and Valiant (2017) and Anandkumar et al. (2015)) are not suitable for handling the incoherence along with the structured sparsity of the factors, and are suitable only for cases wherein all the factors obey the same structural assumptions (incoherence).

[5] `https://github.com/srambhatla/TensorNOODL`; see Chapter 7 for details.



(a) (b) (c)

Figure 3.1: Data samples required by `TensorNOODL` using the number of iterations for convergence (see footnote 6). Panels (a), (b), and (c) show the number of iterations taken by `TensorNOODL` to achieve a target tolerance of 10^{-10} for $\mathbf{A}$ for $J = K = 100$, 300, and 500, respectively across the choices of rank $m = \{50,\ 150,\ 300,\ 450,\ 600\}$ and $\alpha = \beta = \{0.005,\ 0.01,\ 0.05\}$, averaged across three Monte Carlo runs.

required across different choices of the dimension (J, K), rank (m) and sparsity parameters (α, β) averaged across the Monte Carlo runs using the number of iterations T[6]. We observe three interesting trends, also predicted by our analysis. First, in each panel the number of iterations (to achieve target tolerance) decrease as we move from left to right. This is due to increase in data samples with increasing (α, β). Second, looking across panels for a fixed rank and sparsity parameters, the number of iterations decreases with increasing (J, K), also attributed to the increase in available data samples. Finally, we note that as the rank increases, `TensorNOODL` requires more samples, as pointed by our sample complexity requirement. It is worth noting that although we consider the case of fixed (J, K) and (α, β), `TensorNOODL` can also be used when these parameters vary at each iteration. This feature can be especially useful in real-world applications, where the dimensions and sparsity of the tensor may change over iterations. In addition, since $\mathbf{X}^*$ columns can be estimated independently, and further since we only use the non-zero fibers, `TensorNOODL` can be implemented in highly distributed settings.

3.7 Discussion and Conclusions

We propose, to the best of out knowledge, the first algorithm for the exact recovery of CP factors of a structured tensor (an inherently non-convex optimization task) at a

[6] Since our algorithm takes a fresh tensor as an input at each iteration t of the algorithm, the number of iterations T to achieve the target tolerance can be viewed as a surrogate for the sample requirement.

linear rate, where the columns of one of the tensor factors obeys some incoherence assumption, while the other two factors are sparse. Here, we cast the tensor factorization problem as a dictionary learning task to develop a provable algorithm to recover these factors (up to permutation and scaling indeterminacies). The Kronecker-dependence structure induced by the matricization makes a few data samples unusable (since the algorithm requires independent samples for learning). Although, not an issue in practice, this lead to somewhat conservative sample complexity results in theory. Relaxing these requirements, extending the analysis to noisy settings, and using `TensorNOODL` in other structured tensor factorization tasks, are all promising future directions.

Appendices: Provable Structured Tensor Factorization via Dictionary Learning

We summarize the notation used in our work in Appendix 3.A, including with a list of frequently used symbols and their corresponding definitions. Next, in Appendix 3.B, we present the proof of our main result, and organize the the proofs of intermediate results in Appendix 3.C. Additional results are listed in Appendix 3.D. Furthermore, we show the detailed experimental results in Appendix 3.E, along with how to reproduce the numerical results of this work. We also provide the code used to compile these results, with specific recommendation on the parameter setting.

3.A Summary of Notation

We summarizes the definitions of some frequently used symbols in our analysis in Table 3.A.1 and 3.A.2. Also note that, since we show that $\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \le \epsilon_t$ contracts in every step, therefore we fix $\epsilon_t, \epsilon_0 = \mathcal{O}^*(1/\log(n))$ in our analysis.

3.B Proof of Theorem 1

In this section, we present the details of the analysis pertaining to our main result (shown below).

Theorem 3.1 **[Main Result]** *Suppose a tensor $\underline{\mathbf{Z}} \in \mathbb{R}^{n \times J \times K}$ provided to Algorithm 2 at each iteration t admits a decomposition of the form (3.1) with factors $\mathbf{A}^* \in \mathbb{R}^{n \times m}$, $\mathbf{B}^* \in \mathbb{R}^{J \times m}$*

Table 3.A.1: Frequently used symbols: Definitions of Probabilities

Probabilities

Symbol	Definition	Symbol	Definition
γ	$\gamma := \alpha\beta$	$\delta_{\mathbf{B}_i}^{(t)}$	$\delta_{\mathbf{B}_i}^{(t)} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ for any $\epsilon > 0$.
$\delta_{\mathcal{T}}^{(t)}$	$\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$	$\delta_{\beta}^{(t)}$	$2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$
$\delta_{\mathrm{s}}^{(t)}$	$\delta_s^{(t)} = \min(J,K)\exp(-\epsilon^2\alpha\beta m/2(1+\epsilon/3))$ for any $\epsilon > 0$	$\delta_p^{(t)}$	$\delta_p^{(t)} = \exp(-\frac{\epsilon^2}{2}L(1-(1-\gamma)^m))$
$\delta_{\mathrm{IHT}}^{(t)}$	$\delta_{\mathrm{IHT}}^{(t)} = \delta_{\mathcal{T}}^{(t)} + \delta_{\beta}^{(t)}$	$\delta_{\mathrm{NOODL}}^{(t)}$	$\delta_{\mathrm{NOODL}}^{(t)} = \delta_{\mathcal{T}}^{(t)} + \delta_{\beta}^{(t)} + \delta_{\mathrm{HW}} + \delta_{\mathbf{g}_i}^{(t)} + \delta_{\mathbf{g}}^{(t)}$
q_i	$q_i = \mathbf{Pr}[i \in S] = \Theta(\frac{s}{m})$	$q_{i,j}$	$q_{i,j} = \mathbf{Pr}[i,j \in S] = \Theta(\frac{s^2}{m^2})$
p_i	$p_i = \mathbf{E}[\mathbf{X}_{ij}^*\mathrm{sign}(\mathbf{X}_{ij}^*)\vert\mathbf{X}_{ij}^* \neq 0]$	$\delta_{\mathrm{HW}}^{(t)}$	$\delta_{\mathrm{HW}}^{(t)} = \exp(-1/\mathcal{O}(\epsilon_t))$
$\delta_{\mathbf{g}_i}^{(t)}$	$\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(s))$	$\delta_{\mathbf{g}}^{(t)}$	$\delta_{\mathbf{g}}^{(t)} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$

and $\mathbf{C}^* \in \mathbb{R}^{K\times m}$ *and* $\min(J,K) = \Omega(ms^2)$. *Further, suppose that the assumptions* ***A.1-A.6*** *hold. Then, given* $R = \Omega(\log(n))$*, with probability at least* $(1-\delta_{alg})$ *for some small constant* δ_{alg}*, the coefficient estimate* $\widehat{\mathbf{X}}^{(t)}$ *at t-th iteration has the correct signed-support and satisfies*

$$(\widehat{\mathbf{X}}_{i,j}^{(t)} - \mathbf{X}_{i,j}^*)^2 \leq \zeta^2 := \mathcal{O}(s(1-\omega)^{t/2}\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|), \textit{ for all } (i,j) \in \mathrm{supp}(\mathbf{X}^*).$$

Furthermore, for some $0 < \omega < 1/2$*, the estimate* $\mathbf{A}^{(t)}$ *at t-th iteration satisfies*

$$\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\|^2 \leq (1-\omega)^t\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|^2, \textit{ for all } t = 1,2,\ldots.$$

Consequently, Algorithm 3 recovers the supports of the sparse factors $\mathbf{B}$ *and* $\mathbf{C}$ *correctly, and* $\|\widehat{\mathbf{B}}_i - \mathbf{B}_i\|_2 \leq \epsilon_B$ *and* $\|\widehat{\mathbf{C}}_i - \mathbf{C}_i\|_2 \leq \epsilon_C$*, where* $\epsilon_B = \epsilon_C = \mathcal{O}(\frac{\zeta^2}{\alpha\beta})$.

Here, $\delta_{alg} = \delta_{\mathrm{s}} + \delta_p^{(t)} + \delta_{\mathbf{B}_i}^{(t)} + \delta_{\mathrm{NOODL}}$. *Further,* $\delta_{\mathrm{NOODL}}^{(t)} = \delta_{\mathcal{T}}^{(t)} + \delta_{\beta}^{(t)} + \delta_{\mathrm{HW}} + \delta_{\mathbf{g}_i}^{(t)} + \delta_{\mathbf{g}}^{(t)}$*, where* $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-C^2/\mathcal{O}^*(\epsilon_t^2))$*,* $\delta_{\beta}^{(t)} = 2s\exp(-1/\mathcal{O}(\epsilon_t))$*,* $\delta_{\mathrm{HW}}^{(t)} = \exp(-1/\mathcal{O}(\epsilon_t))$*,* $\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(s))$*,* $\delta_{\mathbf{g}}^{(t)} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$*. Furthermore,* $\delta_s^{(t)} = \min(J,K)\exp(-\frac{\epsilon^2\alpha\beta m}{2(1+\epsilon/3)})$ *for any* $\epsilon > 0$*,* $\delta_p^{(t)} = \exp(-\frac{\epsilon^2}{2}L(1-(1-\gamma)^m))$*, and* $\delta_{\mathbf{B}_i}^{(t)} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ *for any* $\epsilon > 0$*. Also,* $\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \leq \epsilon_t$.

Proof. **of Theorem 3.1** As alluded to in our discussion, we achieve tensor factorization by means of dictionary learning. In particular, our algorithm is based on the recent provable dictionary learning algorithm proposed in Rambhatla et al. (2019). It is worth

Table 3.A.2: Frequently used symbols: Notation and Parameters

Symbol	Definition	Symbol	Definition
$(\cdot)^*$	Used to represent the ground-truth matrix.	$(\cdot)^{(t)}$, $\widehat{(\cdot)}^{(t)}$, and $\widehat{(\cdot)}$	Used to represent the estimates formed by the algorithm.
$(\cdot)^{(t)}$	The subscript t is used to represent the estimates at each iteration of the online algorithm.	$(\cdot)^{(r)}$	The subscript r is used to represent the IHT iterates.
$\mathbf{A}_i^{(t)}$	i-th column of $\mathbf{A}^*$ estimate at the t-th iterate.	$\widehat{\mathbf{B}}$ $(\widehat{\mathbf{C}})$	Estimate of $\mathbf{B}^*$ $(\mathbf{C}^*)$ at an iteration of the online algorithm.
$\mathbf{S}^*$	Transposed Khatri-Rao structured (sparse) matrix, $\mathbf{S}^* = (\mathbf{C}^* \odot \mathbf{B}^*)^\top$, its i-th row is given by $\mathbf{C}_i^* \otimes \mathbf{B}_i^*$.	$\mathbf{X}^*$	Sparse matrix formed by collecting non-zero columns of $\mathbf{S}^*$.
p	Number of columns in $\mathbf{X}^*$.	$\mathbf{Z}_1^\top$	Mode-1 unfolding of $\underline{\mathbf{Z}}$, $\mathbf{Z}_1^\top = \mathbf{A}^*(\mathbf{C}^* \odot \mathbf{B}^*)^\top$.
ϵ_t	Upper-bound on column-wise error at the t-th iterate.$\\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\\| \leq \epsilon_t = \mathcal{O}^*(\frac{1}{\log(n)})$.	ϵ_B	Upper-bound on column-wise ℓ_2-error in the estimate $\widehat{\mathbf{B}}$ at each iteration t, $\\|\widehat{\mathbf{B}}_i - \mathbf{B}_i^*\\| \leq \epsilon_B = \mathcal{O}(\frac{\xi^2}{\alpha\beta})$.
ϵ_C	Upper-bound on column-wise ℓ_2-error in the estimate $\widehat{\mathbf{C}}$ at each iteration t, $\\|\widehat{\mathbf{C}}_i - \mathbf{C}_i^*\\| \leq \epsilon_C = \mathcal{O}(\frac{\xi^2}{\alpha\beta})$.	μ	The incoherence between the columns of the factor $\mathbf{A}^*$; see Def. 3.2.
μ_t	Incoherence between the columns of $\mathbf{A}^{(t)}$, $\frac{\mu_t}{\sqrt{n}} = \frac{\mu}{\sqrt{n}} + 2\epsilon_t$.	$\alpha(\beta)$	The probability that an element $\mathbf{B}_{ij}$ ($\mathbf{C}_{ij}$) of $\mathbf{B}$ $(\mathbf{C})$ is non-zero.
ξ	The element-wise upper bound on the error between $\widehat{\mathbf{S}}_{ij}$ and $\mathbf{S}_{ij}^*$, i.e., $\|\mathbf{S}_{ij}^* - \widehat{\mathbf{S}}_{ij}\| \leq \xi$.	s	The number of non-zeros in a column of $\mathbf{S}^*$.
R	The number of IHT iterations at each iteration t of the online algorithm.	T	Total number of online iterations.
δ_R	Decay parameter for each IHT stage, $\delta_R \geq (1-\eta_x)^R$.	δ_T	Element-wise target error tolerance for final estimate of $\mathbf{X}^*$, $\|\widehat{\mathbf{X}}_{ij}^{(T)} - \mathbf{X}_{ij}^*\| \leq \delta_T \forall i \in \text{supp}(\mathbf{X}^*)$.
C	Lower-bound on $\mathbf{x}_i^*$s, $\|\mathbf{X}_{ij}^*\| \geq C$ for $i \in \text{supp}(\mathbf{X}^*)$ and $C \leq 1$	L	$L := \min(J, K)$

noting that, our procedure itself is agnostic to the dictionary learning algorithm, and can be used with any dictionary learning algorithm which exactly recovers both factors of the dictionary learning model. In the following discussion we reinstantiate selective results from Chapter 2, and present the statement of the results here for completeness.

The alternating optimization-based algorithm for online dictionary learning proposed in Chapter 2 (and Rambhatla et al. (2019)) is based on jointly making progress on the dictionary and the coefficient estimates. Here, the authors propose a simple provable algorithm for this matrix factorization task, which recovers the dictionary and the coefficients exactly, at a linear rate, given an appropriate initial estimate of the dictionary.

The proof procedure relies three main steps – 1) Recovering the signed-support of the coefficient estimate, 2) expressing the error incurred by the coefficient estimate as being composed of a component that depends on the initial coefficient estimate (and ensuring that this becomes negligible given sufficient IHT-based coefficient update iterations), and a component that depends on the error in the dictionary, 3) showing that the dictionary makes progress at every iteration of the online algorithm, and maintains the conditions required for the next iteration of the online algorithm.

The authors show that these steps result in removal of the bias in dictionary estimation as compared to the prior art, in addition to providing simultaneous recovery of the sparse coefficient. The separability of the updates across the data samples and the simplicity of the algorithm makes it amenable for large scale distributed (neural) implementations. This coupled with the exact recovery guarantees for both factors makes it particularly suitable for the tensor factorization considered in this work; see Chapter 2 and Rambhatla et al. (2019) for detailed proofs.

In order to leverage these results, we need to get a handle on the sparsity (number of non-zeros in a column of $\mathbf{S}^*$), and characterize the number of usable (independent) data samples available to the algorithm. To this end, the following lemma characterizes the upper bound on the sparsity, k, the number of non-zeros in a column of $\mathbf{S}^*$.

Lemma 3.1. If $m = \Omega(\log(\min(J,K))/\alpha\beta)$ then with probability at least $(1-\delta_s^{(t)})$ the number of non-zeros s, in a column of $\mathbf{S}^*$ are upper-bounded as $s = \mathcal{O}(\alpha\beta m)$, where $\delta_s^{(t)} = \min(J,K)\exp(-\epsilon^2\alpha\beta m/2(1+\epsilon/3))$ for any $\epsilon > 0$.

In line with our intuition, the sparsity scales with the parameters α, β and m.

We now characterize the number of usable data samples available to the algorithm. For this, notice that the i-th row of $\mathbf{S}^*$ can be written as $(\mathbf{C}_i^* \otimes \mathbf{B}_i^*)^\top$. Now, since $\mathbf{B}^*$ and $\mathbf{C}^*$ are sparse, there are a number of columns in $\mathbf{S}^*$ which are degenerate (all-zeros). As a result, the corresponding data samples (columns of $\mathbf{Z}_1^\top$) are also degenerate, and cannot be used for learning. Furthermore, due to the dependence structure in $\mathbf{S}^*$ (discussed in Section 3.5) some of the data samples are dependent on each other, and at least from the theoretical perspective, are not eligible for the learning process. Therefore, we characterize the expected number of viable data samples in the following lemma.

Lemma 3.2. For $L = \min(J, K)$, $\gamma = \alpha\beta$, and any $\epsilon > 0$ and suppose we have

$$L \geq \tfrac{2}{(1-(1-\gamma)^m)\epsilon^2} \log(\tfrac{1}{\delta_p^{(t)}}),$$

then with probability at least $(1 - \delta_p)$,

$$p = L(1-(1-\gamma)^m),$$

where $\delta_p^{(t)} = \exp(-\frac{\epsilon^2}{2}L(1-(1-\gamma)^m))$.

Here, we observe that the number of viable samples increase with number of independent samples $L = \min(J, K)$, sparsity parameter $\gamma = \alpha\beta$, and rank of the decomposition m.

Using Lemma 3.1 and Lemma 3.2, we can now use Theorem 2.1 from Chapter 2 to show that online learning algorithm, which alternates between an Iterative Hard Thresholding (IHT) step to estimate the non-zero columns of $\mathbf{S}^*$, and approximate gradient descent-based update for the dictionary factor, recover $\mathbf{A}^*$ and $\mathbf{S}^*$ (or $\mathbf{X}^*$) exactly; see also Rambhatla et al. (2019).

Specifically, for recovery of the sparse matrix $\mathbf{S}^*$ (or $\mathbf{X}^*$), we leverage the Lemma 2.1 from Chapter 2 to show that at the t-th iteration of the online algorithm, the initial sparse coefficient estimate ($\mathbf{X}^{(0)}$) has the correct signed-support (see Def. 3.5) as $\mathbf{X}^*$ with probability $(1-\delta_{\mathcal{T}}^{(t)})$ given an $(\epsilon_0, 2)$-close estimate $\mathbf{A}^{(0)}$ of the the true factor $\mathbf{A}^*$, for $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$. To use this result, we arrive at the condition that $s = \mathcal{O}(\alpha\beta m) = \mathcal{O}^*\sqrt{n}/\mu\log(n)$, which leads us to assumption **A.4**.

Lemma 3.3 (from Lemma 2.1). **(Signed-support recovery)** Suppose $\mathbf{A}^{(t)}$ is ϵ_t-close to $\mathbf{A}^*$. Then, if $\mu = \mathcal{O}(\log(n))$, $s = \mathcal{O}^*(\sqrt{n}/\mu\log(n))$, and $\epsilon_t = \mathcal{O}^*(1/\sqrt{\log(m)})$, with probability at least $(1-\delta_{\mathcal{T}}^{(t)})$ for each random sample $\mathbf{y} = \mathbf{A}^*\mathbf{x}^*$:

$$\operatorname{sign}(\mathcal{T}_{C/2}((\mathbf{A}^{(t)})^\top\mathbf{y}) = \operatorname{sign}(\mathbf{x}^*),$$

where $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$.

Next, we use Lemma 3.4 to arrive at the conditions on the step size parameter $\eta_x^{(r)}$, and the threshold $\tau^{(r)}$, such that that the IHT-step preserves the correct signed-support with probability $\delta_{\rm IHT}^{(t)}$, for $\delta_{\rm IHT}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)}) + 2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.

Lemma 3.4 (from Lemma 2.2)**. (IHT update step preserves the correct signed-support**) Suppose $\mathbf{A}^{(t)}$ is ϵ_t-close to $\mathbf{A}^*$, $\mu = \mathcal{O}(\log(n))$, $s = \mathcal{O}^*(\sqrt{n}/\mu\log(n))$, and $\epsilon_t = \mathcal{O}^*(1/\log(m))$ Then, with probability at least $(1 - \delta_{\beta}^{(t)} - \delta_{\mathcal{T}}^{(t)})$, each iterate of the IHT-based coefficient update step shown in (3.6) has the correct signed-support, if for a constant $c_1^{(r)}(\epsilon_t, \mu, s, n) = \widetilde{\Omega}(k^2/n)$, the step size is chosen as $\eta_x^{(r)} \le c_1^{(r)}$, and the threshold $\tau^{(r)}$ is chosen as

$$\tau^{(r)} = \eta_x^{(r)}(t_\beta + \tfrac{\mu_t}{\sqrt{n}}\|\mathbf{x}^{(r-1)} - \mathbf{x}^*\|_1) := c_2^{(r)}(\epsilon_t, \mu, s, n) = \widetilde{\Omega}(s^2/n),$$

for some constants c_1 and c_2. Here, $t_\beta = \mathcal{O}(\sqrt{s\epsilon_t})$, $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$,and $\delta_{\beta}^{(t)} = 2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.

To get a handle on the error incurred by the each element of the sparse matrix $\widehat{\mathbf{X}}$, i.e.,

$$|\mathbf{X}^*_{ij} - \widehat{\mathbf{X}}_{ij}| = |\mathbf{S}^*_{ij} - \widehat{\mathbf{S}}_{ij}| \le \xi, \tag{3.9}$$

and derive an expression for estimating the sparse matrix $\widehat{\mathbf{X}}$, we use Lemma 3.5 and 3.6. Here, we use Lemma 3.5 to show that the error in the non-zero elements of $\widehat{\mathbf{X}}$ only depends on the error in the incoherent factor (dictionary) $\mathbf{A}^{(t)}$, which leads us to

$$\xi^2 := \mathcal{O}(s(1-\omega)^{t/2}\|\mathbf{A}_i^{(0)} - \mathbf{A}_i^*\|), \text{ for all } (i,j) \in \operatorname{supp}(\mathbf{X}^*). \tag{3.10}$$

Lemma 3.5 (from Lemma 2.3)**. (Upper-bound on the error in coefficient estimation)** With probability at least $(1 - \delta_{\beta}^{(t)} - \delta_{\mathcal{T}}^{(t)})$ the error incurred by each element $(i_1, j_1) \in \operatorname{supp}(\mathbf{X}^*)$ of the coefficient estimate is upper-bounded as

$$|\widehat{\mathbf{X}}_{i_1j_1} - \mathbf{X}^*_{i_1j_1}| \le \mathcal{O}(t_\beta) + \left((R+1)s\eta_x\tfrac{\mu_t}{\sqrt{n}}\max_{(i,j)}|\mathbf{X}^{(0)}_{ij} - \mathbf{X}^*_{ij}| + |\mathbf{X}^{(0)}_{i_1j_1} - \mathbf{X}^*_{i_1j_1}|\right)\delta_R = \mathcal{O}(t_\beta)$$

where $t_\beta = \mathcal{O}(\sqrt{s\epsilon_t})$, $\delta_R := (1 - \eta_x + \eta_x \frac{\mu_t}{\sqrt{n}})^R$, $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$, $\delta_\beta^{(t)} = 2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$, and μ_t is the incoherence between the columns of $\mathbf{A}^{(t)}$.

Therefore, if the the column-wise error in the dictionary decreases at each iteration t, then the IHT-based sparse matrix estimates also improve progressively. Now, the expression for the coefficient estimates (Lemma 3.6) facilitates the analysis of the dictionary updates.

Lemma 3.6 (from Lemma 2.4). **(Expression for the coefficient estimate at the end of** R**-th IHT iteration)**] With probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_\beta^{(t)})$ the i-th element of the coefficient estimate, for each $i \in \text{supp}(\mathbf{x}^*)$, is given by

$$\widehat{\mathbf{x}}_i := \mathbf{x}_i^{(R)} = \mathbf{x}_i^*(1 - \lambda_i^{(t)}) + \vartheta_i^{(R)}.$$

Here, $|\vartheta_i^{(R)}| = \mathcal{O}(t_\beta)$, where $t_\beta = \mathcal{O}(\sqrt{s\epsilon_t})$. Further, $\lambda_i^{(t)} = |\langle \mathbf{A}_i^{(t)} - \mathbf{A}_i^*, \mathbf{A}_i^* \rangle| \leq \frac{\epsilon_t^2}{2}$, $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)})$ and $\delta_\beta^{(t)} = 2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$.

The IHT-based coefficient estimation step is foundational for the recovery of the sparse tensor factors $\mathbf{B}^*$ and $\mathbf{C}^*$. Before we show that the approximate gradient descent-based update step to recover $\mathbf{A}^*$ makes progress at each iteration of the online algorithm, we first show the correctness of Algorithm 3. This procedure recovers the sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$, given element-wise ξ-close estimate $\widehat{\mathbf{S}}$ of $\mathbf{S}^*$. The following lemma establishes recovery guarantees on the sparse factors using the SVD-based Algorithm 3, up to sign and scaling ambiguity.

Lemma 3.7. Suppose the input $\widehat{\mathbf{S}}$ to Algorithm 3 is entry-wise ζ close to $\mathbf{S}^*$, i.e., $|\widehat{\mathbf{S}}_{ij} - \mathbf{S}_{ij}^*| \leq \zeta$ and has the correct signed-support as $\mathbf{S}^*$. Then with probability atleast $(1 - \delta_{\text{IHT}}^{(t)} - \delta_{\mathbf{B}_i}^{(t)})$, both $\widehat{\mathbf{B}}_i$ and $\widehat{\mathbf{C}}_i$ have the correct support, and $\left\| \frac{\mathbf{B}_i^*}{\|\mathbf{B}_i^*\|} - \pi_i \frac{\widehat{\mathbf{B}}_i}{\|\widehat{\mathbf{B}}_i\|} \right\| = \mathcal{O}(\zeta^2)$ and $\left\| \frac{\mathbf{C}_i^*}{\|\mathbf{C}_i^*\|} - \pi_i \frac{\widehat{\mathbf{C}}_i}{\|\widehat{\mathbf{C}}_i\|} \right\| = \mathcal{O}(\zeta^2)$, where $\delta_{\text{IHT}}^{(t)} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)}) + 2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$ for $\|\mathbf{A}_i^{(t)} - \mathbf{A}_i^*\| \leq \epsilon_t$, and $\delta_{\mathbf{B}_i}^{(t)} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ for any $\epsilon > 0$.

Here, we have used $\delta_{\text{IHT}}^{(t)} = \delta_\beta^{(t)} + \delta_{\mathcal{T}}^{(t)}$ for simplicity. To recover the incoherent (dictionary) factor $\mathbf{A}^*$, we first develop an expression for the expected gradient vector in Lemma 3.8.

Lemma 3.8 (from Lemma 2.5). **(Expression for the expected gradient vector)** Suppose that $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$-near to $\mathbf{A}^*$. Then, the dictionary update step in Algorithm 2 amounts to the following for the j-th dictionary element

$$\mathbf{E}[\mathbf{A}_j^{(t+1)}] = \mathbf{A}_j^{(t)} + \eta_A \mathbf{g}_j^{(t)},$$

where for a small $\widetilde{\gamma}$, $\mathbf{g}_j^{(t)}$ is given by

$$\mathbf{g}_j^{(t)} = q_j p_j \Big((1 - \lambda_j^{(t)}) \mathbf{A}_j^{(t)} - \mathbf{A}_j^* + \tfrac{1}{q_j p_j} \Delta_j^{(t)} \pm \widetilde{\gamma} \Big),$$

$\lambda_j^{(t)} = |\langle \mathbf{A}_j^{(t)} - \mathbf{A}_j^*, \mathbf{A}_j^* \rangle|$, and $\Delta_j^{(t)} := \mathbf{E}[\mathbf{A}_S^{(t)} \vartheta_S^{(R)} \mathrm{sign}(\mathbf{x}_j^*)]$, where $\|\Delta_j^{(t)}\| = \mathcal{O}(\sqrt{m} q_{i,j} p_j \epsilon_t \|\mathbf{A}^{(t)}\|)$.

Now, since we use the empirical gradient estimate, we use Lemma 3.9 to show that the empirical gradient vector concentrates around its mean.

Lemma 3.9 (from Lemma 2.6)**. (Concentration of the empirical gradient vector)** Given $p = \widetilde{\Omega}(mk^2)$ samples, the empirical gradient vector estimate corresponding to the i-th dictionary element, $\widehat{\mathbf{g}}_i^{(t)}$ concentrates around its expectation, i.e.,

$$\|\widehat{\mathbf{g}}_i^{(t)} - \mathbf{g}_i^{(t)}\| \leq o(\tfrac{s}{m} \epsilon_t).$$

with probability at least $(1 - \delta_{\mathbf{g}_i}^{(t)} - \delta_\beta^{(t)} - \delta_{\mathcal{T}}^{(t)} - \delta_{\mathrm{HW}}^{(t)})$, where $\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(s))$.

We then leverage Lemma 3.10 to show that the empirical gradient vector $\widehat{\mathbf{g}}_j^{(t)}$ is correlated with the descent direction (see Def. 3.7), which ensures that the dictionary estimate makes progress at each iteration of the online algorithm.

Lemma 3.10 (from Lemma 2.7)**. (Empirical gradient vector is correlated with the descent direction)** Suppose $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$-near to $\mathbf{A}^*$, $s = \mathcal{O}(\sqrt{n})$ and $\eta_A = \mathcal{O}(m/s)$. Then, with probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_\beta^{(t)} - \delta_{\mathrm{HW}}^{(t)} - \delta_{\mathbf{g}_i}^{(t)})$ the empirical gradient vector $\widehat{\mathbf{g}}_j^{(t)}$ is $(\Omega(k/m), \Omega(m/k), 0)$-correlated with $(\mathbf{A}_j^{(t)} - \mathbf{A}_j^*)$, and for any $t \in [T]$,

$$\|\mathbf{A}_j^{(t+1)} - \mathbf{A}_j^*\|^2 \leq (1 - \rho_{\_}\eta_A)\|\mathbf{A}_j^{(t)} - \mathbf{A}_j^*\|^2.$$

This step also requires closeness that the estimate $\mathbf{A}^{(t)}$ and $\mathbf{A}^*$ are close, both column-wise and in the spectral norm-sense, as per Def 3.1. To this end, we show that the updated dictionary matrix maintain the closeness property. For this, we first show that the gradient matrix concentrates around its mean in Lemma 3.11.

Lemma 3.11 (from Lemma 2.8)**. (Concentration of the empirical gradient matrix)** With probability at least $(1 - \delta_\beta^{(t)} - \delta_{\mathcal{T}}^{(t)} - \delta_{\mathrm{HW}}^{(t)} - \delta_{\mathbf{g}}^{(t)})$, $\|\widehat{\mathbf{g}}^{(t)} - \mathbf{g}^{(t)}\|$ is upper-bounded by $\mathcal{O}^*(\frac{s}{m}\|\mathbf{A}^*\|)$, where $\delta_{\mathbf{g}}^{(t)} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$.

Further, that the closeness property is maintained in Lemma 3.12, as shown below.

Lemma 3.12 (from Lemma 2.9)**.** **($\mathbf{A}^{(t+1)}$ maintains closeness)** Suppose $\mathbf{A}^{(t)}$ is $(\epsilon_t, 2)$ near to $\mathbf{A}^*$ with $\epsilon_t = \mathcal{O}^*(1/\log(n))$, and number of samples used in step t is $p = \widetilde{\Omega}(ms^2)$, then with probability at least $(1 - \delta_{\mathcal{T}}^{(t)} - \delta_{\beta}^{(t)} - \delta_{\text{HW}}^{(t)} - \delta_{\mathbf{g}}^{(t)})$, $\mathbf{A}^{(t+1)}$ satisfies $\|\mathbf{A}^{(t+1)} - \mathbf{A}^*\| \leq 2\|\mathbf{A}^*\|$.

Therefore, the recovery of factor $\mathbf{A}^*$, and the sparse-structured matrix $\mathbf{X}^*$ suceeds with probability $\delta_{\text{NOODL}}^{(t)} = \delta_{\mathcal{T}}^{(t)} + \delta_{\beta}^{(t)} + \delta_{\text{HW}} + \delta_{\mathbf{g}_i}^{(t)} + \delta_{\mathbf{g}}^{(t)}$, where $\delta_{\mathcal{T}}^{(t)} = 2m\exp(-C^2/\mathcal{O}^*(\epsilon_t^2))$, $\delta_{\beta}^{(t)} = 2s\exp(-1/\mathcal{O}(\epsilon_t))$, $\delta_{\text{HW}}^{(t)} = \exp(-1/\mathcal{O}(\epsilon_t))$, $\delta_{\mathbf{g}_i}^{(t)} = \exp(-\Omega(s))$, $\delta_{\mathbf{g}}^{(t)} = (n+m)\exp(-\Omega(m\sqrt{\log(n)})$.

Further, from Lemma 3.1, we have that the columns of $\mathbf{S}^*$ are $s = \mathcal{O}(\alpha\beta m)$ sparse with probability $(1 - \delta_s^{(t)})$, where $\delta_s^{(t)} = \min(J, K)\exp(-\epsilon^2\alpha\beta m/2(1+\epsilon/3))$ for any $\epsilon > 0$, and that with probability at least $(1-\delta_p)$, the number of data samples $p = L(1-(1-\gamma)^m)$, where $\delta_p^{(t)} = \exp(-\frac{\epsilon^2}{2}L(1-(1-\gamma)^m))$ using Lemma 3.1. Furthermore, from Lemma 3.7, we know that Algorithm 3 (which only relies on recovery of $\mathbf{X}^*$) succeeds in recovering $\mathbf{B}^*$ and $\mathbf{C}^*$ with probability $(1 - \delta_{\mathbf{B}_i}^{(t)})$, where $\delta_{\mathbf{B}_i}^{(t)} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ for any $\epsilon > 0$.

Combining all these results we have that, Algorithm 2 succeeds with probability $(1 - \delta_{alg})$, where $\delta_{alg} = \delta_s + \delta_p^{(t)} + \delta_{\mathbf{B}_i}^{(t)} + \delta_{\text{NOODL}}$. Hence, our main result.

A note on independent sample requirement: Since the IHT-based coefficient operates independently on each column of $\mathbf{Y}$ (the non-zero columns of $\mathbf{Z}_1^\top$), the dependence structure of $\mathbf{S}^*$ does not affect this stage. For the dictionary update (in theory) we only use the independent columns of $\mathbf{Y}$, these can be inferred using J and K, and corresponding induced transposed Khatri-Rao structure. In practice however, we don't need to throw away any samples, this is purely to ensure that the independence assumption holds for our finite sample analysis of the algorithm. □

3.C Proof of Intermediate Results

Lemma 3.1 *If $m = \Omega(\log(\min(J,K))/\alpha\beta)$ then with probability at least $(1-\delta_s^{(t)})$ the number of non-zeros, s, in a column of $\mathbf{S}^*$ are upper-bounded as $s = \mathcal{O}(\alpha\beta m)$, where* $\delta_s^{(t)} = \min(J, K)$ $\exp(-\frac{\epsilon^2\alpha\beta m}{2(1+\epsilon/3)})$ *for any $\epsilon > 0$.*

Proof. **of Lemma 3.1** Consider a column of the transposed Khatri-Rao structured matrix $\mathbf{S}^*$ defined as $\mathbf{S}^* = (\mathbf{C}^* \odot \mathbf{B}^*)^\top$. Here, since the entries of factors $\mathbf{B}^*$ and $\mathbf{C}^*$ are independently non-zero with probability α and β, respectively, each entry of a column of $\mathbf{S}^*$ is independently non-zero with probability $\gamma = \alpha\beta$, i.e., $\mathbb{1}_{|\mathbf{S}^*_{ij}|>0} \sim$ Bernoulli(γ). As a result, the number of non-zero elements in a column of $\mathbf{S}^*$ are Binomial(m, γ).

Now, let $\mathbf{s}_{ij}$ be the indicator for the (i,j) element of $\mathbf{S}^*$ being non-zero, defined as

$$\mathbf{s}_{ij} = \mathbb{1}_{|\mathbf{S}^*_{ij}|>0}.$$

Then, the expected number of non-zeros (sparsity) in the j-th column of $\mathbf{S}^*$ are given by

$$\mathbf{E}[\textstyle\sum_{i=1}^{m} \mathbf{s}_{ij}] = \gamma m.$$

Since, γ can be small, we use Lemma 3.13(a) (McDiarmid, 1998) to derive an upper bound on the sparsity for each each column as

$$\mathbf{Pr}[\textstyle\sum_{i=1}^{m} \mathbf{s}_{ij} \geq (1+\epsilon)\gamma m] \leq \exp(-\frac{\epsilon^2 \gamma m}{2(1+\epsilon/3)}).$$

for any $\epsilon > 0$. Union bounding over $L = \min(J, K)$ independent columns of $\mathbf{S}^*$.

$$\mathbf{Pr}[\ \textstyle\bigcup_{j=1}^{L} (\sum_{i=1}^{m} \mathbf{s}_{ij} \leq (1+\epsilon)\gamma m)] \geq 1 - L\exp(-\frac{\epsilon^2 \gamma m}{2(1+\epsilon/3)}).$$

Therefore, we conclude that if $m = \Omega(\log(L)/\gamma)$ then with probability $(1-\delta_s)$ the expected number of non-zeros in a column of $\mathbf{S}^*$ are $\mathcal{O}(\gamma m)$, where $\delta_s = L\exp(-\frac{\epsilon^2 \gamma m}{2(1+\epsilon/3)})$. □

Lemma 3.2 For any $\epsilon > 0$ suppose we have

$$L \geq \frac{2}{(1-(1-\gamma)^m)\epsilon^2} \log(\frac{1}{\delta_p^{(t)}}),$$

for $L = \min(J, K)$ and $\gamma = \alpha\beta$, then with probability at least $(1-\delta_p)$,

$$p = L(1-(1-\gamma)^m),$$

where $\delta_p^{(t)} = \exp(-\frac{\epsilon^2}{2} L(1-(1-\gamma)^m))$.

Proof. **of Lemma 3.2** We begin by evaluating the probability that a column of $\mathbf{S}^*$ has a non-zero element. Let $\mathbf{s}_{ij}$ be the indicator for the (i,j) element of $\mathbf{S}^*$ being non-zero, defined as

$$\mathbf{s}_{ij} = \mathbb{1}_{|\mathbf{S}^*_{ij}|>0}.$$

Further, let w_j denote the number of non-zeros in the j-th column of $\mathbf{S}^*$, defined as

$$w_j = \textstyle\sum_{i=1}^{m} \mathbf{s}_{ij}.$$

Since each element of a column of $\mathbf{S}^*$ is non-zero with probability γ, the probability that the j-th column of $\mathbf{S}^*$ is an all zero vector is,

$$\mathbf{Pr}[w_j = 0] = (1-\gamma)^m.$$

Therefore, the probability that the j-th column of $\mathbf{S}^*$ has at least one non-zero element is given by

$$\mathbf{Pr}[w_j > 0] = 1 - (1-\gamma)^m. \tag{3.11}$$

Now, we are interested in the number of columns with at least one non-zero element among the $L = \min(J, K)$ independent columns of $\mathbf{S}$, which we denote by p. Specifically, we analyze the following sum

$$p = \textstyle\sum_{j=1}^{L} \mathbb{1}_{w_j>0}.$$

Next, using (3.11) $\mathbf{E}[p] = L(1-(1-\gamma)^m)$. Applying the result stated Lemma 3.13 (b),

$$\mathbf{Pr}[\sum_{j=1}^{L} \mathbb{1}_{w_j} \leq (1-\epsilon)\mathbf{E}[p]] \leq \exp(-\tfrac{\epsilon^2 E[p]}{2}) := \delta_p^{(t)}.$$

Therefore, if for any $\epsilon > 0$ we have

$$L \geq \tfrac{2}{(1-(1-\gamma)^m)\epsilon^2}\log(\tfrac{1}{\delta_p^{(t)}})$$

then with probability at least $(1-\delta_p)$, $p = L(1-(1-\gamma)^m)$, where $\delta_p^{(t)} = \exp(-\frac{\epsilon^2}{2}L(1-(1-\gamma)^m))$.

□

Lemma 3.7 Suppose the input $\widehat{\mathbf{S}}$ to Algorithm 3 is entry-wise ζ close to $\mathbf{S}^*$, i.e., $|\widehat{\mathbf{S}}^*_{ij} -$

$\mathbf{S}^*_{ij}| \le \zeta$ and has the correct signed-support as $\mathbf{S}^*$. Then with probability atleast $(1 - \delta^{(t)}_{\text{IHT}} - \delta^{(t)}_{\mathbf{B}_i})$, both $\widehat{\mathbf{B}}_i$ and $\widehat{\mathbf{C}}_i$ have the correct support, and $\left\|\frac{\mathbf{B}^*_i}{\|\mathbf{B}^*_i\|} - \pi_i \frac{\widehat{\mathbf{B}}_i}{\|\widehat{\mathbf{B}}_i\|}\right\| = \mathcal{O}(\zeta^2)$ and $\left\|\frac{\mathbf{C}^*_i}{\|\mathbf{C}^*_i\|} - \pi_i \frac{\widehat{\mathbf{C}}_i}{\|\widehat{\mathbf{C}}_i\|}\right\| = \mathcal{O}(\zeta^2)$, where $\delta^{(t)}_{\text{IHT}} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)}) + 2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$ for $\|\mathbf{A}^{(t)}_i - \mathbf{A}^*_i\| \le \epsilon_t$, and $\delta^{(t)}_{\mathbf{B}_i} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ for any $\epsilon > 0$.

Proof. **of Lemma 3.7** The Iterative Hard Thresholding (IHT) results in an estimate of $\mathbf{X}^*$ which has the correct signed support (Chapter 2) (Rambhatla et al., 2019). As a result, putting back the columns of $\widehat{\mathbf{X}}$ at the respective non-zero column locations of $\mathbf{Z}_1^\top$, we arrive at the estimate $\widehat{\mathbf{S}}$ of $\mathbf{S}^*$, which has the correct signed-support, we denote this estimate by $\widehat{\mathbf{S}}$. To recover the estimates $\widehat{\mathbf{B}}$ and $\widehat{\mathbf{C}}$, we use a SVD-based procedure. Specifically, we note that,

$$\mathbf{S}^{*\top}_{i,:} = \mathbf{C}^*_i \otimes \mathbf{B}^*_i = vec(\mathbf{B}^*_i {\mathbf{C}_i}^{*\top})$$

As a result, the left and right singular vectors of the rank-1 matrix $\mathbf{B}^*_i \mathbf{C}^{*\top}_i$ are the columns $\mathbf{B}^*_i$ and $\mathbf{C}^*_i$, respectively (up to scaling).

Let $\mathbf{M}^{(i)}$ denote the $J \times K$ matrix formed by reshaping the vector $\widehat{\mathbf{S}}^\top_{i,:}$. We choose the appropriately scaled left and right singular vectors corresponding to the largest singular value of $\mathbf{M}^{(i)}$ as our estimates $\widehat{\mathbf{B}}_i$ and $\widehat{\mathbf{C}}_i$, respectively.

First, notice that since $\widehat{\mathbf{S}}^\top_{i,:}$ has the correct sign and support (due to Lemma 3.4), the support of matrix $\mathbf{M}^{(i)}$ is the same as $\mathbf{B}^*_i \mathbf{C}^{*\top}_i$. As a result, the estimates $\widehat{\mathbf{B}}_i$ and $\widehat{\mathbf{C}}_i$ have the correct support, and the error is only due to the scaling ambiguity on the support. This is due to the fact that the principal singular vectors ($\mathbf{u}$ and $\mathbf{v}$) align with the sparsity structure of $\mathbf{M}^{(i)}$ as they solve the following maximization problem also known as variational characterization of svd,

$$\sigma_1^2 = \max_{\|\mathbf{u}\|=1} \mathbf{u}^\top \mathbf{M}^{(i)} {\mathbf{M}^{(i)}}^\top \mathbf{u} = \max_{\|\mathbf{v}\|=1} \mathbf{v}^\top {\mathbf{M}^{(i)}}^\top \mathbf{M}^{(i)} \mathbf{v},$$

where σ_1 denotes the principal singular value. Therefore, since $\mathbf{M}^{(i)}$ has the correct sparsity structure as $\mathbf{B}^*_i {\mathbf{C}_i}^{*\top}$ the resulting $\mathbf{u}$ and $\mathbf{v}$ have the correct supports as well. Here, $\mathbf{u}$ and $\mathbf{v}$ can be viewed as the normalized versions of $\widehat{\mathbf{B}}_i$ and $\widehat{\mathbf{C}}_i$, respectively, i.e., $\mathbf{u} = \widehat{\mathbf{B}}_i / \|\widehat{\mathbf{B}}_i\|$ and $\mathbf{v} = \widehat{\mathbf{C}}_i / \|\widehat{\mathbf{C}}_i\|$.

Let $\mathbf{E} = \mathbf{M}^{(i)} - \mathbf{B}^*_i \mathbf{C}^{*\top}_i$, now since $|\widehat{\mathbf{S}}_{ij} - \mathbf{S}^*_{ij}| \le \zeta$ and, from Lemma 3.4) $\widehat{\mathbf{S}}(i,:)$ has the correct signed-support with probability $(1 - \delta^{(t)}_{\text{IHT}})$, where $\delta^{(t)}_{\text{IHT}} = 2m\exp(-\frac{C^2}{\mathcal{O}^*(\epsilon_t^2)}) +$

$2s\exp(-\frac{1}{\mathcal{O}(\epsilon_t)})$, and further using Claim 11, we have that the expected number of non-zeros in $\widehat{\mathbf{S}}(i,:)$ are $JK\alpha\beta$, with probability atleast $(1-\delta_{\mathbf{B}_i}^{(t)})$, where $\delta_{\mathbf{B}_i}^{(t)} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ for some $\epsilon > 0$, we have

$$\|\mathbf{E}\| \leq \|\mathbf{E}\|_{\mathrm{F}} \leq \sqrt{JK\alpha\beta}\zeta,$$

Then, using the result in Yu et al. (2014), and noting that $\sigma_1(\mathbf{B}_i\mathbf{C}_i^\top) = \|\mathbf{B}_i\|\|\mathbf{C}_i\|$ and letting $\pi_i \in \{-1,1\}$ (to resolve the sign ambiguity), we have that

$$\left\|\frac{\mathbf{B}_i^*}{\|\mathbf{B}_i^*\|} - \pi_i\mathbf{u}\right\| = \left\|\frac{\mathbf{B}_i^*}{\|\mathbf{B}_i^*\|} - \pi_i\frac{\widehat{\mathbf{B}}_i}{\|\widehat{\mathbf{B}}_i\|}\right\| \leq \frac{2^{3/2}(2\|\mathbf{B}_i\|\|\mathbf{C}_i\|+\sqrt{JK\alpha\beta}\zeta)\sqrt{JK\alpha\beta}\zeta}{\|\mathbf{B}_i\|^2\|\mathbf{C}_i\|^2}.$$

Next, since $\mathbf{E}[\mathbf{B}_{ij}^2|(i,j) \in \mathrm{supp}(\mathbf{B})] = 1$ as per our distributional assumptions **Def.3.3**, we have

$$\mathbf{E}[\|\mathbf{B}_{ji}^*\|^2] = \mathbf{E}[\mathbf{B}_{ji}^{*2}|(j,i) \in \mathrm{supp}(\mathbf{B}^*)]\mathbf{Pr}[(j,i) \in \mathrm{supp}(\mathbf{B}^*)] + 0.\mathbf{Pr}[(j,i) \notin \mathrm{supp}(\mathbf{B}^*)] = \alpha$$

Similarly, $\mathbf{E}[\|\mathbf{C}_{ji}^*\|^2] = \beta$. Substituting,

$$\left\|\frac{\mathbf{B}_i^*}{\|\mathbf{B}_i^*\|} - \pi_i\frac{\widehat{\mathbf{B}}_i}{\|\widehat{\mathbf{B}}_i\|}\right\| \leq \frac{2^{3/2}(2\sqrt{JK\alpha\beta}+\sqrt{JK\alpha\beta}\zeta)\sqrt{JK\alpha\beta}\zeta}{JK\alpha\beta} = \mathcal{O}(\zeta^2).$$

□

Claim 11. Suppose $J = \Omega(\frac{1}{\alpha}))$, then with probability at least $(1-\delta_{\mathbf{B}_i}^{(t)})$,

$$\textstyle\sum_{j=1}^{JK}\mathrm{supp}(\mathbf{S}^*(i,j)) = JK\alpha\beta,$$

where $\delta_{\mathbf{B}_i}^{(t)} = \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)})$ for any $\epsilon > 0$.

Proof. **of Claim 11** In this lemma we establish an upper-bound on the number of non-zeros in a row of $\mathbf{S}^*$. The i-th row of $\mathbf{S}^*$ can be written as $\mathrm{vec}(\mathbf{B}_{\mathrm{i}}^*\mathbf{C}_{\mathrm{i}}^{*\top})$.

Since each element of matrix $\mathbf{B}^*$ and $\mathbf{C}^*$ are independently non-zero with probabilities α and β, the number of non-zeros in a column $\mathbf{B}_i^*$ of $\mathbf{B}^*$ are binomially distributed. Let $\mathbf{s}_j$ be the indicator for the j-th element of $\mathbf{B}_i^*$ being non-zero, defined as

$$\mathbf{s}_i = \mathbb{1}_{|\mathbf{B}^*(j,i)|>0}.$$

Then, the expected number of non-zeros (sparsity) in the i-th column of $\mathbf{B}^*$ are given by

$$\mathbf{E}[\textstyle\sum \mathrm{supp}(\mathbf{B}^*_i)] = \mathbf{E}[\sum_{j=1}^{J} \mathbf{s}_j] = J\alpha.$$

Since, α can be small, we use Lemma 3.13(a) (McDiarmid, 1998) to derive an upper bound on the sparsity for each each column as

$$\mathbf{Pr}[\textstyle\sum_{j=1}^{J} \mathbf{s}_j \geq (1+\epsilon)J\alpha] \leq \exp(-\frac{\epsilon^2 J\alpha}{2(1+\epsilon/3)}) := \delta^{(t)}_{\mathbf{B}_i}. \tag{3.12}$$

for any $\epsilon > 0$.

Now we turn to the number of non-zeros in $\mathbf{S}^*_i = \mathrm{vec}(\mathbf{B}^*_{\mathrm{i}}\mathbf{C}^{*\top}_{\mathrm{i}})$. We first note that the j-th column of $\mathbf{B}^*_i\mathbf{C}^{*\top}_i$ is given by $\mathbf{C}(j,i)^*\mathbf{B}^*_i$. This implies that the j-th column can be all-zeros if $\mathbf{C}(j,i)^* = 0$. As a result, the expected number of non-zeros in the j-th column of $\mathbf{B}^*_i\mathbf{C}^{*\top}_i$ can be written as,

$$\begin{aligned}
&\mathbf{E}[\textstyle\sum \mathrm{supp}(\mathbf{C}^*_{ji}\mathbf{B}^*_i)] \\
&= \mathbf{E}[\textstyle\sum \mathrm{supp}(\mathbf{C}^*_{ji}\mathbf{B}^*_i)|\mathbf{C}^*_{ji} \neq 0]\mathbf{Pr}[\mathbf{C}^*_{ji} \neq 0] + \mathbf{E}[\sum \mathrm{supp}(\mathbf{C}^*_{ji}\mathbf{B}^*_i)|\mathbf{C}^*_{ji} = 0]\mathbf{Pr}[\mathbf{C}^*_{ji} = 0] \\
&= \mathbf{E}[\textstyle\sum \mathrm{supp}(\mathbf{C}^*_{ji}\mathbf{B}^*_i)|\mathbf{C}^*_{ji} \neq 0]\mathbf{Pr}[\mathbf{C}^*_{ji} \neq 0] = \mathbf{E}[\sum \mathrm{supp}(\mathbf{B}^*_i)]\mathbf{Pr}[\mathbf{C}^*_{ji} \neq 0].
\end{aligned}$$

Now, from (3.12), we have that if we choose $J = \Omega(\frac{1}{\alpha}))$ with probability atleast $(1-\delta^{(t)}_{\mathbf{B}_i})$, there are $S_i = J\alpha$ non-zeros in a column of $\mathbf{B}^*$. Further since, $\mathbf{Pr}[\mathbf{C}^*_{ji} \neq 0] = \beta$, we have that with probability atleast $(1-\delta^{(t)}_{\mathbf{B}_i})$,

$$\mathbf{E}[\textstyle\sum \mathrm{supp}(\mathbf{C}^*_{ji}\mathbf{B}^*_i)] = J\alpha\beta.$$

Furthermore, since there are K columns in $\mathbf{B}^*_i\mathbf{C}^{*\top}_i$, with probability atleast $(1-\delta^{(t)}_{\mathbf{B}_i})$,

$$\mathbf{E}[\textstyle\sum \mathrm{supp}(\mathrm{vec}(\mathbf{B}^*_i\mathbf{C}^{*\top}_i)] = \mathbf{E}[\sum_{j=1}^{JK} \mathrm{supp}(\mathbf{S}^*(i,j))] = JK\alpha\beta.$$

□

3.D Additional Theoretical Results

Lemma 3.13. Relative Chernoff McDiarmid (1998) Let random variables $w_1,\ldots,w_\ell$ be independent, with $0 \leq w_i \leq 1$ for each i. Let $S_w = \sum_{i=1}^{\ell} w_i$, let $\nu = \mathbf{E}(S_w)$ and let $p = \nu/\ell$, then for any $\epsilon > 0$,

$$(a) \quad \mathbf{Pr}[S_w - \nu \geq \epsilon\nu] \leq \exp(-\epsilon^2\nu/2(1+\varepsilon/3)),$$
$$(b) \quad \mathbf{Pr}[S_w - \nu \leq \epsilon\nu] \leq \exp(-\epsilon^2\nu/2).$$

Lemma 3.14 (Specialized Theorem 4 in Yu et al. (2014) for singular vectors)**.** Given $\mathbf{M}$, $\widetilde{\mathbf{M}} \in \mathbb{R}^{m\times n}$, where $\widetilde{\mathbf{M}} = \mathbf{M}+\mathbf{E}$ and the corresponding SVD of $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ and $\widetilde{\mathbf{M}} = \widetilde{\mathbf{U}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^\top$, the sine of angle between the principal left (and right) singular vectors of matrices $\mathbf{M}$ and $\widetilde{\mathbf{M}}$ is given by

$$\sin\Theta(\mathbf{U}_1, \widetilde{\mathbf{U}}_1) \leq \tfrac{2(2\sigma_1+\|\mathbf{E}\|_2)(\min(\|\mathbf{E}\|_2,\|\mathbf{E}\|_\mathrm{F})}{\sigma_1^2},$$

where σ_1 is the principal singular value corresponding to $\mathbf{U}_1$. Furthermore, there exists $\pi \in -1, 1$ such that

$$\|\mathbf{U}_1 - \pi\widetilde{\mathbf{U}}_1\| \leq \tfrac{2^{3/2}(2\sigma_1+\|\mathbf{E}\|_2)(\min(\|\mathbf{E}\|_2,\|\mathbf{E}\|_\mathrm{F})}{\sigma_1^2}.$$

3.E Experimental Set-up and Additional Experimental Results

In this appendix we detail the specifics of the experiments presented in Section 3.6. In addition, we also presented the simulation results corresponding to $\alpha = \beta = \{0.005, 0.01, 0.05\}$ are shown in Table 3.E.2, 3.E.3, and 3.E.4, respectively.

3.E.1 Experimental Set-up

We analyze the performance of the algorithm across different choices of tensor dimensions (J, K) for a fixed $n = 300$, its rank(m) and the sparsity of factors $\mathbf{B}^*$ and $\mathbf{C}^*$ controlled by parameters (α, β), for recovery of the constituent factors using three Monte-Carlo runs. For each of these runs, we analyze the recovery performance across three choices of dimensions $J = K = \{100,\ 300,\ 500\}$, five choices of rank $m = \{50, 150, 300, 450, 600\}$, and three choices of the sparsity parameters $\alpha = \beta = \{0.005, 0.01, 0.05\}$.

Parameters Setting – We set `TensorNOODL` specific IHT parameters $\eta_x = 0.2$ and $\tau = 0.1$ for all experiments. As recommended by our main result, the dictionary step-size parameter η_A is set proportional to m/k. Since `TensorNOODL`, `Arora(b)`, and `Arora(u)` all rely on an approximate gradient descent strategy for dictionary update, we use the same step-size η_A for a fair comparison. Specifically, the dictionary update step-size parameter (η_A) is set to be the same for `TensorNOODL`, `Arora(b)`, and `Arora(u)` depending upon the choice of rank m, and probabilities (α, β), according to assumption **A.5** and Table 3.E.1. Note that `Mairal` does not employ such a parameter.

Table 3.E.1: Choosing the step-size (η_A) for the dictionary update step. The dictionary update step-size parameter (η_A) is set to be the same for `TensorNOODL`, `Arora(b)`, and `Arora(u)` depending upon the choice of rank m, and probabilities (α, β), according to assumption **A.5**.

Rank, m	**Step-size for dictionary update,** η_A	**Notes**
50	20	For $(\alpha, \beta) = 0.005$, we use $\eta_A = 5$
150	40	–
300	40	–
450	50	–
600	50	–

Data Generation – For each experiment we draw entries of the dictionary factor matrix $\mathbf{A}^* \in \mathbb{R}^{n \times m}$ from $\mathcal{N}(0,1)$, and normalize its columns to be unit-norm. To form $\mathbf{A}^{(0)}$ in accordance with **A.2**, we perturb $\mathbf{A}^*$ with random Gaussian noise and normalized its columns, such that it is column-wise $2/\log(n)$ away from $\mathbf{A}^*$ in ℓ_2 norm sense.

To form the sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$, we assign their entries to the support independently with probability α and β, respectively, and then draw the values on the support from the Rademacher distribution.

Evaluation Metrics – We run all algorithms till one of them achieves target tolerance (error in the factor $\mathbf{A}$, ϵ_T) of 10^{-10}, and report the number of iterations T for each experiment. Note that, in all cases `TensorNOODL` achieves the tolerance first, and in some cases with the algorithms considered in the analysis. Next, since recovery of $\mathbf{A}^*$ and $\mathbf{X}^*$ is vital for the success of the tensor factorization task, we report the relative Frobenius error for each of these matrices, i.e., for a recovered matrix $\widehat{\mathbf{M}}$, we report $\|\widehat{\mathbf{M}} - \mathbf{M}^*\|_{\mathrm{F}}/\|\mathbf{M}^*\|_{\mathrm{F}}$.

In addition, since the dictionary learning task focuses on recovering the sparse matrix $\mathbf{X}^*$, it is agnostic to the transposed Khatri-Rao structure $\mathbf{S}^*$. As a result, for recovering the sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$ is crucial for exact support recovery of $\mathbf{X}^*$. Therefore, we report if the support has been exactly recovered or not.

Reproducible Results – We fix the random seeds (to $42, 26$, and 91) for each Monte Carlo run to ensure reproducibility of the results shown in this work. The experiments were run on a HP Haswell Linux Cluster. The processing of data samples for the sparse coefficients ($\widehat{\mathbf{X}}$) was split across 20 workers (cores), allocated a total of 200 GB RAM. For `Arora(b)`, `Arora(u)`, and `Mairal`, the coefficient recovery was switched between Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) (Beck and Teboulle, 2009), an accelerated proximal gradient descent algorithm, or a stochastic-version of Iterative Shrinkage-Thresholding Algorithm (ISTA) (Chambolle et al., 1998; Daubechies et al., 2004) depending upon the size of the data samples available for learning (see the discussion of the coefficient update step below); see also Beck and Teboulle (2009) for details. We have also made the code available as part of this submission.

Sparse Factor Recovery Considerations – In Arora et al. (2015), the authors present two algorithms – a simple algorithm with a sample complexity of $\widetilde{\Omega}(ms)$ which incurs an estimation bias (`Arora(b)`), and a more involved variant for unbiased estimation of the dictionary whose sample complexity was not established `Arora(u)`. However, these algorithms do not provide guarantees on, or recover the sparse coefficients. As a result, we need to adopt an additional ℓ_1 minimization based coefficient recovery step. Further, the algorithm proposed by Mairal et al. (2009) can be viewed as a variant of regularized alternating least squares algorithm which employs ℓ_1 regularization for the recovery of the transposed Khatri-Rao structured matrix.

To form the coefficient estimates for `Arora(b)`, `Arora(u)`, and `Mairal` '09 we solve the Lasso (Tibshirani, 1996) program using a stochastic-version of Iterative Shrinkage-Thresholding Algorithm (ISTA) (Chambolle et al., 1998; Daubechies et al., 2004) (or Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) (Beck and Teboulle, 2009) if p is small) and report the best estimate (in terms of relative Frobenius error) across 10 values of the regularization parameter. The stochastic projected gradient descent is necessary to make coefficient recovery tractable since size of $\mathbf{X}$ grows quickly with (α, β). Coefficient estimation step is still the slowest step in these algorithms since one has to scan through different values of the regularization parameters. In contrast, `TensorNOODL` does not require such an expensive tuning procedure, while providing recovery guarantees on the recovered coefficients.

Note that in practice ISTA and FISTA can be parallelized as well, but tuning of the regularization parameters is still involves (an expensive) grid search. Arguably even if each step of these algorithms (ISTA and FISTA) take the same amount of time as that of `TensorNOODL`, the search over, say 10, values of the regularization parameters will still be take 10 times the time. As a result, `TensorNOODL` is an attractive choice as it does not involve an expensive tuning procedure.

3.E.2 Additional Results

Table 3.E.2, 3.E.3, and 3.E.4 show the results of the analysis averaged across the three Monte Carlo runs, for $\alpha = \beta = \{0.005, 0.01, 0.05\}$, respectively. We note that for every choice of (J, K), m, and (α, β), `TensorNOODL` emerges as significantly superior to the related techniques. In addition, its support recovery performance is specifically interesting since it is crucial for recovery of the sparse factors $\mathbf{B}^*$ and $\mathbf{C}^*$ via Algorithm 3.

Table 3.E.2: Tensor factorization results $\alpha, \beta = 0.005$ averaged across 3 trials. Here, $T(\text{supp}(\widehat{\mathbf{X}}))$ field shows the number of iterations T to reach the target tolerance, while the categorical field, $\text{supp}(\widehat{\mathbf{X}})$ indicates if the support of the recovered $\widehat{\mathbf{X}}$ matches that of $\mathbf{X}^*$ (Y) or not (N).

(J, K)	**Method**	$m = 50$			$m = 150$			$m = 300$			$m = 450$			$m = 600$		
		$\frac{\lVert\mathbf{A}^*-\mathbf{A}^{(T)}\rVert_F}{\lVert\mathbf{A}^*\rVert_F}$	$\frac{\lVert\mathbf{X}^*-\mathbf{X}^{(T)}\rVert_F}{\lVert\mathbf{X}^*\rVert_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert\mathbf{A}^*-\mathbf{A}^{(T)}\rVert_F}{\lVert\mathbf{A}^*\rVert_F}$	$\frac{\lVert\mathbf{X}^*-\mathbf{X}^{(T)}\rVert_F}{\lVert\mathbf{X}^*\rVert_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert\mathbf{A}^*-\mathbf{A}^{(T)}\rVert_F}{\lVert\mathbf{A}^*\rVert_F}$	$\frac{\lVert\mathbf{X}^*-\mathbf{X}^{(T)}\rVert_F}{\lVert\mathbf{X}^*\rVert_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert\mathbf{A}^*-\mathbf{A}^{(T)}\rVert_F}{\lVert\mathbf{A}^*\rVert_F}$	$\frac{\lVert\mathbf{X}^*-\mathbf{X}^{(T)}\rVert_F}{\lVert\mathbf{X}^*\rVert_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert\mathbf{A}^*-\mathbf{A}^{(T)}\rVert_F}{\lVert\mathbf{A}^*\rVert_F}$	$\frac{\lVert\mathbf{X}^*-\mathbf{X}^{(T)}\rVert_F}{\lVert\mathbf{X}^*\rVert_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$
100	`NOODL`	5.38e-11	2.38e-16	245 (Y)	7.04e-11	2.24e-16	257 (Y)	5.48e-11	5.14e-13	240 (Y)	7.82e-11	1.79e-12	257 (Y)	8.30e-11	6.39e-13	300 (Y)
	`Arora(b)`	1.87e-06	1.14e-05	245 (N)	2.09e-03	1.41e-03	257 (N)	2.70e-03	2.41e-03	240 (N)	3.80e-03	3.20e-03	257 (N)	2.80e-03	3.06e-03	300 (N)
	`Arora(u)`	6.78e-08	1.14e-05	245 (N)	8.94e-05	7.38e-05	257 (N)	1.72e-04	8.76e-05	240 (N)	3.06e-04	1.82e-04	257 (N)	2.52e-04	2.76e-04	300 (N)
	`Mairal`	4.40e-03	2.00e-03	245 (N)	4.90e-03	6.87e-03	257 (N)	6.00e-03	5.10e-03	240 (N)	7.20e-03	6.90e-03	257 (N)	8.27e-03	8.07e-03	300 (N)
300	`NOODL`	5.72e-11	1.13e-12	61 (Y)	6.74e-11	5.44e-13	89 (Y)	9.10e-11	1.27e-12	168 (Y)	9.43e-11	1.56e-12	201 (Y)	9.50e-11	1.63e-12	265 (Y)
	`Arora(b)`	2.13e-03	2.86e-03	61 (N)	5.90e-04	4.50e-04	89 (N)	1.00e-03	1.10e-03	168 (N)	9.77e-04	1.04e-03	201 (N)	1.03e-03	9.36e-04	265 (N)
	`Arora(u)`	2.04e-04	2.70e-04	61 (N)	3.82e-05	4.26e-05	89 (N)	1.04e-04	1.09e-04	168 (N)	1.42e-04	1.68e-04	201 (N)	1.27e-04	1.23e-04	265 (N)
	`Mairal`	2.05e-01	2.28e-01	61 (N)	1.19e-02	1.09e-02	89 (N)	1.07e-02	8.40e-03	168 (N)	1.47e-02	1.39e-02	201 (N)	9.40e-03	1.05e-02	265 (N)
500	`NOODL`	5.49e-11	2.34e-16	50 (Y)	8.15e-11	1.25e-12	76 (Y)	9.27e-11	1.41e-12	160 (Y)	9.77e-11	1.60e-12	196 (Y)	9.72e-11	1.84e-12	264 (Y)
	`Arora(b)`	1.11e-04	1.34e-04	50 (N)	5.75e-04	5.60e-04	76 (N)	6.32e-04	2.71e-03	160 (N)	5.99e-04	5.30e-03	196 (N)	6.04e-04	6.37e-03	264 (N)
	`Arora(u)`	9.75e-06	1.50e-05	50 (N)	4.30e-05	4.73e-05	76 (N)	5.55e-05	2.28e-03	160 (N)	5.91e-05	5.30e-03	196 (N	8.08e-05	6.37e-03	264 (N)
	`Mairal`	1.23e-01	1.10e-01	50 (N)	1.73e-02	1.20e-02	76 (N)	1.44e-02	5.99e-02	160 (N)	3.22e-01	2.87e-01	196 (N)	2.46e-02	1.70e-01	264 (N)

Table 3.E.3: Tensor factorization results $\alpha, \beta = 0.01$ averaged across 3 trials. Here, $T(\mathrm{supp}(\widehat{\mathbf{X}}))$ field shows the number of iterations T to reach the target tolerance, while the categorical field, $\mathrm{supp}(\widehat{\mathbf{X}})$ indicates if the support of the recovered $\widehat{\mathbf{X}}$ matches that of $\mathbf{X}^*$ (Y) or not (N).

(J, K)	**Method**	$m = 50$			$m = 150$			$m = 300$			$m = 450$			$m = 600$		
		$\frac{\lVert \mathbf{A}^* - \mathbf{A}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{A}^* \rVert_\mathrm{F}}$	$\frac{\lVert \mathbf{X}^* - \mathbf{X}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{X}^* \rVert_\mathrm{F}}$	$T(\mathrm{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert \mathbf{A}^* - \mathbf{A}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{A}^* \rVert_\mathrm{F}}$	$\frac{\lVert \mathbf{X}^* - \mathbf{X}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{X}^* \rVert_\mathrm{F}}$	$T(\mathrm{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert \mathbf{A}^* - \mathbf{A}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{A}^* \rVert_\mathrm{F}}$	$\frac{\lVert \mathbf{X}^* - \mathbf{X}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{X}^* \rVert_\mathrm{F}}$	$T(\mathrm{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert \mathbf{A}^* - \mathbf{A}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{A}^* \rVert_\mathrm{F}}$	$\frac{\lVert \mathbf{X}^* - \mathbf{X}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{X}^* \rVert_\mathrm{F}}$	$T(\mathrm{supp}(\widehat{\mathbf{X}})?)$	$\frac{\lVert \mathbf{A}^* - \mathbf{A}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{A}^* \rVert_\mathrm{F}}$	$\frac{\lVert \mathbf{X}^* - \mathbf{X}^{(T)} \rVert_\mathrm{F}}{\lVert \mathbf{X}^* \rVert_\mathrm{F}}$	$T(\mathrm{supp}(\widehat{\mathbf{X}})?)$
100	`NOODL`	5.50e-11	5.66e-13	91 (Y)	7.59e-11	5.28e-13	112 (Y)	4.34e-11	1.62e-12	190 (Y)	9.48e-11	1.78e-12	211 (Y)	7.27e-11	1.94e-12	279 (Y)
	`Arora(b)`	3.93e-03	5.80e-03	91 (N)	2.61e-03	1.58e-03	112 (N)	2.70e-03	3.00e-03	190 (N)	3.30e-03	4.00e-03	211 (N)	3.40e-03	3.37e-03	279 (N)
	`Arora(u)`	4.35e-04	6.77e-04	91 (N)	6.87e-04	1.05e-04	112 (N)	2.98e-04	3.04e-04	190 (N)	8.55e-04	1.27e-03	211 (N)	6.83e-04	6.49e-04	279 (N)
	`Mairal`	4.03e-02	1.26e-02	91 (N)	1.34e-02	1.25e-02	112 (N)	1.18e-02	1.25e-02	190 (N)	8.00e-03	6.60e-03	211 (N)	8.77e-03	9.93e-03	279 (N)
300	`NOODL`	6.78e-11	5.75e-13	51 (Y)	6.35e-11	1.54e-12	76 (Y)	8.64e-11	2.06e-12	158 (Y)	9.43e-11	2.92e-12	192 (Y)	9.33e-11	2.54e-12	252 (Y)
	`Arora(b)`	4.08e-04	4.76e-04	51 (N)	1.03e-03	1.08e-03	76 (N)	1.04e-03	1.17e-02	158 (N)	1.00e-03	1.25e-02	192 (N)	1.13e-03	1.54e-02	252 (N)
	`Arora(u)`	1.99e-05	1.46e-05	51 (N)	1.03e-04	9.59e-05	76 (N)	2.17e-04	1.17e-02	158 (N)	2.22e-04	1.25e-02	192 (N)	2.69e-04	1.54e-02	252 (N)
	`Mairal`	1.64e-01	1.63e-01	51 (N)	2.61e-02	2.64e-02	76 (N)	2.81e-02	1.58e-01	158 (N)	1.39e-01	2.03e-01	192 (N)	1.92e-02	1.83e-01	252 (N)
500	`NOODL`	6.92e-11	8.78e-13	46 (Y)	8.77e-11	1.77e-12	77 (Y)	9.35e-11	2.12e-12	156 (Y)	9.60e-11	2.41e-12	186 (Y)	9.82e-11	2.66e-12	249 (Y)
	`Arora(b)`	3.48e-04	3.28e-04	46 (N)	5.42e-04	6.40e-03	77 (N)	5.69e-04	2.41e-03	156 (N)	6.49e-04	1.20e-02	186 (N)	6.55e-04	1.42e-02	249 (N)
	`Arora(u)`	2.56e-05	3.70e-05	46 (N)	4.81e-05	6.40e-03	77 (N)	1.08e-04	9.30e-03	156 ((N)	1.39e-04	1.20e-02	186 (N)	1.55e-04	1.42e-02	249 (N)
	`Mairal`	1.56e-01	1.53e-01	46 (N)	5.28e-02	1.30e-01	77 (N)	2.53e-02	1.57e-01	156 (N)	6.38e-02	1.54e-01	186 (N)	1.74e-02	1.79e-01	249 (N)

Table 3.E.4: Tensor factorization results $\alpha, \beta = 0.05$ averaged across 3 trials. Here, $T(\text{supp}(\widehat{\mathbf{X}}))$ field shows the number of iterations T to reach the target tolerance, while the categorical field, $\text{supp}(\widehat{\mathbf{X}})$ indicates if the support of the recovered $\widehat{\mathbf{X}}$ matches that of $\mathbf{X}^*$ (Y) or not (N).

(J, K)	**Method**	$m = 50$			$m = 150$			$m = 300$			$m = 450$			$m = 600$		
		$\frac{\\|\mathbf{A}^*-\mathbf{A}^{(T)}\\|_F}{\\|\mathbf{A}^*\\|_F}$	$\frac{\\|\mathbf{X}^*-\mathbf{X}^{(T)}\\|_F}{\\|\mathbf{X}^*\\|_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\\|\mathbf{A}^*-\mathbf{A}^{(T)}\\|_F}{\\|\mathbf{A}^*\\|_F}$	$\frac{\\|\mathbf{X}^*-\mathbf{X}^{(T)}\\|_F}{\\|\mathbf{X}^*\\|_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\\|\mathbf{A}^*-\mathbf{A}^{(T)}\\|_F}{\\|\mathbf{A}^*\\|_F}$	$\frac{\\|\mathbf{X}^*-\mathbf{X}^{(T)}\\|_F}{\\|\mathbf{X}^*\\|_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\\|\mathbf{A}^*-\mathbf{A}^{(T)}\\|_F}{\\|\mathbf{A}^*\\|_F}$	$\frac{\\|\mathbf{X}^*-\mathbf{X}^{(T)}\\|_F}{\\|\mathbf{X}^*\\|_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$	$\frac{\\|\mathbf{A}^*-\mathbf{A}^{(T)}\\|_F}{\\|\mathbf{A}^*\\|_F}$	$\frac{\\|\mathbf{X}^*-\mathbf{X}^{(T)}\\|_F}{\\|\mathbf{X}^*\\|_F}$	$T(\text{supp}(\widehat{\mathbf{X}})?)$
100	`NOODL`	8.03e-11	3.17e-12	46 (Y)	7.71e-11	4.92e-12	63 (Y)	9.66e-11	6.01e-12	110 (Y)	8.92e-11	7.29e-12	115 (Y)	8.71e-11	1.06e-11	131 (Y)
	`Arora(b)`	2.90e-03	3.00e-03	46 (N)	4.60e-03	3.39e-02	63 (N)	5.50e-03	4.89e-02	110 (N)	7.50e-03	6.17e-02	115 (N)	9.16e-03	7.36e-02	131 (N)
	`Arora(u)`	8.97e-04	8.48e-04	46 (N)	1.90e-03	3.40e-02	63 (N)	2.80e-03	4.90e-02	110 (N)	4.40e-03	6.19e-02	115 (N)	5.70e-03	7.40e-02	131 (N)
	`Mairal`	1.57e-01	1.67e-01	46 (N)	3.63e-02	1.54e-01	63 (N)	2.32e-02	1.99e-01	110 (N)	8.79e-02	2.27e-01	115 (N)	2.81e-02	2.56e-01	131 (N)
300	`NOODL`	6.51e-11	3.27e-12	42 (Y)	9.05e-11	5.61e-12	60 (Y)	9.10e-11	7.01e-12	107 (Y)	9.20e-11	8.41-12	110 (Y)	8.49e-11	9.03e-12	128 (Y)
	`Arora(b)`	1.40e-03	1.95e-02	42 (N)	2.50e-03	3.55e-02	60 (N)	3.20e-03	5.04e-02	107 (N)	4.00e-03	6.16e-02	110 (N)	4.90e-03	7.39e-02	128 (N)
	`Arora(u)`	2.48e-04	1.95e-02	42 (N)	6.35e-04	3.56e-02	60 (N)	9.48e-04	5.05e-02	107 (N)	1.40e-03	6.18e-02	110 (N)	1.83e-03	7.42e-02	128 (N)
	`Mairal`	6.24e-02	1.11e-01	42 (N)	3.05e-02	1.59e-01	60(N)	1.91e-02	2.09e-01	107 (N)	4.85e-02	2.19e-01	110 (N)	2.32e-02	2.63e-01	128 (N)
500	`NOODL`	7.72e-11	3.86e-12	42 (Y)	8.44e-11	5.63e-12	59 (Y)	9.64e-11	7.34e-12	106 ((Y)	8.95e-11	8.21e-12	109 (Y)	9.06e-11	9.29e-12	127 (Y)
	`Arora(b)`	1.30e-03	2.02e-02	42 (N)	2.10e-03	3.55e-02	59 (N)	2.80e-03	5.03e-02	106 (N)	3.60e-03	6.21e-02	109 (N)	4.40e-03	7.40e-02	127 (N)
	`Arora(u)`	1.39e-04	2.02e-02	42 (N)	3.82e-04	3.56e-02	59 (N)	5.66e-04	5.05e-02	106 (N)	8.54e-04	6.23e-02	109 (N)	1.10e-03	7.44e-02	127 (N)
	`Mairal`	6.12e-02	1.10e-01	42 (N)	2.93e-02	1.58e-01	59 (N)	1.80e-02	2.11e-01	106 (N)	4.62e-02	2.20e-01	109 (N)	4.05e-02	2.56e-01	127 (N)

Part II

Algorithm-Agnostic Matrix Demixing

Chapter 4

Dictionary-based Generalization of Robust PCA

4.1 Overview

We consider the decomposition of a data matrix assumed to be a superposition of a low-rank matrix and a component which is sparse in a known dictionary, using a convex demixing method. We consider two sparsity structures for the sparse factor of the dictionary sparse component, namely entry-wise and column-wise sparsity, and provide a unified analysis, encompassing both undercomplete and the overcomplete dictionary cases, to show that the constituent matrices can be successfully recovered under some relatively mild conditions on incoherence, sparsity, and rank. We corroborate our theoretical results by presenting empirical evaluations in terms of phase transitions in rank and sparsity, in comparison to related techniques. Investigation of a specific application in hyperspectral imaging is included in Chapter 5.

4.2 Introduction

Leveraging structure of a given dataset is at the heart of all machine learning and data analysis tasks. *A priori* knowledge about the structure often makes the problem well-posed, leading to improvements in the solutions. Perhaps the most common of these, one that is often encountered in practice, is approximate low-rankness of the dataset, which is exploited by the popular principal component analysis (PCA) (Jolliffe, 2002).

The low-rank structure encapsulates the model assumption that the data in fact spans a lower dimensional subspace than the ambient dimension of the data. However, in a number of applications, the data may not be inherently low-rank, but may be decomposed as a superposition of a low-rank component, and a component which has a sparse representation in a known *dictionary*. This scenario is particularly interesting in target identification applications (Rambhatla et al., 2017b; Li et al., 2018b), where the *a priori* knowledge of the target *signatures* (dictionary), can be leveraged for localization.

In this work, we analyze a matrix demixing problem where a data matrix $\mathbf{M} \in \mathbb{R}^{n\times m}$ is formed via a superposition of a low-rank component $\mathbf{L} \in \mathbb{R}^{n\times m}$ of rank-r for $r < \min(n, m)$, and a dictionary sparse part $\mathbf{DS} \in \mathbb{R}^{n\times m}$. Here, the matrix $\mathbf{D} \in \mathbb{R}^{n\times d}$ is an *a priori* known dictionary, and $\mathbf{S} \in \mathbb{R}^{d\times m}$ is an unknown *sparse* coefficient matrix. Specifically, we will study the following model for $\mathbf{M}$:

$$\mathbf{M} = \mathbf{L} + \mathbf{DS}, \tag{4.1}$$

and identify the conditions under which the components $\mathbf{L}$ and $\mathbf{S}$ can be successfully recovered given $\mathbf{M}$ and $\mathbf{D}$ by solving appropriate convex formulations.

We consider the demixing problem described above for two different sparsity models on the matrix $\mathbf{S}$. First, we consider a case where $\mathbf{S}$ has at most s_e total non-zero entries (entry-wise sparse case), and second where $\mathbf{S}$ has s_c non-zero columns (column-wise sparse case). To this end, we develop the conditions under which solving

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda_e \|\mathbf{S}\|_1 \;\; \text{s.t.} \;\; \mathbf{M} = \mathbf{L} + \mathbf{DS}, \tag{D-RPCA(E)}$$

for the entry-wise sparsity case, and

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda_c \|\mathbf{S}\|_{1,2} \; \text{s.t.} \; \mathbf{M} = \mathbf{L} + \mathbf{DS}, \tag{D-RPCA(C)}$$

for the column-wise sparse case, will recover $\mathbf{L}$ and $\mathbf{S}$ for regularization parameters $\lambda_e \geq 0$ and $\lambda_c \geq 0$, respectively, given the data $\mathbf{M}$ and the dictionary $\mathbf{D}$. Here, the known dictionary $\mathbf{D}$ can be overcomplete (*fat*, i.e., $d > n$) or undercomplete (*thin*, i.e., $d \leq n$).

Here, "D-RPCA" refers to "Dictionary based Robust Principal Component Analysis", while the qualifiers "E" and "C" indicate the entry-wise and column-wise sparsity patterns, respectively. In addition, $\|.\|_*$, $\|.\|_1$, and $\|.\|_{1,2}$ refer to the nuclear norm,

ℓ_1- norm of the vectorized matrix, and $\ell_{1,2}$ norm (sum of the ℓ_2 norm of the columns), respectively, which serve as convex relaxations of rank, sparsity, and column-wise sparsity inducing optimization, respectively.

These two types of sparsity patterns capture different structural properties of the dictionary sparse component. The entry-wise sparsity model allows individual data points to span low-dimensional subspaces, still allowing the dataset to span the entire space. In case of the column-wise sparse coefficient matrix **S**, the component **DS** is also column-wise sparse. Therefore, this model effectively captures the structured (which depend upon the dictionary **D**) corruptions in the otherwise low-rank structured columns of data matrix **M**. Note that the non-zero columns of **S** are not restricted to be sparse in the column-wise sparsity model.

4.2.1 Background

A wide range of problems can be expressed in the form described in (4.1). Perhaps the most celebrated of these is principal component analysis (PCA) (Jolliffe, 2002), which can be viewed as a special case of (4.1), with the matrix **D** set to zero. Next, in the absence of the component **L**, the problem reduces to that of sparse recovery (Natarajan, 1995; Donoho and Huo, 2001b; Candès and Tao, 2005); See Rauhut (2010) and references therein for an overview of related works. Further, the popular framework of Robust PCA tackles a case when the dictionary **D** is an identity matrix (Candès et al., 2011; Chandrasekaran et al., 2011); variants include Zhou et al. (2010); Ding et al. (2011); Wright et al. (2013); Chen et al. (2013).

The model described in (4.1) is also closely related to the one considered in Mardani et al. (2013), which explores the overcomplete dictionary setting with applications to detection of network traffic anomalies. However, the analysis therein applies to a case where the dictionary **D** is overcomplete with orthogonal rows, and the coefficient matrix **S** has a small number of non-zero elements per row and column, which may be restrictive assumptions in some applications.

In particular, for the entry-wise case, the model shown in (4.1) is propitious in a number of applications. For example, it can be used for target identification in hyperspectral imaging (Rambhatla et al., 2017b; Li et al., 2018b), and in topic modeling applications to identify documents with certain properties, on similar lines as Min et al. (2010). We analyze and demonstrate the application of this model for a hyperspectral demixing task in an application extension of this work in Rambhatla et al. (2018b).

Further, in source separation tasks, a variant of this model was used in singing voice separation in Huang et al. (2012); Sprechmann et al. (2012). In addition, we can also envision source separation tasks where $\mathbf{L}$ is not low-rank, but can in turn be modeled as being sparse in a known (Starck et al., 2005) or an unknown (Rambhatla and Haupt, 2013b) dictionary.

For the column-wise setting, model (4.1) is also closely related to outlier identification (Xu et al., 2010; Li and Haupt, 2015a,b; Rahmani and Atia, 2015), which is motivated by a number of contemporary "big data" applications. Here, the sparse matrix $\mathbf{S}$, also called outliers in this regime, may be of interest and can be used in identifying malicious responses in collaborative filtering applications (Mehta and Nejdl, 2008), finding anomalous patterns in network traffic (Lakhina et al., 2004) or estimating visually salient regions of images (Itti et al., 1998; Harel et al., 2006; Liu et al., 2007).

4.2.2 Our Contributions

As described above, we propose and analyze a dictionary based generalization of robust PCA as shown in (4.1). Here, we consider two distinct sparsity patterns of $\mathbf{S}$, i.e., entry-wise and column-wise sparse $\mathbf{S}$, arising from different structural assumptions on the dictionary sparse component. Our specific contributions for each sparsity pattern are summarized below.

Entry-wise case: We make the following contributions towards guaranteeing the recovery of $\mathbf{L}$ and $\mathbf{S}$ via the convex optimization problem in D-RPCA(E). First, we analyze the *thin* case (i.e. $d \leq n$), where we assume that the matrix $\mathbf{S}$ has at most $s_e = \mathcal{O}(\frac{m}{r})$ non-zero elements *globally*, i.e., $\|\mathbf{S}\|_0 \leq s_e$ Next, for the *fat* case, we first extend the analysis presented in Mardani et al. (2013) to eliminate the orthogonality constraint on the rows of the dictionary $\mathbf{D}$. Further, we relax the sparsity constraints required by Mardani et al. (2013) on rows and columns of the sparse coefficient matrix $\mathbf{S}$, to study the case when $\|\mathbf{S}\|_0 \leq s_e$ with at most $k = \mathcal{O}(d/\log(n))$ non-zero elements per column (Rambhatla et al., 2016b). Hence, we provide a unified analysis for both the *thin* and the *fat* case, making the model (4.1) amenable to a wide range of applications.

Column-wise case: We propose and analyze a dictionary based generalization of robust PCA, specifically *Outlier Pursuit* (OP) (Xu et al., 2010), wherein the coefficient matrix $\mathbf{S}$ admits a column sparse structure which can be viewed as "outliers"; see also Li et al. (2018b).

Note that, in this case there is an inherent ambiguity regarding the recovery of the true component pair $(\mathbf{L}, \mathbf{S})$ corresponding to the low-rank part and the dictionary sparse component, respectively. Specifically, any pair $(\mathbf{L}_0, \mathbf{S}_0)$ satisfying $\mathbf{M} = \mathbf{L}_0 + \mathbf{D}\mathbf{S}_0 = \mathbf{L} + \mathbf{D}\mathbf{S}$, where $\mathbf{L}_0$ and $\mathbf{L}$ have the same column space, and $\mathbf{S}_0$ and $\mathbf{S}$ have the identical column support, is a solution of D-RPCA(C). To this end, we develop the sufficient conditions under which solving the convex optimization in D-RPCA(C) recovers the column space of the low-rank component $\mathbf{L}$, while identifying the outlier columns of $\mathbf{S}$. Here, the difference between D-RPCA(C) and OP being the inclusion of the known dictionary (Xu et al., 2010).

Next, we demonstrate how the *a priori* knowledge of the dictionary $\mathbf{D}$ helps us identify the corrupted columns via phase transitions in rank and sparsity for recovery of the outlier columns. Specifically, we show that in comparison to OP, D-RPCA(C) works for potentially higher ranks of $\mathbf{L}$, when s_c is a fixed proportion of m.

The *thin* dictionary case – an interesting result: As suggested by Mardani et al. (2013), when the dictionary is *thin*, i.e., $d < n$, one can envision a pseudo-inversed based technique, wherein we pre-multiply both sides in (4.1) with the Moore-Penrose pseudo-inverse $\mathbf{D}^{\dagger} \in \mathbb{R}^{d \times n}$, i.e., $\mathbf{D}^{\dagger}\mathbf{D} = \mathbf{I}$ (this is not applicable for the *fat* case due to the non-trivial null space of the pseudo-inverse). This operation leads to a formulation which resembles the robust PCA (RPCA) (Candès et al., 2011; Chandrasekaran et al., 2011) model for the entry-wise case and Outlier Pursuit (OP) (Xu et al., 2010) for the column-wise case, i.e.,

$$\mathbf{D}^{\dagger}\mathbf{M} = \mathbf{D}^{\dagger}\mathbf{L} + \mathbf{S}, \qquad (\text{RPCA}^{\dagger}) \qquad\qquad \mathbf{D}^{\dagger}\mathbf{M} = \mathbf{D}^{\dagger}\mathbf{L} + \mathbf{S}. \qquad (\text{OP}^{\dagger})$$

An interesting finding of our work is that although this transformation algebraically reduces the entry-wise and column-wise sparsity cases to Robust PCA and OP settings, respectively, the specific model assumptions of Robust PCA and OP may not hold for all choices of dictionary size d and rank r. Specifically, we find that in cases where $d < r$, this pre-multiplication may not lead to a "low-rank" $\mathbf{D}^{\dagger}\mathbf{L}$. This suggests that the notion of "low" or "high" rank is relative to the maximum possible rank of $\mathbf{D}^{\dagger}\mathbf{L}$, which in this case is $\min(d, r)$. Therefore, if $d < r$, $\mathbf{D}^{\dagger}\mathbf{L}$ can be full-rank, and the low-rank assumptions of RPCA and OP may no longer hold. As a result, these two models (the pseudo inversed case and the current work) cannot be used interchangeably for the thin dictionary case. We corroborate these via experimental evaluations presented in

Section 4.5.

The rest of the chapter is organized as follows. We formalize the problem, introduce the notation, and describe various considerations on the structure of the component matrices in Section 4.3. In Section 4.4, we present our main theorems for the entry-wise and column-wise cases along with discussion on the implication of the results, followed by an outline of the analysis in Section 4.B. Numerical evaluations are provided in Section 4.5. Finally, we summarize our contributions and conclude this discussion in Section 4.6 with insights on future work.

4.3 Preliminaries

We start formalizing the problem set-up and introduce model parameters pertinent to our analysis. We begin our discussion with our notion of optimality for the two sparsity modalities; we also summarize the notation in Table 4.A.1 in the appendix.

4.3.1 Optimality of the Solution Pair

For the entry-wise case, we recover the low-rank component $\mathbf{L}$, and the sparse coefficient matrix $\mathbf{S}$, given the dictionary $\mathbf{D}$, and data $\mathbf{M}$ generated according to the model described in (4.1). Recall that s_e is the global sparsity, k denotes the number of non-zero entries in a column of $\mathbf{S}$ when the dictionary is *fat*.

In the the column-wise sparsity setting, due to the inherent ambiguity in the model (4.1), as discussed in Section 4.2.2, we can only hope to recover the column-space for the low-rank matrix and the identities of the non-zero columns for the sparse matrix. Therefore, in this case any solution in the *Oracle Model* (defined below) is deemed to be optimal.

Definition 4.1 (Oracle Model for Column-wise Sparsity Case)**.** Let the pair $(\mathbf{L}, \mathbf{S})$ be the matrices forming the data $\mathbf{M}$ as per (4.1), define the oracle model $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$. Then, any pair $(\mathbf{L}_0, \mathbf{S}_0)$ is in the *Oracle Model* $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$, if $\mathcal{P}_{\mathcal{U}}(\mathbf{L}_0) = \mathbf{L}$, $\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}\mathbf{S}_0) = \mathbf{D}\mathbf{S}$ and $\mathbf{L}_0 + \mathbf{D}\mathbf{S}_0 = \mathbf{L} + \mathbf{D}\mathbf{S} = \mathbf{M}$ hold simultaneously, where $\mathcal{P}_{\mathcal{U}}$ and $\mathcal{P}_{\mathcal{S}_c}$ are projections onto the column space $\mathcal{U}$ of $\mathbf{L}$ and column support $\mathcal{I}_{\mathcal{S}_c}$ of $\mathbf{S}$, respectively.

4.3.2 Conditions on the Dictionary

We require that the dictionary $\mathbf{D}$ follows the *generalized frame property* (GFP) defined as follows.

Definition 4.2. A matrix $\mathbf{D}$ satisfies the *generalized frame property* (GFP), on vectors $\mathbf{v} \in \mathcal{R}$, if for any fixed vector $\mathbf{v} \in \mathcal{R}$ where $\mathbf{v} \neq \mathbf{0}$, we have

$$\alpha_\ell \|\mathbf{v}\|_2^2 \leq \|\mathbf{D}\mathbf{v}\|_2^2 \leq \alpha_u \|\mathbf{v}\|_2^2,$$

where α_ℓ and α_u are the lower and upper *generalized frame bounds* with $0 < \alpha_\ell \leq \alpha_u < \infty$.

The GFP shown above is met as long as the vectors $\mathbf{v}$ are not in the null-space of the matrix $\mathbf{D}$, and $\mathbf{D}$ has a finite $\|\mathbf{D}\|$. Therefore, for the *thin* dictionary setting $d \leq n$ for both entry-wise and column-wise cases $\mathcal{R}$ can be the entire space, and GFP is satisfied as long as $\mathbf{D}$ has full column rank. For example, $\mathbf{D}$ being a *frame*(Duffin and Schaeffer, 1952) suffices; see Heil (2013) for a brief overview of frames.

On the other hand, for the *fat* dictionary setting, we need the space $\mathcal{R}$ to be structured such that the GFP is met for both the entry-wise and column-wise sparsity cases. Specifically, for the entry-wise sparsity case, we also require that the frame bounds α_u and α_ℓ be close to each other. To this end, we assume that $\mathbf{D}$ satisfies the *restricted isomtery property* (RIP) of order $k = \mathcal{O}(d/\log(n))$ with a *restricted isometric constant* (RIC) of δ in this case, and that $\alpha_u = (1 + \delta)$ and $\alpha_\ell = (1 - \delta)$.

4.3.3 Relevant Subspaces

We now define the subspaces relevant for our discussion. For the following discussion, let the pair $(\mathbf{L_0}, \mathbf{S_0})$ denote the solution to D-RPCA(E) in the entry-wise sparse case. Further, for the column-wise sparse setting, let $(\mathbf{L_0}, \mathbf{S_0})$ denote a solution pair in the oracle model $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$ as defined in **D.4.1**, obtained by solving D-RPCA(C).

For the low-rank matrix $\mathbf{L}$, let the compact singular value decomposition (SVD) be defined as

$$\mathbf{L} = \mathbf{U}\Sigma\mathbf{V}^\top,$$

where $\mathbf{U} \in \mathbb{R}^{n \times r}$ and $\mathbf{V} \in \mathbb{R}^{m \times r}$ are the left and right singular vectors of $\mathbf{L}$, respectively, and Σ is the diagonal matrix with singular values on the diagonal. Here, matrices $\mathbf{U}$ and $\mathbf{V}$ each have orthogonal columns, and the non-negative entries $\Sigma_{ii} = \sigma_i$ are arranged in descending order. We define $\mathcal{L}$ as the linear subspace consisting of matrices spanning the same row or column space as $\mathbf{L}$, i.e.,

$$\mathcal{L} := \{\mathbf{U}\mathbf{W}_1^\top + \mathbf{W}_2\mathbf{V}^\top, \mathbf{W}_1 \in \mathbb{R}^{m\times r}, \mathbf{W}_2 \in \mathbb{R}^{n\times r}\}.$$

Next, let $\mathcal{S}_e$ ($\mathcal{S}_c$ for the column-wise sparsity setting) be the space spanned by $d \times m$ matrices with the same non-zero support (column support, denoted as $\text{csupp}(\cdot)$) as $\mathbf{S}$, and let the space $\mathcal{D}$ denote the space spanned by the dictionary sparse component under our model be defined as

$$\mathcal{D} := \{\mathbf{D}\mathbf{H}\}, \text{where} \begin{cases} \mathbf{H} \in \mathcal{S}_e \text{ for entry-wise case,} \\ \text{csupp}(\mathbf{H}) \subseteq \mathcal{I}_{\mathcal{S}_c} \text{ for column-wise case.} \end{cases}$$

Here, $\mathcal{I}_{\mathcal{S}_c}$ denotes the index set containing the non-zero column index set of $\mathbf{S}$ for the column-wise case.

Also, we denote the corresponding complements of the spaces described above by appending '$\perp$'. In addition, we use calligraphic '$\mathcal{P}_{\mathcal{G}}(\cdot)$' to denote the projection operator onto a subspace $\mathcal{G}$, and '$\mathbf{P}_{\mathbf{G}}$' to denote the corresponding projection matrix. For instance, we define $\mathcal{P}_{\mathcal{U}}(\cdot)$ and $\mathcal{P}_{\mathcal{V}}(\cdot)$ as the projection operators corresponding to the column space $\mathcal{U}$ and row space $\mathcal{V}$ of the low-rank component $\mathbf{L}$. Therefore, for a given matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$,

$$\mathcal{P}_{\mathcal{U}}(\mathbf{X}) = \mathbf{P}_{\mathbf{U}}\mathbf{X} \text{ and } \mathcal{P}_{\mathcal{V}}(\mathbf{X}) = \mathbf{X}\mathbf{P}_{\mathbf{V}},$$

where $\mathbf{P}_{\mathbf{U}} = \mathbf{U}\mathbf{U}^\top$ and $\mathbf{P}_{\mathbf{V}} = \mathbf{V}\mathbf{V}^\top$. With this, the projection operators onto, and orthogonal to the subspace $\mathcal{L}$ are respectively defined as

$$\begin{aligned} \mathcal{P}_{\mathcal{L}}(\mathbf{X}) &= \mathbf{P}_{\mathbf{U}}\mathbf{X} + \mathbf{X}\mathbf{P}_{\mathbf{V}} - \mathbf{P}_{\mathbf{U}}\mathbf{X}\mathbf{P}_{\mathbf{V}}, \text{ and} \\ \mathcal{P}_{\mathcal{L}^\perp}(\mathbf{X}) &= (\mathbf{I} - \mathbf{P}_{\mathbf{U}})\mathbf{X}(\mathbf{I} - \mathbf{P}_{\mathbf{V}}). \end{aligned}$$

4.3.4 Incoherence Measures and Other Parameters

We employ various notions of incoherence to identify the conditions under which our procedures succeed. To this end, we first define the incoherence parameter μ, that characterizes the relationship between the low-rank part, $\mathbf{L}$, and the dictionary sparse part **DS** as,

$$\mu := \max_{\mathbf{Z}\in\mathcal{D}\backslash\{\mathbf{0}\}} \tfrac{\|\mathcal{P}_{\mathcal{L}}(\mathbf{Z})\|_{\mathrm{F}}}{\|\mathbf{Z}\|_{\mathrm{F}}}. \tag{4.2}$$

The parameter $\mu \in [0,1]$ is the measure of degree of similarity between the low-rank part and the dictionary sparse component. Here, a larger μ implies that the dictionary sparse component is close to the low-rank part, while a small μ indicates otherwise. In addition, we also define the parameter β_U as

$$\beta_{\mathbf{U}} := \max_{\|\mathbf{u}\|=1} \tfrac{\|(\mathbf{I}-\mathbf{P}_{\mathbf{U}})\mathbf{D}\mathbf{u}\|^2}{\|\mathbf{D}\mathbf{u}\|^2}, \tag{4.3}$$

which measures the similarity between the orthogonal complement of the column-space $\mathcal{U}$ and the dictionary **D**.

The next two measures of incoherence can be interpreted as a way to identify the cases where for **L** with SVD as $\mathbf{L} = \mathbf{U}\Sigma\mathbf{V}^\top$: (a) **U** resembles the dictionary **D**, and/or (b) **V** resembles the sparse coefficient matrix **S**. In these cases, the low-rank part may mimic the dictionary sparse component. To this end, similar to Mardani et al. (2013), we define the following to measure these properties respectively as

$$\text{(a) } \gamma_{\mathbf{U}} := \max_i \tfrac{\|\mathbf{P}_{\mathbf{U}}\mathbf{D}\mathbf{e}_i\|^2}{\|\mathbf{D}\mathbf{e}_i\|^2} \text{ and (b) } \gamma_{\mathbf{V}} := \max_i \|\mathbf{P}_{\mathbf{V}}\mathbf{e}_i\|^2. \tag{4.4}$$

Here, $\gamma_{\mathbf{U}} \leq 1$, and achieves the upper bound when a dictionary element is exactly aligned with the column space $\mathcal{U}$ of **L**. Moreover, $\gamma_{\mathbf{V}}\in [r/nm, 1]$ achieves the upper bound when the row-space of **L** is "spiky", i.e., a certain row of **V** is 1-sparse, meaning that a column of **L** is supported by (can be expressed as a linear combination of) a column of **U**. The lower bound here is attained when it is "spread-out", i.e., each column of **L** is a linear combination of all columns of **U**. In general, our recovery of the two components is easier when the incoherence parameters $\gamma_{\mathbf{U}}$ and $\gamma_{\mathbf{V}}$ are closer to their lower bounds. Further, for notational convenience, we define constants

$$\xi_e := \|\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top\|_\infty \text{ and } \xi_c := \|\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top\|_{\infty,2}. \tag{4.5}$$

Here, ξ_e is the maximum absolute entry of $\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top$, which measures how close columns of **D** are to the singular vectors of **L**. Similarly, for the column-wise case, ξ_c measures the closeness of columns of **D** to the singular vectors of **L** under a different metric (column-wise maximum ℓ_2-norm).

4.4 Main Results

We present the main results corresponding to each sparsity structure of $\mathbf{S}$ in this section. We provide detailed proofs in Appendix 4.B.

4.4.1 Exact Recovery for Entry-wise Sparsity Case

Our main result establishes the existence of a regularization parameter λ_e, for which solving the optimization problem D-RPCA(E) will recover the components $\mathbf{L}$ and $\mathbf{S}$ exactly. To this end, we will show that such a λ_e belongs to a non-empty interval $[\lambda_e^{\min}, \lambda_e^{\max}]$ with $\lambda_e^{\min}$ and $\lambda_e^{\max}$ defined as

$$\lambda_e^{\min} := \tfrac{1+C_e}{1-C_e}\,\xi_e \text{ and } \lambda_e^{\max} := \tfrac{\sqrt{\alpha_\ell}(1-\mu)-\sqrt{r\alpha_u}\mu}{\sqrt{s_e}}, \tag{4.6}$$

where $0 \leq C_e < 1$ is a constant that captures the relationship between different model parameters, and is defined as

$$C_e := \tfrac{c}{\alpha_\ell(1-\mu)^2-c},$$

and c is defined as

$$c := \begin{cases} c_t = \frac{\alpha_u((1+2\gamma_{\mathbf{U}})(\min(s_e,d)+s_e\gamma_{\mathbf{V}})+2\gamma_{\mathbf{V}}\min(s_e,m))}{2} - \frac{\alpha_\ell(\min(s_e,d)+s_e\gamma_{\mathbf{V}})}{2}, & \text{for } d \leq n, \\ c_f = \frac{\alpha_u((1+2\gamma_{\mathbf{U}})(k+s_e\gamma_{\mathbf{V}})+2\gamma_{\mathbf{V}}\min(s_e,m))}{2} - \frac{\alpha_\ell(k+s_e\gamma_{\mathbf{V}})}{2}, & \text{for } d > n. \end{cases}$$

Given these definitions, we formalize the theorem for the entry-wise case as following, and its corresponding analysis is provided in Section 4.B.1.

Theorem 4.1. Suppose $\mathbf{M} = \mathbf{L} + \mathbf{D}\mathbf{S}$, where $\text{rank}(\mathbf{L}) = r$ and $\mathbf{S}$ has at most s_e non-zeros, i.e., $\|\mathbf{S}\|_0 \leq s_e \leq s_e^{\max} := \frac{(1-\mu)^2}{2}\frac{m}{r}$. Given $\mu \in [0,1]$, $\gamma_{\mathbf{U}}$, $\gamma_{\mathbf{V}} \in [r/m, 1]$, ξ_e defined in (4.2), (4.4), (4.5), and any $\lambda_e \in [\lambda_e^{\min}, \lambda_e^{\max}]$ with $\lambda_e^{\max} > \lambda_e^{\min} \geq 0$ defined in (4.6), the dictionary $\mathbf{D} \in \mathbb{R}^{n\times d}$ obeys the generalized frame property **D.4.2** with frame bounds $[\alpha_\ell, \alpha_u]$, solving D-RPCA(E) will recover matrices $\mathbf{L}$ and $\mathbf{S}$ if the following conditions hold:

- For $d \leq n$, $\mathcal{R}$ may contain the entire space and $\gamma_{\mathbf{U}}$ follows

$$\gamma_{\mathbf{U}} \leq \begin{cases} \frac{(1-\mu)^2 - 2s_e\gamma_{\mathbf{V}}}{2s_e(1+\gamma_{\mathbf{V}})}, \text{ for } s_e \leq \min\ (d, s_e^{\max}) \\ \frac{(1-\mu)^2 - 2s_e\gamma_{\mathbf{V}}}{2(d+s_e\gamma_{\mathbf{V}})}, \text{ for } d < s_e \leq s_e^{\max} \end{cases} ; \tag{4.7}$$

- For $d > n > C_1\ k\log(n)$ for a constant C_1, $\mathcal{R}$ consists of all k sparse vectors, and $\gamma_{\mathbf{U}}$ follows

$$\gamma_{\mathbf{U}} \leq \frac{(1-\mu)^2 - 2s_e\gamma_{\mathbf{V}}}{2(k+s_e\gamma_{\mathbf{V}})}. \tag{4.8}$$

Theorem 4.1 establishes the sufficient conditions for the existence of λ_e to guarantee recovery of $(\mathbf{L}, \mathbf{S})$ for both the *thin* and the *fat* cases. The conditions on $\gamma_{\mathbf{U}}$ dictated by (4.7) and (4.8), for the thin and fat case, respectively, arise from ensuring that $\lambda_e^{\min} \geq 0$. Further, the condition $\lambda_e^{\min} < \lambda_e^{\max}$, translates to the following sufficient condition on rank r in terms of the sparsity s_e,

$$r < \left(\sqrt{\frac{\alpha_\ell}{\alpha_u}} \frac{1-\mu}{\mu} - \frac{\xi_e}{\sqrt{\alpha_u}\mu} \frac{1+C_e}{1-C_e} \sqrt{s_e} \right)^2 , \tag{4.9}$$

for the recovery of $(\mathbf{L}, \mathbf{S})$. This relationship matches with our empirical evaluations and will be revisited in Section 4.5.1.

We note that for both, *thin* and *fat* dictionary case, the conditions are closely related to the incoherence measures (μ, $\gamma_{\mathbf{V}}$, and $\gamma_{\mathbf{U}}$) between the low-rank part, $\mathbf{L}$, the dictionary, $\mathbf{D}$, and the sparse component $\mathbf{S}$. In general, smaller sparsity, rank, and incoherence parameters are sufficient for ensuring the recovery of the components for a particular problem. This is in line with our intuition – the more distinct the two components, the easier it should be to tease them apart. Moreover, we observe that the theorem imposes an an upper-bound on the global sparsity, i.e., $s_e \leq s_e^{\max} = \mathcal{O}(\frac{m}{r})$. This bound is similar to the result in Xu et al. (2010), and is due to the deterministic nature of our analysis w.r.t. the locations of the non-zero elements of coefficients $\mathbf{S}$.

4.4.2 Exact Recovery for Column-wise Sparsity Case

Recall that we consider the oracle model in this case as described in **D.**4.1 owing to the intrinsic ambiguity in recovery of $(\mathbf{L}, \mathbf{S})$; see our discussion in Section 4.2.2. To demonstrate its recoverability, the following lemma establishes the sufficient conditions for the existence of an optimal pair $(\mathbf{L}_0, \mathbf{S}_0)$. The proof is provided in Appendix 4.C.2.

Lemma 4.1. Given $\mathbf{M}$, $\mathbf{D}$, and $(\mathcal{L}, \mathcal{S}_c, \mathcal{D})$, any pair $(\mathbf{L}_0, \mathbf{S}_0) \in \{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$ satisfies $\text{span}\{\text{col}(\mathbf{L}_0)\} = \mathcal{U}$ and $\text{csupp}(\mathbf{S}_0) = \mathcal{I}_{\mathcal{S}_c}$ if $\mu < 1$.

Analogous to the entry-wise case, we will show the existence of a non-empty interval $[\lambda_c^{\min}, \lambda_c^{\max}]$ for the regularization parameter λ_c, for which solving D-RPCA(C) recovers an optimal pair as per Lemma 4.1. Here, for a constant $C_c := \frac{\alpha_u}{\alpha_\ell}\frac{1}{(1-\mu)^2}\gamma_{\mathbf{V}}\beta_{\mathbf{U}}$, $\lambda_c^{\min}$ and $\lambda_c^{\max}$ are defined as

$$\lambda_c^{\min} := \tfrac{\xi_c + \sqrt{r s_c \alpha_u}\mu C_c}{1 - s_c C_c} \text{ and } \lambda_c^{\max} := \tfrac{\sqrt{\alpha_\ell}(1-\mu) - \sqrt{r\alpha_u}\mu}{\sqrt{s_c}}. \tag{4.10}$$

Then, our main result for the column-wise case is as follows, and its analysis is provided in Section 4.B.2.

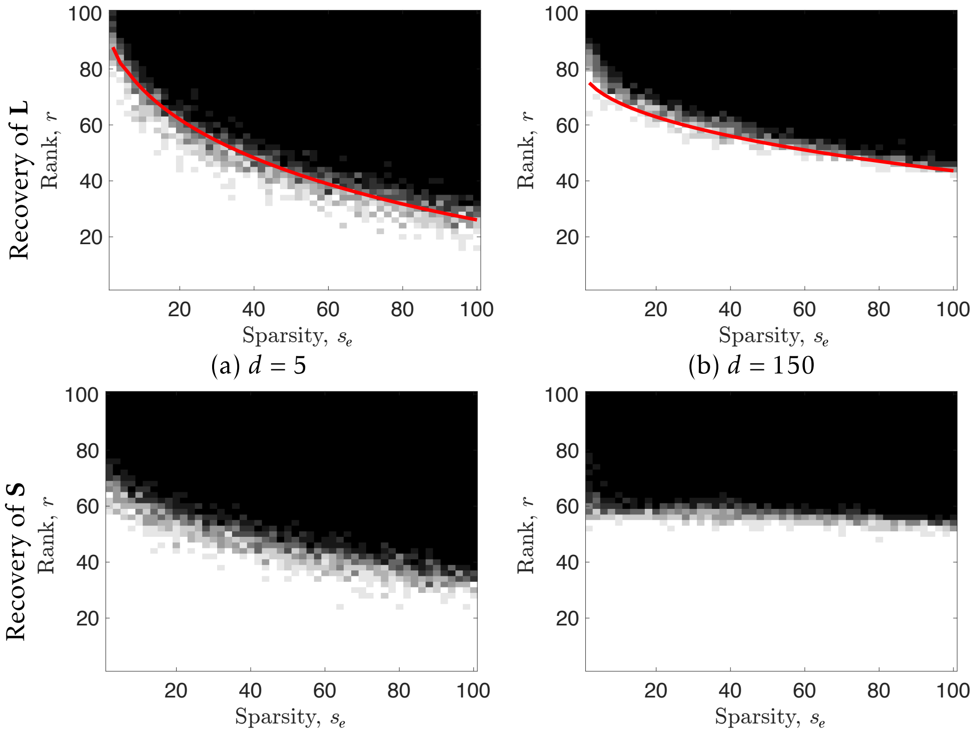
Theorem 4.2. Suppose $\mathbf{M} = \mathbf{L} + \mathbf{DS}$ with $(\mathbf{L}, \mathbf{S})$ defining the oracle model $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$, where $\text{rank}(\mathbf{L}) = r$, $|\mathcal{I}_{\mathcal{S}_c}| = s_c$ for $s_c \leq s_c^{\max} := \frac{\alpha_\ell}{\alpha_u \gamma_{\mathbf{V}}} \cdot \frac{(1-\mu)^2}{\beta_{\mathbf{U}}}$. Given $\mu \in [0, 1)$, $\beta_{\mathbf{U}}$, $\gamma_{\mathbf{V}} \in [r/m, 1]$, ξ_c defined in (4.2), (4.3), (4.4), (4.5), and any $\lambda_c \in [\lambda_c^{\min}, \lambda_c^{\max}]$, for $\lambda_c^{\max} > \lambda_c^{\min} \geq 0$ defined in (4.10), solving D-RPCA(C) will recover a pair of components $(\mathbf{L}_0, \mathbf{S}_0) \in \{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$, if the space $\mathcal{R}$ is structured such that the dictionary $\mathbf{D} \in \mathbb{R}^{n \times d}$ obeys the generalized frame property **D.4.2** with frame bounds $[\alpha_\ell, \alpha_u]$, for $\alpha_\ell > 0$.

Theorem 4.2 states the conditions under which the solution to the optimization problem D-RPCA(C) will be in the oracle model defined in **D.4.1**. The condition on the column sparisty $s_c \leq s_c^{\max}$ is a result of the constraint that $\lambda_c^{\min} \geq 0$. Similar to (4.9), requiring $\lambda_c^{\max} > \lambda_c^{\min}$ leads to the following sufficient condition on the rank r in terms of the sparsity s_c,

$$r < \left(\sqrt{\frac{\alpha_\ell}{\alpha_u}}\frac{1-\mu}{\mu} - \frac{\xi_c}{\sqrt{\alpha_u}\mu}\sqrt{s_c}\right)^2. \tag{4.11}$$

Moreover, suppose that $1 \lesssim \alpha_l \leq \alpha_u \lesssim 1$, which can be easily met by a tight frame when $d < n$, or a RIP type condition when $d > n$. Further, if $\frac{(1-\mu)^2}{\beta_U}$ is a constant, then since $\gamma_{\mathbf{V}} = \Theta(\frac{r}{m})$, we have that $s_c^{\max} = \mathcal{O}(\frac{m}{r})$. This is of the same order with the upper bound of s_c in the Outlier Pursuit (OP) (Xu et al., 2010).

Our numerical results in Section 4.5 further show that D-RPCA(C) can be much more robust than OP, and may recover $\{\mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$ even when the rank of $\mathbf{L}$ is high and the number of outliers s_c is a constant proportion of m.



(a) $d = 5$

(b) $d = 150$

(c) $d = 5$

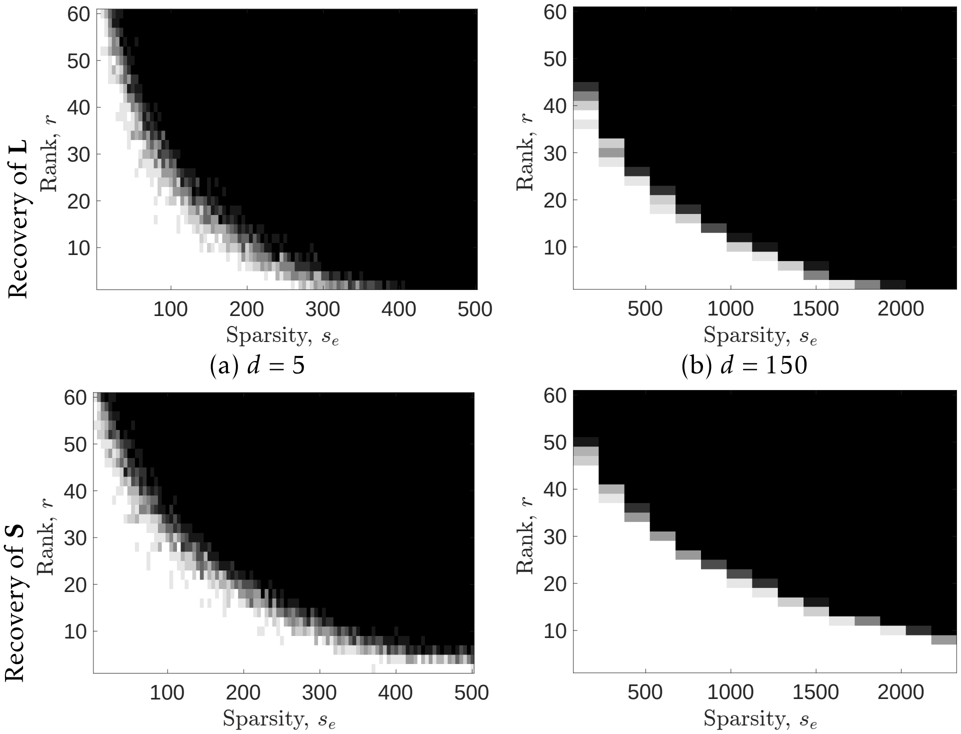
(d) $d = 150$

Figure 4.1: Recovery for varying rank of **L**, sparsity of **S** and number of dictionary elements in **D** as per Theorem 4.1. Each plot shows average recovery across 10 trials for varying ranks and sparsity up to $s_e^{\max} = m$, where $n = m = 100$ and the white region represents correct recovery. We decide success if $\|\mathbf{L} - \widehat{\mathbf{L}}\|_{\mathrm{F}}/\|\mathbf{L}\|_{\mathrm{F}} \leq 0.02$ and $\|\mathbf{S} - \widehat{\mathbf{S}}\|_{\mathrm{F}}/\|\mathbf{S}\|_{\mathrm{F}} \leq 0.02$, where $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{S}}$ are the recovered **L** and **S**, respectively. Panels (a)-(b) show the recovery of the low-rank part **L** and (c)-(d) show the recovery of the sparse part with varying dictionary sizes $d = 5$ and 150, respectively. Also, the predicted trend between rank r and sparsity s_e as per Theorem 4.1, eq.(4.9) is shown in red in panels (a-b).

4.5 Numerical Simulations

In this section, we empirically evaluate the properties of D-RPCA(E) and D-RPCA(C) via phase transition in rank and sparsity, and compare its performance to related techniques, and to the behavior predicted by Theorem 4.1 and Theorem 4.2 in (4.9) and (4.11), respectively [1].

[1] The code is made available at https://github.com/srambhatla/Dictionary-based-Robust-PCA; see Chapter 7 for details.



(a) $d = 5$

(b) $d = 150$

(c) $d = 5$

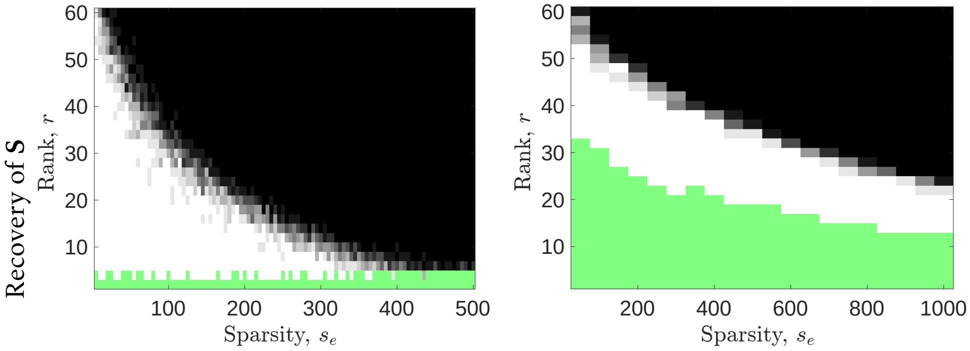
(d) $d = 150$

Figure 4.2: Recovery for varying rank of **L**, sparsity of **S** and number of dictionary elements in **R**. Panels (a)-(b) show the recovery of the low-rank part **L** and (c)-(d) show the recovery of the sparse part with varying dictionary sizes $d = 5$ and 150, respectively. The experimental set-up and the success metric remains the same as in Fig. 4.1.

4.5.1 Entry-Wise Sparsity Case

Experimental Set-up: We employ the accelerated proximal gradient (APG) algorithm outlined in Algorithm 1 of Mardani et al. (2013) and Algorithm 1 of Rambhatla et al. (2018b) to solve the optimization problem D-RPCA(E). For these evaluations, we fix $n = m = 100$, and generate the low-rank part **L** by outer product of two column normalized random matrices of sizes $n \times r$ and $m \times r$, with entries drawn from the standard normal distribution. In addition, we draw s_e non-zero entries of the sparse component **S** from the Rademacher distribution, and the dictionary **D** from the standard normal distribution with normalized columns. We then run 10 Monte-Carlo trials for each pair of rank and sparsity, and for each of these, we scan across 100 values of λ_es in the range of $[\lambda_e^{\min}, \lambda_e^{\max}]$ to find the best pair of $(\mathbf{L}_0, \mathbf{S}_0)$ to compile the results. [2]

[2] For ease of computation we run on modest values of n and m.



(a) $d = 5$ (b) $d = 50$

Figure 4.3: Comparison of phase transitions in rank and sparsity between D-RPCA(E) and RPCA† for recovery of **S** for different dictionary sizes. Panels (a) and (b) correspond to $d = 5$ and $d = 50$, respectively. Experimental set-up and the success metric remains same as Fig. 4.2. The area in green corresponds to recovery by RPCA† where at least 1 out of the 10 Monte-Carlo trials succeeds.

Discussion: Phase transition in rank and sparsity averaged over 10 trials for dictionaries of sizes $d = 5$ (thin) and $d = 150$ (fat), are shown in Fig. 4.1 and Fig. 4.2, respectively. We note from Fig. 4.1 that indeed the empirical relationship between rank and sparsity for the recovery of $(\mathbf{L_0}, \mathbf{S_0})$ has the same trend as predicted by

$$r < \left(\sqrt{\frac{\alpha_\ell}{\alpha_u}} \frac{1-\mu}{\mu} - \frac{\xi_e}{\sqrt{\alpha_u}\mu} \frac{1+C_e}{1-C_e} \sqrt{s_e} \right)^2,$$

as shown in (4.9) in Section 4.4 for $s_e \leq s_e^{\max}$. Here, the parameters corresponding to the predicted trend (shown in red) have been hand-tuned for best fit.

In fact, as shown in Fig. 4.2, this trend continues for sparsity levels much greater than $s_e^{\max}$. This can be potentially attributed to the limitations of the deterministic analysis (on the locations of the non-zero elements of **S**) presented here.

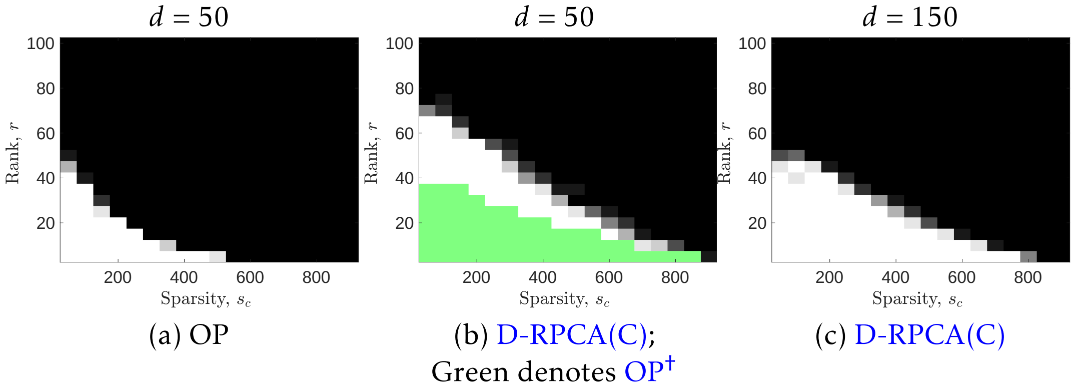
Further, Fig. 4.3 shows the results of RPCA† (in green, shows the area where at least one of the 10 Monte-Carlo simulations succeeds) in comparision to the results obtained by D-RPCA(E) for $d = 5$ and $d = 50$. We observe that D-RPCA(E) outperforms RPCA† across the board. In fact, we notice that the RPCA† technique only succeeds when $r < d$. We believe that this is because when $d < r$ the component $\mathbf{D}^\dagger\mathbf{L}$ is not low-rank (full-rank in this case) w.r.t. the maximum potential rank of $\mathbf{D}^\dagger\mathbf{L}$. As a result, the model assumptions of the robust PCA problem do not apply; see Section 4.2.2. In contrast, the proposed framework of D-RPCA(E) can handle these cases effectively (see Fig. 4.3) since $\mathbf{L}$ is low-rank irrespective of the dictionary size. This phenomenon highlights the applicability of the proposed approach to cases where $d < r$, and simultaneous recovery of the low-rank component in one-shot.

4.5.2 Column-wise Sparsity Case

We now present phase transition in rank r and number of outlierss_c to evaluate the performance of D-RPCA(C). In particular, we compare with Outlier Pursuit (OP) (Xu et al., 2010) that solves D-RPCA(C) with $\mathbf{D} = \mathbf{I}$, and OP† to demonstrate that the *a priori* knowledge of the dictionary provides superior recovery properties.

Experimental Set-up: Again, we employ a variant of the APG algorithm outlined in Algorithm 1 of Mardani et al. (2013) specialized for the column-wise sparsity case to solve the optimization problem D-RPCA(C); see Algorithm 1 of Rambhatla et al. (2018b). We set $n = 100$, $m = 1000$, and for each pair of r and s_c we run 10 Monte-Carlo trials for $r \in \{5, 10, 15 \ldots, 100\}$ and $s_c \in \{50, 100, 150, \ldots, 900\}$. For our experiments, we form $\mathbf{L} = [\mathbf{U}\mathbf{V}^\top \mathbf{0}_{n \times s_c}] \in \mathbb{R}^{n \times m}$, where $\mathbf{U} \in \mathbb{R}^{n \times r}$, $\mathbf{V} \in \mathbb{R}^{(m-s_c) \times r}$ have i.i.d. $\mathcal{N}(0,1)$ entries, which are then column normalized. Next, we generate $\mathbf{S} = [\mathbf{0}_{d \times (m-s_c)} \mathbf{W}] \in \mathbb{R}^{d \times m}$ where each entry of $\mathbf{W} \in \mathbb{R}^{d \times s_c}$ is i.i.d. $\mathcal{N}(0,1)$. Also, the known dictionary $\mathbf{D} \in \mathbb{R}^{n \times d}$ is formed by normalizing the columns of a random matrix with i.i.d. $\mathcal{N}(0,1)$ entries. For each method, we scan through 100 values of the regularization parameter $\lambda_c \in [\lambda_c^{\min}, \lambda_c^{\max}]$ to find a solution pair $(\mathbf{L}_0, \mathbf{S}_0)$ with the best precision, i.e., best detection of the outliers and rejection of false positives. We declare an experiment successful if it acheives a precision i.e. (True Positives/(True Positives + False Positives)) of 0.99 or higher. Here, we threshold the column norms at 2×10^{-3} before we evaluate the precision.

Discussion: Fig. 4.4 (a)-(c) shows the phase transition in rank r and column-sparsity s_c for the outlier identification performance (in terms of precison) of OP for $d = 50$, D-RPCA(C) for $d = 50$ (and OP† in green, marking the region where precision is greater than 0), and D-RPCA(C) for $d = 150$, respectively. We observe that the *a priori* knowledge of the dictionary $\mathbf{D}$ significantly boosts the performance of D-RPCA(C) as compared to OP. This showcases the superior outlier identification properties of the proposed technique D-RPCA(C). Further, similar to the entry-wise case, we note that the pseudo-inversed based technique OP† (in green) fails when $r > d$. For the $d = 150$ case the proposed technique D-RPCA(C) is able to identify the outlier columns with high precision. This is interesting since our technique succeeds even when the outlier



(a) OP

(b) D-RPCA(C); Green denotes OP†

(c) D-RPCA(C)

Figure 4.4: Phase transitions in rank r and column sparsity s_c across 10 Monte-Carlo simulations. Panels (a), (b), (c) show the precision i.e., (True Positives /(True Positives+False Positives)) in identifying the outlier columns of $\mathbf{S}$ for $d = 50$ using (a) OP and (b) OP†, and D-RPCA(C) for $d = 150$, respectively. In addition, panel (b) also shows the performance by OP† for $d = 50$ in green, marking the region where precision is greater than 0, super imposed over D-RPCA(C). Here, we threshold the column norms of the recovered matrix $\mathbf{S}$ at 2×10^{-3} before computing the precision, and a trial is declared successful if it achieves a precision of 0.99 or higher.

columns are not themselves sparse (we draw the entries of the outlier columns from $\mathcal{N}(0,1)$). This corroborates our theoretical assumption that $\mathcal{R}$ needs to be structured such that GFP is met.

Our empirical evaluations paves way to potential improvements via a probabilistic analysis of the model instead of the case considered here. Additionally, the recent results on non-convex low-rank matrix estimation formulations (Tu et al., 2015; Chen and Wainwright, 2015a) may potentially lead to computationally efficient algorithms by replacing the expensive SVD step in every iteration. Exploration of these extensions are left for future work.

4.6 Conclusions and Future Work

We analyze a dictionary based generalization of robust PCA. Here, we model the acquired data as a superposition of a low-rank component and a dictionary sparse component, considering two distinct sparsity patterns – entry-wise sparsity and column-wise sparsity, respectively. Specifically, for the entry-wise sparsity case, we extend the theoretical guarantees presented in Mardani et al. (2013) to a setting wherein the dictionary $\mathbf{D}$ may have arbitrary number of columns, and the coefficient matrix $\mathbf{S}$ has *global* sparsity of s_e, i.e. $\|\mathbf{S}\|_0 = s_e \le s_e^{\max}$, rendering the results useful for a potentially wide range of applications. Further, we propose a column-wise sparsity model, which can be viewed as a dictionary based generalization of Outlier Pursuit (Xu et al., 2010). For this case, we analyze and develop the conditions under which solving a convex program will recover the correct column-space of the low-rank part while identifying the outlier columns in the dictionary sparse part. To corroborate our theoretical results, we provide empirical evaluations via phase transition plots in rank and sparsity.

Appendices: Dictionary-based Generalization of Robust PCA

4.A Summary of Notation

In the following appendices, we provide the proofs of the lemmata employed to establish our main results. We also summarize the notation in Table 4.A.1.

4.B Proof of Main Results

4.B.1 Proofs for Entry-wise Case: Proof of Theorem 4.1

We use dual certificate construction procedure to prove the main result in Theorem. 4.1; the proofs of all lemmata used here are listed in Appendix 4.C.1. To this end, we start by constructing a dual certificate for the convex problem shown in D-RPCA(E). Here, we first show the conditions the dual certificate needs to satisfy via the following lemma.

Lemma 4.2. If there exists a dual certificate $\mathbf{\Gamma} \in \mathbb{R}^{n \times m}$ satisfying

$$\textbf{(C1)}\ \mathcal{P}_{\mathcal{L}}(\mathbf{\Gamma}) = \mathbf{U}\mathbf{V}^\top, \qquad \textbf{(C2)}\ \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{\Gamma}) = \lambda_e \operatorname{sign}(\mathbf{S}_0),$$
$$\textbf{(C3)}\ \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\| < 1,\ \textit{and}\ \textbf{(C4)}\ \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Gamma})\|_\infty < \lambda_e.$$

then the pair $(\mathbf{L}_0, \mathbf{S}_0)$ is the unique solution of D-RPCA(E).

We will now proceed with the construction of the dual certificate which satisfies the conditions outlined by **(C1)-(C4)** by Lemma 4.2. Using the analysis similar to

Table 4.A.1: Summary of important notation and parameters

Matrices	
$\mathbf{M} \in \mathbb{R}^{n\times m}$	The data matrix
$\mathbf{L} \in \mathbb{R}^{n\times m}$	The low-rank matrix with rank-r and singular value decomposition $\mathbf{L} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$
$\mathbf{D} \in \mathbb{R}^{n\times d}$	The known dictionary either *thin* ($d \leq n$) or *fat* ($d > n$)
$\mathbf{S} \in \mathbb{R}^{d\times m}$	The sparse component with the following properties –(1) in case of entry-wise sparsity: s_e non-zero entries and when $d > n$ has at most k non-zeros per column, and (2) in case of column-wise sparsity: s_c non-zero columns
Regularization Parameters	
$\lambda_e \in \mathbb{R}$	The regularization parameter for the entry-wise sparsity case
$\lambda_c \in \mathbb{R}$	The regularization parameter for the column sparsity case
Subspaces	
$\mathcal{L}$	The set of matrices which span the same column or row space as $\mathbf{L}$, i.e., $\mathcal{L} := \{\mathbf{U}\mathbf{W}_1^\top + \mathbf{W}_2\mathbf{V}^\top, \mathbf{W}_1 \in \mathbb{R}^{m\times r}, \mathbf{W}_2 \in \mathbb{R}^{n\times r}\}$.
$\mathcal{S}_e$	The set of matrices with the same support as $\mathbf{S}$ (for the entry-wise sparse case).
$\mathcal{S}_c$	The set of matrices with the same column support as $\mathbf{S}$ (for the column-wise sparse case).
$\mathcal{D}$	The set of matrices whose columns span the subspace spanned by columns of $\mathbf{D}$, i.e. $\mathcal{D} := \{\mathbf{Z} = \mathbf{R}\mathbf{H}, \mathbf{H} \in \mathcal{S}_e \text{ or } \mathbf{H} \in \mathcal{S}_c\}$
$\mathcal{U}$	The column space of $\mathbf{L}$
$\mathcal{V}$	The row space of $\mathbf{L}$
Index Sets	
$\mathcal{I}_{\mathcal{S}_e}$	Support of matrix $\mathbf{S}$ (entry-wise case)
$\mathcal{I}_{\mathcal{S}_c}$	Column support of matrix $\mathbf{S}$ (the outliers)
$\mathcal{I}_{\mathbf{L}}$	Index set of the inliers (column-wise case)
Projection	
$\mathcal{P}_{\mathcal{G}}(\cdot)$	Projection operator corresponding to any subspace $\mathcal{G}$
$\mathbf{P}_{\mathbf{G}}$	Projection matrix corresponding to the operator $\mathcal{P}_{\mathcal{G}}(\cdot)$
Parameters for analysis	
μ	The incoherence parameter between the low-rank component and the dictionary, defined as $\mu := \max\limits_{\mathbf{Z}\in\mathcal{D}\backslash\{\mathbf{0}_{d\times m}\}} \frac{\|\mathcal{P}_{\mathcal{L}}(\mathbf{Z})\|_{\mathrm{F}}}{\|\mathbf{Z}\|_{\mathrm{F}}}$
$\gamma_{\mathbf{V}}$	Defined as $\gamma_{\mathbf{V}} := \max\limits_i \|\mathbf{P}_{\mathbf{V}}\mathbf{e}_i\|^2$
$\gamma_{\mathbf{U}}$	Defined as $\gamma_{\mathbf{U}} := \max\limits_i \frac{\|\mathbf{P}_{\mathbf{U}}\mathbf{D}\mathbf{e}_i\|^2}{\|\mathbf{D}\mathbf{e}_i\|^2}$
$\beta_{\mathbf{U}}$	Defined as $\beta_{\mathbf{U}} := \max\limits_{\|\mathbf{u}\|=1} \frac{\|(\mathbf{I}-\mathbf{P}_{\mathbf{U}})\mathbf{D}\mathbf{u}\|^2}{\|\mathbf{D}\mathbf{u}\|^2}$
ξ_e	Defined as $\xi_e := \|\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top\|_\infty$
ξ_c	Defined as $\xi_c := \|\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top\|_{\infty,2}$
α_ℓ	Lower generalized frame bound
α_u	Upper generalized frame bound

Mardani et al. (2013) (Section V. B.), we construct the dual certificate as

$$\mathbf{\Gamma} = \mathbf{U}\mathbf{V}^\top + (\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V}),$$

for arbitrary $\mathbf{X} \in \mathbb{R}^{n\times m}$. The condition **(C1)** is readily satisfied by our choice of $\mathbf{\Gamma}$. For **(C2)**, we substitute the expression for $\mathbf{\Gamma}$ to arrive at

$$\begin{aligned}\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top) + \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top(\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V}))&\\ = \lambda_e \operatorname{sign}(\mathbf{S}_0).&\end{aligned} \tag{4.12}$$

Letting $\mathbf{Z} := \mathbf{D}^\top(\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V})$ and

$$\mathbf{B}_{\mathcal{S}_\mathbf{e}} := \lambda_e \operatorname{sign}(\mathbf{S}_0) - \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top),$$

we can write (4.12) as $\mathcal{P}_{\mathcal{S}_e}(\mathbf{Z}) = \mathbf{B}_{\mathcal{S}_\mathbf{e}}$. Further, we can vectorize the equation above as $\mathcal{P}_{\mathcal{S}_e}(\text{vec}(\mathbf{Z})) = \text{vec}(\mathbf{B}_{\mathcal{S}_\mathbf{e}})$. Let $\mathbf{b}_{\mathcal{S}_e}$ be a length s_e vector containing elements of $\mathbf{B}_{\mathcal{S}_e}$ corresponding to the support of $\mathbf{S}_0$. Now, note that vec($\mathbf{Z}$) can be represented in terms of a Kronecker product as follows,

$$\text{vec}(\mathbf{Z}) = [(\mathbf{I} - \mathbf{P_V}) \otimes \mathbf{D}^\top(\mathbf{I} - \mathbf{P_U})]\text{vec}(\mathbf{X}).$$

On defining $\mathbf{A} := (\mathbf{I} - \mathbf{P_V}) \otimes \mathbf{D}^\top(\mathbf{I} - \mathbf{P_U}) \in \mathbb{R}^{md\times mn}$, we have vec($\mathbf{Z}$) = $\mathbf{A}$vec($\mathbf{X}$). Further, let $\mathbf{A}_{\mathcal{S}_\mathbf{e}} \in \mathbb{R}^{s\times nm}$ denote the rows of $\mathbf{A}$ that correspond to support of $\mathbf{S}_0$, and let $\mathbf{A}_{\mathcal{S}_\mathbf{e}^\perp}$ correspond to the remaining rows of $\mathbf{A}$. Using these definitions and results, we have $\mathbf{A}_{\mathcal{S}_e}\text{vec}(\mathbf{X}) = \mathbf{b}_{\mathcal{S}_e}$. Thus, for conditions **(C1)** and **(C2)** to be satisfied, we need

$$\text{vec}(\mathbf{X}) = \mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1}\mathbf{b}_{\mathcal{S}_e}. \tag{4.13}$$

Here, the following result ensures the existence of the inverse.

Lemma 4.3. If $\mu < 1$ and $\alpha_\ell > 0$, $\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})$ satisfies the bound $\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e}) \geq \sqrt{\alpha_\ell}(1-\mu)$.

Now, we look at the condition **(C3)** $\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\| < 1$. This is where our analysis departs from Mardani et al. (2013); we write

$$\begin{aligned}\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\| &= \|(\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V})\|\\ &\leq \|\mathbf{X}\| \leq \|\mathbf{X}\|_{\text{F}} \leq \|\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1}\|\|\mathbf{b}_{\mathcal{S}_e}\|_2,\end{aligned}$$

where we have used the fact that $\|(\mathbf{I}-\mathbf{P_U})\| \leq 1$ and $\|(\mathbf{I}-\mathbf{P_V})\| \leq 1$. Now, as $\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1}$ is the pseudo-inverse of $\mathbf{A}_{\mathcal{S}_e}$, i.e., $\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1} = \mathbf{I}$, we have that $\|\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1}\| = 1/\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})$, where $\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})$ is the smallest singular value of $\mathbf{A}_{\mathcal{S}_e}$. Therefore, we have

$$\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\| \leq \frac{\|\mathbf{b}_{\mathcal{S}_e}\|_2}{\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})}. \tag{4.14}$$

The following lemma establishes an upper bound on $\|\mathbf{b}_{\mathcal{S}_e}\|_2$.

Lemma 4.4. $\|\mathbf{b}_{\mathcal{S}_e}\|_2$ satisfies the bound $\|\mathbf{b}_{\mathcal{S}_e}\|_2 \leq \lambda_e\sqrt{s_e} + \sqrt{r\alpha_u}\mu$.

Combining (4.14), Lemma 4.3, and Lemma 4.4, we have

$$\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\| \leq \frac{\lambda_e\sqrt{s_e}+\sqrt{r\alpha_u}\mu.}{\sqrt{\alpha_\ell}(1-\mu)}. \tag{4.15}$$

Now, combining (4.15) and the upper bound on λ_e defined in (4.6), we have that **(C3)** holds.

Now, we move onto finding conditions under which **(C4)** is satisfied by our dual certificate. For this we will bound $\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Gamma})\|_\infty$. Our analysis follows the similar procedure as employed in deriving (16) in Mardani et al. (2013), reproduced here for completeness. First, by the definition of $\mathbf{\Gamma}$ and properties of the $\|.\|_\infty$ norm, we have

$$\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Gamma})\|_\infty \leq \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{Z})\|_\infty + \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{UV})\|_\infty. \tag{4.16}$$

We now focus on simplifying the term $\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{Z})\|_\infty$. By definition of $\mathbf{A}$, and using the fact that $\text{vec}(\mathbf{Z}) = \mathbf{A}\text{vec}(\mathbf{X})$, we have $\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{Z}) = \mathbf{A}_{\mathcal{S}_e^\perp}\text{vec}(\mathbf{X})$, which implies

$$\begin{aligned}\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{Z})\|_\infty &= \|\mathbf{A}_{\mathcal{S}_e^\perp}\text{vec}(\mathbf{X})\|_\infty \\ &= \|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1}\mathbf{b}_{\mathcal{S}_e}\|_\infty,\end{aligned}$$

where we have used the result on $\text{vec}(\mathbf{X})$ shown in (4.13).

Now, since $\|\mathbf{b}_{\mathcal{S}_e}\|_\infty$ can be written as

$$\|\mathbf{b}_{\mathcal{S}_e}\|_\infty = \|\mathbf{B}_{\mathcal{S}_e}\|_\infty = \|\lambda_e\text{sign}(\mathbf{A}_0) - \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{UV}^\top)\|_\infty.$$

Now, using the following upper bound on $\|\mathbf{b}_{\mathcal{S}_e}\|_\infty$,

Lemma 4.5. $\|\mathbf{b}_{\mathcal{S}_e}\|_\infty$ satisfies the bound $\|\mathbf{b}_{\mathcal{S}_e}\|_\infty \leq \lambda_e + \|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{UV}^\top)\|_\infty$.

and on defining

$$\mathbf{Q} := \mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1},$$

we have

$$\begin{aligned}\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{Z})\|_\infty &= \|\mathbf{Q}\mathbf{b}_{\mathcal{S}_e}\|_\infty \leq \|\mathbf{Q}\|_{\infty,\infty}\|\mathbf{b}_{\mathcal{S}_e}\|_\infty\\ &= \|\mathbf{Q}\|_{\infty,\infty}\|\lambda_e \mathrm{sign}(\mathbf{A}_0) - \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\infty,\\ &\leq \|\mathbf{Q}\|_{\infty,\infty}(\lambda_e + \|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\infty),\end{aligned}$$

where we have the following bound for $\|\mathbf{Q}\|_{\infty,\infty}$.

Lemma 4.6. $\|\mathbf{Q}\|_{\infty,\infty}$ satisfies the bound $\|\mathbf{Q}\|_{\infty,\infty} \leq C_e(\alpha_u, \alpha_\ell, \gamma_{\mathbf{U}}, \gamma_{\mathbf{V}}, s_e, d, k, \mu)$, where

$$C_e := \tfrac{c}{\alpha_\ell(1-\mu)^2 - c}$$

where $0 \leq C_e < 1$ and c is defined as

$$c := \begin{cases} c_t = \frac{\alpha_u((1+2\gamma_{\mathbf{U}})(\min(s_e,d)+s_e\gamma_{\mathbf{V}})+2\gamma_{\mathbf{V}}\min(s_e,m))}{2} \\ \qquad\qquad - \frac{\alpha_\ell(\min(s_e,d)+s_e\gamma_{\mathbf{V}})}{2}, & \text{for } d \leq n, \\ c_f = \frac{\alpha_u((1+2\gamma_{\mathbf{U}})(k+s_e\gamma_{\mathbf{V}})+2\gamma_{\mathbf{V}}\min(s_e,m))}{2} \\ \qquad\qquad - \frac{\alpha_\ell(k+s_e\gamma_{\mathbf{V}})}{2}, & \text{for } d > n. \end{cases} \tag{4.17}$$

Combining this with (4.16) and Lemma 4.6, we have

$$\begin{aligned}\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Gamma})\|_\infty \leq C_e\Big(\lambda_e + \|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\infty\Big)&\\ + \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\infty.&\end{aligned} \tag{4.18}$$

By simplifying (4.18), we arrive at the lower bound $\lambda_e^{\min}$ for λ_e as in (4.6), from which **(C4)** holds. Gleaning from the expressions for $\lambda_e^{\max}$ and $\lambda_e^{\min}$, we observe that $\lambda_e^{\max} > \lambda_e^{\min} \geq 0$ for the existence of λ_e that can recover the desired matrices. This completes the proof. □

Characterizing $\lambda_e^{\min}$: In the previous section, we characterized the $\lambda_e^{\min}$ and $\lambda_e^{\max}$ based on the dual certificate construction procedure. For the recovery of the true pair $(\mathbf{L}, \mathbf{S})$, we require $\lambda_e^{\max} > \lambda_e^{\min} \geq 0$. Since $\xi_e \geq 0$ and $c \geq 0$ by definition, we need

$0 \le C_e < 1$ for $\lambda_e^{\min} > 0$, i.e.,

$$c < \tfrac{1}{2}\alpha_\ell(1-\mu)^2 \ge \tfrac{\alpha_\ell}{2}. \tag{4.19}$$

***Conditions for thin* D**: To simplify the analysis we assume, without loss of generality, that $d < m$. Specifically, we will assume that $d \le \frac{m}{\alpha r}$, where $\alpha > 1$ is a constant. With this assumption in mind, we will analyze the following cases for the global sparsity, when $s_e \le d$ and $d < s_e \le m$.

Case 1: $s_e \le d$. For this case c_t is given by

$$c_t = \tfrac{s_e\alpha_u}{2}[(1+2\gamma_{\mathbf{U}})(1+\gamma_{\mathbf{V}}) + 2\gamma_{\mathbf{V}}] - \tfrac{s_e\alpha_\ell}{2}(1+\gamma_{\mathbf{V}})$$

From (4.19), we have $\alpha_\ell(1-\mu)^2 - 2c_t > 0$, which leads to

$$\tfrac{\alpha_u}{\alpha_\ell} < \tfrac{(1-\mu)^2 + s_e(1+\gamma_{\mathbf{V}})}{s_e(1+2\gamma_{\mathbf{U}})(1+\gamma_{\mathbf{V}}) + 2s_e\gamma_{\mathbf{V}}}.$$

As per the GFP of **D.4.2**, we also require that $\alpha_u/\alpha_\ell \ge 1$. Therefore we arrive at

$$\gamma_{\mathbf{U}} < \tfrac{(1-\mu)^2 - 2s_e\gamma_{\mathbf{V}}}{2s_e(1+\gamma_{\mathbf{V}})}.$$

Further, since $\gamma_{\mathbf{U}} \ge 0$, we require the numerator to be positive, and since the lower bound on $\gamma_{\mathbf{V}} \ge \frac{r}{m}$, we have

$$s_e \le \tfrac{(1-\mu)^2}{2}\tfrac{m}{r} := s_e^{\max},$$

which also implies $s_e \le m$. Now, the condition $c_t \ge 0$ implies

$$\tfrac{\alpha_u}{\alpha_\ell} \ge \tfrac{1+\gamma_{\mathbf{V}}}{(1+2\gamma_{\mathbf{U}})(1+\gamma_{\mathbf{V}}) + 2\gamma_{\mathbf{V}}}.$$

Since, the R.H.S. of this inequality is upper bounded by 1 (achieved when $\gamma_{\mathbf{U}}$ and $\gamma_{\mathbf{V}}$ are zero). This condition on c_t is satisfied by our assumption that $\alpha_u/\alpha_\ell \ge 1$.

Case 2: $d < s_e \le m$. For this case, we have

$$c_t = \tfrac{\alpha_u}{2}((1+2\gamma_{\mathbf{U}})(d + s_e\gamma_{\mathbf{V}}) + 2s_e\gamma_{\mathbf{V}}) - \tfrac{\alpha_\ell}{2}(d + s_e\gamma_{\mathbf{V}}).$$

From (4.19), we have

$$\frac{\alpha_u}{\alpha_\ell} < \frac{(1-\mu)^2+(d+s_e\gamma_{\mathbf{V}})}{(1+2\gamma_{\mathbf{U}})(d+s_e\gamma_{\mathbf{V}})+2s_e\gamma_{\mathbf{V}}}.$$

Again, due to the requirement that $\alpha_u/\alpha_\ell \geq 1$, following a similar argument as in the previous case we conclude that

$$\gamma_{\mathbf{U}} \leq \frac{(1-\mu)^2-2s_e\gamma_{\mathbf{V}}}{2(d+s_e\gamma_{\mathbf{V}})} \text{ and } s_e \leq \frac{(1-\mu)^2}{2}\frac{m}{r}.$$

As in the previous case the $c_t \geq 0$ is met by our due to our assumption on the frame bounds.

***Conditions for fat* D**: To simplify the analysis, we suppose that $k < m$. Note that in this case, we require that the coefficient matrix **S** has k-sparse columns. Now, $c = c_f$ is given by

$$c_f := \frac{\alpha_u}{2}((1+2\gamma_{\mathbf{U}})(k+s_e\gamma_{\mathbf{V}})+2\gamma_{\mathbf{V}}s_e) - \frac{\alpha_l}{2}(k+s_e\gamma_{\mathbf{V}})$$

As for the *thin* case, we substitute the expression for c_f in (4.19) as follows

$$\frac{\alpha_u}{\alpha_\ell} < \frac{(1-\mu)^2+(k+s_e\gamma_{\mathbf{V}})}{(1+2\gamma_{\mathbf{U}})(k+s_e\gamma_{\mathbf{V}})+2s_e\gamma_{\mathbf{V}}}$$

Again, by GFP we require that $\alpha_u/\alpha_\ell \geq 1$, therefore we have

$$\gamma_{\mathbf{U}} < \frac{(1-\mu)^2-2s_e\gamma_{\mathbf{V}}}{2(k+s_e\gamma_{\mathbf{V}})} \text{ and } s_e \leq \frac{(1-\mu)^2}{2}\frac{m}{r},$$

Also, the condition that $c_f \geq 0$ is met by the assumption that **D** obeys GFP.

Characterizing $\lambda_e^{\max}$: Further, the condition $\lambda_e^{\min} < \lambda_e^{\max}$, translates to a relationship between rank r, and the sparsity s_e, as shown in (4.9) for $s_e \leq s_e^{\max}$.

4.B.2 Proofs for Column-wise Case: Proof of Theorem 4.2

In this section we prove Theorem 4.2; the proofs of all lemmata are listed in Appendix 4.C.2. The Lagrangian of the nonsmooth optimization problem D-RPCA(C) is

$$\mathcal{F}(\mathbf{L},\mathbf{S},\mathbf{\Lambda}) = \|\mathbf{L}\|_* + \lambda_c\|\mathbf{S}\|_{1,2} + \langle \mathbf{\Lambda}, \mathbf{M}-\mathbf{L}-\mathbf{DS}\rangle, \tag{4.20}$$

where $\mathbf{\Lambda} \in \mathbb{R}^{n\times m}$ is a dual variable. The subdifferentials of (4.20) with respect to $(\mathbf{L}, \mathbf{S})$ are

$$\partial_{\mathbf{L}}\mathcal{F}(\mathbf{L},\mathbf{S},\mathbf{\Lambda}) = \{\mathbf{U}\mathbf{V}^\top + \mathbf{W} - \mathbf{\Lambda}, \|\mathbf{W}\|_2 \leq 1, \mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}\},$$
$$\partial_{\mathbf{S}}\mathcal{F}(\mathbf{L},\mathbf{S},\mathbf{\Lambda}) = \Big\{\lambda_c\mathbf{H} + \lambda_c\mathbf{F} - \mathbf{D}^\top\mathbf{\Lambda}, \mathcal{P}_{\mathcal{S}_c}(\mathbf{H}) = \mathbf{H},$$
$$\mathbf{H}_{:,j} = \frac{\mathbf{S}_{:,j}}{\|\mathbf{S}_{:,j}\|_2}, \mathcal{P}_{\mathcal{S}_c}(\mathbf{F}) = \mathbf{0}, \|\mathbf{F}\|_{\infty,2} \leq 1\Big\}. \tag{4.21}$$

Then we claim that a pair $(\mathbf{L}, \mathbf{S})$ is an optimal point of D-RPCA(C) if and only if the following hold by the optimality conditions:

$$\mathbf{0}_{n\times m} \in \partial_{\mathbf{L}}\mathcal{F}(\mathbf{L},\mathbf{S},\mathbf{\Lambda}) \text{ and} \tag{4.22}$$
$$\mathbf{0}_{d\times m} \in \partial_{\mathbf{S}}\mathcal{F}(\mathbf{L},\mathbf{S},\mathbf{\Lambda}). \tag{4.23}$$

The following lemma states the optimality conditions for the optimal solution pair $(\mathbf{L}, \mathbf{S})$.

Lemma 4.7. Given $\mathbf{M}$ and $\mathbf{D}$, let $(\mathbf{L}, \mathbf{S})$ define the oracle model $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$. Then any solution $(\mathbf{L}_0, \mathbf{S}_0) \in \{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$ is the an optimal solution pair of D-RPCA(C), if there exists a dual certificate $\mathbf{\Gamma} \in \mathbb{R}^{n\times m}$ that satisfies
(C1) $\mathcal{P}_{\mathcal{L}}(\mathbf{\Gamma}) = \mathbf{U}\mathbf{V}^\top$, **(C2)** $\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{\Gamma}) = \lambda_c\mathbf{H}$, where $\mathbf{H}_{:,j} = \mathbf{S}_{:,j}/\|\mathbf{S}_{:,j}\|_2$ for all $j \in \mathcal{I}_{\mathcal{S}_c}$; $\mathbf{0}$, otherwise,
(C3) $\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\|_2 < 1$, and **(C4)** $\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\mathbf{\Gamma})\|_{\infty,2} < \lambda_c$.

We first propose $\mathbf{\Gamma}$ as the dual certificate, where

$$\mathbf{\Gamma} = \mathbf{U}\mathbf{V}^\top + (\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V}), \text{ for any } \mathbf{X} \in \mathbb{R}^{n\times m}.$$

Hence, the condition **(C1)** is readily satisfied by our choice of $\mathbf{\Gamma}$. Now, the condition **(C2)**, defined as $\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{\Gamma}) = \lambda_c\widetilde{\mathbf{S}}$, where $\widetilde{\mathbf{S}}_{:,j} = \frac{\mathbf{S}_{:,j}}{\|\mathbf{S}_{:,j}\|_2}$ for all $j \in \mathcal{I}_{\mathcal{S}_c}$; $\mathbf{0}$, otherwise. Substrituting the expression for $\mathbf{\Gamma}$, we need the following condition to hold

$$\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top) + \mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top(\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V})) = \lambda_c\widetilde{\mathbf{S}}. \tag{4.24}$$

Letting $\mathbf{Z} := \mathbf{D}^\top(\mathbf{I} - \mathbf{P_U})\mathbf{X}(\mathbf{I} - \mathbf{P_V})$ and $\mathbf{B}_{\mathcal{S}_c} := \lambda_c\widetilde{\mathbf{S}} - \mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)$, we have $\mathcal{P}_{\mathcal{S}_c}(\mathbf{Z}) = \mathbf{B}_{\mathcal{S}_c}$.

Further, vectorizing the equation above, we have

$$\mathcal{P}_{\mathcal{S}_c}(\mathrm{vec}(\mathbf{Z})) = \mathbf{b}_{\mathcal{S}_c}, \tag{4.25}$$

where $\mathbf{b}_{\mathcal{S}_c} := \mathrm{vec}(\mathbf{B}_{\mathcal{S}_c})$. Next, by letting $\mathbf{A} := (\mathbf{I} - \mathbf{P}_\mathbf{V}) \otimes \mathbf{D}^\top(\mathbf{I} - \mathbf{P}_\mathbf{U})$, using the definition of $\mathbf{Z}$ and the properties of the Kronecker product we have $\mathrm{vec}(\mathbf{Z}) = \mathbf{A}\mathrm{vec}(\mathbf{X})$. Now, let $\mathbf{A}_{\mathcal{S}_c}$ denote the rows of $\mathbf{A}$ corresponding to the non-zero rows of $\mathrm{vec}(\mathbf{S})$ and $\mathbf{A}_{\mathcal{S}_c^\perp}$ denote the remaining rows, then

$$\mathcal{P}_{\mathcal{S}_c}(\mathrm{vec}(\mathbf{Z})) = \mathbf{A}_{\mathcal{S}_c}\mathrm{vec}(\mathbf{X}). \tag{4.26}$$

From (4.25) and (4.26), we have $\mathbf{A}_{\mathcal{S}_c}\mathrm{vec}(\mathbf{X}) = \mathbf{b}_{\mathcal{S}_c}$. Therefore, we need the following

$$\mathrm{vec}(\mathbf{X}) = \mathbf{A}_{\mathcal{S}_c}^\top(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\mathbf{b}_{\mathcal{S}_c}, \tag{4.27}$$

which corresponds to the least norm solution i.e., $\mathbf{X} = \mathrm{argmin}_\mathbf{X} \|\mathbf{X}\|_\mathrm{F}$, s.t. $\mathbf{A}_{\mathcal{S}_c}\mathrm{vec}(\mathbf{X}) = \mathbf{b}_{\mathcal{S}_c}$). For this choice of $\mathbf{X}$ (4.24) is satisfied and consequently the condition **(C2)**. Here, the existence of the inverse is ensured by the following lemma.

Lemma 4.8. If $\mu < 1$ and $\alpha_\ell > 0$, the minimum singular values of $\mathbf{A}_{\mathcal{S}_c}$ is bounded away from 0 and is given by $\sqrt{\alpha_\ell}(1 - \mu)$

Upon the existence of such $\mathbf{X}$ as defined in (4.27), **(C3)** is satisfied if the following condition holds

$$\begin{aligned}\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\|_2 &= \|(\mathbf{I} - \mathbf{P}_\mathbf{V})\mathbf{X}(\mathbf{I} - \mathbf{P}_\mathbf{U})\|_2 \\ &\leq \|\mathbf{I} - \mathbf{P}_\mathbf{V}\|_2\|\mathbf{X}\|_2\|\mathbf{I} - \mathbf{P}_\mathbf{U}\|_2 = \|\mathbf{X}\|_2 \leq \|\mathbf{X}\|_\mathrm{F} < 1.\end{aligned}$$

From (4.27), this condition translates to

$$\|\mathbf{A}_{\mathcal{S}_c}^\top(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\|\|\|\mathbf{b}_{\mathcal{S}_c}\|_2 < 1.$$

Now, since $\|\mathbf{A}_{\mathcal{S}_c}^\top(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\|\| = 1/\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c})$ (see the analogous analysis for the entry-wise case), we need

$$\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\|_2 \leq \tfrac{\|\mathbf{b}_{\mathcal{S}_c}\|_2}{\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c})} < 1.$$

Now, using Lemma 4.8 and the following bound on $\|\mathbf{b}_{\mathcal{S}_c}\|_2$,

Lemma 4.9. $\|\mathbf{b}_{\mathcal{S}_c}\|_2$ is upper bounded by $\lambda_c\sqrt{s_c}+\sqrt{r\alpha_u}\mu$.

we have that the condition **(C3)** holds if

$$\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Gamma})\|_2 \leq \tfrac{\lambda_c\sqrt{s_c}+\sqrt{r\alpha_u}\mu}{\sqrt{\alpha_\ell}(1-\mu)} < 1,$$

which is satisfied by our choice of $\lambda_c^{\max}$ (4.10). Now, for the condition **(C4)** we need the following condition to hold true:

$$\begin{aligned}&\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\mathbf{\Gamma})\|_{\infty,2}\\ &\leq \|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_{\infty,2}+\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{X}(\mathbf{I}-\mathbf{P}_\mathbf{V}))\|_{\infty,2}\\ &= \|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_{\infty,2}+\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})\|_{\infty,2} < \lambda_c.\end{aligned}$$

Note that, here $\|\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^T\mathbf{U}\mathbf{V}^T)\|_{\infty,2} \leq \xi_c$. Therefore, using the following result,

Lemma 4.10. An upper bound on $\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})\|_{\infty,2}$ is given by $(\lambda_c s_c+\sqrt{r\alpha_u s_c}\mu)C_c$.

the condition **(C4)** implies that,

$$\xi_c + \tfrac{\alpha_u}{\alpha_\ell(1-\mu)^2}\sqrt{s_c}\gamma_\mathbf{V}\beta_\mathbf{U}(\lambda_c\sqrt{s_c}+\sqrt{r\alpha_u}\mu) < \lambda_c.$$

To this end, if we let $C_c := \frac{\alpha_u}{\alpha_\ell(1-\mu)^2}\gamma_\mathbf{V}\beta_\mathbf{U}$, **(C4)** is satisfied by $\lambda_c^{\min}$ defined in (4.10). This completes the proof. □

Characterizing $\lambda_c^{\min}$: From (4.10), we need $\lambda_c^{\min} := \frac{\xi_c+\sqrt{rs_c\alpha_u}\mu C_c}{1-s_cC_c} \geq 0$, where $C_c := \frac{\alpha_u}{\alpha_\ell(1-\mu)^2}\gamma_\mathbf{V}\beta_\mathbf{U} \geq 0$. Then from $s_cC_c < 1$, we require $s_c < s_c^{\max} := \frac{\alpha_\ell(1-\mu)^2}{\alpha_u\gamma_\mathbf{V}\beta_\mathbf{U}}$.

Characterizing $\lambda_c^{\max}$: Since we need $\lambda_c^{\min} < \lambda_c^{\max}$, substituting the expressions for $\lambda_c^{\min}$ and $\lambda_c^{\max}$, and using the fact that $s_cC_c < 1$, we arrive at (4.11).

4.C Proofs of Intermediate Results

4.C.1 Proofs for Entry-wise Case

We present the details of the proofs in this section for the entry-wise case. We first start by deriving the optimality conditions.

Proof of Lemma 4.2. Let $\{\mathbf{L}_0,\mathbf{S}_0\}$ be a solution of the problem posed above. Notice that this pair is not necessarily unique. For example, as shown in proof of Lemma 2 in

Mardani et al. (2013), $\{\mathbf{L}_0 + \mathbf{DH}, \mathbf{S}_0 - \mathbf{H}\}$, with arbitrary $\mathbf{H}$, is another feasible solution of the problem satisfying the optimality conditions (derived in this section).

We begin by writing the Lagrangian, $\mathcal{F}(\mathbf{L}, \mathbf{S}, \boldsymbol{\Lambda})$, for the given problem as follows.

$$\mathcal{F}(\mathbf{L}, \mathbf{S}, \boldsymbol{\Lambda}) = \|\mathbf{L}\|_* + \lambda_e \|\mathbf{S}\|_1 \ + \langle \boldsymbol{\Lambda},\ \mathbf{M} - \mathbf{L} - \mathbf{DS} \rangle,$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{n \times m}$ are the Lagrange multipliers.

Let the singular value decomposition (SVD) of $\mathbf{L}_0$ be represented as $\mathbf{U\Sigma V}^\top$. Then the sub-differential set of $\|\mathbf{L}\|_*$ can be represented as

$$\partial_{\mathbf{L}} \|\mathbf{L}\|_* \Big|_{\mathbf{L}=\mathbf{L}_0} = \{\mathbf{UV}^\top + \mathbf{W} : \|\mathbf{W}\| \leq 1, \mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}\},$$

as shown in Watson (1992). Also, the subdifferential set corresponding to $\|\mathbf{S}\|_1$ is given by

$$\partial_{\mathbf{S}} \|\mathbf{S}\|_1 \Big|_{\mathbf{S}=\mathbf{S}_0} = \{\text{sign}(\mathbf{S}_0) + \mathbf{F} : \|\mathbf{F}\|_\infty \leq 1, \mathcal{P}_{\mathcal{S}_e}(\mathbf{F}) = \mathbf{0}\},$$

Using these results, we write the sub-differential of the Lagrangian with respect to $\mathbf{L}$ and $\mathbf{S}$ at $\{\mathbf{L}_0, \mathbf{S}_0\}$ as

$$\begin{aligned}
\partial_{\mathbf{L}} \mathcal{F}(\mathbf{L}_0, \mathbf{S}_0, \boldsymbol{\Lambda}) &= \{\mathbf{UV}^\top + \mathbf{W} - \boldsymbol{\Lambda} : \|\mathbf{W}\| \leq 1, \mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}\}, \\
\partial_{\mathbf{S}} \mathcal{F}(\mathbf{L}_0, \mathbf{S}_0, \boldsymbol{\Lambda}) &= \{\lambda_{\mathbf{e}} \text{sign}(\mathbf{S}_0) + \lambda_{\mathbf{e}} \mathbf{F} - \mathbf{D}^\top \boldsymbol{\Lambda}, \|\mathbf{F}\|_\infty \leq 1, \\
&\qquad\qquad\qquad \mathcal{P}_{\mathcal{S}_e}(\mathbf{F}) = \mathbf{0}\}.
\end{aligned}$$

Then optimality conditions are

$$\mathbf{0}_{n \times m} \in \partial_{\mathbf{L}} \mathcal{F}(\mathbf{L}_0, \mathbf{S}_0, \boldsymbol{\Lambda}) \text{ and } \mathbf{0}_{d \times m} \in \partial_{\mathbf{S}} \mathcal{F}(\mathbf{L}_0, \mathbf{S}_0, \boldsymbol{\Lambda}),$$

which implies that the dual solution $\boldsymbol{\Lambda}$ must obey the following,

$$\begin{aligned}
\boldsymbol{\Lambda} &\in \mathbf{UV}^\top + \mathbf{W},\ \|\mathbf{W}\| \leq 1,\ \mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}_{n \times m} \text{ and} \\
\mathbf{D}^\top \boldsymbol{\Lambda} &\in \lambda_{\mathbf{e}} \text{sign}(\mathbf{S}_0) + \lambda_{\mathbf{e}} \mathbf{F},\ \|\mathbf{F}\|_\infty \leq 1,\ \mathcal{P}_{\mathcal{S}_e}(\mathbf{F}) = \mathbf{0}_{d \times m}.
\end{aligned}$$

Our aim here is to find the conditions on $\mathbf{W}$ and $\mathbf{F}$ such that the pair $\{\mathbf{L}_0,\ \mathbf{S}_0\}$ is a unique solution to the problem at hand.

Using these conditions, we see that $\mathcal{P}_{\mathcal{L}}(\mathbf{\Lambda}) = \mathbf{U}\mathbf{V}^\top$ and $\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{\Lambda}) = \lambda_e \text{sign}(\mathbf{S}_0)$; these correspond to conditions **(C1)** and **(C2)**, respectively. Now consider a feasible solution $\{\mathbf{L_0} + \mathbf{D}\mathbf{H}, \mathbf{S_0} - \mathbf{H}\}$ for a non-zero $\mathbf{H} \in \mathbb{R}^{d\times m}$. Let $\mathbf{W}$, with $\|\mathbf{W}\| = 1$ and $\mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}$, then by duality of norms,

$$\langle \mathbf{W},\ \mathbf{D}\mathbf{H}\rangle \ = \ \langle \mathbf{W},\ \mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\mathbf{H})\rangle \ = \ \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\mathbf{H})\|_*.$$

Further, let $\mathbf{F}$, with $\|\mathbf{F}\|_\infty = 1$ and $\mathcal{P}_{\mathcal{S}_e}(\mathbf{F}) = \mathbf{0}$, be such that

$$\mathbf{F}_{ij} = \begin{cases} -\text{sign}(\mathbf{H}_{ij}) & \text{, if } \{i,\ j\} \notin \mathcal{S}_e \text{ and } \mathbf{H}_{ij} \neq 0 \\ 0 & \text{, otherwise} \end{cases},$$

where $\mathbf{F}_{ij}$ denotes the $(i,j)^{\text{th}}$ element of $\mathbf{F}$. Then, we arrive at the following simplification for $\langle \mathbf{F},\ \mathbf{H}\rangle$ by duality of norms,

$$\langle \mathbf{F},\ \mathbf{H}\rangle \ = \ \langle \mathbf{F},\ \mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{H})\rangle \ = \ -\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{H})\|_1.$$

We first write the sub-gradient optimality condition,

$$\begin{aligned} \|\mathbf{L}_0 + \mathbf{D}\mathbf{H}\|_* + \lambda_e\ \|\mathbf{S_0} - \mathbf{H}\|_1 \geq\ & \|\mathbf{L}_0\|_* + \lambda_e\ \|\mathbf{S}_0\|_1 \\ & + \langle \mathbf{U}\mathbf{V}^\top + \mathbf{W},\ \mathbf{D}\mathbf{H}\rangle - \langle \lambda_e \text{sign}(\mathbf{S}_0) + \lambda_e \mathbf{F},\ \mathbf{H}\rangle. \end{aligned} \tag{4.28}$$

Next, we use the relationships derived above to simplify the following term:

$$\begin{aligned} &\langle \mathbf{U}\mathbf{V}^\top + \mathbf{W},\ \mathbf{D}\mathbf{H}\rangle - \langle \lambda_e \text{sign}(\mathbf{S}_0) + \lambda_e \mathbf{F},\ \mathbf{H}\rangle, \\ &= \langle \mathbf{W}, \mathbf{D}\mathbf{H}\rangle - \lambda_e \langle \mathbf{F}, \mathbf{H}\rangle + \langle \mathcal{P}_{\mathcal{L}}(\mathbf{\Lambda}), \mathbf{D}\mathbf{H}\rangle - \langle \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{\Lambda}), \mathbf{H}\rangle \\ &= \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\mathbf{H})\|_* + \lambda_e \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{H})\|_1 + \langle \mathcal{P}_{\mathcal{L}}(\mathbf{\Lambda}), \mathbf{D}\mathbf{H}\rangle - \langle \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{\Lambda}), \mathbf{H}\rangle \end{aligned}$$

We now simplify $\langle \mathcal{P}_{\mathcal{L}}(\mathbf{\Lambda}),\ \mathbf{D}\mathbf{H}\rangle - \langle \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{\Lambda}),\ \mathbf{H}\rangle$ using Holder's inequality.

$$\begin{aligned} &\langle \mathcal{P}_{\mathcal{L}}(\mathbf{\Lambda}), \mathbf{D}\mathbf{H}\rangle - \langle \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{\Lambda}), \mathbf{H}\rangle \\ &= \langle \mathbf{\Lambda} - \mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Lambda}),\ \mathbf{D}\mathbf{H}\rangle - \langle \mathbf{D}^\top\mathbf{\Lambda} - \mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Lambda}), \mathbf{H}\rangle \\ &\geq -\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\mathbf{H})\|_* \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Lambda})\| - \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Lambda})\|_\infty \|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{H})\|_1 \end{aligned}$$

Finally, we simplify the optimality condition in shown in (4.28),

$$\begin{aligned}&\|\mathbf{L}_0+\mathbf{DH}\|_* \;+\; \lambda_e\;\|\mathbf{S_0}-\mathbf{H}\|_1\\ &\geq\; \|\mathbf{L}_0\|_*+\lambda_e\;\|\mathbf{S}_0\|_1\; +(1-\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Lambda})\|)\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{DH})\|_*\\ &\qquad +(\lambda_e-\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Lambda})\|_\infty)\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{H})\|_1.\end{aligned}$$

Here, we note that if $\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{\Lambda})\|<1$ and $\|\mathcal{P}_{\mathcal{S}_e^\perp}(\mathbf{D}^\top\mathbf{\Lambda})\|_\infty<\lambda_e$, then the pair $\{\mathbf{L}_0,\mathbf{S}_0\}$ is the unique solution of the problem. Consequently, these are the required necessary conditions **(C3)** and **(C4)**, respectively. □

Proof of Lemma 4.3. First, note that we need $\mathbf{A}_{\mathcal{S}_e}$ to have full row rank, i.e, its smallest singular value should be greater than zero. To this end, we first derive a lower bound on the smallest singular value, $\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})$ of $\mathbf{A}_{\mathcal{S}_e}$ as follows:

$$\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})=\min_{\mathbf{H}\in\mathcal{S}_e\backslash\{\mathbf{0}\}}\frac{\|\mathbf{A}^\top\mathrm{vec}(\mathbf{H})\|}{\|\mathrm{vec}(\mathbf{H})\|}.$$

Now, using the definition of $\mathbf{A}^\top$ and properties of Kronecker products namely, transpose and vectorization of product of three matrices, we have

$$\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})=\min_{\mathbf{H}\in\mathcal{S}_e\backslash\{\mathbf{0}\}}\frac{\|(\mathbf{I}-\mathbf{P_U})\mathbf{DH}(\mathbf{I}-\mathbf{P_V})\|_{\mathrm{F}}}{\|\mathbf{H}\|_{\mathrm{F}}}.$$

Now, since $(\mathbf{I}-\mathbf{P_U})\mathbf{DH}(\mathbf{I}-\mathbf{P_V})=\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{DH})$,

$$\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})=\min_{\mathbf{H}\in\mathcal{S}_e\backslash\{\mathbf{0}\}}\frac{\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{DH})\|_{\mathrm{F}}}{\|\mathbf{DH}\|_{\mathrm{F}}}\frac{\|\mathbf{DH}\|_{\mathrm{F}}}{\|\mathbf{H}\|_{\mathrm{F}}}.$$

Using the GFP, we have the following lower bound:

$$\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})\geq\sqrt{\alpha_\ell}\min_{\mathbf{Z}\in\mathcal{D}\backslash\{\mathbf{0}\}}\frac{\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{Z})\|_{\mathrm{F}}}{\|\mathbf{Z}\|_{\mathrm{F}}}.$$

Further, simplifying using properties of the projection operator, the reverse triangle inequality and the definition of μ,

$$\begin{aligned}\sigma_{\min}(\mathbf{A}_{\mathcal{S}_e})&=\sqrt{\alpha_\ell}\min_{\mathbf{Z}\in\mathcal{D}\backslash\{\mathbf{0}\}}\frac{\|\mathbf{Z}-\mathcal{P}_{\mathcal{L}}(\mathbf{Z})\|_{\mathrm{F}}}{\|\mathbf{Z}\|_{\mathrm{F}}}\\ &\geq\sqrt{\alpha_\ell}\left(1-\max_{\mathbf{Z}\in\mathcal{D}\backslash\{\mathbf{0}\}}\frac{\|\mathcal{P}_{\mathcal{L}}(\mathbf{Z})\|_{\mathrm{F}}}{\|\mathbf{Z}\|_{\mathrm{F}}}\right)=\sqrt{\alpha_\ell}(1-\mu).\end{aligned}$$

Therefore, we note that if $\mu < 1$ and $\alpha_\ell > 0$, $\mathbf{A}_{\mathcal{S}_e}$ has full row rank, and the lower bound on the smallest singular value is given by $\sqrt{\alpha_\ell}(1-\mu)$. □

Proof of Lemma 4.4. We begin with the definition of $\mathbf{b}_{\mathcal{S}_e}$. Since $\|\mathbf{b}_{\mathcal{S}_e}\|_2 = \|\mathbf{B}_{\mathcal{S}_e}\|_\mathrm{F}$ and $\mathbf{B}_{\mathcal{S}_\mathbf{e}} := \lambda_e \mathrm{sign}(\mathbf{S}_0) - \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)$,

$$\begin{aligned}\|\mathbf{b}_{\mathcal{S}_e}\|_2 &= \|\lambda_e \mathrm{sign}(\mathbf{S}_0) - \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F},\\ &\leq \lambda_e\sqrt{s_e} + \|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}.\end{aligned}$$

Now for an upper bound on $\|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}$ we start by analyzing $\|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}^2$,

$$\|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}^2 = |\langle \mathbf{D}^\top\mathbf{U}\mathbf{V}^\top, \, \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\rangle|.$$

Using properties of the inner products and using the fact that $\mathcal{P}_\mathcal{L}(\mathbf{U}\mathbf{V}^\top) = \mathbf{U}\mathbf{V}^\top$,

$$\|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}^2 = |\langle \mathcal{P}_\mathcal{L}(\mathbf{U}\mathbf{V}^\top), \, \mathbf{D}\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\rangle|.$$

Further simplifying using Cauchy Schwarz inequality and the definition of μ we have

$$\begin{aligned}&\|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}^2 \leq \|\mathcal{P}_\mathcal{L}(\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}\|\mathcal{P}_\mathcal{L}(\mathbf{D}\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top))\|_\mathrm{F}\\ &\leq \mu\|\mathbf{U}\mathbf{V}^\top\|_\mathrm{F}\|\mathbf{D}\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}\end{aligned}$$

Now, since $\|\mathbf{U}\mathbf{V}^\top\|_\mathrm{F} = \sqrt{r}$ and using the GFP we have $\|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F} \leq \mu\sqrt{r\alpha_u}$. Therefore, an upper bound for $\|\mathbf{b}_{\mathcal{S}_e}\|_2$ is given by $\|\mathbf{b}_{\mathcal{S}_e}\|_2 \leq \lambda_e\sqrt{s_e} + \sqrt{r\alpha_u}\mu$. □

Proof of Lemma 4.5. Since $\|\mathbf{b}_{\mathcal{S}_e}\|_\infty = \|\mathbf{B}_{\mathcal{S}_e}\|_\infty$ and $\mathbf{B}_{\mathcal{S}_\mathbf{e}} := \lambda_e \mathrm{sign}(\mathbf{S}_0) - \mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)$, we have the upper bound $\|\mathbf{b}_{\mathcal{S}_e}\|_\infty \leq \lambda_e + \|\mathcal{P}_{\mathcal{S}_e}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\infty$. □

Proof of Lemma 4.6. We begin by simplifying the quantity of interest as follows:

$$\begin{aligned}\|\mathbf{Q}\|_{\infty,\infty} &= \|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top(\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)^{-1}\|_{\infty,\infty}\\ &\leq \|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty}\|(\mathbf{I}-(\mathbf{I}-\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top))^{-1}\|_{\infty,\infty}\\ &\leq \tfrac{\|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty}}{1-\|\mathbf{I}-\mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty}}.\end{aligned} \tag{4.29}$$

Now, we derive appropriate bounds on the numerator and the denominator of (4.29) separately. Consider the numerator $\|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty}$. Here, we are interested in the maximum ℓ_1-norm of the rows of $\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top$, i.e.,

$$\|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty} = \max_i \|\mathbf{e}_i^\top \mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_1.$$

Let $\mathcal{I}_{\mathcal{S}_e}$ refer to the support of $\mathbf{S}_0$, and $\bar{\mathcal{I}}_{\mathcal{S}_e}$ to its complement. Then, the expression can be written in terms of $\mathcal{I}_{\mathcal{S}_e}$ and $\bar{\mathcal{I}}_{\mathcal{S}_e}$:

$$\|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty} = \max_{j\in\bar{\mathcal{I}}_{\mathcal{S}_e}} \sum_{\ell\in\mathcal{I}_{\mathcal{S}_e}} |\mathbf{e}_l^\top \mathbf{A}\mathbf{A}^\top \mathbf{e}_j|.$$

Now, $\mathbf{A}$ is defined as $(\mathbf{I}-\mathbf{P_V})\otimes\mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})$, therefore using the property of the product of two Kronecker products and product of projection matrices, $\mathbf{A}\mathbf{A}^\top$ can be written as

$$\mathbf{A}\mathbf{A}^\top = (\mathbf{I}-\mathbf{P_V})\otimes\mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}.$$

We are interested in the $\{\ell, j\}$ entry of $\mathbf{A}\mathbf{A}^\top$. Since, $\mathbf{A}\mathbf{A}^\top$ has a Kronecker product structure, an entry of $\mathbf{A}\mathbf{A}^\top$ is given by the product of elements of the matrices in the Kronecker product, therefore

$$\max_{j\in\bar{\mathcal{I}}_{\mathcal{S}_e}} \sum_{\ell\in\mathcal{I}_{\mathcal{S}_e}} |\mathbf{e}_l^\top \mathbf{A}\mathbf{A}^\top \mathbf{e}_j| = \max_{j_1,j_2\in\bar{\mathcal{I}}_{\mathcal{S}_e}} \sum_{\ell_1,\ell_2\in\mathcal{I}_{\mathcal{S}_e}} g(j_1,j_2,\ell_1,\ell_2), \tag{4.30}$$

where $g(j_1,j_2,\ell_1,\ell_2)$ is given by

$$g(j_1,j_2,\ell_1,\ell_2) = |\mathrm{Tr}(\mathbf{e}_{\ell_2}\mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top(\mathbf{I}-\mathbf{P_V}))|.$$

Now, consider $g(j_1,j_2,\ell_1,\ell_2)$, which can be simplified as

$$\begin{aligned} g(j_1,j_2,\ell_1,\ell_2) = |&\mathrm{Tr}(\mathbf{e}_{\ell_2}\mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top) \\ &- \mathrm{Tr}(\mathbf{e}_{\ell_2}\mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top\mathbf{P_V})|. \end{aligned}$$

Since trace is invariant under cyclic permutations, we have

$$\begin{aligned} g(j_1,j_2,\ell_1,\ell_2) = |&\mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}\mathbb{1}_{\{j_2=\ell_2\}} \\ &- \mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top\mathbf{P_V}\mathbf{e}_{\ell_2}|. \end{aligned}$$

Denote $x := \mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}$ and $y := \mathbf{e}_{j_2}^\top\mathbf{P_V}\mathbf{e}_{\ell_2}$, then we have

$$g(j_1,j_2,\ell_1,\ell_2) = |x\mathbb{1}_{\{j_2=\ell_2\}} - xy|.$$

Now, the following upper bound on $g(j_1,j_2,\ell_1,\ell_2)$ can be evaluated by squaring both sides and simplifying

$$g(j_1,j_2,\ell_1,\ell_2) \leq x\sqrt{\mathbb{1}_{\{j_2=\ell_2\}} + y^2}. \tag{4.31}$$

First consider x, which can be written as $x = x\mathbb{1}_{\{j_1=\ell_1\}} + x\mathbb{1}_{\{j_1\neq\ell_1\}}$. Here, $x\mathbb{1}_{\{j_1=\ell_1\}}$ can be upper bounded as shown below using the GFP

$$x = (\mathbf{e}_{\ell_1}^\top \mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D}\mathbf{e}_{\ell_1}) \leq \mathbf{e}_{\ell_1}^\top \mathbf{D}^\top\mathbf{D}\mathbf{e}_{\ell_1} \leq \alpha_u.$$

Further, we can derive an upper bound on $x\mathbb{1}_{\{j_1\neq\ell_1\}}$ using the paraflelogram law for inner-products as follows.

$$\begin{aligned} x\ &\leq |\mathbf{e}_{j_1}^\top \mathbf{D}^\top\mathbf{D}\mathbf{e}_{\ell_1}| + |\mathbf{e}_{j_1}^\top \mathbf{D}^\top\mathbf{P}_\mathbf{U}\mathbf{D}\mathbf{e}_{\ell_1}| \\ &\leq \tfrac{\alpha_u-\alpha_\ell}{2} + \alpha_u\gamma_\mathbf{U} = \tfrac{\alpha_u(1+2\gamma_\mathbf{U})}{2} - \tfrac{\alpha_\ell}{2}. \end{aligned}$$

Therefore, we have

$$x \leq \alpha_u\mathbb{1}_{\{j_1=\ell_1\}} + (\tfrac{\alpha_u(1+2\gamma_\mathbf{U})}{2} - \tfrac{\alpha_\ell}{2})\mathbb{1}_{\{j_1\neq\ell_1\}}.$$

Now, consider $\sqrt{\mathbb{1}_{\{j_2=\ell_2\}} + y^2}$, since $y = \mathbf{e}_{j_2}^\top\mathbf{P}_\mathbf{V}\mathbf{P}_\mathbf{V}\mathbf{e}_{\ell_2}$, and further, since $\sqrt{a^2+b^2} < (a+b)$ for $a>0$ and $b>0$, we have $\sqrt{\mathbb{1}_{\{j_2=\ell_2\}} + y^2} \leq \mathbb{1}_{\{j_2=\ell_2\}} + \gamma_\mathbf{V}$. Now, substituting in (4.31), i.e., the expression for $g(j_1,j_2,\ell_1,\ell_2)$, we have,

$$\begin{aligned} &g(j_1,j_2,\ell_1,\ell_2) \leq \\ &(\alpha_u\mathbb{1}_{\{j_1=\ell_1\}} + (\tfrac{\alpha_u(1+2\gamma_\mathbf{U})}{2} - \tfrac{\alpha_\ell}{2})\mathbb{1}_{\{j_1\neq\ell_1\}})(\mathbb{1}_{\{j_2=\ell_2\}} + \gamma_\mathbf{V}), \end{aligned}$$

and finally substituting in (4.30) and noting that since $j_1,j_2 \in \bar{\mathcal{I}}_{\mathcal{S}_e}$ and $\ell_1,\ell_2 \in \bar{\mathcal{I}}_{\mathcal{S}_e}$, $\mathbb{1}_{\{j_1=\ell_1\}}\mathbb{1}_{\{j_2=\ell_2\}} = 0$,

$$\begin{aligned} \|\mathbf{A}_{\mathcal{S}_e{}^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty} \leq \max_{j_1,j_2\in\bar{\mathcal{I}}_{\mathcal{S}_e}} &\sum_{\ell_1,\ell_2\in\mathcal{I}_{\mathcal{S}_e}} (\tfrac{\alpha_u(1+2\gamma_\mathbf{U})}{2} - \tfrac{\alpha_\ell}{2})\mathbb{1}_{\substack{\{j_1\neq\ell_1\},\\ \{j_2=\ell_2\}}} \\ &+ \alpha_u\gamma_\mathbf{V}\mathbb{1}_{\{j_1=\ell_1\}} + (\tfrac{\alpha_u(1+2\gamma_\mathbf{U})\gamma_\mathbf{V}}{2} - \tfrac{\alpha_\ell\gamma_\mathbf{V}}{2})\mathbb{1}_{\{j_1\neq\ell_1\}}. \end{aligned} \tag{4.32}$$

Now, for $\mathbf{A}_0 \in \mathbb{R}^{d\times m}$, the maximum number of non-zeros per row is $\min(s_e, m)$, while those in a column are $\min(s_e, d)$ for the *thin* case and $\min(s_e, k)$ for the *fat* case. Then we have

$$\|\mathbf{A}_{\mathcal{S}_e^\perp}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty} \leq c. \tag{4.33}$$

Here, the constant c is as defined in (4.17). Now, to bound the denominator of (4.29), we have

$$\begin{aligned}\|\mathbf{I} - \mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty} &= \max_i \|\mathbf{e}_i^\top(\mathbf{I} - \mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top)\|_1 \\ &= \max_{j,\ell\in\mathcal{S}} |1 - \|\mathbf{e}_j^\top\mathbf{A}\|^2| + \sum_{j\neq\ell} |\langle \mathbf{e}_j^\top\mathbf{A}, \mathbf{e}_l^\top\mathbf{A}\rangle| \end{aligned} \tag{4.34}$$

We proceed to bound $|1 - \|\mathbf{e}_j^\top\mathbf{A}\|^2|$. For this, we derive a lower bound on $\|\mathbf{e}_j^\top\mathbf{A}\|^2$. Note that $\mathbf{e}_j^\top\mathbf{A}$ selects the j-th row of $\mathbf{A}$, which has a Kronecker product structure. Therefore,

$$\begin{aligned}\|\mathbf{e}_j^\top\mathbf{A}\| &= \|(\mathbf{I} - \mathbf{P_U})\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top(\mathbf{I} - \mathbf{P_V})\|_{\mathrm{F}} = \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top)\|_{\mathrm{F}} \\ &\geq \|\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top\| - \|\mathcal{P}_{\mathcal{L}}(\mathbf{D}\mathbf{e}_{j_1}\mathbf{e}_{j_2}^\top)\|_{\mathrm{F}} \geq \sqrt{\alpha_\ell}(1-\mu).\end{aligned}$$

Therefore, since $\mu < 1$ and $\alpha_\ell > 0$, then if $\alpha_\ell \leq \frac{1}{(1-\mu)^2}$, we have $|1 - \|\mathbf{e}_j^\top\mathbf{A}\|^2| \leq 1 - \alpha_\ell(1-\mu)^2$. The analysis for deriving an upper bound for the second term in (4.34) closely follows that used in (4.33), as shown below.

$$\sum_{j\neq\ell} |\langle \mathbf{e}_j^\top\mathbf{A}_{\mathcal{S}_e}, \mathbf{e}_l^\top\mathbf{A}_{\mathcal{S}_e}\rangle| = \sum_{(\ell_1,\ell_2)\in\mathcal{S}\backslash\{(j_1,j_2)\}} g(j_1, j_2, \ell_1, \ell_2) \leq c.$$

Combining these results, we have the following bound for

$$\|\mathbf{I} - \mathbf{A}_{\mathcal{S}_e}\mathbf{A}_{\mathcal{S}_e}^\top\|_{\infty,\infty} \leq 1 - \alpha_\ell(1-\mu)^2 + c.$$

Finally, substituting these results in (4.29) we have $\|\mathbf{Q}\|_{\infty,\infty} \leq C_e := \frac{c}{\alpha_\ell(1-\mu)^2 - c}$, where c is given by (4.17). □

4.C.2 Proofs for Column-wise Case

Proof of Lemma 4.1. We show that for any $(\mathbf{L}_0, \mathbf{S}_0) \in \{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$, if $\mathrm{span}\{\mathrm{col}(\mathbf{L}_0)\} = \mathcal{U}$ and $\mathrm{csupp}(\mathbf{D}\mathbf{S}_0) = \mathcal{I}_{\mathcal{S}_c}$ do not hold simultaneously, then $\mu = 1$.

Let $\mathbf{L} + \mathbf{DS} = \mathbf{M}$, as per our model shown in (4.1). Now, let $(\mathbf{L}_0, \mathbf{S}_0)$ be any other pair in our Oracle Model $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$,

$$\mathbf{L}_0 = \mathbf{L} + \boldsymbol{\Delta}_1 \in \mathcal{U} \;\text{ and }\; \mathbf{DS}_0 = \mathbf{DS} + \boldsymbol{\Delta}_2 \in \mathcal{S}_c,$$

for some $\boldsymbol{\Delta}_1$ and $\boldsymbol{\Delta}_2$, then we have that $\boldsymbol{\Delta}_1 + \boldsymbol{\Delta}_2 = \mathbf{0}$. This implies that $\text{csupp}(\boldsymbol{\Delta}_1) \in \mathcal{S}_c$. Further, this implies that $\mathbf{L}$ and $\mathbf{L}_0$ at least match in the columns indexed by the inliers, i.e., $\mathcal{P}_{\mathcal{I}_\text{L}}(\mathbf{L}) = \mathcal{P}_{\mathcal{I}_\text{L}}(\mathbf{L}_0)$, and we have

$$\mathcal{U} = \text{span}\{\text{col}(\mathbf{L}_0)\} = \text{span}\{\text{col}(\mathcal{P}_{\mathcal{I}_\text{L}}(\mathbf{L}_0))\} = \text{span}\{\text{col}(\mathcal{P}_{\mathcal{I}_\text{L}}(\mathbf{L}))\}.$$

Therefore, $\text{csupp}(\mathbf{DS}_0) \subseteq \mathcal{I}_{\mathcal{S}_c}$. Specifically, this implies that there may exist a $j \in \mathcal{I}_{\mathcal{S}_c}$ for which $\mathbf{DS}_{:,j} - (\boldsymbol{\Delta}_1)_{:,j} = 0$, which will imply that $\mathcal{P}_{\mathcal{U}^\perp}(\mathbf{DS}_{:,j}) = 0$. This condition implies that $\mu = 1$. Therefore, we require $\text{span}\{\text{col}(\mathbf{L}_0)\} = \mathcal{U}$ and $\text{csupp}(\mathbf{DS}_0) = \mathcal{I}_{\mathcal{S}_c}$ to hold simultaneously for $\mu < 1$. □

Proof of Lemma 4.7. Let $(\mathbf{L}_0, \mathbf{S}_0)$ be an optimal solution pair of (D-RPCA(C)). From the optimality conditions (4.22) and (4.23), we seek $\boldsymbol{\Lambda}$ such that

$$\boldsymbol{\Lambda} \in \mathbf{UV}^\top + \mathbf{W} \;\text{ and }\; \mathbf{D}^\top \boldsymbol{\Lambda} \in \lambda_c \mathbf{H} + \lambda_c \mathbf{F}. \tag{4.35}$$

Now consider a feasible solution $\{\mathbf{L_0} + \mathbf{D}\boldsymbol{\Delta}, \mathbf{S_0} - \boldsymbol{\Delta}\}$ for a non-zero $\boldsymbol{\Delta} \in \mathbb{R}^{d \times m}$. Then by the optimality of $(\mathbf{L}_0, \mathbf{S}_0)$ using the subgradient inequality, we have

$$\begin{aligned} \|\mathbf{L}_0 + \mathbf{D}\boldsymbol{\Delta}\|_* + \lambda_c \|\mathbf{S}_0 - \boldsymbol{\Delta}\|_{1,2} \geq \|\mathbf{L}_0\|_* + \lambda_c \|\mathbf{S}_0\|_{1,2} \\ + \langle \mathbf{UV}^\top + \mathbf{W}, \mathbf{D}\boldsymbol{\Delta} \rangle - \lambda_c \langle \mathbf{H} + \mathbf{F}, \boldsymbol{\Delta} \rangle. \end{aligned}$$

Let $G(\boldsymbol{\Delta}) = \langle \mathbf{UV}^\top + \mathbf{W}, \mathbf{D}\boldsymbol{\Delta} \rangle - \lambda_c \langle \mathbf{H} + \mathbf{F}, \boldsymbol{\Delta} \rangle$. We will show that if **(q1)**-**(q4)** hold, then $G(\boldsymbol{\Delta}) > 0$, which proves the optimality of $(\mathbf{L}_0, \mathbf{S}_0)$. Rewrite $G(\boldsymbol{\Delta})$ as

$$G(\boldsymbol{\Delta}) = \langle \mathbf{W}, \mathbf{D}\boldsymbol{\Delta} \rangle - \lambda_c \langle \mathbf{F}, \boldsymbol{\Delta} \rangle + \langle \mathbf{D}^\top \mathbf{UV}^\top - \lambda_c \mathbf{H}, \boldsymbol{\Delta} \rangle. \tag{4.36}$$

Let $\mathbf{W}$, with $\|\mathbf{W}\| = 1$ and $\mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}$, then by duality of norms,

$$\langle \mathbf{W}, \mathbf{D}\boldsymbol{\Delta} \rangle = \langle \mathbf{W}, \mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\boldsymbol{\Delta}) \rangle = \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\boldsymbol{\Delta})\|_*. \tag{4.37}$$

Further, let $\mathbf{F}$, with $\|\mathbf{F}\|_{\infty,2} = 1$ and $\mathcal{P}_{\mathcal{S}_c}(\mathbf{F}) = \mathbf{0}$, be such that

$$\mathbf{F}_{:,j} = \begin{cases} -\frac{\boldsymbol{\Delta}_{:,j}}{\|\boldsymbol{\Delta}_{:,j}\|}, & \text{if } j \notin \mathcal{I}_{\mathcal{S}_c} \text{ and } \boldsymbol{\Delta}_{:,j} \neq 0 \\ 0, & \text{otherwise} \end{cases},$$

where $\mathbf{F}_{:,j}$ denotes the j^{th} column of $\mathbf{F}$. Then, we arrive at the following simplification for $\langle \mathbf{F},\ \boldsymbol{\Delta} \rangle$ by duality of norms,

$$\langle \mathbf{F},\ \boldsymbol{\Delta} \rangle \ = \ \langle \mathbf{F},\ \mathcal{P}_{\mathcal{S}_c^\perp}(\boldsymbol{\Delta}) \rangle \ = \ -\|\mathcal{P}_{\mathcal{S}_c^\perp}(\boldsymbol{\Delta})\|_{1,2}. \tag{4.38}$$

Since $\mathcal{P}_{\mathcal{L}}(\boldsymbol{\Lambda}) = \mathbf{U}\mathbf{V}^\top$ and $\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\boldsymbol{\Lambda}) = \lambda_c \mathbf{H}$ by optimality conditions of (4.35),

$$\begin{aligned} &\langle \mathbf{D}^\top\mathbf{U}\mathbf{V}^\top - \lambda_c\mathbf{H}, \boldsymbol{\Delta} \rangle = -\langle \mathcal{P}_{\mathcal{L}^\perp}(\boldsymbol{\Lambda}), \mathbf{D}\boldsymbol{\Delta} \rangle + \langle \mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\boldsymbol{\Lambda}), \boldsymbol{\Delta} \rangle \\ &\geq -\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\boldsymbol{\Delta})\|_* \|\mathcal{P}_{\mathcal{L}^\perp}(\boldsymbol{\Lambda})\| \\ &\qquad - \|\mathcal{P}_{\mathcal{S}_c^\perp}(\boldsymbol{\Delta})\|_{1,2} \|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\boldsymbol{\Lambda})\|_{\infty,2}, \end{aligned} \tag{4.39}$$

where we use Holder's inequality in the last step.

Combining (4.36), (4.37), (4.38), and (4.39), we have

$$\begin{aligned} G(\boldsymbol{\Delta}) \geq & (1 - \|\mathcal{P}_{\mathcal{L}^\perp}(\boldsymbol{\Lambda})\|) \|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\boldsymbol{\Delta})\|_* \\ & + (\lambda_c - \|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\boldsymbol{\Lambda})\|_{\infty,2}) \|\mathcal{P}_{\mathcal{S}_c^\perp}(\boldsymbol{\Delta})\|_{1,2} \end{aligned}$$

Since we have an arbitrary $\boldsymbol{\Delta}$ with $\boldsymbol{\Delta} \neq \mathbf{0}$ and $(\mathbf{L}_0 + \mathbf{D}\boldsymbol{\Delta}, \mathbf{S}_0 - \boldsymbol{\Delta}) \notin \{\mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$, $\|\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{D}\boldsymbol{\Delta})\|_* = \|\mathcal{P}_{{\mathcal{S}_c}^\perp}(\boldsymbol{\Delta})\|_{1,2} = 0$ does not hold. Therefore, to ensure the uniqueness of the solution $(\mathbf{L}_0, \mathbf{S}_0)$, we need $\|\mathcal{P}_{\mathcal{L}^\perp}(\boldsymbol{\Lambda})\| < 1$ and $\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{D}^\top\boldsymbol{\Lambda})\|_{\infty,2} < \lambda_c$. Hence, any dual certificate which obeys the conditions **(C1)**-**(C4)** guarantees optimality of the solution. □

Proof of Lemma 4.8. We begin by writing the definition of $\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c}^\top)$ as

$$\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c}^\top) = \min_{\mathbf{H} \in \mathcal{S}_c/\{\mathbf{0}_{d\times m}\}} \frac{\|\mathbf{A}^\top \text{vec}(\mathbf{H})\|_2}{\|\text{vec}(\mathbf{H})\|_2}.$$

By the definition of $\mathbf{A}$ and using the property of Kronecker product for multiplication by a vector we have

$$\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c}^\top) = \min_{\mathbf{H} \in \mathcal{S}_c/\{\mathbf{0}_{d\times m}\}} \frac{\|(\mathbf{I} - \mathbf{P}_{\mathbf{U}})\mathbf{D}\mathbf{H}(\mathbf{I} - \mathbf{P}_{\mathbf{V}})\|_{\text{F}}}{\|\mathbf{H}\|_{\text{F}}}.$$

Further $(\mathbf{I}-\mathbf{P}_{\mathbf{U}})\mathbf{DH}(\mathbf{I}-\mathbf{P}_{\mathbf{V}}) = \mathcal{P}_{\mathcal{L}^\perp}(\mathbf{DH})$, and we can write that expression above as follows

$$\begin{aligned}\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c}^\top) &= \min_{\mathbf{H}\in\mathcal{S}_c/\{\mathbf{0}_{d\times m}\}} \tfrac{\|\mathbf{DH}\|_\mathrm{F}}{\|\mathbf{H}\|_\mathrm{F}} \cdot \tfrac{\|(\mathbf{I}-\mathcal{P}_{\mathcal{L}})(\mathbf{DH})\|_\mathrm{F}}{\|\mathbf{DH}\|_\mathrm{F}}\\ &\overset{(i)}{\geq} \sqrt{\alpha_\ell}(1-\max_{\mathbf{Z}\in\mathcal{D}/\{\mathbf{0}_{n\times m}\}} \tfrac{\|\mathcal{P}_{\mathcal{L}}(\mathbf{Z})\|_\mathrm{F}}{\|\mathbf{Z}\|_\mathrm{F}}) \overset{(ii)}{\geq} \sqrt{\alpha_\ell}(1-\mu).\end{aligned}$$

Here (i) is due to the GFP condition **D.4.2** and the reverse triangle inequality, and (ii) from the incoherence property in (4.2). □

Proof of Lemma 4.9. We start by using the correspondence between the vector $\mathbf{b}_{\mathcal{S}_c}$ and the matrix $\mathbf{B}_{\mathcal{S}_c}$, i.e.,

$$\|\mathbf{b}_{\mathcal{S}_c}\|_2 = \|\mathbf{B}_{\mathcal{S}_c}\|_\mathrm{F} = \|\lambda_c\widetilde{\mathbf{S}} - \mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}.$$

Now, since $\widetilde{\mathbf{S}}_{:,j} = \mathbf{S}_{:,j}/\|\mathbf{S}_{:,j}\|_2$ for all $j\in\mathcal{I}_{\mathcal{S}_c}$; and is $\mathbf{0}$ otherwise (i.e., when $j\notin\mathcal{I}_{\mathcal{S}_c}$), using triangle inequality, we have

$$\|\mathbf{b}_{\mathcal{S}_c}\|_2 \leq \lambda_c\sqrt{s_c} + \|\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}. \tag{4.40}$$

Since we have

$$\begin{aligned}&\|\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}^2 \leq \|\mathcal{P}_{\mathcal{L}}(\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F}\|\mathcal{P}_{\mathcal{L}}(\mathbf{D}\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top))\|_\mathrm{F}\\ &\overset{(i)}{\leq} \mu\|\mathbf{U}\mathbf{V}^\top\|_\mathrm{F}\|\mathbf{D}\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F} \overset{(ii)}{\leq} \sqrt{r\alpha_u}\mu\|\mathcal{P}_{\mathcal{S}_c}(\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top)\|_\mathrm{F},\end{aligned} \tag{4.41}$$

where (i) is from subspace incoherence property and (ii) is from the GFP **D.4.2**. Combining (4.40) and (4.41), we have

$$\|\mathrm{vec}(\mathbf{B}_{\mathcal{S}_c})\|_2 \leq \lambda_c\sqrt{s_c} + \sqrt{r\alpha_u}\mu.$$

□

Proof of Lemma 4.10. We begin by analyzing the quantity of interest – $\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})\|_{\infty,2}$, i.e., we are interested in the maximum column norm of the matrix $\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})$. Note that $\mathbf{Z}$ is defined as

$$\mathbf{Z} = \mathbf{D}^\top(\mathbf{I}-\mathbf{P}_{\mathbf{U}})\mathbf{X}(\mathbf{I}-\mathbf{P}_{\mathbf{V}}),$$

and we have $\text{vec}(\mathbf{Z}) = \mathbf{A}\text{vec}(\mathbf{X})$. Further, we have that

$$\mathcal{P}_{\mathcal{S}_c^\perp}(\text{vec}(\mathbf{Z})) = \mathbf{A}_{\mathcal{S}_c^\perp}\text{vec}(\mathbf{X}).$$

Now, observe that the columns of matrix $\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})$ appear as blocks of size $n \times 1$ in the vector $\mathcal{P}_{\mathcal{S}_c^\perp}(\text{vec}(\mathbf{Z}))$. Moreover, the elements of vector $\mathcal{P}_{\mathcal{S}_c^\perp}(\text{vec}(\mathbf{Z}))$ are formed due to the inner product between the rows of Kronecker product structured matrix $\mathbf{A}_{\mathcal{S}_c^\perp}$ and $\text{vec}(X)$. Therefore, to identify a column of $\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})$ we need to focus on the interaction between correponding rows of $\mathbf{A}_{\mathcal{S}_c^\perp}$ and $\text{vec}(\mathbf{X})$.

Consider the Kronecker product structured matrix $\mathbf{A}_{\mathcal{S}_c^\perp}$. Since the rows in $\mathbf{A}_{\mathcal{S}_c^\perp}$ correspond to all rows outside the column support $\mathcal{S}_c$, this corresponds to selecting those rows of $m \times m$ matrix $(\mathbf{I} - \mathbf{P}_\mathbf{V})$ which correspond to $\mathcal{S}_c^\perp$, which we denote by $(\mathbf{I} - \mathbf{P}_\mathbf{V})_{\mathcal{S}_c^\perp}$ i.e.,

$$\mathbf{A}_{\mathcal{S}_c^\perp} = (\mathbf{I} - \mathbf{P}_\mathbf{V})_{\mathcal{S}_c^\perp} \otimes \mathbf{D}^\top(\mathbf{I} - \mathbf{P}_\mathbf{U}).$$

For simplicity of the upcoming analysis, we denote the matrix $(\mathbf{I} - \mathbf{P}_\mathbf{V})$ as

$$(\mathbf{I} - \mathbf{P}_\mathbf{V}) = \begin{bmatrix} v_{11} & \cdots & v_{1m} \\ \vdots & \ddots & \vdots \\ v_{m1} & \cdots & v_{mm} \end{bmatrix}.$$

Using this notation, the j-th block of vector $\mathcal{P}_{\mathcal{S}_c^\perp}(\text{vec}(\mathbf{Z}))$ (which is also the j-th column of $\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})$), can be written as

$$\mathbf{Z}_{:,j} = (v_{j,:} \otimes \mathbf{D}^\top(\mathbf{I} - \mathbf{P}_\mathbf{U}))\text{vec}(\mathbf{X})$$

for some $j \in \mathcal{I}_{\mathcal{S}_c^\perp}$. Now, further since $\text{vec}(\mathbf{X}) := \mathbf{A}_{\mathcal{S}_c}^\top(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\text{vec}(\mathbf{B}_{\mathcal{S}_c})$, therefore we are interested in maximum 2-norm of

$$\mathbf{Z}_{:,j} = (v_{j,:} \otimes \mathbf{D}^\top(\mathbf{I} - \mathbf{P}_\mathbf{U}))\mathbf{A}_{\mathcal{S}_c}^\top(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\text{vec}(\mathbf{B}_{\mathcal{S}_c}),$$

for some $j \in \mathcal{I}_{\mathcal{S}_c^\perp}$. Note that $\mathbf{A}_{\mathcal{S}_c}^\top$ itself is a Kronecker product structured matrix given by

$$\mathbf{A}_{\mathcal{S}_c} = (\mathbf{I} - \mathbf{P}_\mathbf{V})_{\mathcal{S}_c}^\top \otimes (\mathbf{I} - \mathbf{P}_\mathbf{U})\mathbf{D}.$$

Using the mixed product rule for Kronecker products we have

$$\mathbf{Z}_{:,j} = (v_{j,:}(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c}^\top \otimes \mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D})(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\mathbf{b}_{\mathcal{S}_c},$$

for some $j \in \mathcal{I}_{\mathcal{S}_c^\perp}$. Further, since for two matrices $\mathbf{A}$ and $\mathbf{B}$, $\|\mathbf{A}\otimes\mathbf{B}\| = \|\mathbf{A}\|\|\mathbf{B}\|$, we have

$$\begin{aligned}\|\mathbf{Z}_{:,j}\| \leq{}& \|\mathbf{e}_j^\top(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c^\perp}(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c}^\top\| \\ &\times \|\mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D}\|\|(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\|\|\mathbf{b}_{\mathcal{S}_c}\|, \end{aligned} \tag{4.42}$$

where we also use the fact that $v_{j,:} = \mathbf{e}_j^\top(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c^\perp}$. We will now proceed to bound the first term in (4.42). Note that

$$\begin{aligned}&\max_{j\in\mathcal{S}_c^\perp}\|\mathbf{e}_j^\top(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c^\perp}(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c}^\top\|^2 \\ &= \max_{j\in\mathcal{S}_c^\perp}\sum_{i\in\mathcal{S}_c}\langle(\mathbf{I}-\mathbf{P}_\mathbf{V})^\top\mathbf{e}_j, (\mathbf{I}-\mathbf{P}_\mathbf{V})^\top\mathbf{e}_i\rangle^2.\end{aligned}$$

Now, each term in the summation can be bounded as

$$\begin{aligned}&\max_{j\in\mathcal{S}_c^\perp, i\in\mathcal{S}_c}|\langle(\mathbf{I}-\mathbf{P}_\mathbf{V})^\top\mathbf{e}_j, (\mathbf{I}-\mathbf{P}_\mathbf{V})^\top\mathbf{e}_i\rangle| \\ &= \max_{j\in\mathcal{S}_c^\perp, i\in\mathcal{S}_c}|-\langle\mathbf{P}_\mathbf{V}\mathbf{e}_j, \mathbf{P}_\mathbf{V}\mathbf{e}_i\rangle| \leq \|\mathbf{P}_\mathbf{V}\mathbf{e}_j\|\|\mathbf{P}_\mathbf{V}\mathbf{e}_i\| \leq \gamma_\mathbf{V}.\end{aligned}$$

This implies $\|\mathbf{e}_j^\top(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c^\perp}(\mathbf{I}-\mathbf{P}_\mathbf{V})_{\mathcal{S}_c}^\top\| \leq \sqrt{s_c}\gamma_\mathbf{V}$. Further, note that $\|(\mathbf{A}_{\mathcal{S}_c}\mathbf{A}_{\mathcal{S}_c}^\top)^{-1}\| \leq \|\mathbf{A}_{\mathcal{S}_c}^{-1}\|^2 = \frac{1}{\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c})^2}$. Substituting this into (4.42), for a $j \in \mathcal{S}_c^\perp$, we have

$$\|\mathbf{Z}_{:,j}\| \leq \frac{\sqrt{s_c}\gamma_\mathbf{V}}{\sigma_{\min}(\mathbf{A}_{\mathcal{S}_c})^2}\|\mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D}\|\|\mathbf{b}_{\mathcal{S}_c}\|. \tag{4.43}$$

We can further write $\|\mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D}\|$ as follows

$$\|\mathbf{D}^\top(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D}\| = \max_{\|u\|=1}\frac{\|(\mathbf{I}-\mathbf{P}_\mathbf{U})\mathbf{D}u\|^2}{\|\mathbf{D}u\|^2}\|\mathbf{D}u\|^2 \leq \beta_\mathbf{U}\alpha_u.$$

Substituting this result in (4.43), using Lemma 4.8 and Lemma 4.9,

$$\|\mathcal{P}_{\mathcal{S}_c^\perp}(\mathbf{Z})\|_{\infty,2} \leq \sqrt{s_c}C_c(\lambda_c\sqrt{s_c} + \sqrt{r\alpha_u}\mu).$$

□

Chapter 5

Target Localization in Hyperspectral Images

5.1 Overview

We consider the task of localizing targets of interest in a hyperspectral (HS) image based on their spectral signature(s), by posing the problem as two distinct convex demixing task(s). With applications ranging from remote sensing to surveillance, this task of target detection leverages the fact that each material/object possesses its own characteristic spectral response, depending upon its composition. However, since *signatures* of different materials are often correlated, matched filtering-based approaches may not be apply here. To this end, we model a HS image as a superposition of a low-rank component and a dictionary sparse component, wherein the dictionary consists of the *a priori* known characteristic spectral responses of the target we wish to localize, and develop techniques for two different sparsity structures, resulting from different model assumptions. We also present the corresponding recovery guarantees, leveraging our theoretical results from Chapter 4. Finally, we analyze the performance of the proposed approach via experimental evaluations on real HS datasets for a classification task, and compare its performance with related techniques.

5.2 Introduction

Hyperspectral (HS) imaging is an imaging modality which senses the intensities of the reflected electromagnetic waves (responses) corresponding to different wavelengths of the electromagnetic spectra, often invisible to the human eye. As the spectral response associated with an object/material is dependent on its composition, HS imaging lends itself very useful in identifying the said target objects/materials via their characteristic spectra or *signature* responses, also referred to as *endmembers* in the literature. Typical applications of HS imaging range from monitoring agricultural use of land, catchment areas of rivers and water bodies, food processing and surveillance, to detecting various minerals, chemicals, and even presence of life sustaining compounds on distant planets; see Borengasser et al. (2007); Park and Lu (2015), and references therein for details. However, often, these spectral *signatures* are highly correlated, making it difficult to detect regions of interest based on these endmembers. In this work, we present two techniques to localize target materials/objects in a given HS image based on some structural assumptions on the data, using the *a priori* known signatures of the target of interest.

The primary property that enables us to localize a target is the approximate low-rankness of HS images when represented as a matrix, owing to the fact that a particular scene is composed of only a limited type of objects/materials (Keshava and Mustard, 2002). For instance, while imaging an agricultural area, one would expect to record responses from materials like biomass, farm vehicles, roads, houses, water bodies, and so on. Moreover, the spectra of complex materials can be assumed to be a linear mixture of the constituent materials (Keshava and Mustard, 2002; Greer, 2012), i.e. the received HS responses can be viewed as being generated by a linear mixture model (Xing et al., 2012). For the target localization task at hand, this approximate low-rank structure is used to decompose a given HS image into a low-rank part, and a component that is sparse in a known dictionary – a *dictionary sparse* part– wherein the dictionary is composed of the spectral signatures of the target of interest. We begin by formalizing the specific model of interest in the next section.

5.2.1 Model

A HS sensor records the response of a region, which corresponds to a pixel in the HS image as shown in Fig. 5.1, to different frequencies of the electromagnetic spectrum.

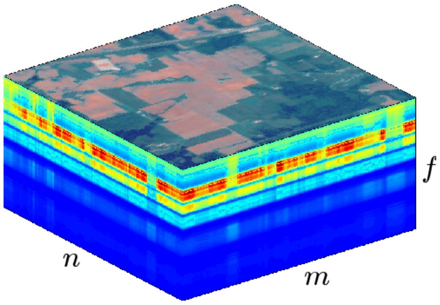


Figure 5.1: The HS image data-cube corresponding to the Indian Pines dataset.

As a result, each HS image $\mathbf{I} \in \mathbb{R}^{n \times m \times f}$, can be viewed as a data-cube formed by stacking f matrices of size $n \times m$, as shown in Fig. 5.1. Therefore, each volumetric element or *voxel*, of a HS image is a vector of length f, and represents the response of the material to f measurement channels. Here, f is determined by the number of channels or frequency bands across which measurements of the reflectances are made.

Formally, let $\mathbf{M} \in \mathbb{R}^{f \times nm}$ be formed by *unfolding* the HS image $\mathbf{I}$, such that, each column of $\mathbf{M}$ corresponds to a voxel of the data-cube. We then model $\mathbf{M}$ as arising from a superposition of a low-rank component $\mathbf{L} \in \mathbb{R}^{f \times nm}$ with rank r, and a dictionary-sparse component, expressed as $\mathbf{DS}$, i.e.,

$$\mathbf{M} = \mathbf{L} + \mathbf{DS}. \tag{5.1}$$

Here, $\mathbf{D} \in \mathbb{R}^{f \times d}$ represents an *a priori* known dictionary composed of appropriately normalized characteristic responses of the material/object (or the constituents of the material), we wish to localize, and $\mathbf{S} \in \mathbb{R}^{d \times nm}$ refers to the *sparse* coefficient matrix (also referred to as *abundances* in the literature). Note that $\mathbf{D}$ can also be constructed by learning a dictionary based on the known spectral signatures of a target; see Olshausen and Field (1997); Aharon et al. (2005); Mairal et al. (2010); Lee et al. (2007).

5.2.2 Our Contributions

In this work, we present two techniques[1] for target detection in a HS image, depending upon different sparsity assumptions on the matrix $\mathbf{S}$, by modeling the data as shown in (5.1). Building on the theoretical results of Chapter 4 (and Rambhatla et al. (2016b); Li et al. (2018b); Rambhatla et al. (2018a)), our techniques operate by forming the dictionary $\mathbf{D}$ using the *a priori* known spectral signatures of the target of interest, and

[1] The code is made available at `github.com/srambhatla/Dictionary-based-Robust-PCA`.

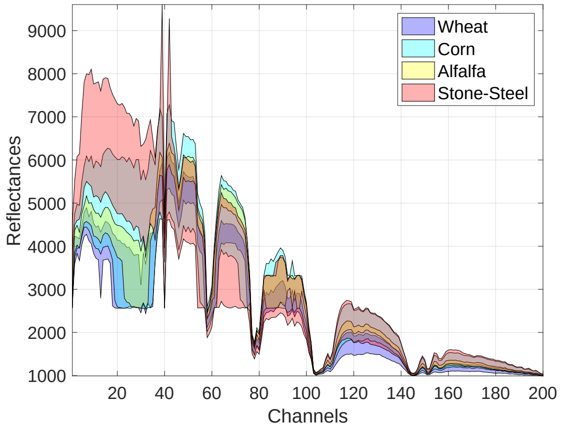


Figure 5.2: Correlated spectral signatures. The spectral signatures of even different materials are highly correlated. Shown here are spectral signatures of classes from the Indian Pines dataset (Baumgardner et al., 2015). Here, the shaded region shows the lower and upper ranges of reflectance values the signatures take.

leveraging the approximate low-rank structure of the data matrix **M** (Rambhatla et al., 2017b). Here, the dictionary **D** can be formed from the *a priori* known signatures directly, or by learning an appropriate dictionary based on target data; see Olshausen and Field (1997); Aharon et al. (2005); Mairal et al. (2010); Lee et al. (2007).

We consider two types of sparsity structures for the coefficient matrix **S**, namely, a) *global* or *entry-wise* sparsity, wherein we let the matrix **S** have s_e non-zero entries globally, and b) *column-wise* sparse structure, where at most s_c columns of the matrix **S** have non-zero elements. The choice of a particular sparsity model depends on the properties of the dictionary matrix **D**. In particular, if the target signature admits a sparse representation in the dictionary, entry-wise sparsity structure is preferred. This is likely to be the case when the dictionary is overcomplete ($f < d$) or *fat*, and also when the target spectral responses admit a sparse representation in the dictionary. On the other hand, the column-wise sparsity structure is amenable to cases where the representation can use all columns of the dictionary. This potentially arises in the cases when the dictionary is undercomplete ($f \geq d$) or *thin*. Note that, in the column-wise sparsity case, the non-zero columns need not be sparse themselves. The applicability of these two modalities is also exhibited in our experimental analysis; see Section 5.6 for further details. Further, we specialize the theoretical results of Chapter 4, to present the conditions under which such a demixing task will succeed under the two sparsity models discussed above; see also Rambhatla et al. (2016b) and Li et al. (2018b).

Next, we analyze the performance of the proposed techniques via extensive experimental evaluations on real-world demixing tasks over different datasets and dictionary choices, and compare the performance of the proposed techniques with related works. This demixing task is particularly challenging since the spectral signatures of even distinct classes are highly correlated to each other, as shown in Fig. 5.2. The shaded region here shows the upper and lower ranges of different classes. For instance, in Fig. 5.2 we observe that the spectral signature of the "Stone-Steel" class is similar to that of class "Wheat". This correlation between the spectral signatures of different classes results in an approximate low-rank structure of the data, captured by the low-rank component $\mathbf{L}$, while the dictionary-sparse component $\mathbf{DS}$ is used to identify the target of interest. We specifically show that such a decomposition successfully localizes the target despite the high correlation between spectral signatures of distinct classes.

Finally, it is worth noting that although we consider *thin* dictionaries ($f \geq d$) for the purposes of this work, since it is more suitable for the current exposition, our theoretical results are also applicable for the *fat* case ($f < d$); see Chapter 4, Rambhatla et al. (2016b), Li et al. (2018b), and Rambhatla et al. (2018a) for further details.

5.2.3 Prior Art

The model shown in (5.1) is closely related to a number of well-known problems. To start, in the absence of the dictionary sparse part $\mathbf{DS}$, (5.1) reduces to the popular problem of principal component analysis (PCA) (Pearson, 1901; Jolliffe, 2002). The problem considered here also shares its structure with variants of PCA, such as robust-PCA (Candès et al., 2011; Chandrasekaran et al., 2011) (with $\mathbf{D} = \mathbf{I}$ for an identity matrix $\mathbf{I}$,) outlier pursuit (Xu et al., 2010) (where $\mathbf{D} = \mathbf{I}$ and $\mathbf{S}$ is column-wise sparse,) and others (Zhou et al., 2010; Ding et al., 2011; Wright et al., 2013; Chen et al., 2013; Li and Haupt, 2015a,b,c, 2016; Li et al., 2016a).

On the other hand, the problem can be identified as that of sparse recovery (Natarajan, 1995; Donoho and Huo, 2001b; Candès and Tao, 2005; Rambhatla and Haupt, 2013b), in the absence of the low-rank part $\mathbf{L}$. Following which, sparse recovery methods for analysis of HS images have been explored in (Moudden et al., 2009; Bobin et al., 2009; Kawakami et al., 2011; Charles et al., 2011). In addition, in a recent work (Giampouras et al., 2016), the authors further impose a low-rank constraint on the coefficient matrix $\mathbf{S}$ for the demixing task. Further, applications of compressive sampling have been explored in Golbabaee et al. (2010), while Xing et al. (2012) analyzes the case where HS images are noisy and incomplete. The techniques discussed above focus on identifying all materials in a given HS image. However, for target localization tasks, it is of interest to identify only specific target(s) in a given HS image. As a result, there is a need for techniques which localize targets based on their *a priori* known spectral signatures.

The model described in (5.1) was introduced in Mardani et al. (2013) as a means to detect traffic anomalies in a network, wherein, the authors focus on a case where the dictionary $\mathbf{D}$ is *overcomplete*, i.e., *fat*, and the rows of $\mathbf{D}$ are orthogonal, e.g., $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$. Here, the coefficient matrix $\mathbf{S}$ is assumed to possess at most k nonzero elements per row and column, and s nonzero elements globally. In a recent work (Rambhatla et al., 2016b) and the accompanying theoretical work (Rambhatla et al., 2018a) (presented in Chapter 4), we analyze the extension of Mardani et al. (2013) to include a case where the dictionary has more rows than columns, i.e., is *thin*, while removing the orthogonality constraint for both the *thin* and the *fat* dictionary cases, when s is small. This case is particularly amenable for the target localization task at hand, since often we aim to localize targets based on a few *a priori* known spectral signatures. To this end, we focus our attention on the *thin* case, although a similar analysis applies for the *fat* case (Rambhatla et al., 2016b); see also Chapter 4 and Rambhatla et al. (2018a).

5.2.4 Related Techniques

To study the properties of our techniques, we compare and contrast their performance with related works. First, as a sanity check, we compare the performance of the proposed techniques with matched filtering-based methods (detailed in Section 5.6). In addition, we compare the performance of our techniques to other closely related methods based on the sparsity assumptions on the matrix $\mathbf{S}$, as described below.

For entry-wise sparse structure: The first method we compare to is based on the observation that in cases where the known dictionary $\mathbf{D}$ is thin, we can multiply (5.1) on the left by the pseudo-inverse of $\mathbf{D}$, say $\mathbf{D}^\dagger$, in which case, the model shown in (5.1) reduces to that of robust PCA, i.e.,

$$\widetilde{\mathbf{M}} = \widetilde{\mathbf{L}} + \mathbf{S}, \tag{RPCA†}$$

where $\widetilde{\mathbf{M}} = \mathbf{D}^\dagger\mathbf{M}$ and $\widetilde{\mathbf{L}} = \mathbf{D}^\dagger\mathbf{L}$. Therefore, in this case, we can recover the sparse matrix $\mathbf{S}$ by robust PCA (Candès et al., 2011; Chandrasekaran et al., 2011), and estimate the low-rank part using the estimate of $\mathbf{DS}$. Note that this is not applicable for the *fat* case due to the non-trivial null space of its pseudo-inverse.

Although at a first glance this seems like a reasonable technique, somewhat surprisingly, it does not succeed for all *thin* dictionaries. Specifically, in cases where r, the rank of $\mathbf{L}$, is greater than the number of dictionary elements d, the pseudo-inversed component $\widetilde{\mathbf{L}}$ is no longer "low-rank." In fact, since the notion of low-rankness is relative to the potential maximum rank of the component, $\widetilde{\mathbf{L}}$ can be close to full-rank. As a result, the robust PCA model shown in RPCA† is no longer applicable and the demixing task may not succeed; see Chapter 4 and Rambhatla et al. (2018a) for details.

Moreover, even in cases where RPCA† succeeds ($r < d$), our proposed one-shot procedure guarantees the recovery of the two components under some mild conditions, while the pseudo-inverse based procedure RPCA† will require a two-step procedure – one to recover the sparse coefficient matrix and other to recover the low-rank component – in addition to a non-trivial analysis of the interaction between $\mathbf{D}^\dagger$ and the low-rank part $\mathbf{L}$. This is also apparent from our experiments shown in Section 5.6, which indicate that optimization based on the model in (5.1) is more *robust* as compared to RPCA† for the classification problem at hand across different choices of the dictionaries.

For column-wise sparse structure: The column-wise sparse structure of the matrix $\mathbf{S}$ results in a column-wise sparse structure of the dictionary-sparse component $\mathbf{DS}$. As a result, the model at hand is similar to that studied in OP (Xu et al., 2010). Specifically, the OP technique is aimed at identifying the outlier columns in a given matrix. However, it fails in cases where the target of interest is not an outlier, as in case of HS data. On the other hand, since the proposed technique uses the dictionary $\mathbf{D}$ corresponding to the spectral signatures of the target of interest to guide the demixing procedure, it results in a spectral signature-driven technique for target localization. This distinction between the two procedures is also discussed in our corresponding theoretical work presented in Chapter 4 Section 4.5, and is exemplified by our experimental results shown in Section 5.6.

Further, as in the entry-wise case, one can also envision a pseudo-inverse based procedure to identify the target of interest via OP (Xu et al., 2010) on the pseudo-inversed data (referred to as OP† in our discussion) i.e.,

$$\widetilde{\mathbf{M}} = \widetilde{\mathbf{L}} + \mathbf{S}, \tag{OP†}$$

where $\widetilde{\mathbf{M}} = \mathbf{D}^\dagger\mathbf{M}$ and $\widetilde{\mathbf{L}} = \mathbf{D}^\dagger\mathbf{L}$, with $\mathbf{S}$ admitting a column-wise sparse structure. However, this variant of OP does not succeed when the rank of the low-rank component is greater than the number of dictionary elements, i.e., $r \geq d$, as in the previous case; see Section 4.5 of Chapter 4 for details.

The rest of the chapter is organized as follows. We formulate the problem and introduce relevant theoretical quantities in Section 5.3, followed by specializing the theoretical results for the current application in Section 5.4. Next, in Section 5.5, we present the specifics of the algorithms for the two cases. In Section 5.6, we describe the experimental set-up and demonstrate the applicability of the proposed approaches via extensive numerical simulations on real HS datasets for a classification task. Finally, we conclude this discussion in Section 5.7.

5.3 Problem Formulation

In this section, we introduce the optimization problem of interest and different theoretical quantities pertinent to our analysis. These are motivated from our analysis in Chapter 4 Section 4.3; we outline these conditions in this section for completeness and instantiate them for our problem of interest.

5.3.1 Optimization problems

Our aim is to recover the low-rank component $\mathbf{L}$ and the sparse coefficient matrix $\mathbf{S}$, given the dictionary $\mathbf{D}$ and samples $\mathbf{M}$ generated according to the model shown in (5.1). Here the coefficient matrix $\mathbf{S}$ can either have an entry-wise sparse structure or a column-wise sparse structure. We now crystallize our model assumptions to formulate appropriate convex optimization problems for the two sparsity structures.

Specifically, depending upon the priors about the sparsity structure of $\mathbf{S}$, and the low-rank property of the component $\mathbf{L}$, we aim to solve the following convex optimization problems, i.e.,

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda_e\|\mathbf{S}\|_1 \text{ s.t. } \mathbf{M} = \mathbf{L} + \mathbf{DS} \tag{D-RPCA(E)}$$

for the entry-wise sparsity case, and

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda_c\|\mathbf{S}\|_{1,2} \text{ s.t. } \mathbf{M} = \mathbf{L} + \mathbf{DS} \tag{D-RPCA(C)}$$

for the column-wise sparse case, to recover $\mathbf{L}$ and $\mathbf{S}$ with regularization parameters $\lambda_e \geq 0$ and $\lambda_c \geq 0$, given the data $\mathbf{M}$ and the dictionary $\mathbf{D}$. Here, the *a priori* known dictionary $\mathbf{D}$ is assumed to be undercomplete (*thin*, i.e., $d \leq f$) for the application at hand. Analysis of a more general case can be found in Chapter 4 (and Rambhatla et al. (2018a)). Further, here "D-RPCA" refers to "dictionary based robust principal component analysis", while the qualifiers "E" and "C" indicate the entry-wise and column-wise sparsity patterns, respectively.

Note that, in the column-wise sparse case there is an inherent ambiguity regarding the recovery of the true component pairs $(\mathbf{L},\mathbf{S})$ corresponding to the low-rank part and the dictionary sparse component, respectively. Specifically, any pair $(\mathbf{L}_0,\mathbf{S}_0)$ satisfying $\mathbf{M} = \mathbf{L}_0 + \mathbf{DS}_0 = \mathbf{L} + \mathbf{DS}$, where $\mathbf{L}_0$ and $\mathbf{L}$ have the same column space, and $\mathbf{S}_0$ and $\mathbf{S}$ have the identical column support, is a solution of D-RPCA(C). To this end, we define the following *oracle model* to characterize the optimality of any solution pair $(\mathbf{L}_0,\mathbf{S}_0)$.

Definition 5.1 (Oracle Model for Column-wise Sparse Case)**.** Let the pair $(\mathbf{L},\mathbf{S})$ be the matrices forming the data $\mathbf{M}$ as per (5.1), define the corresponding oracle model $\{\mathbf{M},\mathcal{U},\mathcal{I}_{\mathcal{S}_c}\}$. Then, any pair $(\mathbf{L}_0,\mathbf{S}_0)$ is in the *Oracle Model* $\{\mathbf{M},\mathcal{U},\mathcal{I}_{\mathcal{S}_c}\}$, if $\mathcal{P}_{\mathcal{U}}(\mathbf{L}_0) = \mathbf{L}$, $\mathcal{P}_{\mathcal{S}_c}(\mathbf{DS}_0) = \mathbf{DS}$ and $\mathbf{L}_0 + \mathbf{DS}_0 = \mathbf{L} + \mathbf{DS} = \mathbf{M}$ hold simultaneously, where $\mathcal{P}_{\mathcal{U}}$ and $\mathcal{P}_{\mathcal{S}_c}$ are projections onto the column space $\mathcal{U}$ of $\mathbf{L}$ and column support $\mathcal{I}_{\mathcal{S}_c}$ of $\mathbf{S}$, respectively.

For this case, we then first establish the sufficient conditions for the existence of a solution based on some incoherence conditions. Following which, our main result for the column-wise case states the sufficient conditions under which solving a convex optimization problem recovers a solution pair $(\mathbf{L}_0,\mathbf{S}_0)$ in the oracle model.

5.3.2 Conditions on the Dictionary

For our analysis, we require that the dictionary $\mathbf{D}$ follows the *generalized frame property* (GFP) defined as follows.

Definition 5.2. A matrix $\mathbf{D}$ satisfies the *generalized frame property* (GFP), on vectors $\mathbf{v} \in \mathcal{R}$, if for any fixed vector $\mathbf{v} \in \mathcal{R}$ where $\mathbf{v} \neq \mathbf{0}$, we have

$$\alpha_\ell \|\mathbf{v}\|_2^2 \leq \|\mathbf{D}\mathbf{v}\|_2^2 \leq \alpha_u \|\mathbf{v}\|_2^2,$$

where α_ℓ and α_u are the lower and upper *generalized frame bounds* with $0 < \alpha_\ell \leq \alpha_u < \infty$.

The GFP is met as long as the vector $\mathbf{v}$ is not in the null-space of the matrix $\mathbf{D}$, and $\|\mathbf{D}\|$ is bounded. Therefore, for the *thin* dictionary setting $d < n$ for both entry-wise and column-wise sparsity cases, this condition is satisfied as long as $\mathbf{D}$ has a full column rank, and $\mathcal{R}$ can be the entire space. For example, $\mathbf{D}$ being a *frame* (Duffin and Schaeffer, 1952) suffices; see Heil (2013) for a brief overview of frames.

5.3.3 Relevant Subspaces

Before we define the relevant subspaces for this discussion, we define a few preliminaries. First, let the pair $(\mathbf{L_0}, \mathbf{S_0})$ be the solution to D-RPCA(E) (the entry-wise sparse case), and for the column-wise sparse case, let the pair $(\mathbf{L_0}, \mathbf{S_0})$ be in the oracle model $\{\mathbf{M}, \mathcal{U}, \mathcal{I}_{\mathcal{S}_c}\}$; see Definition **D.5.1**.

Next, for the low-rank matrix $\mathbf{L}$, let the compact singular value decomposition (SVD) be represented as

$$\mathbf{L} = \mathbf{U}\Sigma\mathbf{V}^\top,$$

where $\mathbf{U} \in \mathbb{R}^{f \times r}$ and $\mathbf{V} \in \mathbb{R}^{nm \times r}$ are the left and right singular vectors of $\mathbf{L}$, respectively, and Σ is a diagonal matrix with singular values arranged in a descending order on the diagonal. Here, matrices $\mathbf{U}$ and $\mathbf{V}$ each have orthogonal columns. Further, let $\mathcal{L}$ be the linear subspace consisting of matrices spanning the same row or column space as $\mathbf{L}$, i.e.,

$$\mathcal{L} := \{\mathbf{U}\mathbf{W}_1^\top + \mathbf{W}_2\mathbf{V}^\top, \mathbf{W}_1 \in \mathbb{R}^{nm \times r}, \mathbf{W}_2 \in \mathbb{R}^{f \times r}\}.$$

Next, let $\mathcal{S}_e$ ($\mathcal{S}_c$) be the space spanned by $d \times nm$ matrices with the same non-zero support (column support, denoted as csupp) as $\mathbf{S}$, and let $\mathcal{D}$ be defined as

$$\mathcal{D} := \{\mathbf{DH}\}, \text{where} \begin{cases} \mathbf{H} \in \mathcal{S}_e \text{ for entry-wise case,} \\ \text{csupp}(\mathbf{H}) \subseteq \mathcal{I}_{\mathcal{S}_c} \text{ for column-wise case.} \end{cases}$$

Here, $\mathcal{I}_{\mathcal{S}_c}$ denotes the index set containing the non-zero column indices of $\mathbf{S}$ for the column-wise sparsity case. In addition, we denote the corresponding complements of the spaces described above by appending '$\perp$'.

We use calligraphic '$\mathcal{P}(\cdot)$' to denote the projection operator onto a subspace defined by the subscript, and '$\mathbf{P}$' to denote the corresponding projection matrix with the appropriate subscripts. Therefore, using these definitions the projection operators onto and orthogonal to the subspace $\mathcal{L}$ are defined as

$$\mathcal{P}_{\mathcal{L}}(\mathbf{L}) = \mathbf{P_U L} + \mathbf{L P_V} - \mathbf{P_U L P_V}$$

and

$$\mathcal{P}_{\mathcal{L}^\perp}(\mathbf{L}) = (\mathbf{I} - \mathbf{P_U})\mathbf{L}(\mathbf{I} - \mathbf{P_V}),$$

respectively.

5.3.4 Incoherence Measures

We also employ various notions of incoherence to identify the conditions under which our procedures succeed. To this end, we first define the incoherence parameter μ that characterizes the relationship between the low-rank part $\mathbf{L}$ and the dictionary sparse part $\mathbf{DS}$, as

$$\mu := \max_{\mathbf{Z} \in \mathcal{D} \backslash \{\mathbf{0}_{d \times nm}\}} \frac{\|\mathcal{P}_{\mathcal{L}}(\mathbf{Z})\|_{\text{F}}}{\|\mathbf{Z}\|_{\text{F}}}. \tag{5.2}$$

The parameter $\mu \in [0,1]$ is the measure of degree of similarity between the low-rank part and the dictionary sparse component. Here, a larger μ implies that the dictionary sparse component is close to the low-rank part. In addition, we also define the parameter β_U as

$$\beta_{\mathbf{U}} := \max_{\|\mathbf{u}\|=1} \frac{\|(\mathbf{I}-\mathbf{P}_{\mathbf{U}})\mathbf{D}\mathbf{u}\|^2}{\|\mathbf{D}\mathbf{u}\|^2}, \tag{5.3}$$

which measures the similarity between the orthogonal complement of the column-space $\mathcal{U}$ and the dictionary $\mathbf{D}$.

The next two measures of incoherence can be interpreted as a way to identify the cases where for $\mathbf{L}$ with SVD as $\mathbf{L} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$: (a) $\mathbf{U}$ resembles the dictionary $\mathbf{D}$, and (b) $\mathbf{V}$ resembles the sparse coefficient matrix $\mathbf{S}$. In these cases, the low-rank part may resemble the dictionary sparse component. To this end, similar to Mardani et al. (2013), we define the following measures to identify these cases as

$$\text{(a) } \gamma_{\mathbf{U}} := \max_i \frac{\|\mathbf{P}_{\mathbf{U}}\mathbf{D}\mathbf{e}_i\|^2}{\|\mathbf{D}\mathbf{e}_i\|^2} \text{ and (b) } \gamma_{\mathbf{V}} := \max_i \|\mathbf{P}_{\mathbf{V}}\mathbf{e}_i\|^2. \tag{5.4}$$

Here, $0 \leq \gamma_{\mathbf{U}} \leq 1$ achieves the upper bound when a dictionary element is exactly aligned with the column space $\mathcal{U}$ of the $\mathbf{L}$, and lower bound when all of the dictionary elements are orthogonal to $\mathcal{U}$. Moreover, $\gamma_{\mathbf{V}} \in [r/nm, 1]$ achieves the upper bound when the row-space of $\mathbf{L}$ is "spiky", i.e., a certain row of $\mathbf{V}$ is 1-sparse, meaning that a column of $\mathbf{L}$ is supported by (can be expressed as a linear combination of) a column of $\mathbf{U}$. The lower bound here is attained when it is "spread-out", i.e., each column of $\mathbf{L}$ is a linear combination of all columns of $\mathbf{U}$. In general, our recovery of the two components is easier when the incoherence parameters $\gamma_{\mathbf{U}}$ and $\gamma_{\mathbf{V}}$ are closer to their lower bounds. In addition, for notational convenience, we define constants

$$\xi_e := \|\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top\|_\infty \text{ and } \xi_c := \|\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top\|_{\infty,2}. \tag{5.5}$$

Here, ξ_e is the maximum absolute entry of $\mathbf{D}^\top\mathbf{U}\mathbf{V}^\top$, which measures how close columns of $\mathbf{D}$ are to the singular vectors of $\mathbf{L}$. Similarly, for the column-wise case, ξ_c measures the closeness of columns of $\mathbf{D}$ to the singular vectors of $\mathbf{L}$ under a different metric (column-wise maximal ℓ_2-norm).

5.4 Theoretical Results

In this section, we specialize our theoretical results presented in Chapter 4 for the HS demixing task. Specifically, we provide the main results corresponding to each sparsity structure of $\mathbf{S}$ for the thin dictionary case considered here. We start with the theoretical results for the entry-wise sparsity case, and then present the corresponding theoretical guarantees for the column-wise sparsity structure; see Chapter 4 for detailed proofs.

5.4.1 Exact Recovery for Entry-wise Sparsity Case

For the entry-wise case, our main result establishes the existence of a regularization parameter λ_e, for which solving the optimization problem D-RPCA(E) will recover the components $\mathbf{L}$ and $\mathbf{S}$ exactly. To this end, we will show that such a λ_e belongs to a non-empty interval $[\lambda_e^{\text{min}}, \lambda_e^{\text{max}}]$, where λ_e^{min} and λ_e^{max} are defined as

$$\lambda_e^{\text{min}} := \tfrac{1+C_e}{1-C_e}\,\xi_e \text{ and } \lambda_e^{\text{max}} := \tfrac{\sqrt{\alpha_\ell}(1-\mu)-\sqrt{r\alpha_u}\mu}{\sqrt{s_e}}. \tag{5.6}$$

Here, $C_e(\alpha_u, \alpha_\ell, \gamma_{\mathbf{U}}, \gamma_{\mathbf{V}}, s_e, d, k, \mu)$ where $0 \leq C_e < 1$ is a constant that captures the relationship between different model parameters, and is defined as

$$C_e := \tfrac{c}{\alpha_\ell(1-\mu)^2 - c},$$

where $c = \frac{\alpha_u}{2}((1+2\gamma_{\mathbf{U}})(\min(s_e,d)+s_e\gamma_{\mathbf{V}})+2\gamma_{\mathbf{V}}\min(s_e,nm)) - \frac{\alpha_\ell}{2}(\min(s_e,d)+s_e\gamma_{\mathbf{V}})$. Given these definitions, we have the following result for the entry-wise sparsity structure.

Theorem 5.1. Suppose $\mathbf{M} = \mathbf{L} + \mathbf{DS}$, where $\text{rank}(\mathbf{L}) = r$ and $\mathbf{S}$ has at most s_e non-zeros, i.e., $\|\mathbf{S}\|_0 \leq s_e \leq s_e^{\text{max}} := \frac{(1-\mu)^2}{2}\frac{nm}{r}$, and the dictionary $\mathbf{D} \in \mathbb{R}^{f\times d}$ for $d \leq f$ obeys the generalized frame property (4.2) with frame bounds $[\alpha_\ell, \alpha_u]$, where $0 < \alpha_\ell \leq \frac{1}{(1-\mu)^2}$, and $\gamma_{\mathbf{U}}$ follows

$$\gamma_{\mathbf{U}} \leq \begin{cases} \frac{(1-\mu)^2 - 2s_e\gamma_{\mathbf{V}}}{2s_e(1+\gamma_{\mathbf{V}})}, \text{ for } s_e \leq \min\ (d, s_e^{\text{max}}) \\ \frac{(1-\mu)^2 - 2s_e\gamma_{\mathbf{V}}}{2(d+s_e\gamma_{\mathbf{V}})}, \text{ for } d < s_e \leq s_e^{\text{max}}. \end{cases} \tag{5.7}$$

Then given $\mu \in [0,1]$, $\gamma_{\mathbf{U}}$ and $\gamma_{\mathbf{V}} \in [r/nm, 1]$, and ξ_e defined in (5.2), (4.4), (5.5), respectively, $\lambda_e \in [\lambda_e^{\text{min}}, \lambda_e^{\text{max}}]$ with $\lambda_e^{\text{max}} > \lambda_e^{\text{min}} \geq 0$ defined in (4.6), solving D-RPCA(E) will recover matrices $\mathbf{L}$ and $\mathbf{S}$.

We observe that the conditions for the recovery of $(\mathbf{L},\mathbf{S})$ are closely related to the incoherence measures (μ, $\gamma_{\mathbf{V}}$, and $\gamma_{\mathbf{U}}$) between the low-rank part, $\mathbf{L}$, the dictionary, $\mathbf{D}$, and the sparse component $\mathbf{S}$. In general, smaller sparsity, rank, and incoherence parameters are sufficient for ensuring the recovery of the components for a particular problem. This is in line with our intuition that the more distinct the two components, the easier it should be to tease them apart. For our HS demixing problem, this indicates that a target of interest can be localized as long as its the spectral signature is appropriately different from the other materials in the scene.

5.4.2 Recovery for Column-wise Sparsity Case

For the column-wise sparsity model, recall that any pair in the oracle model described in **D.5.1** is considered optimal. To this end, we first establish the sufficient conditions for the existence of such an optimal pair $(\mathbf{L}_0,\mathbf{S}_0)$ by the following lemma.

Lemma 5.1. Given $\mathbf{M}$, $\mathbf{D}$, and $(\mathcal{L},\mathcal{S}_c,\mathcal{D})$, any pair $(\mathbf{L}_0,\mathbf{S}_0) \in \{\mathbf{M},\mathcal{U},\mathcal{I}_{\mathcal{S}_c}\}$ satisfies span{col($\mathbf{L}_0$)} $= \mathcal{U}$ and csupp$(\mathbf{S}_0) = \mathcal{I}_{\mathcal{S}_c}$ if $\mu < 1$.

In essence, we need the incoherence parameter μ to be strictly smaller than 1. Next, analogous to the entry-wise case, we show that λ_c belongs to a non-empty interval $[\lambda_c^{\min},\lambda_c^{\max}]$, using which solving D-RPCA(C) recovers an optimal pair in the oracle model **D.5.1** in accordance with Lemma 5.1. Here, for a constant $C_c := \frac{\alpha_u}{\alpha_\ell}\frac{1}{(1-\mu)^2}\gamma_{\mathbf{V}}\beta_{\mathbf{U}}$, $\lambda_c^{\min}$ and $\lambda_c^{\max}$ are defined as

$$\lambda_c^{\min} := \tfrac{\xi_c+\sqrt{rs_c\alpha_u}\mu C_c}{1-s_cC_c} \text{ and } \lambda_c^{\max} := \tfrac{\sqrt{\alpha_\ell}(1-\mu)-\sqrt{r\alpha_u}\mu}{\sqrt{s_c}}. \tag{5.8}$$

This leads us to the following result for the column-wise case.

Theorem 5.2. Suppose $\mathbf{M} = \mathbf{L} + \mathbf{DS}$ with $(\mathbf{L},\mathbf{S})$ defining the oracle model $\{\mathbf{M},\mathcal{U},\mathcal{I}_{\mathcal{S}_c}\}$, where rank$(\mathbf{L}) = r$, $|\mathcal{I}_{\mathcal{S}_c}| = s_c$ for $s_c \leq s_c^{\max} := \frac{\alpha_\ell}{\alpha_u\gamma_{\mathbf{V}}} \cdot \frac{(1-\mu)^2}{\beta_{\mathbf{U}}}$. Given $\mu \in [0,1)$, $\beta_{\mathbf{U}}$, $\gamma_{\mathbf{V}} \in [r/nm,1]$, ξ_c as defined in (5.2), (5.3), (5.4), (5.5), respectively, and any $\lambda_c \in [\lambda_c^{\min},\lambda_c^{\max}]$, for $\lambda_c^{\max} > \lambda_c^{\min} \geq 0$ defined in (5.8), solving D-RPCA(C) will recover a pair of components $(\mathbf{L}_0,\mathbf{S}_0) \in \{\mathbf{M},\mathcal{U},\mathcal{I}_{\mathcal{S}_c}\}$, if the dictionary $\mathbf{D} \in \mathbb{R}^{f\times d}$ obeys the generalized frame property **D**.5.2 with frame bounds $[\alpha_\ell,\alpha_u]$, for $\alpha_\ell > 0$.

Theorem 5.2 outlines the sufficient conditions under which the solution to the optimization problem D-RPCA(C) will be in the oracle model defined in **D.5.1**. Here,

Algorithm 4: APG Algorithm for D-RPCA(E) and D-RPCA(C), adapted from Mardani et al. (2013)

```
Require: $\mathbf{M}$, $\mathbf{D}$, $\lambda$, $v$, $\nu_0$, $\bar{\nu}$, and $L_f = \lambda_{max}([\mathbf{I}\ \mathbf{D}]^\top[\mathbf{I}\ \mathbf{D}])$
Initialize: $\mathbf{L}[0] = \mathbf{L}[-1] = \mathbf{0}_{L\times T}$, $\mathbf{S}[0] = \mathbf{S}[-1] = \mathbf{0}_{F\times T}$, $t[0] = t[-1] = 1$, and set $k = 0$.
  while not converged do
    Generate points $\mathbf{T}_L[k]$ and $\mathbf{T}_S[k]$ using momentum:
      $\mathbf{T}_L[k] = \mathbf{L}[k] + \frac{t[k-1]-1}{t[k]}(\mathbf{L}[k] - \mathbf{L}[k-1])$,
      $\mathbf{T}_S[k] = \mathbf{S}[k] + \frac{t[k-1]-1}{t[k]}(\mathbf{S}[k] - \mathbf{S}[k-1])$.
    Take a gradient step using these points :
      $\mathbf{G}_L[k] = \mathbf{T}_L[k] + \frac{1}{L_f}(\mathbf{M} - \mathbf{T}_L[k] - \mathbf{D}\mathbf{T}_S[k])$,
      $\mathbf{G}_S[k] = \mathbf{T}_S[k] + \frac{1}{L_f}\mathbf{D}^\top(\mathbf{M} - \mathbf{T}_L[k] - \mathbf{D}\mathbf{T}_S[k])$.
    Update Low-rank part via singular value thresholding:
      $\mathbf{U}\Sigma\mathbf{V}^\top = \text{svd}(\mathbf{G}_L[k])$,
      $\mathbf{L}[k+1] = \mathbf{U}\mathcal{S}_{\nu[k]/L_f}(\Sigma)\mathbf{V}^\top$.
    Update the Dictionary Sparse part:
      $\mathbf{S}[k+1] = \begin{cases} \mathcal{S}_{\nu[k]\lambda_e/L_f}(\mathbf{G}_S[k]), & \text{for D-RPCA(E)}, \\ \mathcal{C}_{\nu[k]\lambda_c/L_f}(\mathbf{G}_S[k]), & \text{for D-RPCA(C)}. \end{cases}$
    Update the momentum term parameter $t[k+1]$:
      $t[k+1] = \frac{1+\sqrt{4t^2[k]+1}}{2}$.
    Update the continuation parameter $\nu[k+1]$:
      $\nu[k+1] = \max\{v\nu[k], \bar{\nu}\}$.
    $k \leftarrow k+1$
  end while
return $\mathbf{L}[k]$, $\mathbf{S}[k]$
```

for a case where $1 \lesssim \alpha_l \leq \alpha_u \lesssim 1$, which can be easily met by a tight frame when $f > d$, constant $\frac{(1-\mu)^2}{\beta_U}$, and $\gamma_{\mathbf{V}} = \Theta(\frac{r}{nm})$, we have $s_c^{\max} = \mathcal{O}(\frac{nm}{r})$, which is of same order as in the Outlier Pursuit (OP) (Xu et al., 2010). Moreover, our numerical results in Chapter 4 show that D-RPCA(C) can be much more robust than OP, and may recover $\{\mathcal{U}, \mathcal{I}_{\mathbf{C}}\}$ even when the rank of $\mathbf{L}$ is high and the number of outliers s_c is a constant proportion of m. This implies that, D-RPCA(C) will succeed as long as the dictionary $\mathbf{D}$ can successfully represent the target of interest while rejecting the columns of the data matrix $\mathbf{M}$ corresponding to materials other than the target.

5.5 Algorithmic Considerations

The optimization problems of interest, D-RPCA(E) and D-RPCA(C), for the entry-wise and column-wise case, respectively, are convex but non-smooth. To solve for the components of interest, we adopt the accelerated proximal gradient (APG) algorithm, as shown in Algorithm 4. Note that Mardani et al. (2013) also applied the APG algorithm for D-RPCA(E), and we present a unified algorithm for both sparsity cases for completeness.

5.5.1 Background

The APG algorithm is motivated from a long line of work starting with Nesterov (1983), which showed the existence of a first order algorithm with a convergence rate of $\mathcal{O}(1/k^2)$ for a smooth convex objective, where k denotes the iterations. Following this, Beck and Teboulle (2009) developed the popular fast iterative shrinkage-thresholding algorithm (FISTA) which achieves this convergence rate for convex non-smooth objectives by accelerating the proximal gradient descent algorithm using a *momentum term* (the term $\frac{t[k-1]-1}{t[k]}$ in Algorithm 4) as prescribed by Nesterov (1983). As a result, it became a staple to solve a wide range of convex non-smooth tasks including matrix completion Toh and Yun (2010), and robust PCA (Chen et al., 2009) and its variants (Mardani et al., 2013; Xu et al., 2010). Also, recently Karimi et al. (2016) has shown further improvements in the rate of convergence.

In addition to the momentum term, the APG procedure operates by evaluating the gradient at a point further in the direction pointed by the negative gradient. Along with faster convergence, this insight about the next point minimizes the oscillations around the optimum point; see Beck and Teboulle (2009) and references therein.

5.5.2 Discussion of Algorithm 4

For the optimization problem of interest, we solve an unconstrained problem by transforming the equality constraint to a least-square term which penalizes the fit. In particular, the problems of interest we will solve via the APG algorithm are given by

$$\min_{\mathbf{L},\mathbf{S}} \ \nu\|\mathbf{L}\|_* + \nu\lambda_e\|\mathbf{S}\|_1 + \tfrac{1}{2}\|\mathbf{M}-\mathbf{L}-\mathbf{D}\mathbf{S}\|_F^2 \tag{5.9}$$

for the entry-wise sparsity case, and

$$\min_{\mathbf{L},\mathbf{S}} \; \nu\|\mathbf{L}\|_* + \nu\lambda_c\|\mathbf{S}\|_{1,2} + \tfrac{1}{2}\|\mathbf{M}-\mathbf{L}-\mathbf{D}\mathbf{S}\|_\mathrm{F}^2, \tag{5.10}$$

for the column-wise sparsity case. We note that although for the application at hand, the thin dictionary case with ($f \geq d$) might be more useful in practice, Algorithm 4 allows for the use of fat dictionaries ($f < d$) as well.

Algorithm 4 also employs a continuation technique (Chen et al., 2009), which can be viewed as a "warm start" procedure. Here, we initialize the parameter ν_0 at some large value and geometrically reduced until it reaches a value $\bar{\nu}$. A smaller choice of $\bar{\nu}$ results in a solution which is closer to the optimal solution of the constrained problem. Further, as ν approaches zero, (5.9) and (5.10) recover the optimal solution of D-RPCA(E) and D-RPCA(C), respectively. Moreover, Algorithm 4 also utilizes the knowledge of the smoothness constant L_f (the Lipschitz constant of gradient) to set the step-size parameter.

Specifically, the APG algorithm requires that the gradient of the smooth part,

$$f(\mathbf{L},\mathbf{S}) := \tfrac{1}{2}\|\mathbf{M}-\mathbf{L}-\mathbf{D}\mathbf{S}\|_\mathrm{F}^2 = \tfrac{1}{2}\|\mathbf{M}-\begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}\begin{bmatrix}\mathbf{L}\\ \mathbf{S}\end{bmatrix}\|_\mathrm{F}^2$$

of the convex objectives shown in (5.9) and (5.10) is Lipschitz continuous with minimum Lipschitz constant L_f. Now, since the gradient $\nabla f(\mathbf{L},\mathbf{S})$ with respect to $\begin{bmatrix}\mathbf{L} & \mathbf{S}\end{bmatrix}^\top$ is given by

$$\nabla f(\mathbf{L},\mathbf{S}) = \begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}^\top(\mathbf{M}-\begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}\begin{bmatrix}\mathbf{L}\\ \mathbf{S}\end{bmatrix}),$$

we have that the gradient ∇f is Lipschitz continuous as

$$\|\nabla f(\mathbf{L}_1,\mathbf{S}_1) - \nabla f(\mathbf{L}_2,\mathbf{S}_2)\| \leq L_f\|\begin{bmatrix}\mathbf{L}_1\\ \mathbf{S}_1\end{bmatrix} - \begin{bmatrix}\mathbf{L}_2\\ \mathbf{S}_2\end{bmatrix}\|,$$

where

$$L_f = \|\begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}^\top\begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}\| = \lambda_{\max}(\begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}^\top\begin{bmatrix}\mathbf{I} & \mathbf{D}\end{bmatrix}),$$

as shown in Algorithm 4.

The update of the low-rank component and the sparse matrix $\mathbf{S}$ for the entry-wise case, both involve a soft thresholding step, $\mathcal{S}_\tau(.)$, where for a matrix $\mathbf{Y}$, $\mathcal{S}_\tau(\mathbf{Y}_{ij})$ is defined as

$$\mathcal{S}_\tau(\mathbf{Y}_{ij}) = \text{sgn}(\mathbf{Y}_{ij})\max(|\mathbf{Y}_{ij} - \tau|, 0).$$

In case of the low-rank part we apply this function to the singular values (therefore referred to as *singular value thresholding*) (Toh and Yun, 2010), while for the update of the dictionary sparse component, we apply it to the sparse coefficient matrix $\mathbf{S}$.

The low-rank update step for the column-wise case remains the same as for the entry-wise case. However, for the update of the column-wise case we threshold the columns of $\mathbf{S}$ based on their column norms, i.e., for a column $\mathbf{Y}_j$ of a matrix $\mathbf{Y}$, the column-norm based soft-thresholding function, $\mathcal{C}_\tau(.)$ is defined as

$$\mathcal{C}_\tau(\mathbf{Y}_j) = \max(\mathbf{Y}_j - \tau\mathbf{Y}_j/\|\mathbf{Y}_j\|).$$

5.5.3 Parameter Selection

Since the choice of regularization parameters by our main theoretical results contain quantities (such as incoherence etc.) that cannot be evaluated in practice, we employ a grid-search strategy over the range of admissible values for the low-rank and dictionary sparse component to find the best values of the regularization parameters. We now discuss the specifics of the grid-search for each sparsity case.

Selecting parameters for the entry-wise case

The choice of parameters ν and λ_e in Algorithm 4 is based on the optimality conditions of the optimization problem shown in (5.9). As presented in Mardani et al. (2013), the range of parameters ν and $\nu\lambda_e$ associated with the low-rank part $\mathbf{L}$ and the sparse coefficient matrix $\mathbf{S}$, respectively, lie in $\nu \in \{0, \|\mathbf{M}\|\}$ and $\nu\lambda_e \in \{0, \|\mathbf{D}^\top\mathbf{M}\|_\infty\}$, i.e., for Algorithm 4 $\nu_0 = \|\mathbf{M}\|$.

These ranges for ν and $\nu\lambda_e$ are derived using the optimization problem shown in (5.9). Specifically, we find the largest values of these regularization parameters which yield a $(\mathbf{0}, \mathbf{0})$ solution for the pair $(\mathbf{L}_0, \mathbf{S}_0)$ by analyzing the optimality conditions of

(5.9). This value of the regularization parameter then defines the upper bound on the range. For instance, let $\lambda_* := \nu$ and $\lambda_1 := \nu\lambda_e$, then the optimality condition is given by

$$\lambda_* \partial_{\mathbf{L}} \|\mathbf{L}\|_* - (\mathbf{M} - \mathbf{L} - \mathbf{D}\mathbf{S}) = 0,$$

where the sub-differential set $\partial_{\mathbf{L}}\|\mathbf{L}\|_*$ is defined as

$$\partial_{\mathbf{L}} \|\mathbf{L}\|_* \Big|_{\mathbf{L}=\mathbf{L}_0} = \{\mathbf{U}\mathbf{V}^\top + \mathbf{W} : \|\mathbf{W}\| \leq 1, \mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}\}.$$

Therefore, for a zero solution pair $(\mathbf{L}_0, \mathbf{S}_0)$ we have that

$$\{\lambda_* \mathbf{W} = \mathbf{M} : \|\mathbf{W}\| \leq 1, \mathcal{P}_{\mathcal{L}}(\mathbf{W}) = \mathbf{0}\},$$

which yields the condition that $\|\mathbf{M}\| \leq \lambda_*$. Therefore, the maximum value of λ_* which drives the low-rank part to an all-zero solution is $\|\mathbf{M}\|$.

Similarly, for the dictionary sparse component the optimality condition for choosing λ_1 is given by

$$\lambda_1 \partial_{\mathbf{S}} \|\mathbf{S}\|_1 - \mathbf{D}^\top (\mathbf{M} - \mathbf{L} - \mathbf{D}\mathbf{S}) = 0,$$

where the the sub-differential set $\partial_{\mathbf{S}}\|\mathbf{S}\|_1$ is defined as

$$\partial_{\mathbf{S}} \|\mathbf{S}\|_1 \Big|_{\mathbf{S}=\mathbf{S}_0} = \{\text{sign}(\mathbf{S}_0) + \mathbf{F} : \|\mathbf{F}\|_\infty \leq 1, \mathcal{P}_{\mathcal{S}_e}(\mathbf{F}) = \mathbf{0}\}.$$

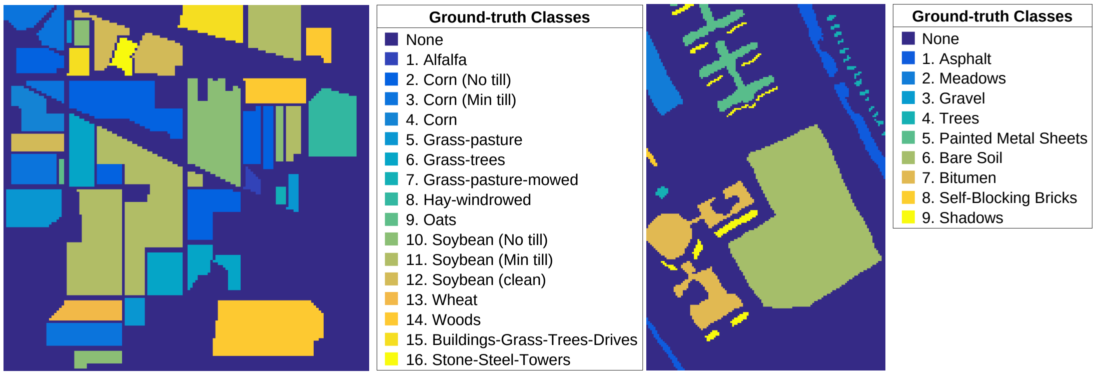
Again, for a zero solution pair $(\mathbf{L}_0, \mathbf{S}_0)$ we need that

$$\{\lambda_1 \mathbf{F} = \mathbf{D}^\top \mathbf{M} : \|\mathbf{F}\|_\infty \leq 1, \mathcal{P}_{\mathcal{S}_e}(\mathbf{F}) = \mathbf{0}\},$$

which implies that $\|\mathbf{D}^\top \mathbf{M}\|_\infty \leq \lambda_1$. Meaning, that the maximum value of λ_1 that drives the dictionary sparse part to zero is $\|\mathbf{D}^\top \mathbf{M}\|_\infty$.

Selecting parameters for the column-wise case

Again, the choice of parameters ν and λ_c is derived from the optimization problem shown in (5.10). In this case, the range of parameters ν and $\nu\lambda_c$ associated with the low-rank part $\mathbf{L}$ and the sparse coefficient matrix $\mathbf{S}$, respectively, lie in $\nu \in \{0, \|\mathbf{M}\|\}$ and $\nu\lambda_e \in \{0, \|\mathbf{D}^\top \mathbf{M}\|_{\infty,2}\}$, i.e., for Algorithm 4 $\nu_0 = \|\mathbf{M}\|$. The range of regularization



(a) Indian Pines

(b) Pavia University

Figure 5.1: Ground-truth classes in the datasets. Panels (a) and (b) show the ground truth classes for the Indian Pines dataset (Baumgardner et al., 2015) and Pavia University dataset (Gamba, 2002), respectively.

parameters are evaluated using the analysis similar to the entry-wise case, by analyzing the optimality conditions for (5.10), instead of (5.9).

5.6 Experimental Evaluation

We now evaluate the performance of the proposed technique on real HS data[2]. We begin by introducing the dataset used for the simulations, following which we describe the experimental set-up and present the results.

5.6.1 Data

Indian Pines Dataset: We first consider the "Indian Pines" dataset (Baumgardner et al., 2015), which was collected over the Indian Pines test site in North-western Indiana in the June of 1992 using the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) (of Technology, 1987) sensor, a popular choice for collecting HS images for various remote sensing applications. This dataset consists of spectral reflectances across 224 bands in wavelength of ranges 400 – 2500 nm from a scene which is composed mostly of agricultural land along with two major dual lane highways, a rail

[2] The code is made available at `https://github.com/srambhatla/Dictionary-based- Robust-PCA`; see Chapter 7 for details.

line and some built structures, as shown in Fig. 5.1(a). The dataset is further processed by removing the bands corresponding to those of water absorption, which results in a HS data-cube with dimensions $\{145 \times 145 \times 200\}$ is as visualized in Fig. 5.1. Here, $n = m = 145$ and $f = 200$. This modified dataset is available as "corrected Indian Pines" dataset (Baumgardner et al., 2015), with the ground-truth containing 16 classes; Henceforth, referred to as the "Indian Pines Dataset". We form the data matrix $\mathbf{M} \in \mathbb{R}^{f \times nm}$ by stacking each voxel of the image side-by-side, which results in a $\{200 \times 145^2\}$ data matrix $\mathbf{M}$. We will analyze the performance of the proposed technique for the identification of the stone-steel towers (class 16 in the dataset), shown in Fig. 5.1(a), which constitutes about 93 voxels in the dataset.

Pavia University Dataset: Acquired using Reflective Optics System Imaging Spectrometer (ROSIS) sensor, the Pavia University Dataset (Gamba, 2002) consists of spectral reflectances across 103 bands (in the range 430 – 860 nm) of an urban landscape over northern Italy. The selected subset of the scene, a $\{201 \times 131 \times 103\}$ data-cube, mainly consists of buildings, roads, painted metal sheets and trees, as shown in Fig. 5.1(b). Note that class-3 corresponding to "Gravel" is not present in the selected data-cube considered here. For our demixing task, we will analyze the localization of target class 5, corresponding to the painted metal sheets, which constitutes 707 voxels in the scene. Note that for this dataset $n = 201$, $m = 131$ and $f = 103$.

5.6.2 Dictionary

We form the known dictionary $\mathbf{D}$ two ways: 1) where a (thin) dictionary is learned based on the voxels using Algorithm 5, and 2) when the dictionary is formed by randomly sampling voxels from the target class. This is to emulate the ways in which we can arrive at the dictionary corresponding to a target – 1) where the *exact signatures* are not available, and/or there is noise, and 2) where we have access to the exact signatures of the target, respectively. Note that, the optimization procedures for D-RPCA(E) and D-RPCA(C) are agnostic to the selection of the dictionary.

In our experiments for case 1), we learn the dictionary using the target class data $\mathbf{Y} \in \mathbb{R}^{f \times p}$ via Algorithm 5, which (approximately) solves the following optimization

Algorithm 5: Dictionary Learning (Mairal et al., 2010; Lee et al., 2007)

```
Require: Data $\mathbf{Y} \in \mathbb{R}^{f \times p}$, regularization parameter $\rho$, and the number of dictionary elements $d$.
Ensure: The dictionary $\mathbf{D} \in \mathbb{R}^{f \times d}$
Initialize: $\widehat{\mathbf{A}} \leftarrow \mathbf{0}_{d \times p}$, $\widehat{\mathbf{D}}$ with $\mathcal{N}(0,1)$ entries and columns normalized to have norm 1, $\widehat{\mathbf{Y}} = \widehat{\mathbf{D}}\widehat{\mathbf{A}}$, and tolerance $\epsilon$ .
  while $\frac{\|\mathbf{Y}-\widehat{\mathbf{Y}}\|_\mathrm{F}}{\|\mathbf{Y}\|_\mathrm{F}} \geq \epsilon$ do
    Update Coefficient Matrix A:
        $$\widehat{\mathbf{A}} = \underset{\mathbf{A}}{\arg.\min} \|\mathbf{Y} - \widehat{\mathbf{D}}\mathbf{A}\|_\mathrm{F}^2 + \rho\|\mathbf{A}\|_1 \qquad (5.11)$$
    Update Dictionary D:
        $$\widehat{\mathbf{D}} = \underset{\mathbf{D}:\|\mathbf{D}_i\|=1}{\arg.\min} \|\mathbf{Y} - \mathbf{D}\mathbf{A}\|_\mathrm{F}^2 \qquad (5.12)$$
    Form Estimate of Data $\widehat{\mathbf{Y}}$:
        $\widehat{\mathbf{Y}} = \widehat{\mathbf{D}}\widehat{\mathbf{A}}$
  end while
return $\widehat{\mathbf{D}}$
```

problem,

$$\widehat{\mathbf{D}} = \underset{\mathbf{D}:\|\mathbf{D}_i\|=1,\mathbf{A}}{\arg.\min} \|\mathbf{Y} - \mathbf{D}\mathbf{A}\|_\mathrm{F}^2 + \rho\|\mathbf{A}\|_1,$$

Algorithm 5 operates by alternating between updating the sparse coefficients (5.11) via FISTA (Beck and Teboulle, 2009) and dictionary (5.12) via the Newton method (Nocedal and Wright, 2006).

For case 2), the columns of the dictionary are set as the known data voxels of the target class. Specifically, instead of learning a dictionary based on a target class of interest, we set it as the exact signatures observed previously. Note that for this case, the dictionary is not normalized at this stage since the specific normalization depends on the particular demixing problem of interest, discussed shortly. In practice, we can store the un-normalized dictionary **D** (formed from the voxels), consisting of actual *signatures* of the target material, and can normalize it after the HS image has been acquired.

5.6.3 Experimental Setup

Normalization of data and the dictionary: For normalizing the data, we divide each element of the data matrix $\mathbf{M}$ by $\|\mathbf{M}\|_\infty$ to preserve the inter-voxel scaling. For the dictionary, in the learned dictionary case, i.e., case 1), the dictionary already has unit-norm columns as a result of Algorithm 5. Further, when the dictionary is formed from the data directly, i.e., for case 2), we divide each element of $\mathbf{D}$ by $\|\mathbf{M}\|_\infty$, and then normalize the columns of $\mathbf{D}$, such that they are unit-norm.

Dictionary selection for the Indian Pines Dataset: For the learned dictionary case, we evaluate the performance of the aforementioned techniques for both entry-wise and column-wise settings for two dictionary sizes, $d = 4$ and $d = 10$, for three values of the regularization parameter ρ, used for the initial dictionary learning step, i.e., $\rho = 0.01,\ 0.1$ and 0.5. Here, the parameter ρ controls the sparsity during the initial dictionary learning step; see Algorithm 5. For the case when dictionary is selected from the voxels directly, we randomly select 15 voxels from the target class-16 to form our dictionary.

Dictionary selection for the Pavia University Dataset: Here, for the learned dictionary case, we evaluate the performance of the aforementioned techniques for both entry-wise and column-wise settings for a dictionary of size $d = 30$ for three values of the regularization parameter ρ, used for the initial dictionary learning step, i.e., $\rho = 0.01,\ 0.1$ and 0.5. Further, we randomly select 60 voxels from the target class-5, when the dictionary is formed from the data voxels.

Comparison with matched filtering (MF)-based approaches: In addition to the robust PCA-based and OP-based techniques introduced in Section 5.2.4, we also compare the performance of our techniques with two MF-based approaches. These MF-based techniques are agnostic to our model assumptions, i.e., entry-wise or column-wise sparsity cases. Therefore, the following description of these techniques applies to both sparsity cases.

For the first MF-based technique, referred to as MF, we form the inner-product of the column-normalized data matrix $\mathbf{M}$, denoted as $\mathbf{M}_n$, with the dictionary $\mathbf{D}$, i.e., $\mathbf{D}^\top\mathbf{M}_n$, and select the maximum absolute inner-product per column. For the second MF-based technique, MF†, we perform matched filtering on the pseudo-inversed data $\widetilde{\mathbf{M}} = \mathbf{D}^\dagger\mathbf{M}$. Here, the matched filtering corresponds to finding maximum absolute entry for each column of the column-normalized $\widetilde{\mathbf{M}}$. Next, in both cases we scan through 1000 threshold values between $(0,1]$ to generate the results.

Performance Metrics: We evaluate the performance of these techniques via the receiver operating characteristic (ROC) plots. ROC plots are a staple for analysis of classification performance of a binary classifier in machine learning; see James et al. (2013) for details. Specifically, it is a plot between the true positive rate (TPR) and the false positive rate (FPR), where a higher TPR (close to 1) and a lower FPR (close to 0) indicate that the classifiier performs detects all the elements in the class while rejecting those outside the class.

A natural metric to gauge good performance is the area under the curve (AUC) metric. It indicates the area under the ROC curve, which is maximized when TPR = 1 and FPR = 0, therefore, a higher AUC is preferred. Here, an AUC of 0.5 indicates that the performance of the classifier is roughly as good as a coin flip. As a result, if a classifier has an AUC < 0.5, one can improve the performance by simply inverting the result of the classifier. This effectively means that AUC is evaluated after “flipping” the ROC curve. In other words, this means that the classifier is good at rejecting the class of interest, and taking the complement of the classifier decision can be used to identify the class of interest.

In our experiments, MF-based techniques often exhibit this phenomenon. Specifically, when the dictionary contains element(s) which resemble the average behavior of the spectral signatures, the inner-product between the normalized data columns and these dictionary elements may be higher as compared to other distinguishing dictionary elements. Since, MF-based techniques rely on the maximum inner-product between the normalized data columns and the dictionary, and further since the spectral signatures of even distinct classes are highly correlated; see, for instance Fig. 5.2, where MF-based approaches in these cases can effectively reject the class of interest. This leads to an AUC < 0.5. Therefore, as discussed above, we invert the result of the classifier (indicated as $(\cdot)_*$ in the tables) to report the best performance. If using MF-based techniques, this issue can potentially be resolved in practice by removing the dictionary elements which tend to resemble the average behavior of the spectral signatures.

Table 5.1: Entry-wise sparsity model for the Indian Pines Dataset. Simulation results are presented for our proposed approach (D-RPCA(E)), robust-PCA based approach on transformed data $\mathbf{D}^{\dagger}\mathbf{M}$ (RPCA†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^{\dagger}\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning procedure ρ; see Algorithm 5 . Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, "$*$" denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance).

(a) Learned dictionary, $d = 4$

d	ρ	Method	Threshold	Performance at best operating point		AUC
				TPR	FPR	
4	0.01	**D-RPCA(E)**	0.300	**0.979**	**0.023**	**0.989**
		RPCA†	0.650	0.957	0.049	0.974
		MF$_*$	N/A	0.957	0.036	0.994
		MF$^{\dagger}_*$	N/A	0.914	0.104	0.946
	0.1	**D-RPCA(E)**	0.800	**0.989**	0.017	0.997
		RPCA†	0.800	0.989	0.014	0.997
		MF	N/A	0.989	0.016	0.998
		MF†	N/A	0.989	**0.010**	**0.998**
	0.5	**D-RPCA(E)**	0.600	**0.968**	**0.031**	**0.991**
		RPCA†	0.600	0.935	0.067	0.988
		MF	N/A	0.548	0.474	0.555
		MF$^{\dagger}_*$	N/A	0.849	0.119	0.939

(b) Learned dictionary, $d = 10$

d	ρ	Method	Threshold	Performance at best operating point		AUC
				TPR	FPR	
10	0.01	**D-RPCA(E)**	0.600	0.935	0.060	0.972
		RPCA†	0.700	**0.978**	**0.023**	**0.990**
		MF$_*$	N/A	0.624	0.415	0.681
		MF$^{\dagger}_*$	N/A	0.569	0.421	0.619
	0.1	**D-RPCA(E)**	0.500	**0.968**	**0.029**	**0.993**
		RPCA†	0.500	0.871	0.144	0.961
		MF$_*$	N/A	0.688	0.302	0.713
		MF†	N/A	0.527	0.469	0.523
	0.5	**D-RPCA(E)**	1.000	**0.978**	**0.031**	**0.996**
		RPCA†	2.200	0.849	0.113	0.908
		MF	N/A	0.807	0.309	0.781
		MF$^{\dagger}_*$	N/A	0.527	0.465	0.539

(c) Dictionary by sampling voxels, $d = 15$

d	Method	Threshold	Performance at best operating point		AUC
			TPR	FPR	
15	**D-RPCA(E)**	0.300	**0.989**	**0.021**	**0.998**
	RPCA†	3.000	0.849	0.146	0.900
	MF	N/A	0.957	0.085	0.978
	MF†	N/A	0.796	0.217	0.857

(d) Average performance

Method	TPR		FPR		AUC	
	Mean	St.Dev.	Mean	St.Dev.	Mean	St.Dev.
D-RPCA(E)	**0.972**	**0.019**	**0.030**	**0.014**	**0.991**	**0.009**
RPCA†	0.919	0.061	0.079	0.055	0.959	0.040
MF	0.796	0.179	0.234	0.187	0.814	0.178
MF†	0.739	0.195	0.258	0.192	0.775	0.207

5.6.4 Parameter Setup for the Algorithms

Entry-wise sparsity case: We evaluate and compare the performance of the proposed method D-RPCA(E) with RPCA† (described in Section 5.2.3), MF, and MF†. Specifically, we evaluate the performance of these techniques via the receiver operating characteristic (ROC) plot for the Indian Pines dataset and the Pavia University dataset, with the results shown in Table 5.1(a)-(d) and Table 5.2(a)-(c), respectively.

For the proposed technique, we employ the accelerated proximal gradient (APG)

Table 5.2: Entry-wise sparsity model and Pavia University Dataset. Simulation results are presented for the proposed approach (D-RPCA(E)), robust-PCA based approach on transformed data (RPCA†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^\dagger\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning step ρ. Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, " $*$ " denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance).

(a) Learned dictionary, $d = 30$

d	ρ	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
				TPR	**FPR**	
30	0.01	**D-RPCA(E)**	0.150	**0.989**	**0.015**	**0.992**
		RPCA†	0.700	0.849	0.146	0.925
		MF	N/A	0.929	0.073	0.962
		MF†	N/A	0.502	0.498	0.498
	0.1	**D-RPCA(E)**	0.050	**0.982**	**0.019**	**0.992**
		RPCA†	3.000	0.638	0.374	0.664
		MF	N/A	0.979	0.053	0.986
		MF†	N/A	0.620	0.381	0.660
	0.5	**D-RPCA(E)**	0.080	**0.982**	**0.019**	**0.992**
		RPCA†	2.500	0.635	0.381	0.671
		MF	N/A	0.980	0.159	0.993
		MF$^\dagger_*$	N/A	0.555	0.447	0.442

(b) Dictionary by sampling voxels, $d = 60$

d	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
			TPR	**FPR**	
60	**D-RPCA(E)**	0.060	**0.986**	0.016	**0.995**
	RPCA†	1.000	0.799	0.279	0.793
	MF	N/A	0.980	**0.011**	0.994
	MF†	N/A	0.644	0.355	0.700

(c) Average performance

Method	**TPR**		**FPR**		**AUC**	
	Mean	**St.Dev.**	**Mean**	**St.Dev.**	**Mean**	**St.Dev.**
D-RPCA(E)	**0.984**	**0.003**	**0.014**	**0.002**	**0.993**	**0.001**
RPCA†	0.730	0.110	0.295	0.110	0.763	0.123
MF	0.967	0.025	0.074	0.062	0.983	0.0149
MF†	0.580	0.064	0.420	0.065	0.575	0.125

algorithm shown in Algorithm 4 and discussed in Section 5.5 to solve the optimization problem shown in D-RPCA(E). Similarly, for RPCA† we employ the APG algorithm with transformed data matrix $\widetilde{\mathbf{M}}$, while setting $\mathbf{D} = \mathbf{I}$.

With reference to selection of tuning parameters for the APG solver for (D-RPCA(E)) (RPCA†, respectively), we choose $v = 0.95$, $\nu = \|\mathbf{M}\|$ ($\nu = \|\widetilde{\mathbf{M}}\|$), $\bar{\nu} = 10^{-4}$, and scan through 100 values of λ_e in the range $\lambda_e \in (0, \|\mathbf{D}^\top\mathbf{M}\|_\infty / \|\mathbf{M}\|]$ ($\lambda_e \in (0, \|\widetilde{\mathbf{M}}\|_\infty / \|\widetilde{\mathbf{M}}\|]$), to generate the ROCs. We threshold the resulting estimate of the sparse part $\mathbf{S} \in \mathbb{R}^{d \times nm}$ based on its column norm. We choose the threshold such that the AUC metric is maximized for both cases (D-RPCA(E) and RPCA†).

Column-wise sparsity case: For this case, we evaluate and compare the performance of the proposed method D-RPCA(C) with OP† (as described in Section 5.2.3), MF, and MF†. The results for the Indian Pines dataset and the Pavia University dataset as shown in Table 5.3(a)-(d) and Table 5.4(a)-(c), respectively.

As in the entry-wise sparsity case, we employ the accelerated proximal gradient (APG) algorithm presented in Algorithm 4 to solve the optimization problem shown

Table 5.3: Column-wise sparsity model and Indian Pines Dataset. Simulation results are presented for the proposed approach (D-RPCA(C)), Outlier Pursuit (OP) based approach on transformed data (OP†), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^\dagger\mathbf{M}$ (MF†), across dictionary elements d, and the regularization parameter for initial dictionary learning step ρ. Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, "$*$" denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance).

(a) Learned dictionary, $d = 4$

d	ρ	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
				TPR	**FPR**	
4	0.01	**D-RPCA(C)**	0.905	**0.989**	**0.014**	**0.998**
		OP†	0.895	0.989	0.015	0.998
		MF$_*$	N/A	0.656	0.376	0.611
		MF$^\dagger_*$	N/A	0.624	0.373	0.639
	0.1	**D-RPCA(C)**	0.805	**0.989**	**0.013**	**0.998**
		OP$^\dagger_*$	1.100	0.720	0.349	0.682
		MF$_*$	N/A	0.742	0.256	0.780
		MF†	N/A	0.828	0.173	0.905
	0.5	**D-RPCA(C)**	1.800	**0.989**	**0.010**	**0.998**
		OP†	1.300	0.989	0.012	0.998
		MF	N/A	0.548	0.474	0.556
		MF$^\dagger_*$	N/A	0.849	0.146	0.939

(b) Learned dictionary, $d = 10$

d	ρ	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
				TPR	**FPR**	
10	0.01	**D-RPCA(C)**	0.800	**0.946**	**0.016**	**0.993**
		OP†	1.300	0.946	0.060	0.988
		MF$_*$	N/A	0.946	0.060	0.987
		MF$^\dagger_*$	N/A	0.527	0.468	0.511
	0.1	**D-RPCA(C)**	0.550	**0.979**	**0.029**	**0.997**
		OP†	0.800	0.893	0.112	0.928
		MF$_*$	N/A	0.688	0.302	0.714
		MF†	N/A	0.527	0.470	0.523
	0.5	**D-RPCA(C)**	1.400	**0.989**	**0.037**	**0.997**
		OP†	0.800	0.807	0.148	0.847
		MF	N/A	0.807	0.309	0.781
		MF$^\dagger_*$	N/A	0.527	0.468	0.539

(c) Dictionary by sampling voxels, $d = 15$

d	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
			TPR	FPR	
15	**D-RPCA(C)**	0.800	0.989	0.018	0.998
	OP†	2.200	0.882	0.126	0.900
	MF	N/A	0.957	0.085	0.978
	MF†	N/A	0.796	0.217	0.857

(d) Average performance

Method	**TPR**		**FPR**		**AUC**	
	Mean	St.Dev.	Mean	St.Dev.	Mean	St.Dev.
D-RPCA(C)	**0.981**	**0.016**	**0.020**	**0.010**	**0.997**	**0.002**
OP†	0.889	0.099	0.117	0.115	0.906	0.114
MF	0.763	0.151	0.266	0.149	0.772	0.166
MF†	0.668	0.151	0.331	0.148	0.702	0.192

in D-RPCA(C). Similarly, for OP† we employ the APG with transformed data matrix $\widetilde{\mathbf{M}}$, while setting $\mathbf{D} = \mathbf{I}$. For the tuning parameters for the APG solver for (D-RPCA(C)) (OP†, respectively), we choose $v = 0.95$, $\nu = \|\mathbf{M}\|$ ($\nu = \|\widetilde{\mathbf{M}}\|$), $\bar{\nu} = 10^{-4}$, and scan through 100 λ_cs in the range $\lambda_c \in (0, \|\mathbf{D}^\top\mathbf{M}\|_{\infty,2}/\|\mathbf{M}\|]$ ($\lambda_c \in (0, \|\widetilde{\mathbf{M}}\|_{\infty,2}/\|\widetilde{\mathbf{M}}\|]$), to generate the ROCs. As in the previous case, we threshold the resulting estimate of the sparse part $\mathbf{S} \in \mathbb{R}^{d\times nm}$ based on its column norm.

Table 5.4: Column-wise sparsity model and Pavia University Dataset. Simulation results for the proposed approach (D-RPCA(C)), Outlier Pursuit (OP) based approach ($\text{OP}^\dagger$), matched filtering (MF) on original data $\mathbf{M}$, and matched filtering on transformed data $\mathbf{D}^\dagger\mathbf{M}$ ($\text{MF}^\dagger$), across dictionary elements d, and the regularization parameter for initial dictionary learning step ρ. Threshold selects columns with column-norm greater than threshold such that AUC is maximized. For each case, the best performing metrics are reported in bold for readability. Further, " $*$ " denotes the case where ROC curve was "flipped" (i.e. classifier output was inverted to achieve the best performance).

(a) Learned dictionary, $d = 30$

d	ρ	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
				TPR	**FPR**	
30	0.01	**D-RPCA(C)**	0.065	**0.990**	**0.015**	**0.991**
		$\mathbf{OP}^\dagger$	0.800	0.7581	0.3473	0.705
		MF	N/A	0.929	0.073	0.962
		$\mathbf{MF}^\dagger$	N/A	0.502	0.50	0.498
	0.1	**D-RPCA(C)**	0.070	**0.996**	**0.022**	**0.994**
		$\mathbf{OP}^\dagger$	0.100	0.989	0.3312	0.904
		MF	N/A	0.979	0.053	0.986
		$\mathbf{MF}^\dagger$	N/A	0.62	0.3814	0.66
	0.5	**D-RPCA(C)**	0.035	**0.983**	**0.017**	**0.995**
		$\mathbf{OP}^\dagger$	0.200	0.940	0.264	0.887
		MF	N/A	0.980	0.160	0.993
		$\mathbf{MF}^\dagger_*$	N/A	0.555	0.447	0.442

(b) Dictionary by sampling voxels, $d = 60$

d	**Method**	**Threshold**	**Performance at best operating point**		**AUC**
			TPR	FPR	
60	**D-RPCA(C)**	0.020	**0.993**	0.022	**0.994**
	$\mathbf{OP}^\dagger$	0.250	0.963	0.264	0.907
	MF	N/A	0.980	**0.011**	0.994
	$\mathbf{MF}^\dagger$	N/A	0.644	0.355	0.700

(c) Average performance

Method	**TPR**		**FPR**		**AUC**	
	Mean	St.Dev.	Mean	St.Dev.	Mean	St.Dev.
D-RPCA(C)	**0.990**	**0.006**	**0.015**	**0.003**	**0.993**	**0.002**
$\mathbf{OP}^\dagger$	0.912	0.105	0.302	0.044	0.850	0.098
MF	0.97	0.025	0.074	0.063	0.984	0.015
$\mathbf{MF}^\dagger$	0.580	0.064	0.4208	0.065	0.575	0.124

5.6.5 Analysis

Table 5.1–5.2 and Table 5.3–5.4 show the ROC characteristics and the classification performance of the proposed techniques D-RPCA(E) and D-RPCA(C), for two datasets under consideration, respectively, under various choices of the dictionary $\mathbf{D}$ and regularization parameter ρ for Algorithm 5. We note that both proposed techniques D-RPCA(E) and D-RPCA(C) on an average outperform competing techniques, emerging as the most reliable techniques across different dictionary choices for the demixing task at hand; see Tables 5.1(d), 5.2(c), 5.3(d), and 5.4(c).

Further, the performance of D-RPCA(C) is slightly better than D-RPCA(E). This can be attributed to the fact that the column-wise sparsity model does not require the columns of $\mathbf{S}$ to be sparse themselves. As alluded to in Section 5.2.2, this allows for higher flexibility in the choice of the dictionary elements for the thin dictionary case.

In addition, we see that the matched filtering-based techniques (and even $\text{OP}^\dagger$ based technique for $d = 4$ and $\rho = 0.1$ in Table 5.3) exhibit "flip" or inversion of the

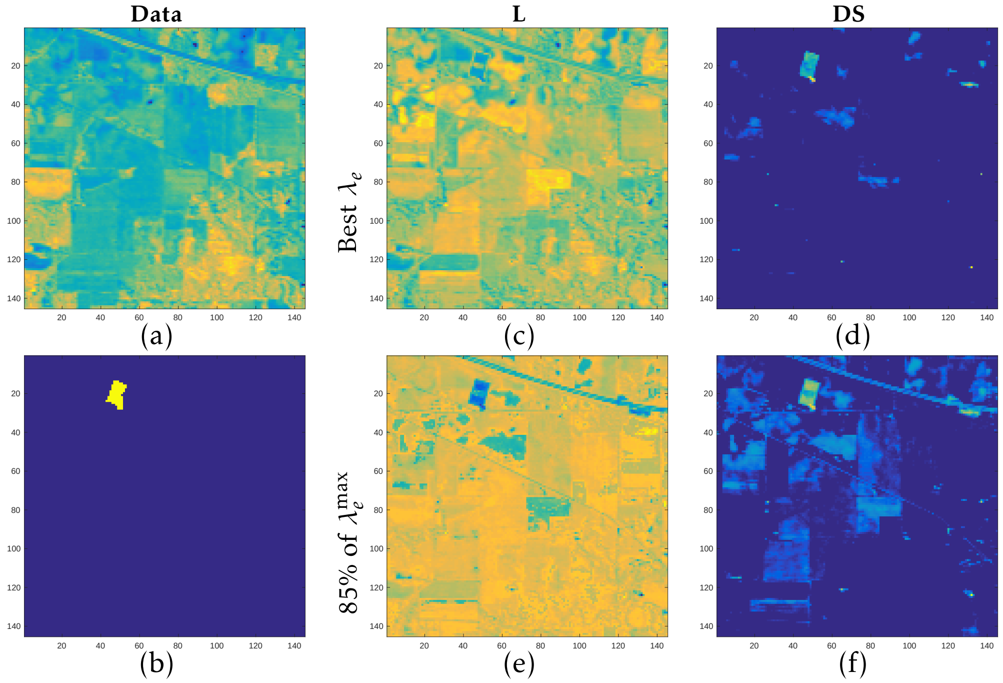


Figure 5.2: Recovery of the low-rank component **L** and the dictionary sparse component **DS** for different values of λ for the proposed technique at $f = 50$-th channel of the (Baumgardner et al., 2015) (shown in panel (a)) corresponding to the results shown in Table 5.1(c). Panel (b) corresponds to the ground truth for class-16. Panel (c) and (d) show the recovery of the low-rank part and dictionary sparse part for a λ at the best operating point. While, panels (e) and (f) show the recovery of these components at $\lambda_e = 85\%$ of $\lambda_e^{\max}$. Here, $\lambda_e^{\max}$ denotes the maximum value λ_e can take; see Section 5.5.3.

ROC curve. As described in Section 5.6.3, this phenomenon is an indicator that a classifier is better at rejecting the target class. In case of MF-based technique, this is a result of a dictionary that contains an element that resembles the average behavior of the spectral responses. A similar phenomenon is at play in case of the OP† for $d = 4$ and $\rho = 0.1$ in Table 5.3. Specifically, here the inversion indicates that the dictionary is capable of representing the columns of the data **M** effectively, which leads to an increase in the corresponding column norms in their representation $\widehat{\mathbf{M}}$. Coupled with the fact that the component **L** is no longer low-rank for this thin dictionary case (see our discussion in Section 5.2.4), this results in rejection of the target class. On the other hand, our techniques D-RPCA(E) and D-RPCA(C) do not suffer from this issue. Moreover, note that across all the experiments, the thresholds for RPCA† and OP† are higher than their D-RPCA counterparts. This can also be attributed to the pre-multiplication by the pseudo-inverse of the dictionary $\mathbf{D}^{\dagger}$, which increases column norms based on the leading singular values of **D**. Therefore, using D-RPCA(E), when the target spectral response admits a sparse representation, and D-RPCA(C), otherwise, yield consistent and superior results as compared to related techniques considered in this work.

There are other interesting recovery results which warrant our attention. Fig. 5.2 shows the low-rank and the dictionary sparse component recovered by D-RPCA(E) for two different values of λ_e, for the case where we form the dictionary by randomly sampling the voxels (Table 5.1(c)) for the Indian Pines Dataset (Baumgardner et al., 2015). Interestingly, we recover the rail tracks/roads running diagonally on the top-right corner, along with some low-density housing; see Fig 5.2 (f). This is because the *signatures* we seek (stone-steel towers) are similar to the signatures of the materials used in these structures. This further corroborates the applicability of the proposed approach in detecting the presence of a particular spectral *signature* in a HS image. However, this also highlights potential drawback of this technique. As D-RPCA(E) and D-RPCA(C) are based on identifying materials with similar composition, it may not be effective in distinguishing between very closely related classes, say two agricultural crops, also indicated by our theoretical results.

5.7 Conclusions

We present a generalized robust PCA-based technique to localize a target in a HS image, based on the *a priori* known spectral *signature* of the material we wish to localize. We model the data as being composed of a low-rank component and a dictionary-sparse component, and consider two different sparsity patterns corresponding to different structural assumptions on the data, where the dictionary contains the *a priori* known spectral *signatures* of the target. We adapt the theoretical results of Chapter 4, to present the conditions under which such decompositions recover the two components for the HS demixing task. Further, we evaluate and compare the performance of the proposed method via experimental evaluations for a classification task for different choices of the dictionary on real HS image datasets, and demostrate the applicability of the proposed techniques for a target localization in HS images.

Part III

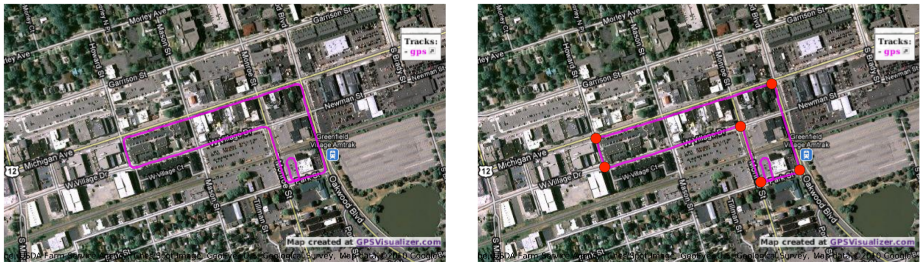
Application-Focused Techniques

Chapter 6

Lidar-Based Topological Mapping and Localization via Tensor Decompositions

6.1 Overview

We propose a technique to develop (and localize in) topological maps from light detection and ranging (Lidar) data. Localizing an autonomous vehicle with respect to a reference map in real-time is crucial for its safe operation. Owing to the rich information provided by Lidar sensors, these are emerging as a promising choice for this task. However, since a Lidar outputs a large amount of data every fraction of a second, it is progressively harder to process the information in real-time. Consequently, current systems have migrated towards faster alternatives at the expense of accuracy. To overcome this inherent trade-off between latency and accuracy, we propose a technique to develop topological maps from Lidar data using the orthogonal Tucker3 tensor decomposition. Our experimental evaluations demonstrate that in addition to achieving a high compression ratio as compared to full data, the proposed technique, *TensorMap*, also accurately detects the position of the vehicle in a graph-based representation of a map. We also analyze the robustness of the proposed technique to Gaussian and translational noise, thus initiating explorations into potential applications of tensor decompositions in Lidar data analysis.



(a) (b)

Figure 6.1: The Ford Dataset (Pandey et al., 2011). Panels (a) and (b) show the trajectory traced by the vehicle, and nodes of a representative topological map (in red), respectively.

6.2 Introduction

Autonomous vehicles are gaining significant traction due to the advent of smaller footprint, yet fast processors. One of the major steps in autonomous vehicle navigation is to keep track of the state of the vehicle which, among other things, includes the position of the vehicle with respect to the global frame of reference. For this, vehicles often employ a wide range of sensors like GPS, cameras and inertial measurement units (IMU). However, these sensors usually do not provide the accuracies required to establish safe (and stable) operation.

The advances in Lidar technology coupled with its increasing affordability have made it the most popular sensor for tracking position with millimeter accuracies. However, the Lidar technology comes with its own set of drawbacks. Each *scan* (the range measurements received by the sensors at different angles of azimuth and elevation) obtained by the Lidar sensor is a point cloud containing millions of data points. Although this data provides very accurate details about the operating environment, the sheer volume of the data thrown at the processor every fraction of a second, often forces us to choose between speed of operation (latency) and accuracy.

One way of addressing this issue is to develop efficient representations of the map. To develop these representations, often a map as the one shown in Fig. 6.1 (a), can be viewed as a graph with nodes as turns/landmarks, with roads as the edges or segments of the graph. Such a map is known as a *topological map*; Fig. 6.1 (b) shows an example of the nodes in such a map. The problem of localization then becomes a problem of identifying which segment the vehicle is on, and how far along in the segment it is positioned.

6.2.1 Prior-Art

Building topological maps for localization using imaging-based techniques has gained traction in recent times since these are inexpensive to implement and faster to process (Siagian and Itti, 2009; Wang et al., 2006; Fraundorfer et al., 2007; Booij et al., 2007; Chang et al., 2010; Milford and Wyeth, 2012; Schindler et al., 2007; Angeli et al., 2009), as compared to Lidar sensors. However, these vision-based techniques are sensitive to changing weather and illumination (day and night).

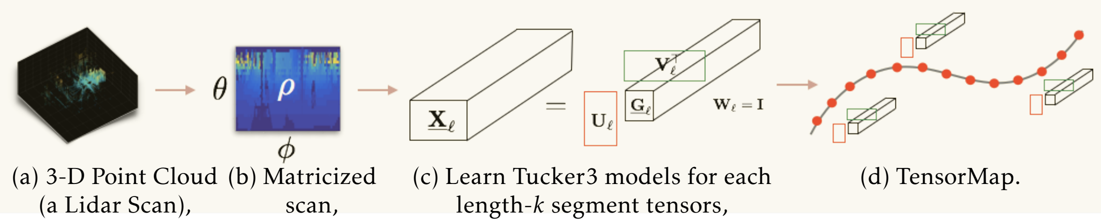
The process of identifying the rigid body transformation that aligns a scan with a map is known as *scan matching*, and is a very effective choice for localization. Significant advances have been made in the area of developing better and accurate representations for scan-matching using Lidar data (Besl and McKay, 1992; Biber and Straßer, 2003; Morris et al., 2005; Myronenko and Song, 2010; Mueller et al., 2011), but the time, and computational overhead, associated with it are still prohibitive. The state-of-the-art techniques deal with the computational overhead by acquiring Lidar data at lower rate in order to operate in real-time (Zhang and Singh, 2014, 2015).

On the other hand, low rank tensor models, specifically Tucker3 (Tucker and Ledyard, 1966) decomposition, popularized by the higher-order singular value decomposition (HO-SVD) technique(Lathauwer et al., 2000), have gained success in a wide variety of applications; see Kolda and Bader (2009); Sidiropoulos et al. (2017) and the references therein for details. Viewed as a generalization of SVD, here the tensor is factorized as *core* tensor multiplied by factor matrices in each dimension (mode); the size of the matrices controlling the respective mode ranks (collectively, the so-called multilinear rank of the tensor). In addition to compressing approximately low multi-linear rank tensors, this decomposition exhibits an interesting property – the core tensor is *all orthogonal*, i.e., each slice of this tensor is orthogonal to all the other slices; see Lathauwer et al. (2000) for details.

It is worth noting that recently, Li et al. (2017) employed tensor models to classify objects in a Lidar scan based on dictionary learning. As opposed to this work, our aim here is to localize a vehicle on a map using the Lidar scans.

6.2.2 Summary of Our Technique

In this work, we present a tensor decompositions-based technique for building topological maps using Lidar data. To this end, we first represent the 3D-point cloud Lidar


θ
ρ
ϕ
$\underline{\mathbf{X}}_\ell$
$=$
$\mathbf{U}_\ell$
$\underline{\mathbf{G}}_\ell$
$\mathbf{V}_\ell^\top$
$\mathbf{W}_\ell = \mathbf{I}$

(a) 3-D Point Cloud (a Lidar Scan), (b) Matricized scan, (c) Learn Tucker3 models for each length-k segment tensors, (d) TensorMap.

Figure 6.2: Learning the topological map. We represent each 3-D point cloud corresponding to each Lidar scan (a), as a matrix (b) after conversion to polar coordinates. We aggregate the matricized scans to form length-k segment tensors $\underline{\mathbf{X}}_\ell$, and learn the orthogonal Tucker3 models on each of these (shown in panels (c) and (d)).

scans as a 3-way tensor. Next, we learn orthogonal Tucker3 models on partitions of this tensor by exploiting the approximate low multi-linear rank structure, arising from the fact that scans in a local neighborhood – specifically straight paths – are similar; see Fig. 6.2. Further, we develop a technique to localize in this map by leveraging the "all-orthogonal" property of the aforementioned tensor decomposition; see Fig. 6.3. To the best of our knowledge, this is the first application to exploit the orthogonality of the core tensor slices.

6.2.3 Our Contributions

We make the following contributions: 1) we develop TensorMap[1]: a technique to build Lidar-based topological maps using tensor decompositions and perform localization in them, 2) we analyze the efficiency of the proposed representation in terms of its space complexity in comparison to using the full Lidar data, 3) we show the performance of TensorMap for a localization task on real Lidar data, and 4) we demonstrate the robustness properties of the proposed technique to different types of simulated noise (Gaussian and translational).

The rest of the chapter is organized as follows. We formulate the problem and describe TensorMap in Section 6.3. In Section 6.4, we discuss parameter selection, simulations results, and other applications, and provide a few concluding remarks in Section 6.5.

[1] Details about the implementation can be found at `https://github.com/srambhatla/TensorMap`; see Chapter 7 for details.

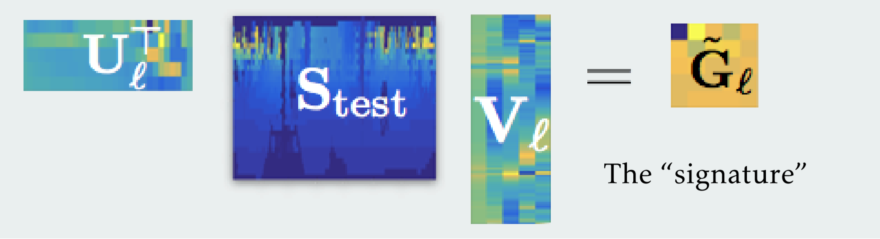


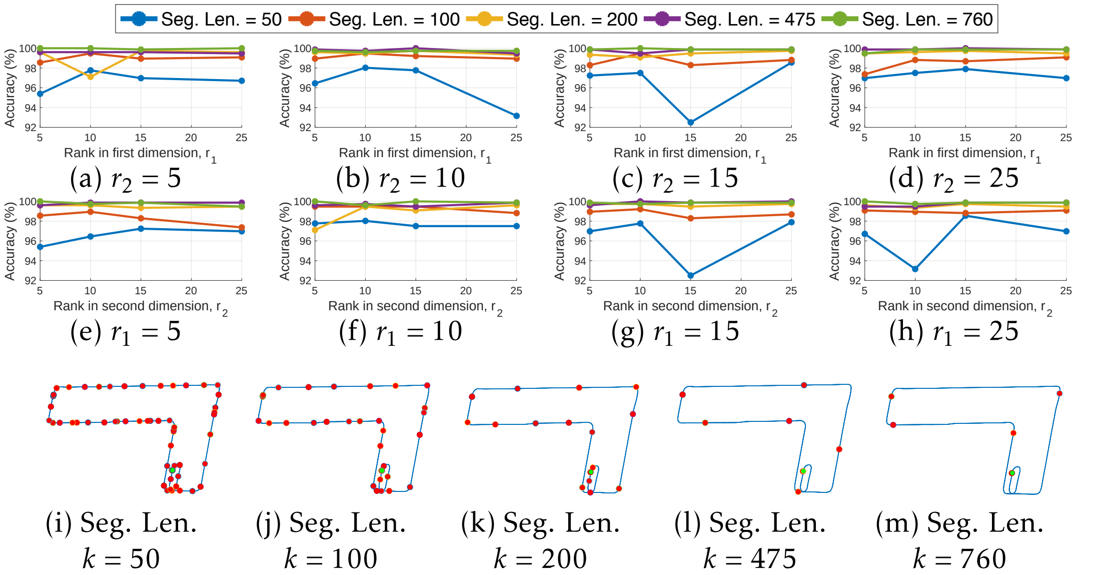
Figure 6.3: Localizing based on a scan. Each test scan, after matricization (as described in Section 6.3.3), is processed by each $\mathbf{U}_\ell$ and $\mathbf{V}_\ell$ to form "signatures" $\widetilde{\mathbf{G}}_\ell$, which are then compared (in Frobenius norm sense) to the core tensors $\underline{\mathbf{G}}_\ell$ of TensorMap for best match.

6.3 Problem formulation

We illustrate TensorMap using the Ford campus vision and Lidar dataset Pandey et al. (2011), henceforth referred to as "the Ford Dataset." The Ford Dataset contains a set of 3800 Lidar scans corresponding to a loop in downtown Dearborn, Michigan. The trajectory of the scans collected by the Ford Dataset is shown in Fig. 6.1(a). The data is collected using a Velodyne 3D-Lidar scanner which has a vertical field of view (FOV) of 26.3° (apx. from $-25°$ to 4°) and a lateral FOV of 360° (from $[-180°, 180°]$), with the Lidar spinning at 10 Hz.

6.3.1 Modeling Lidar data as a Tensor

Each scan in the dataset is a list of about 77,000 *returns* or a point cloud represented in 3D Cartesian coordinates i.e. (x, y, z) corresponding to the position of objects reflecting the incident laser, as shown in Fig. 6.2 (a). Here, the number of returns per scan depends on the scene. To represent Lidar scans as a tensor, we first convert the the data to polar coordinates, which results in a list of returns expressed as (ρ, θ, ϕ), where ρ is the range, θ is the elevation and ϕ is the azimuth. Next, we form a matrix with rows corresponding to elevation angles θ, and columns corresponding to azimuth angles ϕ, by rounding these to whole angles (this discretization is a design choice). Then, for each entry in the list of returns in polar coordinates, we place the range values (ρ) at the rounded-off (θ, ϕ) location, as shown in Fig. 6.2 (b). Due to this quantization (of the θ and ϕ), multiple returns may get mapped to a single entry of the matrix. For the given sensor, θ is restricted between $[-25°, 4°]$ and ϕ between $[-180°, 180°]$. Therefore, each scan is transformed to a 30×361 matrix, and collecting all the scans, results in a



(a) $r_2 = 5$ (b) $r_2 = 10$ (c) $r_2 = 15$ (d) $r_2 = 25$

(e) $r_1 = 5$ (f) $r_1 = 10$ (g) $r_1 = 15$ (h) $r_1 = 25$

(i) Seg. Len. $k = 50$ (j) Seg. Len. $k = 100$ (k) Seg. Len. $k = 200$ (l) Seg. Len. $k = 475$ (m) Seg. Len. $k = 760$

Figure 6.1: Effect of choice of $\{r_1, r_2, k\}$ on the performance accuracy. Panels (a-d) show the effect of choice of segment lengths k and varying r_1 for fixed $r_2 = 5, 10, 15$, and 25, respectively. Similarly, panels (e-h) show the effect of choice of segment lengths $k$and r_2 for fixed $r_1 = 5, 10, 15$, and 25, respectively. Here, segment lengths k considered are $50, 100, 200, 475$, and 760. Panels (i)-(m) show the nodes for each segment corresponding to choice of k (in red), with the start/end point of the path denoted in green.

tensor $\underline{\mathbf{X}}$. Therefore, for the Ford data set $\underline{\mathbf{X}} \in 30 \times 361 \times 3800$.

6.3.2 Building TensorMap

For learning the topological map, we use the orthogonal Tucker decomposition to exploit the low mode-rank (in two of the three modes) structure of the tensor. Lidar data is particularly amenable to this model because the scene at each step is highly correlated to the previous one. To leverage this relationship, let $\underline{\mathbf{X}}$ denote a tensor in $\mathbb{R}^{\mathrm{I}\times\mathrm{J}\times\mathrm{K}}$ containing all scans corresponding to a map. Next, let $\underline{\mathbf{X}}_\ell \in \mathbb{R}^{\mathrm{I}\times\mathrm{J}\times\mathrm{k}}$ denote length-k disjoint partitions of $\underline{\mathbf{X}}$ for each $\ell = \{1, 2, \ldots, \mathrm{L}\}$ for $\mathrm{L} = \mathrm{K}/k$, where we assume that k divides K perfectly; see Fig. 6.2(c). As a result, we have short tensors $\underline{\mathbf{X}}_\ell$ for each length-k segment along the path whose orthogonal Tucker3 decomposition can be written as

$$\mathrm{vec}(\underline{\mathbf{X}}_\ell) = (\mathbf{U}_\ell \otimes \mathbf{V}_\ell \otimes \mathbf{W}_\ell)\bar{\mathbf{g}}_\ell.$$

Here, "$\otimes$" denotes kronecker product, $\mathbf{U}_\ell \in \mathbb{R}^{\mathrm{I}\times r_1}$, $\mathbf{V}_\ell \in \mathbb{R}^{\mathrm{J}\times r_2}$ and $\mathbf{W}_\ell \in \mathbb{R}^{\mathrm{k}\times\mathrm{k}}$ denote the factors where $r_1 \le \mathrm{I}$ and $r_2 \le \mathrm{J}$, and $\bar{\mathbf{g}}_\ell$ denotes the vectorized core tensor $\underline{\mathbf{G}}_\ell$ shown in

Fig. 6.2(c). Note that, to preserve the position information we do not compress along the third dimension of the segment tensor $\underline{\mathbf{X}_\ell}$, i.e., we set $\mathbf{W}_\ell = \mathbf{I}$, where $\mathbf{I}$ denotes an $\mathrm{k} \times \mathrm{k}$ identity matrix. The core tensor $\underline{\mathbf{G}_\ell} \in \mathbb{R}^{\mathrm{r}_1 \times \mathrm{r}_2 \times \mathrm{k}}$ along with factors $\mathbf{U}_\ell$ and $\mathbf{V}_\ell$ corresponding to each segment form the TensorMap, as shown in Fig. 6.2(d).

6.3.3 Localizing in TensorMap

Since each $\mathrm{r}_1 \times \mathrm{r}_2$ slice of the core tensor $\underline{\mathbf{G}_\ell} \in \mathbb{R}^{\mathrm{r}_1 \times \mathrm{r}_2 \times \mathrm{k}}$ (corresponding to the scans in a segment) is orthogonal to the other slices, each slice of the core tensor can be viewed as a “signature” of the associated scan. As shown in Fig. 6.3, we exploit this property for localization. Specifically, to localize any test scan (point cloud), we first convert it into a matrix $\mathbf{S}_{\text{test}}$ as described in Section 6.3.1. Next, we form “signature” $\widetilde{\mathbf{G}}_\ell$ corresponding to $\mathbf{S}_{\text{test}}$ as

$$\widetilde{\mathbf{G}}_\ell = \mathbf{U}_\ell^\top \mathbf{S}_{\text{test}} \mathbf{V}_\ell,$$

for all $\ell \in \{1, 2, \ldots, \mathrm{L}\}$. Then, we find the closest matching core tensor slice $\mathbf{G}_\ell$ (in Frobenius norm sense) across all segments. This process identifies the scan that is a closest match to the test scan, hence also identifies the segment.

6.3.4 Memory Considerations

We consider the space complexity of TensorMap for its implementation on real-world systems and embedded platforms. We propose to learn a orthogonal Tucker3 model for each length-k segment, and there are L such models to be learnt. Therefore, the total number of memory units required to store TensorMap are,

$$\mathrm{L}(\mathrm{I}\mathrm{r}_1 + \mathrm{J}\mathrm{r}_2) + \mathrm{K}\mathrm{r}_1\mathrm{r}_2.$$

This storage requirement is significantly smaller than the original tensor, i.e. IJK, for small values of r_1, r_2 and L. Note that we do not store $\mathbf{W}_\ell$ since in each case it is an identity matrix.

Interestingly, the expression above supports longer segments which still yield a lower error for smaller r_1 and r_2. In the context of maps, this means that scans of a segment should be accumulated as long as they are similar to each other. Therefore, suitable segment length is closely related to the number of straight line paths in the map. Note that, although we consider a fixed segment length for the current

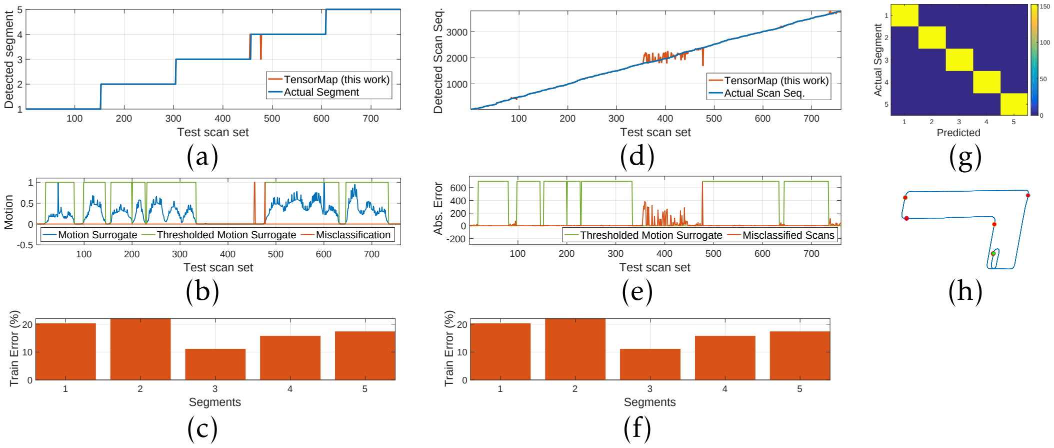


Figure 6.1: Performance of TensorMap on the Ford Dataset with $\{r_1, r_2, k\}$ chosen as $\{5, 5, 760\}$, respectively. Panel (a) shows the classification of test scans into segments. The corresponding surrogate for velocity (blue), the decision of vehicle movement (green), and the errors made by TensorMap (red) are shown in panel (b). Notice how majority of the errors occur when the vehicle is stationary. Panels (c) and (f) show the relative error between the original segment tensor and the model learnt by TensorMap. Panel (d) shows the scan classification performance of the technique, actual test set (blue) the closest (Frobenius norm) train set scan found by TensorMap. The corresponding decision of vehicle movement (green) and the errors made (red) are shown in (e). Panel (g) shows the confusion matrix corresponding to the classification of test scans to segments shown in (a), and (h) shows the nodes of TensorMap (red) superimposed on the actual map (blue).

exposition, there is no requirement that the segments be of equal length. We leave exploration of these extensions to future work.

6.4 Numerical Evaluations

6.4.1 Experimental Set-up

We evaluate the performance of TensorMap based on its classification accuracy of assigning test scans to their respective segments, using a 80 : 20 - Train : Test split of scans in each segment. To this end, we first learn orthogonal Tucker representations (TensorMap) on the training data for each segment using the HO-SVD algorithm (Lathauwer et al., 2000; Kolda and Bader, 2009). We also analyze the within-segment classification performance by analyzing the train scan sequence which was found closest to the test sequence.

6.4.2 Selecting the Parameters

There are a few design parameters that we need to choose, namely the length of the segment k, and the number of columns r_1 and r_2 in factors $\mathbf{U}_\ell$ and $\mathbf{V}_\ell$, respectively. To find the best choice(s), we search over various values of r_1, r_2, and k, to arrive at a $\{r_1, r_2, k\}$ which yields highest accuracy, while being efficient in terms of the storage requirements.

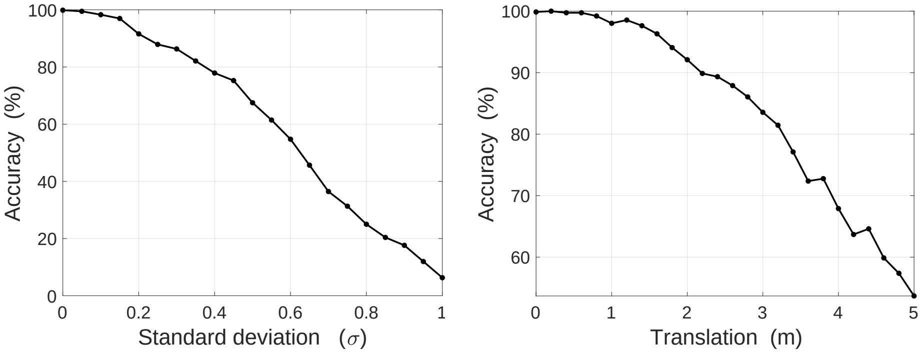
Fig. 6.1 shows accuracies over different choices of $\{r_1, r_2\}$, and segment lengths k. We observe that for a specific choice of r_1 and r_2, the segment classification performance is better for longer segments as compared to shorter ones. This is because scans in shorter segments are very similar to those in neighboring segments; see Fig. 6.1 (i)-(m). Also, although longer segments choices sometimes perform better for larger values of r_1 and r_2, we prefer smaller r_1 and r_2 to reduce the computational and memory overhead. Overall, by this analysis, we arrive at the choice of $\{5, 5, 760\}$ for $\{r_1, r_2, k\}$, respectively.

6.4.3 Results

In Fig. 6.1, we present the results for $\{r_1, r_2, k\}$ chosen as $\{5, 5, 760\}$, respectively. We observe that our method identifies the test segments accurately, except for two scans; see Fig. 6.1(a). To investigate these misclassifications, we turn to Fig. 6.1(b), which shows the relationship of the errors with the motion surrogate, which is formed by evaluating the norm of change in 6-DOF pose – provided by the Ford Dataset – of the vehicle. We observe that the errors seem to arise only when the vehicle is stationary. This is due to the fact that the scan acquisition process does not stop when the vehicle is not moving. As a result, scenes in consecutive segments can be very similar to each other. However, attributing scans to any one of the these segments does not adversely effect the localization performance. Therefore, to account for this effect we report errors on parts where the vehicle is moving, using the motion surrogate.

In panel Fig. 6.1(d) and (e), we show the actual train scan (scan sequence number) found to be the closest to the test set and the misclassified scans, respectively. We note that when the vehicle is in motion, TensorMap indeed performs very well. In practice, we can run TensorMap only when the vehicle is in motion, holding the currently estimated value when the vehicle is stopped.

We also report the error between the original segment tensor and the orthogonal



(a) (b)

Figure 6.2: Effect of two types of noise on accuracy. (a) Effect of zero-mean Gaussian noise of variance σ^2, added to each point, and (b) effect of translations (in meters) to the right (simulated).

Tucker3 model learnt in Fig. 6.1(c) and (f), replicated to improve readability. Further, Fig. 6.1 panel (g) shows the corresponding confusion matrix for segment classification problem shown in Fig. 6.1(a). Also, the topological map learnt is shown in panel (h). Notice that the nodes of this topological map are not spaced uniformly, this is due to the movement of the vehicle.

6.4.4 Effect of Gaussian noise and Translations

We now study the effect of Gaussian noise and translations on the performance of TensorMap. Here, we generate the noisy tensor by adding zero-mean Gaussian random noise of variance σ^2 to each coordinate of the Lidar scan, and process these noisy Lidar scans using the procedure described in Section 6.3.1.

Fig. 6.2 (a) shows the effect of adding zero-mean Gaussian random noise of variance σ^2 to each coordinate of the returns (point cloud) on accuracy. We notice that although the technique seems to be robust to lower levels of noise, the performance degrades with increasing σ. This is because the "signatures" are heavily dependent on the relative position of objects in the environment. This is somewhat reassuring, it points to the fact that TensorMap is basing its decision on the relative placement of features, leveraged at the classification stage.

Next we study the effect of a second, perhaps more challenging type of noise: translations. Fig. 6.2 (b) shows the effect of successively shifting the test sequence to the right on the accuracy (%). We notice that the technique is successful up-to a translation of about 1m, beyond which, the performance quickly degrades. Note that a similar effect can be observed for translations to the left. The translations we consider here are *artificially* generated, in practice the effect of translation may be worse. This is because, the Lidar "sees" additional objects in the direction of translation; posing a potential challenge for our approach.

6.4.5 Compression Ratio

Finally, we analyze the compression ratio of the proposed technique in terms of number of elements to be stored. For the given choice of parameters we achieve the ratio of TensorMap : Tensor representation : Lidar Scan representation of about 1 : 400 : 8300. This significant improvement in terms of memory requirement enables use of TensorMap in real-world applications.

6.4.6 Other Applications and Future Work

Applications of TensorMap also include secure and efficient location communication by transmission of the "signatures" (which in the current case are just 5×5 matrices), these "signatures" can be viewed as encoded location information. These can be directly understood by the sender and receiver(s), who have access to the *a priori* known topological map. Further, as alluded to in Section 6.2, TensorMap can be used for coarse localization before scan-matching thus reducing the associated computational and storage overhead, potentially making scan-matching viable for real-time localization. Further, TensorMap can also be used to detect false loop-closures while scan-matching.

Future work includes fusing data from other sensors to improve the robustness of TensorMap in order to develop techniques for localization, and comparison of such a technique with related works. Also, as alluded to in this discussion, using unequal segment lengths, instead of the fixed ones considered here, remains a potential direction.

6.5 Conclusions

Lidar scan-matching provides the most accurate information about the position of the autonomous vehicle, yet it is computationally expensive, prohibiting its use in real-time localization. Popular techniques reduce the rate of data acquisition to cope with this overhead. In this work, we present a technique based on tensor decompositions for building efficient (in terms of space complexity) graph representations of maps. Our preliminary investigation of the proposed technique via experimental evaluations on real-world Lidar data for a localization task shows promising results, and opens exciting avenues for future explorations, in order to make autonomous vehicle navigation safer and more stable.

Part IV

Tools

Chapter 7

Software Resources

7.1 Reproducible Research

In spirit of reproducible research, we fix the random seed for our experiments (when applicable). In addition, we have released the code on `GitHub` for evaluation and future explorations.

7.2 Software Packages Developed

We now provide a brief overview of the packages developed along with links to the code repositories.

7.2.1 NOODL: Neurally plausible alternating Optimization-based Online Dictionary Learning

The code corresponding to our dictionary learning algorithm, described in Chapter 2, is made available at `https://github.com/srambhatla/NOODL`. The code-base is implemented in `MATLAB` and `Python`.

The implementation covers both the vanilla and distributed versions of the algorithm. In the distributed version, we employ `MATLAB`'s `spmd` to distribute the processing of the data samples across workers. This is especially useful for processing large datasets.

We also provide the code corresponding to our comparative experiments. This can be used to reproduce the results shown in Chapter 2. For these, we also provide implementation of FISTA Beck and Teboulle (2009) and stochastic ISTA (without the acceleration). These are used for the sparse approximation step by competing techniques.

In addition, by leveraging our neural architecture, we also provide a completely parallelized implementation of `NOODL` via `TensorFlow`. This implementation showcases how our algorithm can be used where high throughput is especially important.

7.2.2 TensorNOODL: NOODL for Structured Tensor Decomposition

The code corresponding to the structured tensor decomposition task (presented in Chapter 3) is made available at https://github.com/srambhatla/TensorNOODL. The implementation is in MATLAB, the code scales according to the number of workers made available. We also provide recommendations on the step-size (for the dictionary update step) for different dictionary sizes. The implementation relies on `MATLAB`'s `spmd` command to process the samples.

7.2.3 D-RPCA: Dictionary-based Robust PCA

This package contains the code corresponding to the phase transition plots and the target localization task in `MATLAB`. Details of specific functions and their use is made available at https://github.com/srambhatla/Dictionary-based-Robust-PCA.

For the target localization task, the package contains option to use a dictionary containing a few signatures corresponding to the target object or to learn a specified number of signatures from the hyperspectral images using a dictionary learning algorithm.

7.2.4 TensorMap: Lidar-Based Topological Mapping and Localization via Tensor Decompositions

We provide the code to build TensorMaps at https://github.com/srambhatla/TensorMap. This implementation includes code to 1) tensorize Lidar point-clouds, 2) learn TensorMap using Tucker3 decomposition, and 3) localize (using a Lidar scan) in a given TensorMap.

Chapter 8

Discussion and Future Work

As learning algorithms continue to revolutionize various areas, the question of their correctness and reliability will become central for ensuring their effectiveness. Notwithstanding the success of black-box solutions, questions such as: what and how these algorithms learn and how they make decisions, impede their deployment in critical application areas. To address this need for *Safe AI*, we considered different learning problems and establish guarantees on their performance. Our explorations in each aspect of the learning problem ecosystem (Fig. 1.2) advocate for principled learning algorithm design, and pave way for use of these techniques in critical applications like healthcare, navigation, legal, finance, etc.

8.1 Discussion

We gain following insights from our analysis and exploration of each aspect of the learning problem ecosystem.
Provable Dictionary Learning and Tensor Factorization – Through our work on the dictionary learning problem (Chapter 2), where alternating minimization-based heuristics work so well in practice that the problem is widely viewed as being "solved", we exposed the current gaps in the theoretical analysis which only focused on dictionary recovery, and developed a provable algorithm for exact recovery of both factors (with appropriate initialization). Our work here showcased the virtues of considering a joint optimization problem in case of multiple unknowns. This view of the problem led to

a real-world-ready guaranteed algorithm, also explaining the success of popular alternating minimization-based heuristics.

Leveraging these dictionary learning results we developed an algorithm for the recovering the Canonical Polyadic (CP) factors of a structured tensor (Chapter 3), wherein two of the factors are sparse and the third obeys some incoherence conditions. Our algorithm, to the best of our knowledge, is the first algorithm for recovery of the CP factors of such a structured tensor (up to scalings, permutations and sign-flips), and finds applications in a number of data analytics tasks; see Chapter 3 for details.

Generalization of Robust PCA – Our work on developing a dictionary-based generalization of robust PCA (Chapter 4 and Chapter 5) demonstrates how to leverage prior knowledge in learning tasks. Our model is especially useful in applications where the aim is to localize a particular target of interest, without having to learn the specifics of the rest of the data. We leverage our our algorithm-agnostic theoretical results and consider the task of identifying targets in an hyperspectral image (Chapter 5), where we show the advantages of using our methods, while outperforming related techniques.

Tensor Decompositions for Lidar-based Navigation – Finally, our application-focused efforts demonstrate the use of tensor decomposition techniques for efficiently learning a topological map for navigation using Lidar data (Chapter 6). Our technique is especially useful in overcoming the challenge of navigating in feature-scarce environment while being accurate and efficient (in terms of storage requirements).

8.2 Future Work

Our efforts in area of provable algorithms have led to some interesting observations and questions, which pave way, and constitute promising future directions.

Explaining Success of Random Initializations – Our current theoretical results for NOODL (Chapter 2) rely on an appropriate initialization to guarantee linear convergence of estimates to the ground-truth factors. However, our experimental evaluations indicate that NOODL also works with random initializations albeit does not have linear convergent behavior until it meets the conditions of our analysis. It is surprising that NOODL should recover the ground-truth factors in this case, and we still don't know why. Theoretically, the inherent non-convexity of the problem means that we

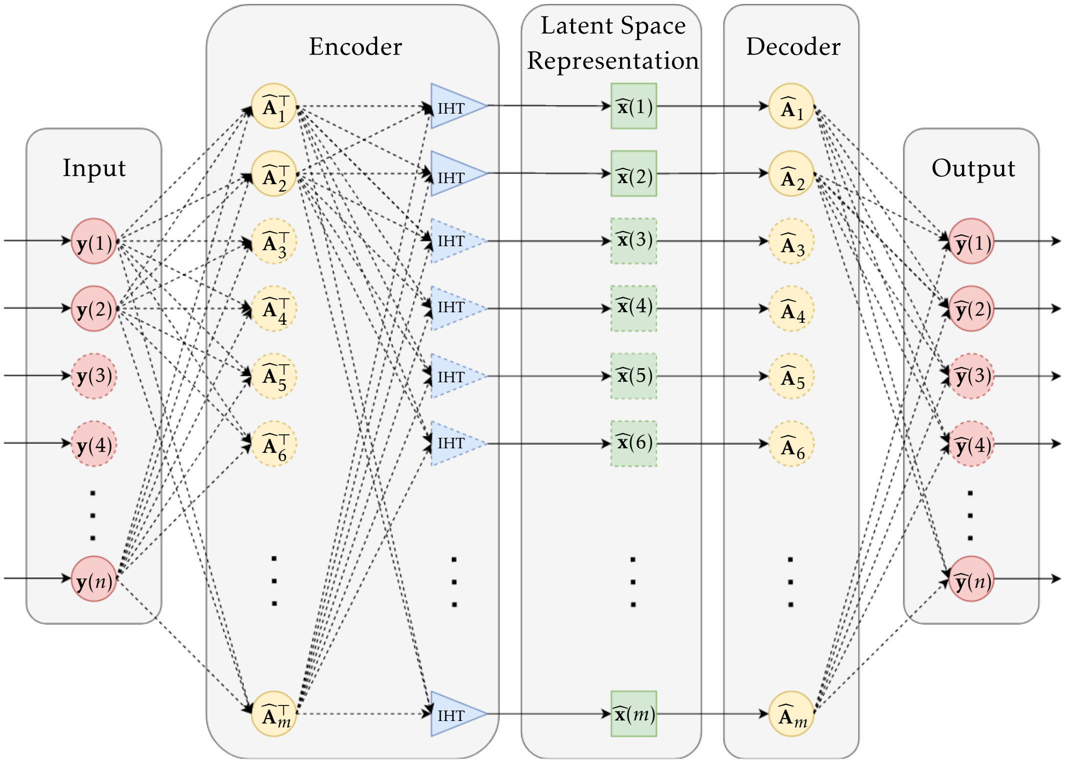

Input
$\mathbf{y}(1)$
$\mathbf{y}(2)$
$\mathbf{y}(3)$
$\mathbf{y}(4)$
$\mathbf{y}(n)$
Encoder
$\widehat{\mathbf{A}}_1^\top$
$\widehat{\mathbf{A}}_2^\top$
$\widehat{\mathbf{A}}_3^\top$
$\widehat{\mathbf{A}}_4^\top$
$\widehat{\mathbf{A}}_5^\top$
$\widehat{\mathbf{A}}_6^\top$
$\widehat{\mathbf{A}}_m^\top$
IHT
Latent Space Representation
$\widehat{\mathbf{x}}(1)$
$\widehat{\mathbf{x}}(2)$
$\widehat{\mathbf{x}}(3)$
$\widehat{\mathbf{x}}(4)$
$\widehat{\mathbf{x}}(5)$
$\widehat{\mathbf{x}}(6)$
$\widehat{\mathbf{x}}(m)$
Decoder
$\widehat{\mathbf{A}}_1$
$\widehat{\mathbf{A}}_2$
$\widehat{\mathbf{A}}_3$
$\widehat{\mathbf{A}}_4$
$\widehat{\mathbf{A}}_5$
$\widehat{\mathbf{A}}_6$
$\widehat{\mathbf{A}}_m$
Output
$\widehat{\mathbf{y}}(1)$
$\widehat{\mathbf{y}}(2)$
$\widehat{\mathbf{y}}(3)$
$\widehat{\mathbf{y}}(4)$
$\widehat{\mathbf{y}}(n)$

Figure 8.1: Future Work: Developing analysis for a one layer Sparse Autoencoder. Figure shows how each sample of the data is processed by first forming a latent space (sparse) representation. Next, the discrepancy between the input $\mathbf{y}$ and the output $\widehat{\mathbf{y}}$ is used to update the weights (dictionary elements).

can only guarantee convergence to a local stationary point. It will be interesting to devise new analysis of NOODL under random initializations. Such an analysis can also be useful to analyze contemporary deep learning architectures.

Using Dependent Samples for Learning – Both NOODL (Chapter 2) and TensorNOODL (Chapter 3) require us to use independent (fresh) samples, which is primarily due to the concentration results we employ. Although in practice both algorithms can be used in batch (offline) settings, it will be interesting to translate the results of online setting to the case of dependent samples. Any work on using dependence will be important in extending these advances to the current exploration of deep learning structures.

Analysis for Autoencoders – NOODL's (Chapter 2) learning procedure, i.e. alternating between an encoding and a decoding step, is analogous to that of a one layer (one encoder and one decoder) sparse autoencoder; see Fig. 8.1 for the proposed architecture. As a result, it can be used to devise analysis for such deep learning structures, and also develop new encoding stratergise for training autoencoders.

Domain Adaptation and Transfer Learning Algorithms – Another promising direction is to bake existing knowledge about the data into NOODL (Chapter 2) . This will help us to leverage side information instead of starting from scratch. Our current efforts for this semi-supervised version of NOODL are aimed to develop an initialization algorithm.

References

Agarwal, A., Anandkumar, A., Jain, P., Netrapalli, P. and Tandon, R. (2014). Learning sparsely used overcomplete dictionaries. In *COLT*.

Aharon, M., Elad, M. and Bruckstein, A. (2005). K-SVD: Design of Dictionaries for Sparse Representation. *In Proceedings of SPARS* 9–12.

Aharon, M., Elad, M. and Bruckstein, A. (2006). k-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, **54** 4311–4322.

Allen, G. (2012). Sparse higher-order principal components analysis. In *Artificial Intelligence and Statistics*.

Anandkumar, A., Ge, R., Hsu, D., Kakade, S. M. and Telgarsky, M. (2014). Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, **15** 2773–2832.

Anandkumar, A., Ge, R. and Janzamin, M. (2015). Learning overcomplete latent variable models through tensor methods. In *Conference on Learning Theory*.

Anandkumar, A., Jain, P., Shi, Y. and Niranjan, U. N. (2016). Tensor vs. matrix methods: Robust tensor decomposition under block sparse perturbations. In *Artificial Intelligence and Statistics*.

Angeli, A., Doncieux, S., Meyer, J. A. and Filliat, D. (2009). Visual topological slam and global localization. In *2009 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Arora, S., Ge, R., Ma, T. and Moitra, A. (2015). Simple, efficient, and neural algorithms for sparse coding. In *COLT*.

Arora, S., Ge, R. and Moitra, A. (2014). New algorithms for learning incoherent and overcomplete dictionaries. In *COLT*.

Barak, B., Kelner, J. A. and Steurer, D. (2015). Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. ACM.

Barthélemy, Q., Gouy-Pailler, C., Isaac, Y., Souloumiac, A., Larue, A. and Mars, J. I. (2013). Multivariate temporal dictionary learning for eeg. *Journal of neuroscience methods*, **215** 19–28.

Baumgardner, M. F., Biehl, L. L. and Landgrebe, D. A. (2015). 220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3, dataset available via http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes.
https://purr.purdue.edu/publications/1947/1

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, **2** 183–202.

Becker, H., Albera, L., Comon, P., Gribonval, R., Wendling, F. and Merlet, I. (2015). Brain-source imaging: From sparse to tensor models. *IEEE Signal Processing Magazine*, **32** 100–112.

Besl, P. J. and McKay, N. D. (1992). Method for registration of 3-d shapes. In *Robotics-DL tentative*. International Society for Optics and Photonics.

Biber, P. and Strasser, W. (2003). The normal distributions transform: A new approach to laser scan matching. In *2003 IEEE International Conference on Intelligent Robots and Systems (IROS)*, vol. 3. IEEE.

Blumensath, T. and Davies, M. E. (2009). Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, **27** 265–274.

Bobin, J., Moudden, Y., Starck, J. L. and Fadili, J. (2009). Sparsity constraints for hyperspectral data analysis: Linear mixture model and beyond.
http://dx.doi.org/10.1117/12.826131

Booij, O., Terwijn, B., Zivkovic, Z. and Kröse, B. (2007). Navigation using an appearance based topological map. In *2007 IEEE International Conference on Robotics and Automation*. IEEE.

Borengasser, M., Hungate, W. S. and Watkins, R. (2007). *Hyperspectral remote sensing: principles and applications*. CRC press.

Candes, E. and Romberg, J. (2007). Sparsity and incoherence in compressive sampling. *Inverse problems*, **23** 969.

Candès, E. J., Li, X., Ma, Y. and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM (JACM)*, **58** 11.

Candès, E. J., Li, X. and Soltanolkotabi, M. (2015). Phase retrieval via wirtinger flow: Theory and algorithms. *IEEE Transactions on Information Theory*, **61** 1985–2007.

Candès, E. J. and Tao, T. (2005). Decoding by linear programming. *IEEE Transactions on Information Theory*, **51** 4203–4215.

Chambolle, A., Vore, R. A. D., Lee, N. Y. and Lucier, B. J. (1998). Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage. *IEEE Transactions on Image Processing*, **7** 319–335.

Chandrasekaran, V., Sanghavi, S., Parrilo, P. A. and Willsky, A. S. (2011). Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, **21** 572–596.

Chang, C. K., Siagian, C. and Itti, L. (2010). Mobile robot vision navigation & localization using gist and saliency. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Charles, A. S., Olshausen, B. A. and Rozell, C. J. (2011). Learning sparse codes for hyperspectral imagery. *IEEE Journal of Selected Topics in Signal Processing*, **5** 963–978.

Chatterji, N. and Bartlett, P. L. (2017). Alternating minimization for dictionary learning with random initialization. In *Advances in Neural Information Processing Systems*.

Chen, M., Ganesh, A., Lin, Z., Ma, Y., Wright, J. and Wu, L. (2009). Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. *Coordinated Science Laboratory Report no. UILU-ENG-09-2214*.

Chen, S. S., Donoho, D. L. and Saunders, M. A. (1998). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, **20** 33–61. https://doi.org/10.1137/S1064827596304010

Chen, Y., Jalali, A., Sanghavi, S. and Caramanis, C. (2013). Low-rank matrix recovery from errors and erasures. *IEEE Transactions on Information Theory*, **59** 4324–4337.

Chen, Y. and Wainwright, M. (2015a). Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*.

Chen, Y. and Wainwright, M. J. (2015b). Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *CoRR*, **abs/1509.03025**.

Comon, P. (1994). Independent component analysis, a new concept? *Signal processing*, **36** 287–314.

Daubechies, I., Defrise, M. and Mol, C. D. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, **57** 1413–1457.

Deburchgraeve, W., Cherian, P. J., Vos, M. D., Swarte, R. M., Blok, J. H., Visser, G. H., Govaert, P. and Huffel, S. V. (2009). Neonatal seizure localization using parafac decomposition. *Clinical Neurophysiology*, **120** 1787–1796.

Ding, X., He, L. and Carin, L. (2011). Bayesian robust principal component analysis. *IEEE Transactions on Image Processing*, **20** 3419–3430.

Donoho, D., Elad, M. and Temlyakov, V. N. (2006). Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, **52** 6–18.

Donoho, D. L. and Huo, X. (2001a). Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, **47** 2845–2862.

Donoho, D. L. and Huo, X. (2001b). Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, **47** 2845–2862.

Duffin, R. J. and Schaeffer, A. C. (1952). A class of nonharmonic fourier series. *Transactions of the American Mathematical Society*, **72** 341–366.

Elad, M. (2010). *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. 1st ed. Springer Publishing Company, Incorporated.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, **15** 3736–3745.

Engan, K., Aase, S. O. and Husoy, J. H. (1999). Method of optimal directions for frame design. In *In Proceedings of 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing.*, vol. 5. IEEE.

Fraundorfer, F., Engels, C. and Nistér, D. (2007). Topological mapping, localization and navigation using image collections. In *2007 IEEE/RSJ International Conference on intelligent Robots and Systems (IROS)*. IEEE.

Fuller, W. A. (2009). *Measurement error models*, vol. 305. John Wiley & Sons.

Gamba, P. (2002). Pavia centre and university dataset. http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes#Pavia_Centre_and_University.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A. and Brendel, W. (2019). Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*.
https://openreview.net/forum?id=Bygh9j09KX

Geng, Q. and Wright, J. (2014). On the local correctness of ℓ_1-minimization for dictionary learning. In *2014 IEEE International Symposium on Information Theory (ISIT),*. IEEE.

Gershgorin, S. A. (1931). Uber die abgrenzung der eigenwerte einer matrix 749–754.

Giampouras, P. V., Themelis, K. E., Rontogiannis, A. A. and Koutroumbas, K. D. (2016). Simultaneously sparse and low-rank abundance matrix estimation for hyperspectral image unmixing. *IEEE Transactions on Geoscience and Remote Sensing*, **54** 4775–4789.

Golbabaee, M., Arberet, S. and Vandergheynst, P. (2010). Distributed compressed sensing of hyperspectral images via blind source separation. In *Forty Fourth Asilomar Conference on Signals, Systems and Computers*.

Greer, J. B. (2012). Sparse demixing of hyperspectral images. *IEEE Transactions on Image Processing*, **21** 219–228.

Gregor, K. and LeCun, Y. (2010). Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*. Omnipress.

Gribonval, R. and Schnass, K. (2010). Dictionary identification and sparse matrix-factorization via ℓ_1 -minimization. *IEEE Transactions on Information Theory*, **56** 3523–3539.

Grundlehner, B., Brown, L., Penders, J. and Gyselinckx, B. (2009). The design and analysis of a real-time, continuous arousal monitor. In *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*. IEEE.

Hanson, D. and Wright, F. T. (1971). A bound on tail probabilities for quadratic forms in independent random variables. *The Annals of Mathematical Statistics*, **42** 1079–1083.

Harel, J., Koch, C. and Perona, P. (2006). Graph-based visual saliency. In *Advances in Neural Information Processing Systems*.

Haupt, J. and Nowak, R. (2006). Signal reconstruction from noisy random projections. *IEEE Transactions on Information Theory*, **52** 4036–4048.

Heil, C. (2013). What is ... a frame? *Notices of the American Mathematical Society*, **60**.

Hillar, C. J. and Lim, L. H. (2013). Most tensor problems are np-hard. *Journal of the ACM (JACM)*, **60** 45.

Håstad, J. (1990). Tensor rank is np-complete. *Journal of Algorithms*, **11** 644 – 654. http://www.sciencedirect.com/science/article/pii/0196677490900146

Huang, F. and Anandkumar, A. (2015). Convolutional dictionary learning through tensor factorization. In *Feature Extraction: Modern Questions and Challenges*.

Huang, K., Sidiropoulos, N. D. and Liavas, A. P. (2016). A flexible and efficient algorithmic framework for constrained matrix and tensor factorization. *IEEE Transactions on Signal Processing*, **64** 5052–5065.

Huang, P. S., Chen, S. D., Smaragdis, P. and Hasegawa, M. J. (2012). Singing-voice separation from monaural recordings using robust principal component analysis. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),*. IEEE.

Huang, Y., Paisley, J., Lin, Q., Ding, X., Fu, X. and Zhang, X. P. (2014). Bayesian nonparametric dictionary learning for compressed sensing mri. *IEEE Transactions on Image Processing*, **23** 5007–5019.

Itti, L., Koch, C. and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, **20** 1254–1259.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An Introduction to Statistical Learning*, vol. 112. Springer.

Jenatton, R., Gribonval, R. and Bach, F. (2012). Local stability and robustness of sparse dictionary learning in the presence of noise. Research report. https://hal.inria.fr/hal-00737152

Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

Jung, A., Eldar, Y. and Görtz, N. (2014). Performance limits of dictionary learning for sparse coding. In *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*. IEEE.

Jung, A., Eldar, Y. C. and Grtz, N. (2016). On the minimax risk of dictionary learning. *IEEE Transactions on Information Theory*, **62** 1501–1515.

Karimi, H., Nutini, J. and Schmidt, M. (2016). Linear convergence of gradient and proximal-gradient methods under the Polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, Cham.

Kawakami, R., Matsushita, Y., Wright, J., Ben-Ezra, M., Tai, Y. W. and Ikeuchi, K. (2011). High-resolution hyperspectral imaging via matrix factorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Keshava, N. and Mustard, J. F. (2002). Spectral unmixing. *IEEE Signal Processing Magazine*, **19** 44–57.

Kolda, T. G. and Bader, B. (2009). Tensor decompositions and applications. *SIAM review*, **51** 455–500.

Kolda, T. G. and Mayo, J. R. (2011). Shifted power method for computing tensor eigenpairs. *SIAM Journal on Matrix Analysis and Applications*, **32** 1095–1124.

Kreutz-Delgado, K., Murray, J. F., Rao, B. D., Engan, K., Lee, T. and Sejnowski, T. J. (2003). Dictionary learning algorithms for sparse representation. *Neural computation*, **15** 349–396.

Kruskal, J. B. (1977). Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications*, **18** 95–138.

Lakhina, A., Crovella, M. and Diot, C. (2004). Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM Computer Communication Review*, vol. 34. ACM.

Lathauwer, L. D., Moor, B. D. and Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, **21** 1253–1278.

Le, Q. V., Karpenko, A., Ngiam, J. and Ng, A. Y. (2011). Ica with reconstruction cost for efficient overcomplete feature learning. In *Advances in Neural Information Processing Systems*.

Lee, H., Battle, A., Raina, R. and Ng, A. Y. (2007). Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*.

Lee, K., Tak, S. and Ye, J. C. (2010). A data-driven sparse glm for fmri analysis using sparse dictionary learning with mdl criterion. *IEEE Transactions on Medical Imaging*, **30** 1076–1089.

Lewicki, M. S. and Sejnowski, T. J. (2000). Learning overcomplete representations. *Neural Comput.*, **12** 337–365.
http://dx.doi.org/10.1162/089976600300015826

Li, N., Pfeifer, N. and Liu, C. (2017). Tensor-based sparse representation classification for urban airborne lidar points. *Remote Sensing*, **9**.

Li, X. and Haupt, J. (2015a). Identifying outliers in large matrices via randomized adaptive compressive sampling. *IEEE Transactions on Signal Processing*, **63** 1792–1807.

Li, X. and Haupt, J. (2015b). Locating salient group-structured image features via adaptive compressive sensing. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*.

Li, X. and Haupt, J. (2015c). Outlier identification via randomized adaptive compressive sampling. In *IEEE International Conference on Acoustic, Speech and Signal Processing*.

Li, X. and Haupt, J. (2016). A refined analysis for the sample complexity of adaptive compressive outlier sensing. In *IEEE Workshop on Statistical Signal Processing*.

Li, X., Ren, J., Rambhatla, S., Xu, Y. and Haupt, J. (2018a). Robust pca via dictionary based outlier pursuit. In *2018 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*. IEEE.

Li, X., Ren, J., Rambhatla, S., Xu, Y. and Haupt, J. (2018b). Robust pca via dictionary based outlier pursuit. In *2018 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP),*. IEEE.

Li, X., Ren, J., Xu, Y. and Haupt, J. (2016a). An efficient dictionary based robust pca via sketching. *Technical Report*.

Li, X., Wang, Z., Lu, J., Arora, R., Haupt, J., Liu, H. and Zhao, T. (2016b). Symmetry, saddle points, and global geometry of nonconvex matrix factorization. *arXiv preprint arXiv:1612.09296*.

Li, X., Zhao, T., Arora, R., Liu, H. and Haupt, J. (2016c). Stochastic variance reduced optimization for nonconvex sparse learning. In *International Conference on Machine Learning*.

Li, Z., Uschmajew, A. and Zhang, S. (2015). On convergence of the maximum block improvement method. *SIAM Journal on Optimization*, **25** 210–233.

Liu, T., Sun, J., Zheng, N., Tang, X. and Shum, H. (2007). Learning to detect a salient object. In *Proc. CVPR*.

Ma, T., Shi, J. and Steurer, D. (2016). Polynomial-time tensor decompositions with sum-of-squares. In *57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE.

Mailhé, B., Gribonval, R., Bimbot, F., Lemay, M., Vandergheynst, P. and Vesin, J. M. (2009). Dictionary learning for the sparse modelling of atrial fibrillation in ecg signals. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE.

Mairal, J., Bach, F., Ponce, J. and Sapiro, G. (2009). Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*. ACM.

Mairal, J., Bach, F., Ponce, J. and Sapiro, G. (2010). Online Learning for Matrix Factorization and Sparse Coding. *Journal of Machine Learning Research*, **11** 19–60.

Mallat, S. G. and Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, **41** 3397–3415.

Mardani, M., Mateos, G. and Giannakis, G. B. (2013). Recovery of low-rank plus compressed sparse matrices with application to unveiling traffic anomalies. *IEEE Transactions on Information Theory*, **59** 5186–5205.

Martínez-Montes, s. E., Sánchez-Bornot, J. M. and Valdés-Sosa, P. A. (2008). Penalized parafac analysis of spontaneous eeg recordings. *Statistica Sinica* 1449–1464.

McDiarmid, C. (1998). Concentration. In *Probabilistic methods for algorithmic discrete mathematics*. Springer, 195–248.

Mehta, B. and Nejdl, W. (2008). Attack resistant collaborative filtering. In *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*.

Milford, M. J. and Wyeth, G. F. (2012). Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Min, K., Zhang, Z., Wright, J. and Ma, Y. (2010). Decomposing background topics from keywords by principal component pursuit. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. CIKM '10, ACM, New York, NY, USA.
http://doi.acm.org/10.1145/1871437.1871475

Mohlenkamp, M. J. (2013). Musings on multilinear fitting. *Linear Algebra and its Applications*, **438** 834 – 852. Tensors and Multilinear Algebra.

Morris, A., Silver, D., Ferguson, D. and Thayer, S. (2005). Towards topological exploration of abandoned mines. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Moudden, Y., Bobin, J., Starck, J. L. and Fadili, J. M. (2009). Dictionary learning with spatio-spectral sparsity constraints. In *Signal Processing with Adaptive Sparse Structured Representations(SPARS)*.

Mueller, A., Himmelsbach, M., Luettel, T., Hundelshausen, F. V. and Wuensche, H. J. (2011). Gis-based topological robot localization through lidar crossroad detection. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE.

Mueller, F. and Lockerd, A. (2001). Cheese: tracking mouse movement activity on websites, a tool for user modeling. In *CHI'01 extended abstracts on Human factors in computing systems*. ACM.

Myronenko, A. and Song, X. (2010). Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32** 2262–2275.

Natarajan, B. K. (1995). Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, **24** 227–234.

NESTEROV, Y. (1983). A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Soviet Mathematics Doklady 27* 72–376.

NOCEDAL, J. and WRIGHT, S. J. (2006). *Numerical Optimization*. 2nd ed. Springer, New York, NY, USA.

OF TECHNOLOGY, J. P. L. N. A. C. I. (1987). Airborne Visible/Infrared Imaging Spectrometer. Available at http://aviris.jpl.nasa.gov/.

OLSHAUSEN, B. A. and FIELD, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, **37** 3311–3325.

PANDEY, G., MCBRIDE, J. R. and EUSTICE, R. M. (2011). Ford campus vision and lidar data set. *International Journal of Robotics Research*, **30** 1543–1552.

PAPALEXAKIS, E. E., SIDIROPOULOS, N. D. and BRO, R. (2013). From k-means to higher-way co-clustering: Multilinear decomposition with sparse latent factors. *IEEE transactions on signal processing*, **61** 493–506.

PARK, B. and LU, R. (2015). *Hyperspectral imaging technology in food and agriculture*. Springer.

PEARSON, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, **2** 559–572.
https://doi.org/10.1080/14786440109462720

RAHMANI, M. and ATIA, G. (2015). Randomized robust subspace recovery for high dimensional data matrices. *arXiv preprint arXiv:1505.05901*.

RAMBHATLA, S. (2012). Semi-blind source separation via sparse representations and online dictionary learning. Masters Thesis, *University of Minnesota – Twin Cities, Minneapolis, MN.*

RAMBHATLA, S. and HAUPT, J. (2013a). Semi-blind source separation via sparse representations and online dictionary learning. In *Asilomar Conference on Signals, Systems and Computers, 2013*, vol. abs/1212.0451. IEEE.
https://arxiv.org/abs/1212.0451

Rambhatla, S. and Haupt, J. (2013b). Semi-blind source separation via sparse representations and online dictionary learning. In *Signals, Systems and Computers, 2013 Asilomar Conference on*. IEEE.

Rambhatla, S., Li, X. and Haupt, J. (2016a). A dictionary based generalization of robust PCA. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, vol. abs/1902.08171. IEEE.
https://arxiv.org/abs/1902.08171

Rambhatla, S., Li, X. and Haupt, J. (2016b). A dictionary based generalization of robust PCA. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE.

Rambhatla, S., Li, X. and Haupt, J. (2017a). Target-based hyperspectral demixing via generalized robust PCA. In *51st Asilomar Conference on Signals, Systems, and Computers, 2017*, vol. abs/1902.11111.
https://arxiv.org/abs/1902.11111

Rambhatla, S., Li, X. and Haupt, J. (2017b). Target-based hyperspectral demixing via generalized robust PCA. In *51st Asilomar Conference on Signals, Systems, and Computers, ACSSC 2017, Pacific Grove, CA, USA, October 29 - November 1, 2017*.
https://doi.org/10.1109/ACSSC.2017.8335372

Rambhatla, S., Li, X. and Haupt, J. (2019). NOODL: Provable online dictionary learning and sparse coding. In *International Conference on Learning Representations (ICLR)*.
https://openreview.net/forum?id=HJeu43ActQ

Rambhatla, S., Li, X., Ren, J. and Haupt, J. (2018a). A Dictionary-Based Generalization of Robust PCA Part I: Study of Theoretical Properties. **abs/1902.08304**.
https://arxiv.org/abs/1902.08304

Rambhatla, S., Li, X., Ren, J. and Haupt, J. (2018b). A Dictionary-Based Generalization of Robust PCA Part II: Applications to Target Localization in Hyperspectral Imaging. **abs/1902.10238**.
https://arxiv.org/abs/1902.10238

Rambhatla, S., Sidiropoulos, N. and Haupt, J. (2018c). Tensormap: Lidar-based topological mapping and localization via tensor decompositions. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, vol. abs/1902.10226. IEEE. https://arxiv.org/abs/1902.10226

Ramirez, I., Sprechmann, P. and Sapiro, G. (2010). Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Ranzato, M., Boureau, Y. and LeCun, Y. (2008). Sparse feature learning for deep belief networks. In *Advances in Neural Information Processing Systems (NIPS)*. 1185–1192.

Rauhut, H. (2010). Compressive sensing and structured random matrices. *Theoretical foundations and numerical methods for sparse recovery*, **9** 1–92.

Razaviyayn, M., Hong, M. and Luo, Z. Q. (2013). A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, **23** 1126–1153.

Rudelson, M. and Vershynin, R. (2013). Hanson-wright inequality and sub-gaussian concentration. *Electronic Communications in Probability*, **18**.

Schindler, G., Brown, M. and Szeliski, R. (2007). City-scale location recognition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Schramm, T. and Steurer, D. (2017). Fast and robust tensor decomposition with applications to dictionary learning. In *Conference on Learning Theory*.

Sharan, V. and Valiant, G. (2017). Orthogonalized als: A theoretically principled tensor decomposition algorithm for practical use. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17, JMLR.org. http://dl.acm.org/citation.cfm?id=3305890.3306001

Siagian, C. and Itti, L. (2009). Biologically inspired mobile robot vision localization. *IEEE Transactions on Robotics*, **25** 861–873.

Sidiropoulos, N. D. and Bro, R. (2000). On the uniqueness of multilinear decomposition of n-way arrays. *Journal of Chemometrics: A Journal of the Chemometrics Society*, **14** 229–239.

Sidiropoulos, N. D., Lathauwer, L. D., Fu, X., Huang, K., Papalexakis, E. E. and Faloutsos, C. (2017). Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, **65** 3551–3582.

Silveira, F., Eriksson, B., Sheth, A. and Sheppard, A. (2013). Predicting audience responses to movie content from electro-dermal activity signals. In *ACM international joint conference on Pervasive and ubiquitous computing*. ACM.

Spielman, D. A., Wang, H. and Wright, J. (2012). Exact recovery of sparsely-used dictionaries. In *Conference on Learning Theory*.

Sprechmann, P., Bronstein, A. M. and Sapiro, G. (2012). Real-time online singing voice separation from monaural recordings using robust low-rank modeling. In *ISMIR*.

Starck, J. L., Moudden, Y., Bobin, J., Elad, M. and Donoho, D. L. (2005). Morphological component analysis. In *Optics & Photonics 2005*. International Society for Optics and Photonics.

Sun, R. and Luo, Z. Q. (2016). Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, **62** 6535–6579.

Sun, W. W., Lu, J., Liu, H. and Cheng, G. (2017). Provable sparse tensor decomposition. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **79** 899–916.

Tang, G. and Shah, P. (2015). Guaranteed tensor decomposition: A moment approach. In *International Conference on Machine Learning*.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288.

Toh, K. C. and Yun, S. (2010). An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of optimization*, **6** 15.

Tosic, I., Jovanovic, I., Frossard, P., Vetterli, M. and Duric, N. (2010). Ultrasound tomography with learned dictionaries. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. CONF.

Tropp, J. (2015). An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, **8** 1–230.

Tu, S., Boczar, R., Soltanolkotabi, M. and Recht, B. (2015). Low-rank solutions of linear matrix equations via procrustes flow. *arXiv preprint arXiv:1507.03566*.

Tucker, L. R. and Ledyard, R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, **31** 279–311.

Uschmajew, A. (2012). Local convergence of the alternating least squares algorithm for canonical tensor approximation. *SIAM Journal on Matrix Analysis and Applications*, **33** 639–652.

Wainwright, M. J. (2009). Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (Lasso). *IEEE Transactions on Information Theory*, **55** 2183–2202.

Wang, J., Zha, H. and Cipolla, R. (2006). Coarse-to-fine vision-based localization by indexing scale-invariant features. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, **36** 413–422.

Watson, G. (1992). Characterization of the subdifferential of some matrix norms. *Linear Algebra and its Applications*, **170** 33 – 45.
http://www.sciencedirect.com/science/article/pii/0024379592904072

Winkler, J. K., Fink, C., Toberer, F., Enk, A., Deinlein, T., Hofmann-Wellenhof, R., Thomas, L., Lallas, A., Blum, A., Stolz, W. and Haenssle, H. A. (2019). Association Between Surgical Skin Markings in Dermoscopic Images and Diagnostic Performance of a Deep Learning Convolutional Neural Network for Melanoma Recognition. *JAMA Dermatology*.
https://doi.org/10.1001/jamadermatol.2019.1735

Wright, J., Ganesh, A., Min, K. and Ma, Y. (2013). Compressive principal component pursuit. *Information and Inference*, **2** 32–68.

Xing, Z., Zhou, M., Castrodad, A., Sapiro, G. and Carin, L. (2012). Dictionary learning for noisy and incomplete hyperspectral images. *SIAM Journal on Imaging Sciences*, **5** 33–56.
http://dx.doi.org/10.1137/110837486

Xu, H., Caramanis, C. and Sanghavi, S. (2010). Robust pca via outlier pursuit. In *Neural Information Processing Systems*.

Yu, Y., Wang, T. and Samworth, R. J. (2014). A useful variant of the davis–kahan theorem for statisticians. *Biometrika*, **102** 315–323.

Yuan, X., Li, P. and Zhang, T. (2016). Exact recovery of hard thresholding pursuit. In *Advances in Neural Information Processing Systems*.

Zhang, J. and Singh, S. (2014). Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems Conference (RSS)*.

Zhang, J. and Singh, S. (2015). Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE.

Zhang, T. and Golub, G. (2001). Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, **23** 534–550.

Zhou, Z., Li, X., Wright, J., Candès, E. J. and Ma, Y. (2010). Stable principal component pursuit. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*. IEEE.

Appendix A

Acronyms

Table A.1: Acronyms

Acronym	Meaning
ICA	Independent Component Analysis
LASSO	Least Absolute Shrinkage and Selection Operator
MCA	Morphological Component Analysis
NNMF	Non-Negative Matrix Factorization
HS	Hyper-Spectral
Lidar	Light Detection and Ranging
MF	Matched Filtering
OP	Outlier Pursuit
PCA	Principal Component Analysis
RLC	Resistance-Inductance-Capacitance
RPCA	Robust Principal Component Analysis
NOODL	Neurally plausible alternating Optimization-based Online Dictionary Learning
TensorNOODL	Neurally plausible alternating Optimization-based Online Dictionary Learning for structured Tensor factorization
TensorMap	An algorithm for building Lidar-based Topological Maps via Tensor Decompositions and Localizing in them
D-RPCA	Dictionary-based Robust Principal Component Analysis

Continued on next page

Table A.1 – continued from previous page

Acronym	Meaning
D-RPCA(E)	Dictionary-based Robust Principal Component Analysis with Entry-wise sparsity
D-RPCA(C)	Dictionary-based Robust Principal Component Analysis with Column-wise sparsity
SBMCA	Semi-Blind Morphological Component Analysis
SVD	Singular Value Decomposition