Provable and efficient algorithms for robust subspace learning and tracking

by

Praneeth Kurpad Narayanamurthy

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Electrical Engineering (Communications and Signal Processing)

Program of Study Committee:
Namrata Vaswani, Major Professor
Chinmay Hegde
Songting Luo
Jin Tian
Zhengdao Wang

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2021

DEDICATION

To my inimitable grandfather, Late. Chintakunta Surya Prakash Rao.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

ACKNOWLEDGMENTS

"If I have seen further it is by standing on the shoulders of Giants"

– Sir Isaac Newton (citation needed)

First and foremost, I wish to express my gratitude to my advisor Prof. Namrata Vaswani for her guidance, support, and inspiration throughout my Ph.D. degree. The countless hours I spent brainstorming ideas and experiments with her has been some of my most memorable times in Ames. Her attention to detail, ability to see proofs several steps ahead, and commitment to research, are a few things I aspire to emulate in my research career. I appreciate the time she spent in molding me into a better researcher and writer. In addition to being a great academic mentor, she has always been kind and patient with me through some very trying times of my life.

I have also greatly enjoyed the interactions with my committee members. I wish to thank Prof. Chinmay Hegde for introducing me to several ideas as part of his Data Science course. I have particularly enjoyed late-night, pre-deadline, and unplanned conversations at the library with him. His enthusiasm for science, and his teaching style is something I will always look up to. I also wish to thank Prof. Songting Luo for his amazing course on Numerical Linear Algebra, and several post-class discussions that have shaped my thesis. Prof. Jin Tian has taught me much of the Machine Learning I know and the material from this course has played a critical role in the way I think. Prof. Zhengdao Wang has been a great mentor and I have learnt much (many times, in hindsight!) from him through his incisive questions at seminars, and his excellent course on Deep Learning.

I wish to thank Prof. Nicola Elia, Prof. Oliver Eulenstein, Prof. Leslie Hogben, and Prof. Eric Weber for patiently answering all my questions in class. I also wish to thank Prof. Aditya

Ramamoorthy for his course on Random Processes. It was undoubtedly the best course I took at Iowa State University.

A huge thanks to Ardhendu, Han, Hooshang, Li, Mohammadreza, Sara, and Songtao – all of whom I incessantly bombarded with questions and seemingly arbitrary discussions but they always provided me with wise, helpful feedback. I have learnt much from you. I would like to thank my other friends at Coover Hall that made the long days at Coover enjoyable: Abhishek, Anindya, Ameya, Amit, Amitanghsu, Ashraf, Gauri, Hooman, Kostas, Koushik, Krishna, Pan, Qi, Vahid, Vahid-Seyyed, Viraj, Rahul, Shubhanwit, Thanh, and Zhengyu. I wish you all the best, and hope the connections remain. A special thanks to Vahid for the soccer and snooker sessions, and the induction into the Aluvial trivia gang.

Outside Coover Hall, I have been fortunate to share several Bridge, and mafia (secret-H?) parties with a large circle of friends, including Amar, Aishwarya, Arpa, Ashirwad, Carolyn, Diskhant, Ganesh, Jyothsna, Niranjana, Payas, Prathamesh, Pratyush, Pratyasha, Raghunandan, Roshni, Shravan, Shrikant, Soori, Souvik, Vignesh, and Vishal. These sessions helped wind-down the incredibly hard graduate life. I would like to especially thank Ganesh for all the (pseudo) jam sessions, impromptu commiseration over food, FIFA nights, and initiating engaging, insightful discussions. I also thank my roommate Payas for his patience during my numerous (long and probably incoherent) rants about life, binge-watching sessions, and, for being a great roommate in general. Finally, I owe a great deal to my friend Gauri for introducing me to Nuit Blanche, for being a constant support through the low times of my life. I will fondly remember the innumerable cups of tea, (daily) HyVee stops, board-games, and "debates" on everthing under the sun.

I wish to thank my friends across the country: Babita, Deepthi, Juju, JK, Koli, Ole, Rakshith, Shreyas, Shruthi, Suchita, Turre, and Viji for being a family away from home. I eagerly looked forward to our trips and you guys never failed to reinvigorate me.

I wish to thank my relatives and cousins for always believing in me. A special thanks to my grandmother, Late. D. Gowramma for her love and affection. Last but not the least, I wish to thank my parents, Gayathri, and Narayana Murthy for their constant love, affection, and support.

This thesis is the culmination of your sacrifices, and I am forever indebted to you. Thank you for all that you continue to do.

If I have missed anyone, I apologize, and thank you for understanding.

ABSTRACT

In the past decades, there has been an explosion in the amount of data that is generated. This calls for development of efficient algorithms to uncover useful information from massive datasets. Although several recent advances in computation allows for faster processing, efficient communication and storage and so on, it is the need of the hour to develop intelligent algorithms that minimize resource utilization, and does so in a near real-time fashion. A commonly observed theme in the Signal Processing and Machine Learning is to exploit the fact that most real-world (extremely high dimensional) data exhibits a simple, succinct, low-dimensional representation. In other words, the data lies close to some low-dimensional structure of the ambient space. In this thesis, we consider two such low-dimensional structures: sparsity and low-rank. Specifically, we develop provable algorithms for the problem of Subspace Tracking (ST) under several constraints. First we study robust ST wherein the data is corrupted by arbitrary outliers. Next, we consider the setting where part of the data is missing (due to issues in transmission or storage). Finally, we develop algorithms that also deal with distributed data.

CHAPTER 1. INTRODUCTION

In the current big-data age, there has been an explosion in the amount of data generated all around us. This can be attributed to the accelerated development of efficient acquisition, transmission, storage and computational modules. Concurrently, the quest of high-dimensional statistical signal processing, and machine learning (ML) researchers has been the development of efficient data-processing algorithms. A commonly observed theme in the aforementioned area is as follows: although the observed data is high-dimensional, typically, most real world data approximately lies in a significantly lower-dimensional ambient space. Motivated by this intuition, the ML community has actively developed provable, and efficient (in terms of sample-complexity, robustness, and computational) algorithms to learn this underlying latent space. However, a key challenge that needs to be addressed is that somewhere in the data-processing pipeline, it is inevitable to avoid corruption of this data, either in terms of missing data, or in terms of outliers that seep in. Another striking feature of traditional datasets is the presence of temporal structure owing to the fact that most data is acquired over time from possibly multi-modal sensors, which can be modeled as time-series data. The overarching goal of this work is to develop provable, robust, and efficient algorithms for low-dimensional structural recovery problem from time-series data.

The problem of *estimating and tracking* a low-dimensional linear subspace from time-series data has garnered significant interest in the signal processing and automatic control communities in the past three decades [2, 18, 13]. However, to the best of our knowledge, all convergence results for this problem were either asymptotic, assumed a *single* underlying subspace, or only provided partial guarantees. My research provides an attempt at resolving this long standing problem, and in addition, we also design and analyze provable, and non-asymptotic algorithms that are also robust.

Model-Based Robust Subspace Tracking. In Chapter 2, we first consider a linear superposition of a low-rank (r-dimensional subspace in n dimensions) and sparse structure for spa-

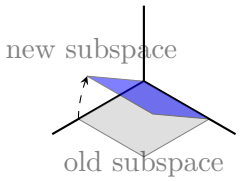


Figure 1.1: Subspace change example in 3D with $r = 2$.

tiotemporal data. In the offline setting, this is commonly referred to as Robust Principal Component Analysis [1] (RPCA). The dynamic version of this problem is referred to as the *Robust Subspace Tracking* (RST) problem [17]. This model is applicable for problems such as Video Layering (separating a video into foreground and background layers), social-network structure identification, and recommendation system design to name a few. In this section, for ease of analysis, we assume that the underlying subspaces can change every so often (in a piecewise constant fashion), but impose a constraint on *how* the changes occur. Formally, we assume that whenever the subspace changes, only 1 out of the r directions changes (see Fig. 1.1 for a simple schematic). We develop an algorithm dubbed simple-Recursive Projected Compressive Sensing (s-ReProCS) based on the ReProCS framework [12] to track the (a) sparse outliers, (b) the *true* low-dimensional data, and (c) the underlying subspace. We show that using a "good enough" initialization, and under standard RPCA/RST assumptions: incoherence of the subspaces, a lower-bound on most outlier magnitudes, mild statistical assumptions on the subspace coefficients, and the subspace change model mentioned above, s-ReProCS is able obtain ε-accurate estimates (of the low-dimensional and sparse vectors, and the underlying subspaces) using just $\mathcal{O}(r \log n \log(1/\varepsilon))$ samples. Additionally, we show that by exploiting the statistical assumptions, we can tolerate a larger fraction of outliers per-row (increase from $\mathcal{O}(1/r)$ to $\mathcal{O}(1)$) in the sparse matrix. Finally, the running time of our algorithm is equal to (upto constant factors) running a rank r-vanilla SVD on the data matrix. The results have been published in IEEE ISIT 2018 [8] and IEEE Transactions on Information Theory [9].

Nearly Optimal Robust Subspace Tracking. A significant drawback of the work described in Chapter 2 was the restrictive subspace change model. In an attempt to relax this assumption,

we propose a modified algorithm referred to as Nearly Optimal RST (NORST). In this work, we assume that the subspace either (a) follows a piecewise constant model but when the subspace does change, it can do so arbitrarily or (b) the subspace is allowed to change at each time, but only by a *little* and have abrupt changes at certain times. Again, we show that under standard RST assumptions, NORST is able to obtain ε-accurate estimates of the underlying subspaces using just $\mathcal{O}(r \log n \log(1/\varepsilon))$ samples. Even with perfect data, estimating a r dimensional subspace in n dimensions requires r samples, and thus our upper bound is only logarithmic factors away from the lower bound. Akin to the results in Chapter 2, our proposed method has an improved outlier tolerance, and the running time (upto constant factors) is the same as running a vanilla SVD on the data matrix.

A critical component of the proof technique involved developing finite sample guarantees for Principal Components Analysis (PCA) in data-dependent noise. Although PCA has been exhaustively studied in the last several decades, most results assume that the noise is uncorrelated (if not independent) with the true data. We consider the setting where the noise can be correlated with the data and provide finite sample guarantees for the SVD solution. In particular, we assume that the noise depends linearly on the data. A key application of the PCA in (sparse) data-dependent noise is in ReProCS based RST. We build upon [14] and provide improved sample-complexity analysis, and a less restrictive data-dependent noise model . These results have been published Allerton 2017 [15] and ISIT 2018 [16].

Using the overall proof and algorithmic skeleton mentioned above, we provide an online algorithm that solves static RPCA in a fast, sample- and memory- efficient manner (published in ICASSP 2018 [6]). A preliminary version of NORST appears in ICML 2018 [7] and the complete paper has been accepted to appear in IEEE Journal of Special Areas in Information Theory [10].

Subspace Tracking from Incomplete Data in Presense of Outliers. In Chapters 2 and 3 we consider the fully observed setting, i.e., we do not account for missing data which may not be practically valid. In Chapter 4, we consider the problem of estimating, and tracking the underlying subspaces when part of the data is missing. The *static* version of this problem is commonly referred

to as *Matrix Completion* (*Robust Matrix Completion* in the presence of outliers). While (Robust) Matrix Completion has been extensively studied in the literature, to the best of our knowledge, there were no finite-sample, *complete* guarantees for the Subspace Tracking with missing entries (STMiss) problem. We show that through a simple modification of our approach for solving RST, the proposed method can also deal with missing data. In particular, we show that under mild and easily interpretable assumptions, the proposed method is fast, sample efficient, and provably correct. Furthermore, while most Matrix Completion methods require that the set of observed entries follow uniform random sampling scheme (i.e., each entry is observed independently of all others with a fixed probability), our algorithm can tolerate deterministic patterns. The tradeoff is that our method requires a larger number of observed entries[1]. The STMiss guarantee has been published in ISIT 2019 [5] and the Robust STMiss problem has been published in ICASSP 2019 [3]. The complete result has been published in IEEE Transactions on Signal Processing [4].

Federated Over-Air Subspace Learning. In Chapters 2 through 4, we implicitly assume that all the data is available at a central node. However, in most practical settings, it is more natural to consider a decentralized setting such that the data is collected in a distributed fashion. Owing to the enormous quantity of acquired data sharing the raw data to a central server is communication-inefficient, but also raises privacy concerns. To alleviate this, in Chapter 5 we analyze the previously discussed RST problem, but in a federated, over-air setting. Federated Learning [19] refers to a paradigm wherein the data is distributed across K peer nodes and the nodes can only share *summary statistics* of their raw data with the central server. For the communication protocol, we consider the newly developed wireless over-air transmission modality that allows for synchronous transmission by the peer nodes as it is K times time- and bandwidth- efficient. However, the central server only receives a sum (superposition) of the individual transmissions and the received sum is corrupted by additive channel noise. We develop an algorithm called Federated Over-Air Robust Subspace Tracking with Missing data (Fed-OA-STMiss) to solve RST while obeying the constraints of federated, over-air communication. In particular, we show that under standard

[1] An equivalent tradeoff can also observed for RPCA wherein, if the support of the sparse matrix is chosen in a probabilistic manner, the tolerable fraction of outliers is larger.

RST assumptions and i.i.d. Gaussian *iteration noise*, with high probability, Fed-OA-RSTMiss computes an ε-accurate subspace estimate (an r dimensional subspace in n dimensions) using just $\mathcal{O}(r \log n \log(1/\varepsilon))$ samples. As in the previous sections, we also show that the running time if equal to (upto constant factors) that of performing a rank-r vanilla SVD on the data matrix. A preliminary version of this work is under review in IEEE Transactions on Signal Processing [11]. We are currently extending this work to also provide a guarantee for differentially private RST in a distributed setting.

Note: Throughout this work, we have tried our best to keep the notation consistent, but each chapter of this work is to be treated independently.

1.1 References

[1] Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *J. ACM 58*, 3 (2011).

[2] Comon, P., and Golub, G. H. Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE 78*, 8 (1990), 1327–1343.

[3] Narayanamurthy, P., Daneshpajooh, V., and Vaswani, N. Provable memory-efficient online robust matrix completion. In *IEEE Int. Conf. Acoust., Speech and Sig. Proc. (ICASSP)* (2019), IEEE, pp. 7918–7922.

[4] Narayanamurthy, P., Daneshpajooh, V., and Vaswani, N. Provable subspace tracking from missing data and matrix completion. *IEEE Transactions on Signal Processing* (2019), 4245–4260.

[5] Narayanamurthy, P., Daneshpajooh, V., and Vaswani, N. Provable subspace tracking with missing entries. In *IEEE Intl. Symp. Info. Th. (ISIT)* (2019).

[6] Narayanamurthy, P., and Vaswani, N. A fast and memory-efficient algorithm for robust pca (merop). In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2018), IEEE, pp. 4684–4688.

[7] Narayanamurthy, P., and Vaswani, N. Nearly optimal robust subspace tracking. In *International Conference on Machine Learning* (2018), pp. 3701–3709.

[8] Narayanamurthy, P., and Vaswani, N. Provable dynamic robust pca or robust subspace tracking. In *2018 IEEE International Symposium on Information Theory (ISIT)* (2018), pp. 376–380.

[9] Narayanamurthy, P., and Vaswani, N. Provable dynamic robust pca or robust subspace tracking. *IEEE Transactions on Information Theory 65*, 3 (2019), 1547–1577.

[10] Narayanamurthy, P., and Vaswani, N. Fast robust subspace tracking via pca in sparse data-dependent noise. *Journal of Selected Areas in Information Theory* (2021).

[11] Narayanamurthy, P., Vaswani, N., and Ramamoorthy, A. Federated over-air subspace tracking from incomplete and corrupted data. *arXiv preprint arXiv:2002.12873 (IEEE Transactions on Signal Processing)* (2020).

[12] Qiu, C., Vaswani, N., Lois, B., and Hogben, L. Recursive robust pca or recursive sparse recovery in large but structured noise. *IEEE Trans. Info. Th.* (August 2014), 5007–5039.

[13] Vaswani, N., Bouwmans, T., Javed, S., and Narayanamurthy, P. Robust subspace learning: Robust pca, robust subspace tracking, and robust subspace recovery. *IEEE signal processing magazine 35*, 4 (2018), 32–55.

[14] Vaswani, N., and Guo, H. Correlated-pca: Principal components' analysis when data and noise are correlated. In *NIPS* (2016).

[15] Vaswani, N., and Narayanamurthy, P. Finite sample guarantees for pca in non-isotropic and data-dependent noise. In *Allerton Conf. on Commun., Control, and Comput.* (2017).

[16] Vaswani, N., and Narayanamurthy, P. Pca in sparse data-dependent noise. In *ISIT* (2018), pp. 641–645.

[17] Vaswani, N., and Narayanamurthy, P. Static and dynamic robust pca and matrix completion: A review. *Proceedings of the IEEE 106*, 8 (2018), 1359–1379.

[18] Yang, B. Asymptotic convergence analysis of the projection approximation subspace tracking algorithms. *Signal Processing 50* (1996), 123–136.

[19] Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST) 10*, 2 (2019), 1–19.

CHAPTER 2. MODEL-BASED ROBUST SUBSPACE TRACKING

Praneeth Narayanamurthy and Namrata Vaswani

Dept. of Electrical and Computer Engineering, Iowa State University, Ames, IA, 50010

Modified from a manuscript published in *IEEE Transactions on Information Theory*

Abstract

Dynamic robust PCA refers to the dynamic (time-varying) extension of robust PCA (RPCA). It assumes that the true (uncorrupted) data lies in a low-dimensional subspace that can change with time, albeit slowly. The goal is to track this changing subspace over time in the presence of sparse outliers. We develop and study a novel algorithm, that we call simple-ReProCS, based on the recently introduced Recursive Projected Compressive Sensing (ReProCS) framework. Our work provides the first guarantee for dynamic RPCA that holds under weakened versions of standard RPCA assumptions, slow subspace change and a lower bound assumption on most outlier magnitudes. Our result is significant because (i) it removes the strong assumptions needed by the two previous complete guarantees for ReProCS-based algorithms; (ii) it shows that it is possible to achieve significantly improved outlier tolerance, compared with all existing RPCA or dynamic RPCA solutions by exploiting the above two simple extra assumptions; and (iii) it proves that simple-ReProCS is online (after initialization), fast, and, has near-optimal memory complexity.

2.1 Introduction

Principal Components Analysis (PCA) is a widely used dimension reduction technique in a variety of scientific applications. Given a set of data vectors, PCA tries to finds a smaller dimensional subspace that best approximates a given dataset. According to its modern definition [5], robust PCA (RPCA) is the problem of decomposing a given data matrix into the sum of a low-rank matrix

(true data) and a sparse matrix (outliers). The column space of the low-rank matrix then gives the desired principal subspace (PCA solution). In recent years, the RPCA problem has been extensively studied, e.g., [5, 6, 14, 22, 20, 34, 33]. A common application of RPCA is in video analytics in separating video into a slow-changing background image sequence (modeled as a low-rank matrix) and a foreground image sequence consisting of moving objects or people (sparse) [5]. Dynamic RPCA refers to the dynamic (time-varying) extension of RPCA [22, 11, 34]. It assumes that the true (uncorrupted) data lies in a low-dimensional subspace that can change with time, albeit slowly. This is a more appropriate model for long data sequences, e.g., surveillance videos. The goal is to track this changing subspace over time in the presence of sparse outliers. Hence this problem can also be referred to as *robust subspace tracking.*

2.1.1 Notation and Problem Setting

Notation. We use bold lower case letters to denote vectors, bold upper case letters to denote matrices, and calligraphic letters to denote sets or events. We use the interval notation $[a, b]$ to mean all of the integers between a and b, inclusive, and $[a, b) := [a, b-1]$. We will often use $\mathcal{J}$ to denote a time interval and $\mathcal{J}^\alpha$ to denote a time interval of length α. We use $\mathbb{1}_S$ to denote the indicator function for statement S, i.e. $\mathbb{1}_S = 1$ if S holds and $\mathbb{1}_S = 0$ otherwise. We use $\|\cdot\|$ without a subscript to denote the l_2 norm of a vector or the induced l_2 norm of a matrix. For other l_p norms, we use $\|\cdot\|_p$. For a set $\mathcal{T}$, we use $\boldsymbol{I}_\mathcal{T}$ to refer to an $n \times |\mathcal{T}|$ matrix of columns of the identity matrix indexed by entries in $\mathcal{T}$. For a matrix $\boldsymbol{A}$, $\boldsymbol{A}'$ denotes its transpose and $\boldsymbol{A}_\mathcal{T} := \boldsymbol{A}\boldsymbol{I}_\mathcal{T}$ is the sub-matrix of $\boldsymbol{A}$ that contains the columns of $\boldsymbol{A}$ indexed by entries in $\mathcal{T}$. Also, we use $\boldsymbol{A}^i$ to denote its i-th row. We use $\lambda_{\min}(.)$ ($\sigma_{\min}(.)$) to denote the minimum eigen (singular) value of a matrix. Similarly for $\lambda_{\max}(.)$ and $\sigma_{\max}(.)$. We use $\delta_s(\boldsymbol{A})$ to denote the s-restricted isometry constant (RIC) [4] of $\boldsymbol{A}$.

A matrix with mutually orthonormal columns is referred to as a *basis* matrix and is used to represent the subspace spanned by its columns. For basis matrices $\hat{\boldsymbol{P}}$, $\boldsymbol{P}$, we use

$$\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) := \|(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{P}\|$$

to quantify the *subspace error (SE)* between their respective column spans. This measures the sine of the maximum principal angle between the subspaces. When $\hat{\boldsymbol{P}}$ and $\boldsymbol{P}$ are of the same size, then SE(.) is symmetric, i.e., $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) = \text{SE}(\boldsymbol{P}, \hat{\boldsymbol{P}})$. We use $\boldsymbol{P}_\perp$ to denote a basis matrix for the orthogonal complement of span($\boldsymbol{P}$).

For a matrix $\boldsymbol{M}$, we use basis($\boldsymbol{M}$) to denote a basis matrix whose columns span the same subspace as the columns of $\boldsymbol{M}$.

The letters c and C denote different numerical constants in each use; c is used for constants less than one and C for those equal to or greater than one.

Dynamic RPCA or Robust Subspace Tracking Problem Statement. At each time t, we observe $\boldsymbol{y}_t \in \mathbb{R}^n$ that satisfies

$$\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{x}_t + \boldsymbol{v}_t, \text{ for } t = 1, 2, \dots, d \tag{2.1}$$

where $\boldsymbol{x}_t$ is the sparse outlier vector, $\boldsymbol{\ell}_t$ is the true data vector that lies in a fixed or slowly changing low-dimensional subspace of $\mathbb{R}^n$, and $\boldsymbol{v}_t$ is small unstructured noise or modeling error. To be precise, $\boldsymbol{\ell}_t = \boldsymbol{P}_{(t)}\boldsymbol{a}_t$ where $\boldsymbol{P}_{(t)}$ is an $n \times r$ *basis matrix* with $r \ll n$ and with $\|(\boldsymbol{I} - \boldsymbol{P}_{(t-1)}\boldsymbol{P}_{(t-1)}{}')\boldsymbol{P}_{(t)}\|$ small compared to $\|\boldsymbol{P}_{(t)}\| = 1$ (slow subspace change). We use $\mathcal{T}_t$ to denote the support set of $\boldsymbol{x}_t$ and we let $s := \max_t |\mathcal{T}_t|$. Given an initial subspace estimate, $\hat{\boldsymbol{P}}_0$, the goal is to track span($\boldsymbol{P}_{(t)}$) within a short delay of each subspace change. The initial estimate can be obtained by applying any static (batch) RPCA technique, e.g., PCP [5] or AltProj [20], to the first t_{train} data frames, $\boldsymbol{Y}_{[1,t_{\text{train}}]}$. A by-product of our solution approach is that the true data vectors $\boldsymbol{\ell}_t$, the sparse outliers $\boldsymbol{x}_t$, and their support sets $\mathcal{T}_t$ can also be tracked on-the-fly. In many practical applications, in fact, $\boldsymbol{x}_t$ or $\mathcal{T}_t$ is often the quantity of interest.

We also assume that (i) $|\mathcal{T}_t|/n$ is upper bounded, (ii) $\mathcal{T}_t$ changes enough over time so that any one index is not part of the outlier support for too long, (iii) the columns of $\boldsymbol{P}_{(t)}$ are dense (non-sparse), and (iv) the subspace coefficients $\boldsymbol{a}_t$ are element-wise bounded, mutually independent, zero mean, have identical and diagonal covariance matrices, and are independent of the outlier supports $\mathcal{T}_t$. We quantify everything in Sec. 2.2.

Subspace Change Assumption. To ensure that the number of unknowns is not too many (see the discussion in Sec. 2.1.3), we will further assume that the subspace $\text{span}(\boldsymbol{P}_{(t)})$ is *piecewise constant* with time, i.e.,

$$\boldsymbol{P}_{(t)} = \boldsymbol{P}_{(t_j)} \quad \text{for all } t \in [t_j, t_{j+1}),\ j = 0, 1, \ldots, J, \tag{2.2}$$

with $t_0 = 1$ and $t_{J+1} = d$. Let $\boldsymbol{P}_j := \boldsymbol{P}_{(t_j)}$. At each change time, t_j, the change is "slow". This means two things:

1. First, at each t_j, only one direction can change with the rest of the subspace remaining fixed, i.e.,

$$\text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j) = \text{SE}(\boldsymbol{P}_{j-1,\text{ch}}, \boldsymbol{P}_{j,\text{rot}}) \tag{2.3}$$

   where $\boldsymbol{P}_{j-1,\text{ch}}$ is *a* direction from $\text{span}(\boldsymbol{P}_{j-1})$ that "changes" at t_j and $\boldsymbol{P}_{j,\text{rot}}$ is its "rotated" version. Thus $\text{span}(\boldsymbol{P}_{j-1}) = \text{span}([\boldsymbol{P}_{j-1,\text{fix}}, \boldsymbol{P}_{j-1,\text{ch}}])$ and $\text{span}(\boldsymbol{P}_j) = \text{span}([\boldsymbol{P}_{j-1,\text{fix}}, \boldsymbol{P}_{j,\text{rot}}])$ where $\boldsymbol{P}_{j-1,\text{fix}}$ is an $n \times (r-1)$ matrix that denotes the part of the subspace that remains "fixed" at t_j.

   Of course at different t_j's, the changing directions could be different.

2. Second, the angle of change is small, i.e., for a $\Delta \ll 1$,

$$\text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j) = \text{SE}(\boldsymbol{P}_{j-1,\text{ch}}, \boldsymbol{P}_{j,\text{rot}}) \leq \Delta. \tag{2.4}$$

Equivalent generative model. With the above model,

$$\boldsymbol{P}_{j,\text{new}} := \frac{(\boldsymbol{I} - \boldsymbol{P}_{j-1,\text{ch}}\boldsymbol{P}_{j-1,\text{ch}}{}')\boldsymbol{P}_{j,\text{rot}}}{\text{SE}(\boldsymbol{P}_{j-1,\text{ch}}, \boldsymbol{P}_{j,\text{rot}})}$$

is the newly added direction at t_j, $\theta_j := \cos^{-1} |\boldsymbol{P}_{j-1,\text{ch}}{}'\boldsymbol{P}_{j,\text{rot}}|$ is the angle by which $\boldsymbol{P}_{j-1,\text{ch}}$ gets rotated out-of-plane (towards $\boldsymbol{P}_{j,\text{new}}$ which lies in $\text{span}(\boldsymbol{P}_{j-1})^\perp$) to get $\boldsymbol{P}_{j,\text{rot}}$. Without loss of generality, assume $0 \leq \theta_j \leq \pi/2$. Thus,

- $|\sin\theta_j| = \sin\theta_j = \text{SE}(\boldsymbol{P}_{j-1,\text{ch}}, \boldsymbol{P}_{j,\text{rot}}) = \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j) \leq \Delta$, and

- $\boldsymbol{P}_{j,\text{del}} := \boldsymbol{P}_{j-1,\text{ch}} \sin\theta_j - \boldsymbol{P}_{j,\text{new}} \cos\theta_j$ is the direction that got deleted at t_j.

We have the following equivalent generative model for getting $\boldsymbol{P}_j$ from $\boldsymbol{P}_{j-1}$: let $\boldsymbol{U}_j$ be an $r \times r$ rotation matrix,

$$\begin{aligned} \boldsymbol{P}_j &= [\underbrace{(\boldsymbol{P}_{j-1}\boldsymbol{U}_j)_{[1,r-1]}}_{\boldsymbol{P}_{j-1,\text{fix}}}, \boldsymbol{P}_{j,\text{rot}}], \text{ where} \\ \boldsymbol{P}_{j,\text{rot}} &:= \underbrace{(\boldsymbol{P}_{j-1}\boldsymbol{U}_j)_r}_{\boldsymbol{P}_{j-1,\text{ch}}} \cos\theta_j + \boldsymbol{P}_{j,\text{new}} \sin\theta_j \end{aligned} \tag{2.5}$$

For a simple example of this in 3D ($n = 3$), see Fig. 2.1.

To make our notation easy to remember, we try to explain its meaning better. Consider the change at t_j. The direction from $\text{span}(\boldsymbol{P}_{j-1})$ that changes is denoted by $\boldsymbol{P}_{j-1,\text{ch}}$. This changes by getting rotated (out-of-plane) by a small angle θ_j towards a new out-of-plane direction $\boldsymbol{P}_{j,\text{new}}$ to get the changed/rotated direction $\boldsymbol{P}_{j,\text{rot}}$. Here "plane" refers to the hyperplane $\text{span}(\boldsymbol{P}_{j-1})$. The basis for the $r-1$-dimensional subspace of $\text{span}(\boldsymbol{P}_{j-1})$ that does not change at t_j is $\boldsymbol{P}_{j-1,\text{fix}}$. So $\boldsymbol{P}_j = [\boldsymbol{P}_{j-1,\text{fix}}, \boldsymbol{P}_{j,\text{rot}}]$.

The span of left singular vectors of $\boldsymbol{L}$ is contained in, or equal to, $\text{span}([\boldsymbol{P}_0, \boldsymbol{P}_{1,\text{new}}, \boldsymbol{P}_{2,\text{new}}, \ldots, \boldsymbol{P}_{J,\text{new}}])$. Equality holds if $\boldsymbol{P}_{j,\text{new}}$ is orthogonal to $\text{span}([\boldsymbol{P}_0, \boldsymbol{P}_{1,\text{new}}, \ldots, \boldsymbol{P}_{j-1,\text{new}}])$ for each j.

In this work we have assumed the simplest possible model on subspace change where, at a change time, only one direction can change. Observe though that, at different change times, the changing direction could be different and hence, over a long period of time, the entire subspace could change. This simple model can be generalized to $r_{\text{ch}} > 1$ directions changing; see the last appendix in the ArXiv posting of this work. It is also possible to study the most general case where $r_{\text{ch}} = r$ and hence no model is assumed for subspace change (only a bound on the maximum principal angle of the change). This requires significant changes to both the algorithm and the guarantee; it is studied in follow-up work [17].

Relation to original RPCA. To connect with the original RPCA problem [5, 14, 20], define the $n \times d$ data matrix $\boldsymbol{Y} := [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots \boldsymbol{y}_d] := \boldsymbol{L} + \boldsymbol{X} + \boldsymbol{V}$ where $\boldsymbol{L}$, $\boldsymbol{X}$, $\boldsymbol{V}$ are simi-

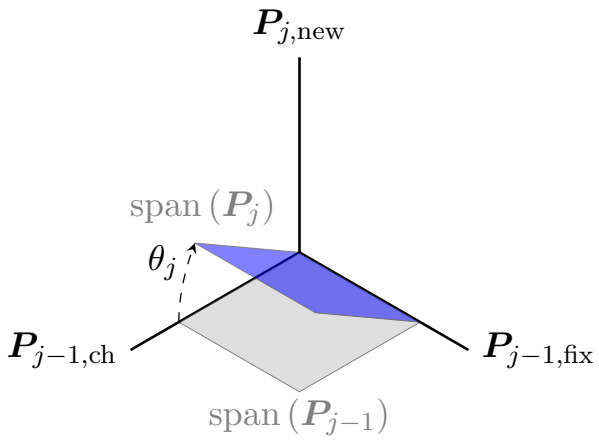


Figure 2.1: Subspace change example in 3D with $r = 2$.

larly defined. Let $r_{_L}$ denote the rank of $\boldsymbol{L}$ and use max-outlier-frac-col and max-outlier-frac-row to denote the maximum fraction of outliers per column and per row of $\boldsymbol{Y}$. RPCA results bound max(max-outlier-frac-row, max-outlier-frac-col). For dynamic RPCA, we will define max-outlier-frac-row slightly differently. It will be the maximum fraction per row of any $n \times \alpha$ sub-matrix of $\boldsymbol{Y}$ with α consecutive columns. Here α denotes the number of frames used in each subspace update. We will denote this by max-outlier-frac-row$^{\alpha}$ to indicate the difference. Since α is large enough (see (2.9)), the two definitions are only a little different. The dynamic RPCA assumption of a bound on $\max_t |\mathcal{T}_t|/n$ is equivalent to bounding max-outlier-frac-col since max-outlier-frac-col $= \max_t |\mathcal{T}_t|/n$. The requirement of $\mathcal{T}_t$'s changing enough is equivalent to a bound on max-outlier-frac-row$^{\alpha}$. As we explain later, the denseness assumption on the $\boldsymbol{P}_{(t)}$'s is similar to the denseness (incoherence) of left singular vectors of $\boldsymbol{L}$ assumed by all standard RPCA solutions, while the assumptions on $\boldsymbol{a}_t$'s replace the right singular vectors' incoherence assumption of standard RPCA.

2.1.2 Related Work and our Contributions

Related Work. We briefly mention all related work here, but provide a detailed discussion later in Sec. 2.3. There is very little work on other solutions for provably correct dynamic RPCA. This includes our early work on a *partial guarantee* (guarantee required assumptions on intermedi-

ate algorithm estimates) [22] and later complete correctness results [16, 34] for more complicated ReProCS-based algorithms. We refer to all of these as "original-ReProCS". It also includes older work on modified-PCP, which is a batch solution for RPCA with partial subspace knowledge, and which can be shown to also provably solve dynamic RPCA in a piecewise batch fashion [35]. The original-ReProCS guarantees require strong assumptions on how the outlier support changes (need a very specific model inspired by a video moving object); their subspace change assumptions are unrealistic; and their subspace tracking delay (equal to the required delay between subspace change times) is very large. On the other hand, the Modified-PCP guarantee [35] requires the outlier support to be uniformly randomly generated (strong assumption; for video, it means that the moving objects need to be single pixel wide and should be jumping around randomly from frame to frame); requires a different stronger assumption on subspace change; and cannot detect subspace change automatically. Other than the above, there is some work on online algorithms for RPCA. The only work that comes with some guarantee, although it is a *partial guarantee*, is an online stochastic optimization based solver for the PCP convex program (ORPCA) [10]. Its guarantee assumed that the basis matrix for the subspace estimate at each t was full rank. To our best knowledge, there is no follow-up work on a complete correctness result for it. There is also much work on empirical online solutions for RPCA, e.g., [12], and older work, e.g., [15, 23]. From a practical standpoint, any online algorithm will implicitly also provide a tracking solution. However, as shown in Sec. 2.6, the solution is not as good as that of ReProCS which explicitly exploits slow subspace change.

The standard RPCA problem has been extensively studied [5, 6, 14, 20, 33, 7]. We discuss these works in detail in Sec. 2.3. A summary is provided in Table 2.1. Briefly, these either need outlier fractions in each row and each column of the observed data matrix to be $O(1/r_L)$ (AltProj [20], GD [33], NO-RMC [7], PCP result of [6, 14], denoted PCP(H)) or need the outlier support to be uniformly randomly generated (PCP result of [5], denoted PCP(C)). Moreover, all these are batch solutions with large memory complexity $O(nd)$.

Contributions. We develop a simple algorithm, termed simple-ReProCS or s-ReProCS, for provably solving the robust subspace tracking or dynamic RPCA problem described earlier. We

Table 2.1: Comparing s-ReProCS with other RPCA solutions with complete guarantees. For simplicity, we ignore all dependence on condition numbers. In this table r_L is the rank of the entire matrix $\boldsymbol{L}$, while r is the maximum rank of any sub-matrices of consecutive columns of $\boldsymbol{L}$ of the form $\boldsymbol{L}_{[t_j,t_{j+1})}$ and thus $r \le r_L$. We show the unrealistic assumptions in red.

Algorithm	Outlier tolerance	Assumptions	Memory, Time,	# params.
PCP (C) [5] (offline)	max-outlier-frac-row $\in \mathcal{O}(1)$ max-outlier-frac-col $\in \mathcal{O}(1)$	outlier support: uniform random $r_L \le c\min(n,d)/\log^2 n$	Memory: $\mathcal{O}(nd)$ Time: $\mathcal{O}(nd^2\frac{1}{\epsilon})$	zero
PCP (H) [14] (offline)	max-outlier-frac-row $\in \mathcal{O}(1/r_L)$ max-outlier-frac-col $\in \mathcal{O}(1/r_L)$		Memory: $\mathcal{O}(nd)$ Time: $\mathcal{O}(nd^2\frac{1}{\epsilon})$	2
AltProj [20], (offline)	max-outlier-frac-row $= O(1/r_L)$ max-outlier-frac-col $\in O(1/r_L)$		Memory: $\mathcal{O}(nd)$ Time: $\mathcal{O}(ndr_L^2\log\frac{1}{\epsilon})$	2
RPCA-GD [33] (offline)	max-outlier-frac-row $\in \mathcal{O}(1/r_L^{1.5})$ max-outlier-frac-col $\in \mathcal{O}(1/r_L^{1.5})$		Memory: $\mathcal{O}(nd)$ Time: $\mathcal{O}(ndr_L\log\frac{1}{\epsilon})$	5
NO-RMC [7] (offline)	max-outlier-frac-row $\in O(1/r_L)$ max-outlier-frac-col $\in \mathcal{O}(1/r_L)$	$Cn \ge d \ge cn$	Memory: $\mathcal{O}(nd)$ Time: $\mathcal{O}(nr_L^3\log^2 n\log^2\frac{1}{\epsilon})$	3
s-ReProCS (online) (this work)	max-outlier-frac-row$^\alpha \in \mathcal{O}(1)$ max-outlier-frac-col $\in \mathcal{O}(1/r)$	**most outlier magnitudes lower bounded slow subspace change first Cr samples: AltProj assumptions**	**Memory:** $\mathcal{O}(nr\log n)$ **Time:** $\mathcal{O}(ndr\log\frac{1}{\epsilon})$ **Detect delay:** $2\alpha = Cr\log n$ **Tracking Delay:** $K\alpha = Cr\log n\log(1/\epsilon)$	**4**

also develop its offline extension that can be directly compared with the standard RPCA results. Simple-ReProCS is based on the ReProCS framework [22]. Our main contribution is the *first correctness guarantee* for dynamic RPCA that *holds under weakened versions of standard RPCA assumptions, slow subspace change, and a lower bound on most outlier magnitudes (this lower bound is proportional to the rate of subspace change).* We say "weakened" because our guarantee implies that, after initialization, s-ReProCS can tolerate an order-wise larger fraction of outliers per row than all existing approaches, without requiring the outlier support to be uniformly randomly generated or without needing any other model on support change. It allows max-outlier-frac-row$^\alpha \in O(1)$ (instead of $O(1/r_L)$). For the video application, this implies that it tolerates slow moving and occasionally static foreground objects much better than other approaches. This fact is also backed up by comparisons on real videos, see Sec. 2.6 and also see [25].

A second key contribution is the algorithm itself. Unlike original-ReProCS [16, 34], s-ReProCS ensures that the estimated subspace dimension is bounded by $(r+1)$ at all times without needing the complicated cluster-EVD step. More importantly, s-ReProCS is provably fast and memory-efficient:

its time complexity is comparable to that of SVD for vanilla PCA, and its memory complexity is near-optimal and equal to $O(nr \log n \log(1/\tilde{\varepsilon}))$ where $\tilde{\varepsilon}$ is the desired subspace recovery accuracy. This is near-optimal because nr is the memory needed to output an r-dimensional subspace estimate in $\mathbb{R}^n$, and the complexity is within log factors of the optimal. To our best knowledge, s-ReProCS is the first provably correct RPCA or dynamic RPCA solution that is *as fast as the best RPCA solution in terms of computational complexity without requiring the data matrix to be nearly square* and *has near-optimal memory complexity.* We provide a tabular comparison of guarantees of offline s-ReProCS with other provable RPCA solutions in Table 2.1. We compare s-ReProCS with other online or tracking solutions for RPCA or dynamic RPCA in Table 2.2 (original-ReProCS, modified-PCP, follow-up work on RePrOCS-NORST [19, 17], ORPCA and GRASTA).

We give a significantly shorter and simpler proof than that for the earlier guarantees for ReProCS-based methods. We do this by first separately *proving a result for the problem of "correlated-PCA" or "PCA in data-dependent noise" [26, 27] with partial subspace knowledge.* This result given in Theorem 2.7 of Sec. 2.5.1 may also be of independent interest.

2.1.3 The need for a piecewise constant model on subspace change

We explain why the piecewise-constant subspace change model is needed. Even if the observed data were perfect (no noise/outlier/missing-data, i.e., we observed $\boldsymbol{\ell}_t$, and all measurements were linearly independent) and the previous subspace were exactly known, in order to obtain a correct r-dimensional estimate[1] for each $\boldsymbol{P}_{(t)}$, one would need at least r samples. Of course, to just find the newly added direction $\boldsymbol{P}_{j,\text{new}}$ and use an $(r+1)$-dimensional estimate, one sample would suffice in this ideal setting (doing this will be especially problematic if the subspace changes at each time because it will mean the estimated subspace dimension will keep growing as $r+t$ at time t). Our actual setting is not this ideal one: we know the previous subspace only up to ϵ error and we observe $\boldsymbol{y}_t$ which is a noisy and outlier-corrupted version of $\boldsymbol{\ell}_t$. This is why, in our setting, more than one data samples are needed even to accurately estimate the newly added direction. Since we get only

[1] requires finding both the newly added direction, $\boldsymbol{P}_{j,\text{new}}$, *and* the deleted direction, $\boldsymbol{P}_{j,\text{del}}$

one observed data vector $\boldsymbol{y}_t$ at each time, the only way to have enough data samples for estimating each subspace is to assume that $\boldsymbol{P}_{(t)}$ is piecewise constant with time, i.e., it satisfies (2.2). In fact, our required lower bound on $t_{j+1} - t_j$ is only a little more than r (see Theorem 2.2), thus making our model a good approximation to slow continuous subspace change.

Furthermore, the following point should be mentioned. In the entire literature on subspace tracking (both with and without outliers, and with and without even missing data), there is no model for subspace change for which there are any provable guarantees. There is no work on provable subspace tracking with outliers (robust subspace tracking) except our own previous work which also used the piecewise constant subspace change model. The subspace tracking (ST) problem (without outliers), and with or without missing data, has been extensively studied [31, 32, 1, 2, 8, 3]; however, all existing guarantees are asymptotic results for the statistically stationary setting of data being generated from a *single unknown* subspace. Moreover, most of these also make assumptions on intermediate algorithm estimates. For a longer discussion of this, please see [28].

2.1.4 Chapter Organization

The proposed algorithm, simple-ReProCS, and its performance guarantees, Theorem 2.2, are given in Sec. 2.2. We discuss the related work in detail in Sec. 2.3 and explain how our guarantee compares with other provable results on RPCA or dynamic RPCA from the literature. Sec. 2.4 provides the main ideas that lead to the proof of Theorem 2.2. We prove Theorem 2.2 under the assumption that the subspace change times are known in Sec. 2.5. This proof helps illustrate all the ideas of the actual proof but with minimal notation. The general proof of Theorem 2.2 is given in Appendix 2.9. Theorem 2.2 relies on a guarantee for PCA in data-dependent noise [26, 27] when partial subspace knowledge is available. This result is proved in Appendix 2.10. We provide detailed empirical evaluation evaluation of simple-ReProCS in Sec. 2.6. We conclude and discuss future directions in Sec. 2.7.

Table 2.2: Comparing s-ReProCS with online or tracking approaches for RPCA. We show the unrealistic assumptions in red. Here, f denotes the condition number of $\boldsymbol{\Lambda}$, r is the maximum dimension of the subspace at any time, and r_L refers to the rank of matrix $\boldsymbol{L}$. Thus $r \leq r_L$. Here, s-ReProCS-no-delete refers to Algorithm 4 without the subpace deletion step.

Algorithm	Outlier tolerance	Assumptions	Memory, Time
orig-ReProCS [34, 16] (online)	max-outlier-frac-row$^{\alpha} \in \mathcal{O}(1/f^2)$ max-outlier-frac-col $\in O(1/r_L)$	outlier support: moving object model, unrealistic subspace change model, changed eigenvalues small for some time, outlier mag. lower bounded, $x_{\min} \geq 14[c\gamma_{\text{new}} + \sqrt{\varepsilon}(\sqrt{r} + \sqrt{c})]$ where, γ_{new} quantifies slow subspace change init data: AltProj assumptions, $d \geq Cr^2/\epsilon^2$	Memory: $O(nr^2/\epsilon^2)$ Time: $\mathcal{O}(ndr \log \frac{1}{\epsilon})$ Detect Delay: $2\alpha = \frac{Cr^2 \log n}{\epsilon^2}$ Tracking Delay: $K\alpha = \frac{Cr^2 \log n \log(1/\epsilon)}{\epsilon^2}$
Modified-PCP [35] (piecewise batch)	max-outlier-frac-row$^{\alpha} \in \mathcal{O}(1)$ max-outlier-frac-col $\in \mathcal{O}(1)$	outlier support: uniform random unrealistic subspace change model $r_L \leq c \min(n,d)/\log^2 n$	Memory: $\mathcal{O}(nr \log^2 n)$ Time: $\mathcal{O}(\frac{ndr \log^2 n}{\epsilon})$ Detect delay: ∞
ORPCA [10]	Has a partial guarantee – assumes algorithm estimates at each time t are full rank		
GRASTA [12]	Has no theoretical guarantees		
s-ReProCS (online) (this work)	max-outlier-frac-row$^{\alpha} \in \boldsymbol{\mathcal{O}(1/f^2)}$ max-outlier-frac-col $\in \boldsymbol{\mathcal{O}(1/r)}$	**most outlier magnitudes lower bounded** $x_{\min} \geq 15C(2\tilde{\varepsilon}\sqrt{r\lambda^+} + \Delta\sqrt{\lambda_{\text{ch}}})$ **slow subspace change** **first Cr samples: AltProj assumptions**	**Memory:** $\mathcal{O}(nr \log n)$ **Time:** $\mathcal{O}(ndr \log \frac{1}{\epsilon})$ **Detect delay:** $2\alpha = Cr \log n$ **Tracking Delay:** $K\alpha = Cr \log n \log(1/\epsilon)$
s-ReProCS-no-delete (online) (this work)	max-outlier-frac-row$^{\alpha} \in \boldsymbol{\mathcal{O}(1)}$ max-outlier-frac-col $\in \boldsymbol{\mathcal{O}(1/r_L)}$	**most outlier magnitudes lower bounded** $x_{\min} \geq 15C(2\tilde{\varepsilon}\sqrt{r\lambda^+} + \Delta\sqrt{\lambda_{\text{ch}}})$ **slow subspace change** **first Cr samples: AltProj assumptions**	**Memory:** $\mathcal{O}(nr \log n)$ **Time:** $\mathcal{O}(ndr \log \frac{1}{\epsilon})$ **Detect delay:** $2\alpha = Cr \log n$ **Tracking Delay:** $K\alpha = Cr \log n \log(1/\epsilon)$
ReProCS-NORST [18, 17] (*online*) (follow-up to this work)	max-outlier-frac-row $= \mathcal{O}(1/f^2)$ max-outlier-frac-col $= \mathcal{O}(1/r)$	outlier mag. lower bounded $x_{\min} \geq C_1\sqrt{r\lambda^+}(\Delta + 2\tilde{\varepsilon})$ slow subspace change or fixed subspace first Cr samples: AltProj assumptions	Memory: $\mathcal{O}(nr \log n \log \frac{1}{\epsilon})$ Time: $\mathcal{O}(ndr \log \frac{1}{\epsilon})$ Detect delay: $Cr \log n$ Tracking Delay: $Cr \log n \log(1/\epsilon)$

2.2 The simple-ReProCS Algorithm and its Guarantee

2.2.1 Simple-ReProCS (s-ReProCS)

S-ReProCS proceeds as follows. The initial subspace is assumed to be accurately known (obtained using AltProj or PCP). At time t, if the previous subspace estimate, $\hat{\boldsymbol{P}}_{(t-1)}$, is accurate enough, because of slow subspace change, projecting $\boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{\ell}_t + \boldsymbol{v}_t$ onto its orthogonal complement will nullify most of $\boldsymbol{\ell}_t$. Moreover, $\|\boldsymbol{v}_t\|$ is small (by assumption). We compute $\tilde{\boldsymbol{y}}_t := \boldsymbol{\Psi}\boldsymbol{y}_t$ where $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$. Thus, $\tilde{\boldsymbol{y}}_t = \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{b}_t$ where $\boldsymbol{b}_t := \boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$ and $\|\boldsymbol{b}_t\|$ is small. Recovering $\boldsymbol{x}_t$ from $\tilde{\boldsymbol{y}}_t$ is thus a traditional compressive sensing (CS) / sparse recovery problem

in small noise [4]. This is solvable because incoherence (denseness) of $\boldsymbol{P}_{(t)}$'s and slow subspace change implies [22] that $\boldsymbol{\Psi}$ satisfies the restricted isometry property [4]. We compute $\hat{\boldsymbol{x}}_{t,cs}$ using l_1 minimization followed by thresholding based support estimation to get $\hat{\mathcal{T}}_t$. A Least Squares (LS) based debiasing step on $\hat{\mathcal{T}}_t$ returns the final $\hat{\boldsymbol{x}}_t$. We then estimate $\boldsymbol{\ell}_t$ as $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$. We refer to the above step as *Projected Compressive Sensing (CS)*. As explained in [25, 28], this can also be understood as solving a *Robust Regression* problem[2].

The $\hat{\boldsymbol{\ell}}_t$'s are used for the *Subspace Update* step which involves (i) detecting subspace change; (ii) obtaining improved estimates of the changed direction(s) by K steps of projection-SVD [22], each done with a new set of α frames of $\hat{\boldsymbol{\ell}}_t$; and (iii) a simple SVD based subspace re-estimation step, done with another new set of α frames. This is done to remove the deleted direction and get an r-dimensional estimate of the new subspace. We explain the subspace change detection strategy in Sec. 2.4.2. Suppose the change is detected at $\hat{t}_j$. The k-th projection-SVD step involves computing $\hat{\boldsymbol{P}}_{j,\text{rot},k}$ as the top singular vector of $(\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}')[\hat{\boldsymbol{\ell}}_{\hat{t}_j+(k-1)\alpha}, \hat{\boldsymbol{\ell}}_{\hat{t}_j+(k-1)\alpha+1}, \dots, \hat{\boldsymbol{\ell}}_{\hat{t}_j+k\alpha-1}]$ and setting $\hat{\boldsymbol{P}}_{(t)} = \hat{\boldsymbol{P}}_{j,k} := [\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},k}]$. For ease of understanding, we summarize a basic version of s-ReProCS in Algorithm 1. This assumes that the change times t_j are known, i.e., that $\hat{t}_j = t_j$. The actual algorithm that detects changes automatically is longer and is given as Algorithm 4 in Sec. 2.4.2. *We both analyze and implement this one.*

The above approach works because, every time the subspace changes, with high probability (whp), the change can be detected within a short delay, and after that, the K projection-SVD steps help get progressively improved estimates of the changed/rotated direction $\boldsymbol{P}_{j,\text{rot}}$. The final simple SVD step re-estimates the entire subspace in order to delete $\boldsymbol{P}_{j,\text{del}}$, from the estimate.

The estimates of the subspace or of $\boldsymbol{\ell}_t$'s are improved in offline mode as follows. At $t = \hat{t}_j + K\alpha$, the K projection-SVD steps are complete and hence the subspace estimate at this time is accurate enough whp. At this time, *offline s-ReProCS* (last line of Algorithm 4) goes back and sets $\hat{\boldsymbol{P}}_{(t)} \leftarrow$

[2]The above step equivalently solves for $\tilde{\boldsymbol{a}}, \tilde{\boldsymbol{x}}$ that satisfy $\boldsymbol{y}_t = \hat{\boldsymbol{P}}_{t-1}\tilde{\boldsymbol{a}} + \tilde{\boldsymbol{x}} + \boldsymbol{b}_t$ with $\tilde{\boldsymbol{x}}$ being sparse and $\|\boldsymbol{b}_t\|$ being small. This is the approximate robust regression problem where columns of $\hat{\boldsymbol{P}}_{t-1}$ are the regressors/predictors, $\tilde{\boldsymbol{x}}$ is the sparse outliers and $\boldsymbol{b}_t$ is the small "noise" or model inaccuracy.

$[\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},K}]$ for all $t \in [\hat{t}_{j-1} + K\alpha, \hat{t}_j + K\alpha)$. It also uses this to get improved estimates of $\hat{\boldsymbol{x}}_t$ and $\hat{\boldsymbol{\ell}}_t$ for all these times t.

2.2.2 Assumptions and Main Result

Incoherence (denseness) of columns of $\boldsymbol{P}_j$'s. In order to separate the $\boldsymbol{\ell}_t$'s from the sparse outliers $\boldsymbol{x}_t$, we need an assumption that ensures that the $\boldsymbol{\ell}_t$'s are themselves not sparse. One way to ensure this is to assume μ-incoherence [5] of the basis matrix for the subspace spanned by the columns of $\boldsymbol{P}_{j-1}$ and $\boldsymbol{P}_j$, i.e., assume that

$$\max_{j=1,2,\ldots,J} \max_{i=1,2,\ldots,n} \| \text{basis}([\boldsymbol{P}_{j-1}, \boldsymbol{P}_j])^i \| \leq \sqrt{\frac{\mu(r+1)}{n}} \tag{2.6}$$

for a $\mu \geq 1$ but not too large (assumed to be a numerical constant henceforth). Because of our subspace change model, the subspace spanned by the columns of $[\boldsymbol{P}_{j-1}, \boldsymbol{P}_j]$ has dimension $r+1$. In fact, $\text{basis}([\boldsymbol{P}_{j-1}, \boldsymbol{P}_j]) = [\boldsymbol{P}_{j-1}, \boldsymbol{P}_{j,\text{new}}]$.

It is easy to see that (2.6), along with the bound on max-outlier-frac-col assumed in Theorem 2.2 given below (max-outlier-frac-col $\leq 0.01/(2\mu(r+1))$), implies that (2.13) given later holds[3]. Our result actually only needs (2.13), but that is complicated to state and explain. Hence we use the above stronger but well-understood assumption.

Assumption on principal subspace coefficients $\boldsymbol{a}_t$. We assume that the $\boldsymbol{a}_t$'s are zero mean, mutually independent, *element-wise bounded* random variables (r.v.), have identical and diagonal covariance matrix denoted $\boldsymbol{\Lambda}$, and are independent of the outlier supports $\mathcal{T}_t$. Here element-wise bounded means that there exists a numerical constant η, such that

$$\max_{j=1,2,\ldots r} \max_t \frac{(\boldsymbol{a}_t)_j^2}{\lambda_j(\boldsymbol{\Lambda})} \leq \eta.$$

For most bounded distributions, η is a little more than one, e.g., if the entries of $\boldsymbol{a}_t$ are zero mean uniform, then $\eta = 3$. As we explain later the above assumptions of $\boldsymbol{a}_t$ replace the right singular vectors' incoherence assumption used by all standard RPCA solutions.

[3]This is true because (i) for any basis matrix $\boldsymbol{P}$, $\max_{\mathcal{T}:|\mathcal{T}|\leq 2s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}\|^2 \leq 2s \max_i \|\boldsymbol{I}_i{}'\boldsymbol{P}\|^2$ [22], here $s = \text{max-outlier-frac-col} \cdot n$; (ii) if $\tilde{\boldsymbol{P}}$ is such that $\text{span}(\tilde{\boldsymbol{P}}) \subseteq \text{span}(\boldsymbol{P})$, then $\|\boldsymbol{I}_{\mathcal{T}}{}'\tilde{\boldsymbol{P}}\|^2 \leq \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}\|^2$; and (iii) both $\text{span}(\boldsymbol{P}_j)$ and $\text{span}(\boldsymbol{P}_{j,\text{new}})$ are contained in the span of $\text{basis}([\boldsymbol{P}_{j-1}, \boldsymbol{P}_j])$. In fact $\text{basis}([\boldsymbol{P}_{j-1}, \boldsymbol{P}_j]) = [\boldsymbol{P}_{j-1}, \boldsymbol{P}_{j,\text{new}}]$. Thus, using the max-outlier-frac-col bound, $\max_{\mathcal{T}:|\mathcal{T}|\leq 2s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_j\|^2 \leq 2s\mu(r+1)/n = 2\text{max-outlier-frac-col}\mu(r+1) \leq 0.01$ and the same also holds for $\max_{\mathcal{T}:|\mathcal{T}|\leq 2s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_{j,\text{new}}\|^2$.

Outlier fractions bounded. Similar to earlier RPCA works, we also need outlier fractions to be bounded. However, we need different bounds on this fraction per column and per row. The row bound can be much larger[4]. Since the ReProCS subspace update step operates on mini-batches of data of size α (i.e. on $n \times \alpha$ sub-matrices of consecutive columns), we need to bound max-outlier-frac-row for each such sub-matrix. We denote this by max-outlier-frac-row$^\alpha$.

Definition 2.1.

1. *For a time interval, $\mathcal{J}$, define*

$$\gamma(\mathcal{J}) := \max_{i=1,2,\dots,n} \frac{1}{|\mathcal{J}|} \sum_{t \in \mathcal{J}} \mathbb{1}_{\{i \in \mathcal{T}_t\}}. \tag{2.7}$$

   Thus $\gamma(\mathcal{J})$ is the maximum outlier fraction in any row of the sub-matrix $\boldsymbol{Y}_{\mathcal{J}}$ of $\boldsymbol{Y}$. Let $\mathcal{J}^\alpha$ denote a time interval of duration α. Define

$$\textit{max-outlier-frac-row}^\alpha := \max_{\mathcal{J}^\alpha \subseteq [t_1, d]} \gamma(\mathcal{J}^\alpha). \tag{2.8}$$

2. *Define max-outlier-frac-col* $:= \max_t |\mathcal{T}_t|/n$.

3. *Let* $x_{\min} := \min_t \min_{i \in \mathcal{T}_t} |(\boldsymbol{x}_t)_i|$ *denote the minimum outlier magnitude.*

4. *Use λ^- and λ^+ to denote the minimum and maximum eigenvalues of $\boldsymbol{\Lambda}$ and $f := \frac{\lambda^+}{\lambda^-}$ its condition number.*

5. *Split $\boldsymbol{a}_t$ as $\boldsymbol{a}_t = \begin{bmatrix} \boldsymbol{a}_{t,\text{fix}} \\ \boldsymbol{a}_{t,\text{ch}} \end{bmatrix}$ where $\boldsymbol{a}_{t,\text{ch}}$ is the scalar coefficient corresponding to the changed direction. Similarly split its diagonal covariance matrix as $\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\Lambda}_{\text{fix}} & \boldsymbol{0} \\ \boldsymbol{0} & \lambda_{\text{ch}} \end{bmatrix}$.*

6. *Let $\tilde{\varepsilon}$ denote the bound on initial subspace error, i.e., let* $\text{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \le \tilde{\varepsilon}$.

7. *For numerical constants C that are re-used to denote different numerical values, define*

$$K := \lceil C \log(\Delta/\tilde{\varepsilon}) \rceil, \ \textit{and} \ \alpha \ge \alpha_* := C f^2 (r \log n). \tag{2.9}$$

[4]One practical application where this is useful is for slow moving or occasionally static video foreground moving objects. For a stylized example of this, see Model 2.19 given in Sec. 2.6.

Main Result. We can now state our main result. For ease of understanding, we provide a table explaining various symbols, and assumptions required for Theorem 2.2 in Table 2.3.

Theorem 2.2. *Consider simple-ReProCS given in Algorithm 4. Assume that* $\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \leq \tilde{\varepsilon}$ *with* $\tilde{\varepsilon} f \leq 0.01 \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$.

1. *(statistical assumptions) assumptions on* $\boldsymbol{a}_t$*'s hold;*
2. *(subspace change)*
   (a) (2.2), (2.3), *and* (2.4) *hold with* $t_{j+1} - t_j > (K+3)\alpha$ *where* K *and* α *are defined above in* (2.9),
   (b) Δ *satisfies* $C(2\tilde{\varepsilon}\sqrt{r\lambda^+} + \Delta\sqrt{\lambda_{\mathrm{ch}}}) < x_{\min}/15$ *with* $C = \sqrt{\eta}$;
3. *(outlier fractions and left incoherence)*
   (a) (2.6) *holds and max-outlier-frac-col* $\leq \rho_{\mathrm{col}} := \frac{0.01}{2\mu(r+1)}$,
   (b) *max-outlier-frac-row*$^{\alpha} \leq \rho_{\mathrm{row}} := \frac{0.01}{f^2}$;
4. *(noise* $\boldsymbol{v}_t$*)* $\boldsymbol{v}_t$*'s are zero mean, mutually independent, independent of the* $\boldsymbol{x}_t$*'s and* $\boldsymbol{\ell}_t$*'s, and satisfy* $\|\boldsymbol{v}_t\|^2 \leq 0.1\tilde{\varepsilon}^2 r\lambda^+$ *and* $\|\mathbb{E}[\boldsymbol{v}_t \boldsymbol{v}_t{}']\| \leq 0.1\tilde{\varepsilon}^2\lambda^+$;
5. *(algorithm parameters) set* K *and* α *as in* (2.9), $\xi = x_{\min}/15$, $\omega_{supp} = x_{\min}/2$, $\omega_{evals} = 5\tilde{\varepsilon}^2 f\lambda^+$;

then, with probability at least $1 - 12dn^{-12}$, *at all times,* t,

1. $\hat{\mathcal{T}}_t = \mathcal{T}_t$,
2. $t_j \leq \hat{t}_j \leq t_j + 2\alpha$,

3. $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) \leq$

$$\begin{cases} 2\tilde{\varepsilon} + \Delta & \text{if } t \in [t_j, \hat{t}_j + \alpha) \\ 1.2\tilde{\varepsilon} + (0.5)^{k-2} 0.06\Delta & \text{if } t \in [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha) \\ 2\tilde{\varepsilon} & \text{if } t \in [\hat{t}_j + K\alpha, \hat{t}_j + K\alpha + \alpha) \\ \tilde{\varepsilon} & \text{if } t \in [\hat{t}_j + K\alpha + \alpha, t_{j+1}) \end{cases}$$

4. *and* $\|\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t\| = \|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\| \leq C(\tilde{\varepsilon}\sqrt{r\lambda^+} + \mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)})\sqrt{\lambda_{\mathrm{ch}}}$ *with* $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)})$ *bounded as above.*

Consider offline s-ReProCS (last line of Algorithm 4). At all t,

$$\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}^{\mathrm{offline}}, \boldsymbol{P}_{(t)}) \leq 2\tilde{\varepsilon}, \text{ and } \|\hat{\boldsymbol{\ell}}_t^{\mathrm{offline}} - \boldsymbol{\ell}_t\| \leq 2.4\tilde{\varepsilon}\|\boldsymbol{\ell}_t\|.$$

The upper bound on $\boldsymbol{v}_t$ and the lower bound on $x_{\min}$ can be relaxed significantly to get a more complicated result which we state in the corollary below.

Corollary 2.3. *Let* $x_{\min,t} := \min_{i \in \mathcal{T}_t} |(\boldsymbol{x}_t)_i|$ *denote the minimum outlier magnitude at time* t *and define the time intervals*

- $\mathcal{J}_0 = [t_j, \hat{t}_j)$ *(interval before the change gets detected),*
- $\mathcal{J}_k := [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$ *(k-th subspace update uses data from this interval) for* $k = 1, 2, 3, \ldots, K$,
- *and* $\mathcal{J}_{K+1} := [\hat{t}_j + K\alpha, \hat{t}_j + K\alpha + \alpha)$ *(final SVD-based re-estimation step uses data from this interval).*

All conclusions of Theorem 2.2 hold if the following hold instead of assumptions 1b, 4, and 5 of Theorem 2.2:

$\boldsymbol{v}_t$*'s are zero mean, mutually independent, independent of the* $\boldsymbol{x}_t$*'s and* $\boldsymbol{\ell}_t$*'s,* $\|\boldsymbol{v}_t\| \leq b_{v,t}$, $\|\mathbb{E}[\boldsymbol{v}_t \boldsymbol{v}_t{}']\| \leq b_{v,t}^2/r$, $x_{\min,t}$ *and* $b_{v,t}$ *satisfy the following:*

1. *for* $t \in \mathcal{J}_0 \cup \mathcal{J}_1$, $b_{v,t} = C(2\tilde{\varepsilon}\sqrt{r\lambda^+} + 0.11\Delta\sqrt{\lambda_{\mathrm{ch}}})$, *and* $x_{\min,t} \geq 30 b_{v,t}$,

2. *for* $t \in \mathcal{J}_k$, $b_{v,t} = C(2\tilde{\varepsilon}\sqrt{r\lambda^+} + 0.5^{k-2}0.06\Delta\sqrt{\lambda_{\text{ch}}})$, *and* $x_{\min,t} \geq 30b_{v,t}$, *for* $k = 2, \ldots, K$,

3. *for* $t \in \mathcal{J}_{K+1}$, $b_{v,t} = C(\tilde{\varepsilon}\sqrt{r\lambda^+})$ *and* $x_{\min,t} \geq 30b_{v,t}$

with $C = \sqrt{\eta}$; *and we set* $\omega_{supp,t} = x_{\min,t}/2$, *and* $\xi_t = x_{\min,t}/15$ *(alternatively, one can also set* $\omega_{supp,t}$ *and* ξ_t *to be proportional to* $b_{v,t}$ *which itself is proportional to the bound on* $\|\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t\|$ *is each interval).*

Proof. We explain the ideas leading to the proof in Sec. 2.4. Instead of first proving Theorem 2.2 and then Corollary 2.3, we directly only prove the latter. The proof of the former is almost the same and is immediate once the latter proof can be understood. For notational simplicity, we first prove the results under the assumption $\hat{t}_j = t_j$ in Sec. 2.5. The proof without assuming $\hat{t}_j = t_j$ is given in Appendix 2.9. □

With the above corollary, the following remark is immediate.

Remark 2.4 (Bi-level outliers)**.** *The lower bound on outlier magnitudes can be relaxed to the following which only requires that most outlier magnitudes are lower bounded, while the others have small enough magnitudes so that their squared sum is upper bounded: Assume that the outlier magnitudes are such that the following holds:* $\boldsymbol{x}_t$ *can be split as* $\boldsymbol{x}_t = (\boldsymbol{x}_t)_{small} + (\boldsymbol{x}_t)_{large}$ *with the two components having disjoint supports and being such that,* $\|(\boldsymbol{x}_t)_{small}\| \leq b_{v,t}$ *and the smallest nonzero entry of* $(\boldsymbol{x}_t)_{large}$ *is greater than* $30b_{v,t}$ *with* $b_{v,t}$ *as defined in Corollary 2.3. If the above is true, and if the vectors* $(\boldsymbol{x}_t)_{small}$ *are zero mean, mutually independent, and independent of* $\boldsymbol{\ell}_t$*'s and of the support of* $(\boldsymbol{x}_t)_{large}$*, then all conclusions of Theorem 2.2 hold except the exact support recovery conclusion (this gets replaced by exact recovery of the support of* $(\boldsymbol{x}_t)_{large}$*).*

This remark follows by replacing $\boldsymbol{v}_t$ *by* $\boldsymbol{v}_t + (\boldsymbol{x}_t)_{small}$ *and* $\boldsymbol{x}_t$ *by* $(\boldsymbol{x}_t)_{large}$ *in Corollary 2.3.*

Remark 2.5. *The first condition (accurate initial estimate) can be satisfied by applying any standard RPCA solution, e.g., PCP, AltProj, or GD, on the first* $t_{\text{train}} = Cr$ *data frames. This requires assuming that* $t_1 \geq Cr$, *and that* $\boldsymbol{Y}_{[1,t_{\text{train}}]}$ *has outlier fractions in any row or column bounded by* c/r. *Moreover, it is possible to significantly relax the initial estimate requirement to only requiring*

Algorithm 1 Simple-ReProCS (with t_j known). We state this first for simplicity. The actual automatic version is given later in Algorithm 4. Let $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \dots, \hat{\boldsymbol{\ell}}_t]$.

Input: $\hat{\boldsymbol{P}}_0$, $\boldsymbol{y}_t$, **Output**: $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}_{(t)}$, **Parameters:** ω_{supp}, K, α, ξ, r, t_j's
$\hat{\boldsymbol{P}}_{(t_{\text{train}})} \leftarrow \hat{\boldsymbol{P}}_0$; $j \leftarrow 1$, $k \leftarrow 1$
for $t > t_{\text{train}}$ **do**
  $(\hat{\boldsymbol{x}}_t, \hat{\mathcal{T}}_t) \leftarrow \text{ProjCS}(\hat{\boldsymbol{P}}_{(t-1)}, \boldsymbol{y}_t)$ ▷ Algorithm 2
  $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$.
  $(\hat{\boldsymbol{P}}_{(t)}, \hat{\boldsymbol{P}}_j, j, k) \leftarrow \text{SubUp}(\hat{\boldsymbol{L}}_{t;\alpha}, \hat{\boldsymbol{P}}_{j-1}, t, t_j, j, k, \hat{\boldsymbol{P}}_{(t-1)})$ ▷ Algorithm 3
end for

that $\text{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \le c/\sqrt{r}$ *if we use* K *iterations of the approach of follow-up work [17] to improve the estimate of* $\boldsymbol{P}_0$ *until a* $\tilde{\varepsilon}$ *accurate estimate is obtained, and then run s-ReProCS. For this to work, we will need a larger lower bound on* $x_{\min}$ *for the initial period.*

Remark 2.6 (Connecting to the left incoherence of standard RPCA solutions)**.** *With minor changes, our left incoherence assumption, (2.6), can be replaced by something that is very close to the one used by all standard RPCA solutions. Instead of (2.6), we can assume* μ*-incoherence of* $\text{basis}([\boldsymbol{P}_0, \boldsymbol{P}_{1,\text{new}}, \boldsymbol{P}_{2,\text{new}}, \dots, \boldsymbol{P}_{J,\text{new}}])$*. This implies that*[5] *the RHS of (2.6) is bounded by* $\sqrt{\mu(r+J)/n}$*. With this, the only change to Theorem 2.2 will be that we will need max-outlier-frac-col* $\le 0.01/(2\mu(r+J))$*.*

If $\boldsymbol{P}_{j,\text{new}}$ *is orthogonal to* $\text{span}([\boldsymbol{P}_0, \boldsymbol{P}_{1,\text{new}}, \dots, \boldsymbol{P}_{j-1,\text{new}}])$ *for each* j*, then the matrix* $[\boldsymbol{P}_0, \boldsymbol{P}_{1,\text{new}}, \boldsymbol{P}_{2,\text{new}}, \dots, \boldsymbol{P}_{J,\text{new}}]$ *is itself a basis matrix, its span is* equal *to that of the left singular vectors of* $\boldsymbol{L}$*, and its rank* $r + J = r_L$*. In this case, the above assumption and the corresponding required bound on max-outlier-frac-col are exactly the same as those used by the standard RPCA solutions.*

2.2.3 Discussion

In this section, we discuss the various implications of our result, the speed and memory guarantees, explain how to set algorithm parameters, and finally discuss its limitations.

[5] *follows because the union of the spans of* $\boldsymbol{P}_{j-1}$ *and* $\boldsymbol{P}_j$ *is contained in the span of* $[\boldsymbol{P}_0, \boldsymbol{P}_{1,\text{new}}, \boldsymbol{P}_{2,\text{new}}, \dots, \boldsymbol{P}_{J,\text{new}}]$.

Algorithm 2 Projected CS (ProjCS)

function ProjCS($\hat{\boldsymbol{P}}_{(t-1)}$, $\boldsymbol{y}_t$)
  $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$
  $\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$
  $\hat{\boldsymbol{x}}_{t,cs} \leftarrow \arg\min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}}\|_1$ s.t $\|\tilde{\boldsymbol{y}}_t - \boldsymbol{\Psi}\tilde{\boldsymbol{x}}\| \le \xi$
  $\hat{\mathcal{T}}_t \leftarrow \{i : |\hat{\boldsymbol{x}}_{t,cs}| > \omega_{supp}\}$
  $\hat{\boldsymbol{x}}_t \leftarrow \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\tilde{\boldsymbol{y}}_t$
  return $\hat{\boldsymbol{x}}_t$, $\hat{\mathcal{T}}_t$
end function

Algorithm 3 Subspace Update (SubUpd).

function SubUp($\hat{\boldsymbol{L}}_{t;\alpha}$, $\hat{\boldsymbol{P}}_{j-1}$, t, t_j, j, k, $\hat{\boldsymbol{P}}_{(t-1)}$)
  if $t = t_j + u\alpha$ for $u = 1,\ 2,\ \cdots, K+1$ **then**
    $\boldsymbol{B} \leftarrow (\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}')\hat{\boldsymbol{L}}_{t;\alpha}$
    $\hat{\boldsymbol{P}}_{j,\text{rot},k} \leftarrow SVD_1[\boldsymbol{B}]$ ▷ subspace addition: via K steps of projection-SVD
    $\hat{\boldsymbol{P}}_{(t)} \leftarrow [\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},k}]$, $k \leftarrow k+1$.
    if $k = K+1$ **then**
      $\hat{\boldsymbol{P}}_j \leftarrow SVD_r[\hat{\boldsymbol{L}}_{t;\alpha}]$ ▷ subspace deletion: via subspace re-estimation using simple SVD
      $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_j$, $j \leftarrow j+1$, $k \leftarrow 1$.
    end if
  else
    $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$
  end if
  return $\hat{\boldsymbol{P}}_{(t)}$, $\hat{\boldsymbol{P}}_j$, j, k
end function

Subspace change detection and tracking with short delay. Theorem 2.2 shows that, whp, the subspace change gets detected within a delay of at most $2\alpha = Cf^2(r\log n)$ frames, and the subspace gets estimated accurately within at most $(K+3)\alpha = Cf^2(r\log n)\log(1/\tilde{\varepsilon})$ frames. Each column of the low rank matrix is recovered with a small time-invariant bound without any delay. If offline processing is allowed, with a delay of at most $(K+3)\alpha$, we can guarantee all recoveries within normalized error $\tilde{\varepsilon}$, or, in fact, with minor modifications, within any $\epsilon = c\tilde{\varepsilon}$ for $c < 1$ (also see the limitations' discussion). Notice that the required delay between subspace change times is more than r by only logarithmic factors (assuming f does not grow with n or r). Since the previous subspace is not exactly known (is known within error at most $\tilde{\varepsilon}$), at each update step, we *do* need to estimate an r-dimensional subspace, and not a one-dimensional one. Hence it is not clear if

the required delay can be reduced any further. Moreover, the delay required for the deletion step cannot be less than r even in the ideal case when $\boldsymbol{\ell}_t$ is directly observed.

Bi-level outliers. Consider the upper bound on Δ (amount of subspace change). Observe that the upper bound essentially depends on the ratio between $x_{\min}$ (minimum outlier magnitude) and $\sqrt{\lambda_{\text{ch}}}$. Read another way, this means that $x_{\min}$ needs to be lower bounded. On first glance, this may seem counter-intuitive since sufficiently small magnitude corruptions should not be problematic. This is actually true. Sufficiently small magnitude corruptions get classified as the small noise $\boldsymbol{v}_t$. Moreover, as noted in Corollary 2.3 and Remark 2.4, our result actually allows "bi-level" corruptions/outliers that need to satisfy a much weaker requirement than this: the large-outliers have magnitude that is "large enough", while the rest are such that the squared sum of their magnitudes is "small enough". The threshold for both "large enough" and "small enough" decreases with each subspace update step.

Order-wise improvement in allowed upper bound on maximum number of outliers per row. As pointed out in [20], solutions for standard RPCA (that only assume incoherence of left and right singular vectors of $\boldsymbol{L}$ and nothing else, i.e., no outlier support model) cannot tolerate[6] a bound on maximum outlier fractions in any row or any column that is larger than $1/r_{\scriptscriptstyle L}$. However observe that simple-ReProCS can tolerate max-outlier-frac-row$^{\alpha} \in O(1)$ (this assumes f is a constant). This is a significant improvement over all existing RPCA results with important practical implications for video analytics. This is possible is because s-ReProCS uses extra assumptions, we explain their next.

The need for extra assumptions. s-ReProCS recovers the sparse outliers $\boldsymbol{x}_t$ first and then the true data $\boldsymbol{\ell}_t$, and does this at each time t. Let $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$. When recovering $\boldsymbol{x}_t$, it exploits two facts: (a) the subspace of $\boldsymbol{\ell}_t$, $\boldsymbol{P}_{(t)}$, satisfies the denseness/incoherence property, and (b) "good" knowledge of the subspace of $\boldsymbol{\ell}_t$ (either from initialization or from the previous subspace's

[6]The reason is this: let ρ_{row} = max-outlier-frac-row, one can construct a matrix $\boldsymbol{X}$ with ρ_{row} outliers in some rows that has rank equal to $1/\rho_{\text{row}}$. A simple way to do this would be to let the support and nonzero entries of $\boldsymbol{X}$ be constant for $\rho_{\text{row}}d$ columns before letting either of them change. Then the rank of $\boldsymbol{X}$ will be $d/(\rho_{\text{row}}d) = 1/\rho_{\text{row}}$. If $1/\rho_{\text{row}} < r_{\scriptscriptstyle L} = \text{rank}(\boldsymbol{L})$, $\boldsymbol{X}$ will wrongly get classified as the low-rank component. This is why we need $\rho_{\text{row}} = \text{max-outlier-frac-row} < 1/r_{\scriptscriptstyle L}$. A similar argument can be used for max-outlier-frac-col.

estimate and slow subspace change) is available. Using these two facts one can show that $\mathbf{\Psi}$ satisfies the RIP property, and that the "noise" seen by the compressive sensing step, $\boldsymbol{b}_t := \mathbf{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$, is small. This, along with a guarantee for CS, helps ensure that the error in recovering $\boldsymbol{x}_t$ is upper bounded[7] by $C\|\boldsymbol{b}_t\|$. This, in turn, means that, to correctly recover the support of $\boldsymbol{x}_t$, the minimum large-outlier magnitude needs to be larger than $C\|\boldsymbol{b}_t\|$. This is where the $x_{\min}$ or $x_{\min,t}$ lower bound comes from[8].

Correct outlier support recovery is needed to ensure that the subspace estimate can be improved with each subspace update step. In particular, it helps ensure that the error vectors $\boldsymbol{e}_t := \boldsymbol{x}_t - \hat{\boldsymbol{x}}_t$ in a given subspace update interval are mutually independent when conditioned on the $\boldsymbol{y}_t$'s from all past intervals. This fact also relies on the mutual independence assumption on the $\boldsymbol{a}_t$'s. Moreover, mutual independence, along with the element-wise boundedness and identical covariances assumption, on the $\boldsymbol{a}_t$'s helps ensure that we can use matrix Bernstein [24] and Vershynin's sub-Gaussian result (bounds singular values of matrices with independent sub-Gaussian rows) [29] for obtaining the desired concentration bounds on the subspace recovery error in each step. As explained below, the above assumptions on $\boldsymbol{a}_t$ replace the right incoherence assumption. Finally, because ReProCS is updating the subspace using just the past α estimates of $\hat{\boldsymbol{\ell}}_t$'s, in order to show that each such step improves the estimate we need to bound max-outlier-frac-row$^{\alpha}$ (instead of just max-outlier-frac-row).

Time and Memory Complexity. Observe that the s-ReProCS algorithm needs memory of order $n\alpha$ in online mode and $Kn\alpha$ in offline mode. Assuming $\alpha = \alpha_*$, even in offline mode, its memory complexity is near-optimal and equal to $O(nr \log n \log(1/\tilde{\varepsilon}))$. Also, observe that the time complexity of s-ReProCS is $O(ndr \log(1/\tilde{\varepsilon}))$. We explain this in Appendix 2.14. These claims

[7] Since the individual vector $\boldsymbol{b}_t$ does not have any structure that can be exploited, the error in recovering $\boldsymbol{x}_t$ cannot be made lower than this. However the $\boldsymbol{b}_t$'s arranged into a matrix do form a low-rank matrix whose approximate rank can be shown to be one (under our current subspace change model). If we try to exploit this structure we end up with the modified-PCP approach studied in earlier work [35]. This needs the uniform random support assumption [35].

[8] If there were a way to bound the element-wise error of the CS step (instead of the l_2 norm of the error), we could relax the $x_{\min}$ lower bound significantly. It is not clear if this is possible though.

assume that f is constant with n, r. If the dependence on f is included both will be multiplied by f^2.

Subspace and outlier assumptions' tradeoff. When there are fewer outliers in the data or when outliers are easy to detect, one would expect to need weaker assumptions on the true data subspace or its rate of change. This is indeed true. For the original RPCA results, this is encoded in the condition max(max-outlier-frac-row, max-outlier-frac-col) $\leq c/(\mu r_L)$ where μ quantifies not-denseness of both left and right singular vectors. From Theorem 2.2, this is also how max-outlier-frac-col, μ (not-denseness of only left singular vectors) and r_L are related for dynamic RPCA. On the other hand, for our result, max-outlier-frac-row$^\alpha$ and the lower bound on $x_{\min}$ govern the allowed rate of subspace change. The latter relation is easily evident from the bound on Δ. If $x_{\min}$ is larger (outliers are large magnitude and hence easy to detect), a larger Δ can be tolerated. The relation of max-outlier-frac-row to rate of change is not evident from the way the guarantee is stated in Theorem 2.2. The reason is we have assumed max-outlier-frac-row$^\alpha \leq \rho_{\text{row}} = 0.01/f^2$ and used that to get a simple expression for K. If we did not do this, we would need K to satisfy

$$c_1\Delta(c_2 f\sqrt{\rho_{\text{row}}})^K + 0.2\tilde{\varepsilon} \leq \tilde{\varepsilon}.$$

With this, K needs to be $K = \lceil \frac{1}{-\log(c_2 f\sqrt{\rho_{\text{row}}})} \log(\frac{c_1\Delta}{0.8\tilde{\varepsilon}}) \rceil$. Recall that we need $t_{j+1} - t_j \geq (K+3)\alpha$. Thus, a smaller ρ_{row} (smaller max-outlier-frac-row$^\alpha$) means one of two things: either a larger Δ (more change at each subspace change time) can be tolerated while keeping K, and hence the lower bound on the delay between change times, the same; or, for Δ fixed, a smaller lower bound is needed on the delay between change time. The above can be understood by carefully checking the proof[9] of Theorem 2.7.

The need for detecting subspace change. As pointed out by an anonymous reviewer, it may not be clear to a reader why we need to explicitly detect subspace change (instead of just always doing subspace update at regularly spaced intervals). The change detection is needed for two key reasons. First, in the projection-SVD step for subspace update, we use $\hat{\boldsymbol{P}}_{j-1}$ as the best

[9]The multiplier 0.4 of q_{rot} in its first claim is obtained by setting $\rho_{\text{row}} = 0.01/f^2$. If we do not do this, 0.4 will get replaced by $c_2 f\sqrt{\rho_{\text{row}}}$.

estimate of the previous subspace. We let $\hat{\boldsymbol{P}}_{j-1}$ be the final $\tilde{\varepsilon}$-accurate subspace estimate obtained after K projection-SVD steps and then one subspace re-estimation step. To know when the K updates are over, we need to know when the first update of the new subspace occurred, or in other words, we need an estimate of when the subspace change occurred. Second, in the current algorithm, because we detect change, we can choose to use the $\hat{\boldsymbol{\ell}}_t$'s from the next α-frame interval, i.e. $[\hat{\boldsymbol{\ell}}_{\hat{t}_j+1}, \hat{\boldsymbol{\ell}}_{\hat{t}_j+2}, \ldots, \hat{\boldsymbol{\ell}}_{\hat{t}_j+\alpha}]$, for the first subspace update. This ensures that the $\hat{\boldsymbol{\ell}}_t$'s from the interval that contains t_j (some of the $\boldsymbol{\ell}_t$'s in this interval come from $\boldsymbol{P}_{j-1}$ while others come from $\boldsymbol{P}_j$) is never used further in any subspace update. The is essential because, if these are used, one will get an incorrect subspace estimate (something in between $\boldsymbol{P}_{j-1}$ and $\boldsymbol{P}_j$) and one whose error cannot easily be bounded. If subspace change is never detected, this cannot be ensured.

Algorithm parameters. Observe from Theorem 2.2 that we need knowledge of only 4 model parameters - r, λ^+, λ^- and $x_{\min}$ - to set our algorithm parameters. The initial dataset used for estimating $\hat{\boldsymbol{P}}_0$ (using PCP/AltProj) can also be used to get an accurate estimate of r, λ^- and λ^+ using standard techniques (maximum likelihood applied to the AltProj estimate of $[\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots, \boldsymbol{\ell}_{t_{\text{train}}}]$). Thus one really only needs to set $x_{\min}$. If continuity over time is assumed, a simple heuristic is to let it be time-varying and use $\min_{i \in \hat{\mathcal{T}}_{t-1}} |(\hat{\boldsymbol{x}}_{t-1})_i|$ as its estimate at time t. This approach in fact allows us to estimate $x_{\min,t}$ and thus allows for larger unstructured noise, $\boldsymbol{v}_t$, levels as allowed by Corollary 2.3.

The most interesting point for practice though is that of Remark 2.4. It indicates that when a subspace change is detected but not estimated, starting at the previous 2α frames, one should use a larger value of the support estimation threshold ω_{supp}. After each subspace update step, ω_{supp} should be decreased roughly exponentially.

Dependence on f. Observe that f appears in our guarantee in the bound on max-outlier-frac-row$^\alpha$ and in the expression for α. The max-outlier-frac-row$^\alpha$ bound is stated that way only for simplicity. Actually, for all time instants except the α-length period when the subspace re-estimation (for deletion) step is run, we only need max-outlier-frac-row$^\alpha \leq 0.01$. We need the tighter bound max-outlier-frac-row$^\alpha \leq \rho_{\text{row}} = 0.01/f^2$ only for the simple SVD based subspace re-

estimation (deletion) step to work (i.e., only for $t \in [\hat{t}_j + K\alpha, \hat{t}_j + K\alpha + \alpha)$). Thus, if offline ReProCS were being used to solve the standard RPCA type problem (where r_L is nicely bounded), one could choose to never run the subspace deletion step. This will mean that the resulting algorithm (s-ReProCS-no-delete) will need max-outlier-frac-col $< c/\mu r_L$, but then max-outlier-frac-row < 0.01 will suffice (the bound would not depend on $1/f^2$). The α expression governs required delay between subspace change times, tracking delay, and time and memory complexity. If the deletion step is removed, the dependence of α on f will not disappear, but will weaken (it will linearly depend on f not on f^2).

We now try to relate f to the condition number of $\boldsymbol{L}$. Observe that f is the condition number of $\mathbb{E}[\boldsymbol{L}_j \boldsymbol{L}_j{}']$ for any j. The condition number of the entire matrix $\boldsymbol{L}$ can be much larger when slow subspace change holds (Δ is small). To see this, let κ^2 denote the condition number of $\mathbb{E}[\boldsymbol{L}\boldsymbol{L}']$, so that, whp, κ is approximately the condition number of $\boldsymbol{L}$. It is not hard to to see that[10], in the worst case (if $\lambda^- = \lambda_{\text{ch}}$), $\kappa^2 = \frac{f}{1-\sqrt{1-2c\Delta^2}} \approx C\frac{f}{\Delta^2}$ when Δ is small. Thus, if Δ is small, $\kappa \approx C\sqrt{f}/\Delta$ can be much larger than f. The guarantees of many of the RPCA solutions such as RPCA-GD [33] depend on κ.

Relating our assumptions to right incoherence of $\boldsymbol{L}_j := \boldsymbol{L}_{[t_j, t_{j+1})}$ [14]. We repeat this discussion from [17]. From our assumptions, $\boldsymbol{L}_j = \boldsymbol{P}_j \boldsymbol{A}_j$ with $\boldsymbol{A}_j := [\boldsymbol{a}_{t_j}, \boldsymbol{a}_{t_j+1}, \dots \boldsymbol{a}_{t_{j+1}-1}]$, the columns of $\boldsymbol{A}_j$ are zero mean, mutually independent, have identical covariance $\boldsymbol{\Lambda}$, $\boldsymbol{\Lambda}$ is diagonal, and are element-wise bounded as specified by Theorem 2.2. Let $d_j := t_{j+1} - t_j$. Define a diagonal matrix Σ with (i,i)-th entry σ_i and with $\sigma_i^2 := \sum_t (a_t)_i^2/d_j$. Define a $d_j \times r$ matrix $\tilde{\boldsymbol{V}}$ with the t-th entry of the i-th column being $(\tilde{\boldsymbol{v}}_i)_t := (\boldsymbol{a}_t)_i/(\sigma_i\sqrt{d_j})$. Then, $\boldsymbol{L}_j = \boldsymbol{P}_j \Sigma \tilde{\boldsymbol{V}}'$ and each column of $\tilde{\boldsymbol{V}}$ is unit 2-norm. Also, from the bounded-ness assumption, $(\tilde{\boldsymbol{v}}_i)_t^2 \le \eta \frac{\lambda_i}{\sigma_i^2} \cdot \frac{1}{d_j}$ where η is a numerical constant. Observe that $\boldsymbol{P}_j \Sigma \tilde{\boldsymbol{V}}'$ is not exactly the SVD of $\boldsymbol{L}_j$ since the columns of $\tilde{\boldsymbol{V}}$ are not necessarily exactly mutually orthogonal. However, if d_j is large enough, using the assumptions on

[10] To understand this, suppose that there is only one subspace change and suppose that the intervals are equal, i.e., $d - t_1 = t_1 - t_0$. Then, $\mathbb{E}[\boldsymbol{L}\boldsymbol{L}']/d = \boldsymbol{P}_{0,\text{fix}} \boldsymbol{\Lambda}_{\text{fix}} \boldsymbol{P}_{0,\text{fix}}{}' + [\boldsymbol{P}_{0,\text{ch}} \ \boldsymbol{P}_{1,\text{new}}] \boldsymbol{B} [\boldsymbol{P}_{0,\text{ch}}{}' \ \boldsymbol{P}_{1,\text{new}}{}']'$ where $\boldsymbol{B} = \lambda_{\text{ch}} \begin{bmatrix} (0.5 + 0.5\cos^2\theta_1) & -0.5\sin\theta_1\cos\theta_1 \\ -0.5\sin\theta_1\cos\theta_1 & 0.5\sin^2\theta_1 \end{bmatrix}$. The maximum eigenvalue of $\mathbb{E}[\boldsymbol{L}\boldsymbol{L}']/d$ is λ^+. Its minimum eigenvalue is the minimum eigenvalue of $\boldsymbol{B}$ which can be computed as $(1 - \cos\theta_1)\lambda_{\text{ch}}$. In the worst case $\lambda_{\text{ch}} = \lambda^-$. When the intervals are not equal, this gets replaced by $(1 - \sqrt{1 - 2c\sin^2\theta_1})\lambda^-$ for a $c < 1$. This is at most $(1 - \sqrt{1 - 2c\Delta^2})$.

$\boldsymbol{a}_t$, one can argue using any law of large numbers' result (e.g., Hoeffding inequality), that (i) the columns of $\tilde{\boldsymbol{V}}$ are approximately mutually orthogonal whp, and (ii) $\sigma_i^2 \geq 0.99\lambda_i$ whp. Thus, our assumptions imply that, whp, $\tilde{\boldsymbol{V}}$ is a basis matrix and $(\tilde{\boldsymbol{v}}_i)_t^2 \leq C/d_j$.

With the above, one can interpret $\tilde{\boldsymbol{V}}$ as an "approximation" to the right singular vectors of $\boldsymbol{L}_j$ and then the above bound on $(\tilde{\boldsymbol{v}}_i)_t^2$ is the same as the right incoherence condition assumed by [14]. It is slightly stronger than what is assumed by [5, 20] and others (these do not require a bound on each entry but on each row, they require that the squared norm of each row of the matrix of right singular vectors be bounded by Cr/d_j).

Ideally we would like to work with the exact SVD of $\boldsymbol{L}_j$, however this is much harder to analyze using our statistical assumptions on the $\boldsymbol{a}_t$'s. To see this, suppose $\boldsymbol{A}_j \overset{\text{SVD}}{=} \boldsymbol{U}\Sigma\boldsymbol{V}'$, then $\boldsymbol{L}_j \overset{\text{SVD}}{=} (\boldsymbol{P}_j\boldsymbol{U})\Sigma\boldsymbol{V}'$ is the exact SVD of $\boldsymbol{L}_j$. Here $\boldsymbol{U}$ is an $r \times r$ orthonormal matrix. Now it is not clear how to relate the element-wise bounded-ness assumption on $\boldsymbol{a}_t$'s to an assumption on entries of $\boldsymbol{V}$, since now there is no easy expression for each entry of $\boldsymbol{V}$ or of the entries of Σ in terms of $\boldsymbol{a}_t$ ($\boldsymbol{U}$ is an unknown matrix that can have all nonzero entries in general).

Limitations of our guarantees. s-ReProCS needs a few extra assumptions beyond slow subspace change and what static RPCA solutions need: (i) instead of a bound on outlier fractions per row of the entire data matrix (which is what standard RPCA methods assume), it needs such a bound for every sub-matrix of α consecutive columns; (ii) it makes statistical assumptions on the principal subspace coefficients $\boldsymbol{a}_t$ (with mutual independence being the strongest requirement); (iii) it needs to lower bound $x_{\min}$; and (iv) it uses $\tilde{\varepsilon}$ to denote both the initial subspace error as well as the final recovery error achieved after a subspace update is complete. Here (i) is needed because ReProCS is an online algorithm that uses α frames at a time to update the subspace, and one needs to show that each update step provides an improved estimate compared to the previous one. However, since α is large enough, requiring a bound max-outlier-frac-row$^{\alpha}$ is not too much stronger than requiring the same bound on the outlier fractions per row of the entire $n \times d$ matrix $\boldsymbol{Y}$. In fact, if we compare the various RPCA solutions with storage complexity fixed at $O(n\alpha) = O(nr \log n)$, i.e., if we implement the various static RPCA solutions for every new batch of α frames of data,

then, the static RPCA solutions will also need to bound max-outlier-frac-row$^{\alpha}$ defined in (2.8). As discussed earlier, these will require a much tighter bound of c/r though. (iv) is assumed for simplicity. What we can actually prove is something slightly stronger: if the initial error is $\tilde{\varepsilon}$, and if $\epsilon = c\tilde{\varepsilon}$ for a constant c which may be less than one, then, without any changes, we can guarantee the final subspace error to be below such an ϵ. More generally, as long as the initial error $\tilde{\varepsilon} \leq \Delta$, it is possible to achieve final error ϵ for *any* $\epsilon > 0$ if we assume that $t_1 - t_{\text{train}} > K\alpha$, assume a slightly larger lower bound on $x_{\min}$, and if we modify our initialization procedure to use the approach of follow-up work [17].

Limitations (ii) and (iii) are artifacts of our proof techniques. The mutual independence can be replaced by an autoregressive model on the $\boldsymbol{a}_t$'s by borrowing similar ideas from [34]. The mutual independence and zero mean assumption on the $\boldsymbol{a}_t$'s is valid for the video analytics' application if we let $\boldsymbol{\ell}_t$ be the mean-subtracted background image at time t. Then, $\boldsymbol{\ell}_t$ models independent zero-mean background image variations about a fixed mean image, e.g., variations due to lighting variations or due to moving curtains; see Fig. 2.3. This type of mean subtraction (with an estimate of the mean background image computed from training data) is commonly done in practice in many practical applications where PCA is used; it is also done in our video experiments shown later. (iii) is needed because our proof first tries to show exact outlier support recovery by solving a CS problem to recover the outliers from the projected measurements, followed by thresholding. It should be possible to relax this by relaxing the exact support recovery requirement which, in turn, will require other significant changes. For example, it may be possible to do this if one is able to do a deterministic analysis. It may be possible to also completely eliminate it if we replace the CS step by thresholding with carefully decreasing thresholds in each iteration (borrow the idea of AltProj); however, we may then require the same tight bound on max-outlier-frac-row that AltProj needs. By borrowing the stagewise idea of AltProj, it may also be possible to remove all dependence on f.

2.3 Discussion of Related Work

Limitations of earlier ReProCS-based guarantees [22, 16, 34]. In [22], we introduced the ReProCS idea and proved a *partial* guarantee for it. We call it a partial guarantee because it needed to assume something about the intermediate subspace estimates returned by the algorithm. However, this work is important because it developed a nice framework for proving guarantees for dynamic RPCA solutions. Both our later complete guarantees [16, 34] as well as the current result build on this framework.

The current work is a significant improvement over the complete guarantees obtained in [16, 34] for two other ReProCS-based algorithms for three reasons. (i) The earlier works needed very specific assumptions on how the outlier support could change (needed an outlier support model inspired by video moving objects). Our result removes such a requirement and instead only needs a bound on the fraction of outliers per column of the data matrix and on the fraction per row of an α-consecutive-column sub-matrix of the data matrix (for α large enough). (ii) The subspace change model assumed in these earlier papers can be interpreted as the current model (given in Sec. 2.2) with $\theta_j = 90^\circ$ or equivalently with $\Delta = 1$. This is an unrealistic model for slow subspace change, e.g., in 3D, it implies that the subspace changes from the x-y plane to the y-z plane. Instead, our current model allows changes from x-y plane to a slightly tilted x-y plane as shown in Fig. 2.1. This modification is more realistic *and* it allows us to replace the upper bound on λ_{ch} required by the earlier results by a similar bound on $\lambda_{\text{ch}}\Delta^2$ (see assumption 1b of Theorem 2.2). Since Δ quantifies rate of subspace change, this new requirement is much weaker. It can be satisfied by assuming that Δ is small, without making any assumption on λ_{ch}. (iii) The required minimum delay between subspace change times in the earlier results depended on $1/\epsilon^2$ where ϵ is the desired final subspace error after a subspace update is complete. This is a strong requirement. Our current result removes this unnecessarily strong dependence. The delay now only depends on $(-\log \epsilon)$ which makes it much smaller. It also implies that the memory complexity of simple-ReProCS is near-optimal. (iv) Unlike [16, 34], we analyze a simple ReProCS-based algorithm that ensures that the estimated subspace dimension is bounded by $(r+1)$, without needing the complicated cluster-

SVD algorithm. This is why our guarantee allows outlier fractions per column to be below c/r. The work of [16] needed this to be below c/r_L while [34] needed an extra assumption (clustered eigenvalues). For long data sequences, c/r can be much larger than c/r_L. We provide a detailed comparison of these assumptions in Table 2.2.

Complete guarantees for other dynamic RPCA or RPCA solutions. Another approach that solves dynamic RPCA, but in a piecewise batch fashion, is modified-PCP (mod-PCP) [35]. The guarantee for mod-PCP was proved using ideas borrowed from [5] for PCP. Thus, like [5], it also needs uniformly randomly generated support sets which is an unrealistic requirement. For the video application, this requires that foreground objects are single pixel wide and move around the entire image completely randomly over time. This is highly impractical. In Table 2.1, we provide a comparison of our current guarantees for simple-ReProCS and its offline version with those for original-ReProCS [22, 16, 34], modified-PCP [35], as well as with those for solutions for standard RPCA - [5] (referred to as PCP(C)), [14] (PCP(H), this strictly improves upon [6]), AltProj [20], RPCA via gradient descent (GD) [33] and nearly-optimal robust matrix completion (NO-RMC) [7]. The table also contains a speed and memory complexity comparison. Offline s-ReProCS can be interpreted as a solution for standard RPCA. From Table 2.1, it is clear that for data that satisfies slow subspace change and the assumption that outlier magnitudes are either large or very small, and that is such that its first t_{train} frames, $\boldsymbol{Y}_{[1,t_{\text{train}}]}$, satisfy AltProj (or PCP) assumptions, s-ReProCS and offline s-ReProCS have the following advantages over other methods.

1. For the data matrix after t_{train}, i.e., for $\boldsymbol{Y}_{[t_{\text{train}}+1,d]}$, ReProCS needs the weakest bound on max-outlier-frac-row$^{\alpha}$ without requiring uniformly randomly generated outlier support sets. This is comparable to the bound needed by PCP(C) or mod-PCP but both assume uniform random outlier supports which is a very strong requirement.

2. The memory complexity of s-ReProCS is significantly better than that of all other published methods for RPCA that provably work, and is nearly optimal.

3. Both in terms of time complexity order (Table 2.1) and experimentally (see Sec. 2.6), s-ReProCS and its offline counterpart are among the fastest, while having the best, or nearly the best, performance experimentally as well. Order-wise, only NO-RMC [7] is faster than s-ReProCS. However, NO-RMC needs the data matrix to be nearly square, i.e., it needs $c_1 n \geq d \geq c_2 n$. This is a very strong requirement that often does not hold: for the video application it requires that the number of video frames d be roughly as large as n (number of pixels in one image frame). The reason is that NO-RMC deliberately under-samples the data matrix $\boldsymbol{Y}$ by randomly throwing away some of its entries and using only the rest even when all are available. In other words, it always solves the robust matrix completion (RPCA with missing entries) problem and this is what results in a significant speed-up, but this is also why it needs $d \approx n$.

4. s-ReProCS can automatically detect subspace change and then also track it with a short delay, while the other approaches (except original-ReProCS) cannot. Notice that s-ReProCS also needs a weaker upper bound of c/r on max-outlier-frac-col while the batch techniques (PCP, AltProj, GD, NO-RMC) applied to the entire matrix $\boldsymbol{L}$ need this to be below c/r_L. Of course, if the batch techniques are applied on pieces of data $\boldsymbol{Y}_j := \boldsymbol{Y}_{[t_j, t_{j+1})}$, they also need the same looser bound of c/r on max-outlier-frac-col and their memory complexity improves too. However, the batch methods do not have a way to estimate the change times t_j, while s-ReProCS does. Moreover, since the other methods (except modified-PCP) do not have a way to use the previous subspace information, if the pieces chosen are are too small, e.g., if the methods are applied on α-frames at a time, their performance is much worse then when the entire dataset is used jointly.

Comparison with follow-up work on ReProCS-NORST [17]. As compared to ReProCS-NORST, which is the algorithm studied in our follow-up work [17] (which allows all r directions of the subspace to change at each t_j), simple-ReProCS has three advantages: (i) it is faster, (ii) it needs a weaker lower bound on $x_{\min}$ (its required lower bound essentially does not depend on r if $\tilde{\varepsilon}$ is very small), and (iii) if it is used to solve the standard RPCA problem (estimate span of

Algorithm 4 The actual simple-ReProCS algorithm, this is the one that is studied in our guarantees and also implemented in our experiments. Let $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \dots, \hat{\boldsymbol{\ell}}_t]$.

Input: $\hat{\boldsymbol{P}}_0$, $\boldsymbol{y}_t$, **Output**: $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}_{(t)}$
Parameters: ω_{supp}, K, α, ξ, r, ω_{evals}
Let $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \dots, \hat{\boldsymbol{\ell}}_t]$.
$\hat{\boldsymbol{P}}_{(t_{\text{train}})} \leftarrow \hat{\boldsymbol{P}}_0$; $j \leftarrow 1$, $k \leftarrow 1$
for $t > t_{\text{train}}$ **do**
  $(\hat{\boldsymbol{x}}_t, \hat{\mathcal{T}}_t) \leftarrow \text{ProjCS}(\hat{\boldsymbol{P}}_{(t-1)}, \boldsymbol{y}_t)$ ▷ Algorithm 2
  $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$.
  $(\hat{\boldsymbol{P}}_{(t)}, \hat{\boldsymbol{P}}_j, \hat{t}_j, k, j, \text{phase}) \leftarrow \text{AutoSubUpd}(\hat{\boldsymbol{L}}_{t;\alpha}, \hat{\boldsymbol{P}}_{j-1}, t, \hat{t}_{j-1}, j, k, \text{phase}, \hat{\boldsymbol{P}}_{(t-1)})$ ▷ Algorithm 5
end for
Offline ReProCS: At $t = \hat{t}_j + K\alpha$, for all $t \in [\hat{t}_{j-1} + K\alpha, \hat{t}_j + K\alpha - 1]$,
$\hat{\boldsymbol{P}}_{(t)}^{\text{offline}} \leftarrow [\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},K}]$;
$\hat{\boldsymbol{x}}_t^{\text{offline}} \leftarrow \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{y}_t$ where $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}' - \hat{\boldsymbol{P}}_{j,\text{rot},K}\hat{\boldsymbol{P}}_{j,\text{rot},K}{}'$;
$\hat{\boldsymbol{\ell}}_t^{\text{offline}} \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t^{\text{offline}}$.

columns of the entire matrix $\boldsymbol{L}$), we can eliminate the r-SVD based subspace re-estimation (deletion) step. With this change, (a) the required upper bound on max-outlier-frac-row$^\alpha$ for s-ReProCS does not depend on the condition number f (just max-outlier-frac-row$^\alpha \le 0.01$ suffices), and (b) its time complexity improves by a factor of r (if the initialization step is ignored) compared to ReProCS-NORST. Of course it will mean s-ReProCS-no-delete will need max-outlier-frac-col $< c/(r+J)$ which is slightly stronger.

Since ReProCS-NORST allows all r directions of the subspace to change, it also has many advantages over s-ReProCS: its subspace tracking delay is near-optimal, and it allows for a weaker initialization assumption.

2.4 Why s-ReProCS works: main ideas of our proof

In this section we explain the main ideas of our proof, first for the t_j known case, and then explain why the subspace change detection step works.

Algorithm 5 Automatic Subspace Update

function AUTOSUBUPD($\hat{\boldsymbol{L}}_{t;\alpha}$, $\hat{\boldsymbol{P}}_{j-1}$, t, $\hat{t}_{j-1}$, j, k, phase, $\hat{\boldsymbol{P}}_{(t-1)}$)
  $\hat{t}_{j-1,fin} \leftarrow \hat{t}_{j-1} + K\alpha + \alpha - 1$
  if phase = detect and $t = \hat{t}_{j-1,fin} + u\alpha$ **then**
    $\boldsymbol{B} \leftarrow (\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}')\hat{\boldsymbol{L}}_{t,\alpha}$
    if $\sigma_{\max}(\boldsymbol{B}) \geq \sqrt{\alpha\omega_{evals}}$ **then**
      phase $\leftarrow$ update, $\hat{t}_j \leftarrow t$,
    end if
    $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$
  end if
  if phase = update **then**
    $(\hat{\boldsymbol{P}}_{(t)}, \hat{\boldsymbol{P}}_j, k) \leftarrow$ SUBUP($\hat{\boldsymbol{L}}_{t;\alpha}$, $\hat{\boldsymbol{P}}_{j-1}$, t, $\hat{t}_{j-1}$, j, k, phase, $\hat{\boldsymbol{P}}_{(t-1)}$) ▷ Algorithm 3
  end if
  if $k = K + 1$ **then**
    phase $\leftarrow$ detect
  end if
  return $\hat{\boldsymbol{P}}_{(t)}$, $\hat{\boldsymbol{P}}_j$, $\hat{t}_j$, j, k, phase
end function

2.4.1 Why s-ReProCS with t_j known works

To understand things simply, first assume that $\hat{t}_j = t_j$, i.e., the subspace change times are known. Consider Algorithm 1. At each time t this consists of three steps - projected Compressive Sensing (CS) to estimate $\boldsymbol{x}_t$, estimating $\boldsymbol{\ell}_t$ by subtraction, and subspace update. Consider projected CS. This is analyzed in Lemma 2.15. At time t, suppose that we have access to $\hat{\boldsymbol{P}}_{(t-1)}$ which is a good estimate of the previous subspace, span($\boldsymbol{P}_{(t-1)}$). Because of slow subspace change, this is also a good estimate of span($\boldsymbol{P}_{(t)}$). Its first step projects $\boldsymbol{y}_t$ orthogonal to $\hat{\boldsymbol{P}}_{(t-1)}$ to get $\tilde{\boldsymbol{y}}_t$. Recall that $\tilde{\boldsymbol{y}}_t = \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{b}_t$ where $\boldsymbol{b}_t := \boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$ is small. Using the incoherence (denseness) assumption and span($\boldsymbol{P}_{(t-1)}$) being a good estimate of span($\boldsymbol{P}_{(t)}$), it can be argued that the restricted isometry constant (RIC) [4] of $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$ will be small. Using [4, Theorem 1.2], this, along with $\|\boldsymbol{b}_t\|$ being small, ensures that l_1 minimization will produce an accurate estimate, $\hat{\boldsymbol{x}}_{t,cs}$, of $\boldsymbol{x}_t$. The support estimation step with a carefully chosen threshold, $\omega_{supp} = x_{\min}/2$, and a lower bound on $x_{\min}$ then ensures exact support recovery, i.e., $\hat{\mathcal{T}}_t = \mathcal{T}_t$. With this, the LS step output, $\hat{\boldsymbol{x}}_t$, satisfies

$\hat{\boldsymbol{x}}_t = \boldsymbol{x}_t + \boldsymbol{e}_t$ with

$$\begin{aligned} \boldsymbol{e}_t &:= \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'(\boldsymbol{\ell}_t + \boldsymbol{v}_t) \\ &= \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t) \end{aligned} \tag{2.10}$$

and with $\|\boldsymbol{e}_t\|$ being small. Computing $\hat{\boldsymbol{\ell}}_t := \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$, then also gives a good estimate of $\boldsymbol{\ell}_t$ that satisfies $\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t + \boldsymbol{v}_t - \boldsymbol{e}_t$ with $\boldsymbol{e}_t$ as above.

The subspace update step uses $\hat{\boldsymbol{\ell}}_t$'s to update the subspace. Since $\boldsymbol{e}_t$ satisfies (2.10), $\boldsymbol{e}_t$ depends on $\boldsymbol{\ell}_t$; thus the error/noise, $\boldsymbol{v}_t - \boldsymbol{e}_t$, in the "observed data" $\hat{\boldsymbol{\ell}}_t$ used for the subspace update step depends on the true data $\boldsymbol{\ell}_t$. Because of this, the subspace update does not involve a PCA or an incremental PCA problem in the traditionally studied setting (data and corrupting noise/error being independent or uncorrelated). It is, in fact, an instance of PCA when the noise/error, $\boldsymbol{v}_t - \boldsymbol{e}_t$, in the observed data $\hat{\boldsymbol{\ell}}_t$ depends on the true data $\boldsymbol{\ell}_t$. This problem was studied in [26, 27] where it was referred to as "correlated-PCA" or "PCA in data-dependent noise". Using this terminology, our subspace update problem (estimating $\boldsymbol{P}_j$ using $\hat{\boldsymbol{P}}_{j-1}$) is a problem of PCA in data-dependent noise with partial subspace knowledge. To simplify our analysis, we first study this more general problem and obtain a guarantee for it in Theorem 2.7 in Sec. 2.5.1. This theorem along with Lemma 2.15 (that analyzes the projected-CS step discussed above) help obtain a guarantee for the k-th projection-SVD step in Lemma 2.16. The $k = 1$ and $k > 1$ cases are handled separately. The main assumption required for applying Theorem 2.7 holds because $\boldsymbol{e}_t$ is sparse with support $\mathcal{T}_t$ that changes enough (max-outlier-frac-row$^{\alpha}$ bound of Theorem 2.2 holds). The subspace deletion via simple SVD step of subspace update is studied in Lemma 2.17. This step solves a problem of PCA in data-dependent noise and so it directly uses the results from [27].

To understand the flow of the proof, consider the interval $[t_j, t_{j+1})$. Assume that, before t_j, the previous subspace has been estimated with error $\tilde{\varepsilon}$, i.e., we have $\hat{\boldsymbol{P}}_{j-1}$ with $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \tilde{\varepsilon}$. We explain below that this implies that, under the theorem's assumptions, we get $\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \tilde{\varepsilon}$ before t_{j+1}. We remove the subscripts j in some of this discussion. Define the interval $\mathcal{J}_k := [t_j + (k-1)\alpha, t_j + k\alpha)$. Suppose also that $\boldsymbol{v}_t = 0$.

1. Before the first projection-SVD step (which is done at $t = t_j + \alpha$), i.e., for $t \in \mathcal{J}_1$, we have no estimate of $\boldsymbol{P}_{\text{new}}$, and hence only a crude estimate of $\boldsymbol{P}_{\text{rot}}$. In particular, we can only get the bound $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{\text{rot}}) = \text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{\text{rot}}) \leq \tilde{\varepsilon} + |\sin\theta|$ for this interval.

   - As a result, the bound on the "noise", $\boldsymbol{b}_t$, seen by the projected-CS step is also the largest for this interval, we have $\|\boldsymbol{b}_t\| \leq C(\tilde{\varepsilon}\sqrt{r\lambda^+} + |\sin\theta|\sqrt{\lambda_{\text{ch}}})$. Using the CS guarantee, followed by ensuring exact support recovery (as explained above), this implies that $\boldsymbol{e}_t$ satisfies (2.10) and that we get a similar bound on the final CS step error: $\|\boldsymbol{e}_t\| \leq C(\tilde{\varepsilon}\sqrt{r\lambda^+} + 0.11|\sin\theta|\sqrt{\lambda_{\text{ch}}})$. The factor of 0.11 in the second term of this bound is obtained because, for this interval, $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}'$ and so $\boldsymbol{\Psi}\boldsymbol{P}_{\text{new}} \approx \boldsymbol{P}_{\text{new}}$ and $\boldsymbol{P}_{\text{new}}$ is dense, see (2.13). Thus one can show that $\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{P}_{\text{new}}\|_2 \leq 0.11$.

   - This bound on $\boldsymbol{e}_t$, along with using the critical fact that $\boldsymbol{e}_t$ satisfies (2.10) (is sparse) and its support $\mathcal{T}_t$ changes enough (the max-outlier-frac-row$^\alpha$ bound of Theorem 2.2 holds), ensures that we get a better estimate of $\boldsymbol{P}_{\text{rot}}$ after the first projection-SVD step. This is what allows us to apply Theorem 2.7. Using it we can show that $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{\text{rot}}) = \text{SE}([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},1}], \boldsymbol{P}_{\text{rot}}) \leq 0.1\tilde{\varepsilon} + 0.06|\sin\theta|$ for $t \in \mathcal{J}_2$. See proof of $k = 1$ case of Lemma 2.16 and Fact 2.14.

2. Thus we have a much better estimate of $\boldsymbol{P}_{\text{rot}}$ for $t \in \mathcal{J}_2$ than for $\mathcal{J}_1$. Because of this, $\|\boldsymbol{b}_t\|$ is smaller, and hence $\|\boldsymbol{e}_t\|$ is smaller for $t \in \mathcal{J}_2$. This, along with the sparsity and changing support, $\mathcal{T}_t$, of $\boldsymbol{e}_t$, ensures an even better estimate at the second projection-SVD step. We can show that $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{\text{rot}}) = \text{SE}([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},2}], \boldsymbol{P}_{\text{rot}}) \leq 0.1\tilde{\varepsilon} + 0.5 \cdot 0.06|\sin\theta|$ for $t \in \mathcal{J}_3$. See proof of $k > 1$ case of Lemma 2.16 and Fact 2.14.

3. Proceeding this way, we show that $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{\text{rot}}) = \text{SE}([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},k}], \boldsymbol{P}_{\text{rot}}) \leq 0.1\tilde{\varepsilon} + 0.5^{k-2}(0.06|\sin\theta|)$ after the k-th projection-SVD step. Picking K appropriately, gives $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{\text{rot}}) \leq \tilde{\varepsilon}$ after K steps, i.e., at $t = t_j + K\alpha$. In all the above intervals, $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) \leq \tilde{\varepsilon} + \text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{\text{rot}})$. Thus at $t = t_j + K\alpha$, $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) \leq 2\tilde{\varepsilon}$.

4. At $t = t_j + K\alpha$, $\hat{\boldsymbol{P}}_{(t)}$ contains $(r+1)$ columns. The subspace re-estimation via simple SVD step re-estimates $\boldsymbol{P}_j$ in order to delete the deleted direction, $\boldsymbol{P}_{\text{del}}$, from $\hat{\boldsymbol{P}}_{(t)}$. The output of this step is $\hat{\boldsymbol{P}}_j$ (the final estimate of $\text{span}(\boldsymbol{P}_j)$). Thus, at $t = t_j + K\alpha + \alpha$, $\hat{\boldsymbol{P}}_{(t)} = \hat{\boldsymbol{P}}_j$ and we can show that it satisfies $\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) = \text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \le \tilde{\varepsilon}$. See Lemma 2.17. The re-estimation is done at this point because, for times t in this interval, $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\| = \|\boldsymbol{e}_t\| \le 2.4\tilde{\varepsilon}\|\boldsymbol{\ell}_t\|$. For PCA in data-dependent noise, simple SVD needs $\alpha \ge (q/\epsilon)^2 f^2 (r \log n)$ where q is the error/noise to signal ratio and ϵ is the final desired error level. For our problem, the "noise" is $\boldsymbol{e}_t$ and thus $q = 2.4\tilde{\varepsilon}$ and $\epsilon = \tilde{\varepsilon}$. Since q/ϵ is a constant, $\alpha \ge \alpha_* = Cf^2 r \log n$ suffices when simple SVD is applied at this time.

When $\boldsymbol{v}_t \neq 0$, almost all of the above discussion remains the same. The reason is this: in the main theorem, we assume $\|\boldsymbol{v}_t\|^2 \le 0.1 r \tilde{\varepsilon}^2 \lambda^+$ with $c < 1$ and so even though we have to deal with $\boldsymbol{v}_t$ in $\boldsymbol{b}_t$ and $\boldsymbol{e}_t$ expressions, and in the α expression, the changes required are only to the numerical constants. In Corollary 2.3, we have carefully chosen the bound on $\|\boldsymbol{v}_t\|$ to equal the bound on $\|\boldsymbol{e}_t\|$ modulo constants. Thus, once again, only numerical constants change, everything else remains the same.

2.4.2 Why automatic subspace change detection and Automatic Simple-ReProCS works

The subspace change detection approach is summarized in Algorithm 4. This idea is motivated by a similar idea first used in our earlier works [16, 34]. The algorithm toggles between the "detect" phase and the "update" phase. It starts in the "detect" phase. If the j-th subspace change is detected at time t, we set $\hat{t}_j = t$. At this time, the algorithm enters the "update" (subspace update) phase. We then repeat the K projection-SVD steps and the one subspace re-estimation via simple SVD step from Algorithm 1 with the following change: the k-th projection-SVD step is now done at $t = \hat{t}_j + k\alpha - 1$ (instead of at $t = t_j + k\alpha - 1$) and the subspace re-estimation is done at $t = \hat{t}_j + K\alpha + \alpha - 1 := \hat{t}_{j,fin}$. Thus, at $t = \hat{t}_{j,fin}$, the subspace update is complete. At this time, the algorithm enters the "detect" phase again. To understand the change detection

strategy, consider the j-th subspace change. Assume that the previous subspace $\boldsymbol{P}_{j-1}$ has been accurately estimated by $t = \hat{t}_{j-1,fin}$ and that $\hat{t}_{j-1,fin} < t_j$. Let $\hat{\boldsymbol{P}}_* := \hat{\boldsymbol{P}}_{j-1}$ denote this estimate. At this time, the algorithm enters the "detect" phase in order to detect the next (j-th) change. Let $\boldsymbol{B}_t := (\boldsymbol{I} - \hat{\boldsymbol{P}}_* \hat{\boldsymbol{P}}_*{}')[\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \ldots, \hat{\boldsymbol{\ell}}_t]$. For every $t = \hat{t}_{j-1,fin} + u\alpha$, $u = 1, 2, \ldots$, we detect change by checking if the maximum singular value of $\boldsymbol{B}_t$ is above a pre-set threshold, $\sqrt{\omega_{evals}\alpha}$, or not.

We claim that, whp, under Theorem 2.2 assumptions, this strategy has no false detects and correctly detects change within a delay of at most 2α frames. The former is true because, for any t for which $[t - \alpha + 1, t] \subseteq [\hat{t}_{j-1,fin}, t_j)$, all singular values of the matrix $\boldsymbol{B}_t$ will be close to zero (will be of order $\sqrt{\tilde{\varepsilon}}$) and hence its maximum singular value will be below $\sqrt{\omega_{evals}\alpha}$. Thus, whp, $\hat{t}_j \geq t_j$. To understand why the change *is* correctly detected within 2α frames, first consider $t = \hat{t}_{j-1,fin} + \left\lceil \frac{t_j - \hat{t}_{j-1,fin}}{\alpha} \right\rceil \alpha := t_{j,*}$. Since we assumed that $\hat{t}_{j-1,fin} < t_j$ (the previous subspace update is complete before the next change), t_j lie in the interval $[t_{j,*} - \alpha + 1, t_{j,*}]$. Thus, not all of the $\boldsymbol{\ell}_t$'s in this interval will be generated from span($\boldsymbol{P}_j$). Thus, depending on where in the interval t_j lies, the algorithm may or may not detect the change at this time. However, in the *next* interval, i.e., for $t \in [t_{j,*} + 1, t_{j,*} + \alpha]$, all of the $\boldsymbol{\ell}_t$'s will be generated from span($\boldsymbol{P}_j$). We can prove that, whp, $\boldsymbol{B}_t$ for this time t *will* have maximum singular value that is above the threshold. Thus, if the change is not detected at $t_{j,*}$, whp, it *will* get detected at $t_{j,*} + \alpha$. Hence one can show that, whp, either $\hat{t}_j = t_{j,*}$, or $\hat{t}_j = t_{j,*} + \alpha$, i.e., $t_j \leq \hat{t}_j \leq t_j + 2\alpha$. To see the actual proof of these claims, please refer to Appendix 2.9 where we prove our main result without assuming t_j known.

2.5 Proving Theorem 2.2 with assuming $\hat{t}_j = t_j$

In Table 2.4, we summarize all the new symbols and terms used in our proof. This, along with Table 2.3 given earlier, should help follow the proof details without having to refer back to earlier sections. To give a simpler proof first, we prove Theorem 2.2 under the assumption that $\hat{t}_j = t_j$ below. The proof without this assumption is given in Appendix 2.9. With assuming $\hat{t}_j = t_j$, we are studying Algorithm 1. Recall from Sec. 2.4 that the subspace update step involves solving a

Table 2.3: List of Symbols and Assumptions used in the Main Result 2.2, and Corollary 2.3. (Note: We show that whp, $\hat{t}_j \geq t_j$ and $\hat{t}_j + (K+1)\alpha \leq t_{j+1}$ and hence, whp, $\mathcal{J}_0, \mathcal{J}_{K+2}$ are non-empty intervals.

Observations: $\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{x}_t + \boldsymbol{v}_t$, where, $\boldsymbol{\ell}_t = \boldsymbol{P}_{(t)}\boldsymbol{a}_t = \begin{bmatrix}\boldsymbol{P}_{j-1,\text{fix}} \boldsymbol{P}_{j,\text{rot}}\end{bmatrix} \begin{bmatrix}\boldsymbol{a}_{t,\text{fix}} \\ \boldsymbol{a}_{t,\text{ch}}\end{bmatrix}$ for $t \in [t_j, t_{j+1})$, $\mathcal{T}_t$ is support of $\boldsymbol{x}_t$.

Subspace Change		**Principal Subspace Coefficients, $\boldsymbol{a}_t$'s**	
$\boldsymbol{P}_{(t)} = \boldsymbol{P}_{(t_j)} = \boldsymbol{P}_j$ for all $t \in [t_j, t_{j+1}),\ j = 0, 1, \ldots, J$		element-wise bounded, zero mean,	
$\boldsymbol{P}_{\text{ch}} \equiv \boldsymbol{P}_{j-1,\text{ch}}$	Changing direction from span$(\boldsymbol{P}_{j-1})$ at t_j	mutually independent with identical covariance (See Sec. 2.2.2)	
$\boldsymbol{P}_{\text{fix}} \equiv \boldsymbol{P}_{j-1,\text{fix}}$	Fixed directions from span$(\boldsymbol{P}_{j-1})$ at t_j	$\mathbb{E}[\boldsymbol{a}_t\boldsymbol{a}_t'] := \boldsymbol{\Lambda} = \begin{bmatrix}\boldsymbol{\Lambda}_{\text{fix}} & \boldsymbol{0} \\ \boldsymbol{0} & \lambda_{\text{ch}}\end{bmatrix}$	
$\boldsymbol{P}_{\text{new}} \equiv \boldsymbol{P}_{j-1,\text{new}}$	New direction from span$(\boldsymbol{P}_{j-1,\perp})$ added at t_j	λ^+	$\lambda_{\max}(\boldsymbol{\Lambda})$
$\boldsymbol{P}_{\text{rot}} \equiv \boldsymbol{P}_{j-1,\text{rot}}$	Rotated version of $\boldsymbol{P}_{\text{ch}}$	λ^-	$\lambda_{\min}(\boldsymbol{\Lambda})$
$\text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j) = \text{SE}(\boldsymbol{P}_{j-1,\text{ch}}, \boldsymbol{P}_{j,\text{rot}}) \leq \Delta$		$f := \lambda^+/\lambda^-$	Condition Number of $\boldsymbol{\Lambda}$
$\boldsymbol{P}_{j,\text{new}} := \frac{(\boldsymbol{I} - \boldsymbol{P}_{j-1,\text{ch}}\boldsymbol{P}_{j-1,\text{ch}}')\boldsymbol{P}_{j,\text{rot}}}{\text{SE}(\boldsymbol{P}_{j-1,\text{ch}}, \boldsymbol{P}_{j,\text{rot}})}$			
See (2.5) for equivalent generative model.			
$\max_j \max_i \lVert \text{basis}([\boldsymbol{P}_{j-1}, \boldsymbol{P}_j])^i \rVert \leq \sqrt{\frac{\mu(r+1)}{n}}$ which implies (2.13) holds.			

Outliers		**Intervals for j-th subspace change and tracking**	
$x_{\min,t} := \min_{i \in \mathcal{T}_t} \lvert(\boldsymbol{x}_t)_i\rvert$	Min. outlier magnitude at t	$\mathcal{J}_0 := [t_j, \hat{t}_j)$	interval before change detected
$x_{\min} := \min_t x_{\min,t}$	Min. outlier magnitude	$\mathcal{J}_k := [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$	k-th subspace update interval
$s :=$ max-outlier-frac-col $\cdot\, n$	Cardinality of support set of $\boldsymbol{x}_t$	$\mathcal{J}_{K+1} := [\hat{t}_j + K\alpha, \hat{t}_j + (K+1)\alpha)$	SVD-re-estimation interval
max-outlier-frac-row$^\alpha \leq \rho_{\text{row}}$	See (2.8)	$\mathcal{J}_{K+2} := [\hat{t}_j + (K+1)\alpha, t_{j+1})$	Final interval
$\rho_{\text{row}} = 0.01/f^2$			
max-outlier-frac-col $\leq \rho_{\text{col}} = \frac{0.01}{2\mu(r+1)}$	See Theorem 2.2		

problem of PCA in data-dependent noise when partial subspace knowledge is available. We provide a guarantee for this problem in Sec. 2.5.1 and use it in our proof in Sec. 2.5.4.

For the entire proof we will use the equivalent subspace change model described in (2.5). Clearly $|\sin\theta_j| \leq \Delta$ by our assumption.

2.5.1 PCA in data-dependent noise with partial subspace knowledge

The Problem. We are given a set of α frames of observed data $\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{z}_t$, with $\lVert\boldsymbol{z}_t\rVert^2 \leq b_z^2$ and $\lVert\mathbb{E}[\boldsymbol{z}_t\boldsymbol{z}_t']\rVert \leq \lambda_z^+ := c_1 b_z^2/r$; $\boldsymbol{w}_t = \boldsymbol{M}_t\boldsymbol{\ell}_t$, $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t$ and with $\boldsymbol{P}$ satisfying

$$\boldsymbol{P} = [\boldsymbol{P}_{\text{fix}}, \boldsymbol{P}_{\text{rot}}], \text{ where } \boldsymbol{P}_{\text{rot}} := (\boldsymbol{P}_{\text{ch}} \cos\theta + \boldsymbol{P}_{\text{new}} \sin\theta),$$

Table 2.4: List of symbols and their associated meaning for understanding the proof of Theorems 2.2 and 2.7. The complete definitions can be found in Definitions 2.12 and 2.20. We also provide the location of the proof for each of events/scalars where applicable in parenthesis.

Symbol	Meaning
Preliminaries (Stated in Definitions 2.12 and 2.20)	
$\theta := \theta_j$	Angle of j-th subspace change
$\boldsymbol{P}_* := \boldsymbol{P}_{j-1}$, $\boldsymbol{P}_{\text{new}} := \boldsymbol{P}_{j,\text{new}}$, $\boldsymbol{P}_{\text{ch}} := \boldsymbol{P}_{j-1,\text{ch}}$, $\boldsymbol{P}_{\text{fix}} := \boldsymbol{P}_{j-1,\text{fix}}$	Parts of the j-th subspace
$\boldsymbol{P}_{\text{rot}} := \boldsymbol{P}_{j,\text{rot}} := (\boldsymbol{P}_{\text{ch}} \cos\theta + \boldsymbol{P}_{\text{new}} \sin\theta)$, $\boldsymbol{P} := \boldsymbol{P}_j$	
$\hat{\boldsymbol{P}}_* := \hat{\boldsymbol{P}}_{j-1}$, $\hat{\boldsymbol{P}} := \hat{\boldsymbol{P}}_j$	Estimates of j-th subspace
$\hat{\boldsymbol{P}}_{\text{rot},k} := \hat{\boldsymbol{P}}_{j,\text{rot},k}$	k-th estimate of $\boldsymbol{P}_{\text{rot}}$.
$\boldsymbol{e}_t := \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$ (Proved in Lemma 2.15)	Expression for error, $\boldsymbol{e}_t = \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}} + \boldsymbol{v}_t$
Scalars (Derived in Fact 2.14)	
$\zeta_0^+ := \tilde{\varepsilon} + \lvert\sin\theta\rvert$	Bound on $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}_*)$, i.e., before subspace update
$\zeta_1^+ := 0.4 \cdot 1.2((0.1 + \tilde{\varepsilon})\lvert\sin\theta\rvert + \tilde{\varepsilon}) + 0.11\tilde{\varepsilon}$	Bound on $\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},1}], \boldsymbol{P}_{\text{rot}})$, i.e., after 1st subspace update
$\zeta_k^+ := 0.4 \cdot (1.2\zeta_{k-1}^+) + 0.11\tilde{\varepsilon}$	Bound on $\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}], \boldsymbol{P}_{\text{rot}})$, i.e., after k-th subspace update
Events – t_j known (Proved in Sec. 2.5.4)	
$\Gamma_0 := \{\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \le \tilde{\varepsilon}\}$	Previous subspace, $\boldsymbol{P}_*$ is $\tilde{\varepsilon}$-accurately estimated
$\Gamma_k := \Gamma_{k-1} \cap \{\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}], \boldsymbol{P}_{\text{rot}}) \le \zeta_k^+\}$	All k subspace update steps work
$\Gamma_{K+1} := \Gamma_K \cap \{\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le \tilde{\varepsilon}\}$	Current subspace, $\boldsymbol{P}$ is $\tilde{\varepsilon}$-accurately estimated
Events – t_j unknown (This section is only used in Appendix 2.9)	
$\hat{t}_{j-1,fin} := \hat{t}_{j-1} + (K+1)\alpha - 1$	Time at which deletion step is complete
$t_{j,*} = \hat{t}_{j-1,fin} + \left\lceil \frac{t_j - \hat{t}_{j-1,fin}}{\alpha} \right\rceil \alpha$	First possible time instant at which subspace change can be detected
$\text{Det0} := \{\hat{t}_j = t_{j,*}\}$ (Proved in Appendix 2.9)	Subspace Change detected within α-frames
$\text{Det1} := \{\hat{t}_j = t_{j,*} + \alpha\}$ (Proved in Appendix 2.9)	Subspace Change detected after α, but before 2α frames
$\text{ProjSVD}_k := \{\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}]) \le \zeta_{\text{rot},k}^+\}$ (Proved in Lemma 2.16)	k-th Proj SVD works
$\text{ProjSVD} := \cap_{k=1}^K \text{ProjSVD}_k$	All K Proj SVD steps work
$\text{Del} := \{\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le \tilde{\varepsilon}\}$ (Proved in Lemma 2.17)	Deletion Step works
NoFalseDets (Proved in Appendix 2.9)	No false detection of subspace change
$\Gamma_{0,\text{end}} := \{\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \le \tilde{\varepsilon}\}$ (Proved in Claim 2.13)	Previous subspace estimated within $\tilde{\varepsilon}$-accuracy
$\Gamma_{j,\text{end}}$ (Proved in Claim 2.13)	All previous j subspaces estimated within $\tilde{\varepsilon}$-accuracy
Notation for PCA in data-dependent noise: Theorem 2.7 (Proved in Appendix 2.10)	
$\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{z}_t$	Observations: True data - $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t = [\boldsymbol{P}_{\text{fix}}, \boldsymbol{P}_{\text{rot}}]\boldsymbol{a}_t$; Data-dep noise - $\boldsymbol{w}_t$; Modeling error - $\boldsymbol{z}_t$
$\boldsymbol{w}_t = \boldsymbol{M}_t\boldsymbol{\ell}_t = \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}\boldsymbol{\ell}_t$	Data-dependent noise,
$\lVert\boldsymbol{M}_{1,t}\boldsymbol{P}_*\rVert \le q_0$ and $\lVert\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\rVert \le q_{\text{rot}}$	Assumptions on Data-dependency matrices
$\lVert\boldsymbol{M}_{2,t}\rVert \le 1$ and $\left\lVert\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}'\right\rVert \le b_0 = 0.01$	
$\lVert\boldsymbol{z}_t\rVert \le b_z := q_0\sqrt{r\lambda^+} + q_{\text{rot}}\sqrt{\lambda_{\text{ch}}}$, $\lVert\mathbb{E}[\boldsymbol{z}_t\boldsymbol{z}_t\rVert \le \lambda_z^+ := b_z^2/r$	Assumptions on modeling error.

$\boldsymbol{P}_{\text{fix}} = (\boldsymbol{P}_*\boldsymbol{U}_0)\boldsymbol{I}_{[1,r-1]}$, $\boldsymbol{P}_{\text{ch}} = (\boldsymbol{P}_*\boldsymbol{U}_0)\boldsymbol{I}_r$, and $\boldsymbol{U}_0$ is an $r \times r$ rotation matrix. Also, the $\boldsymbol{a}_t$'s are zero mean, mutually independent, element-wise bounded r.v.'s with identical and diagonal covariance $\boldsymbol{\Lambda}$, and independent of the matrices $\boldsymbol{M}_t$. The matrices $\boldsymbol{M}_t$ are *unknown*.

Let $\mathcal{J}^\alpha$ denote the α-frame time interval for which the $\boldsymbol{y}_t$'s are available. We also have access to a partial subspace estimate $\hat{\boldsymbol{P}}_*$ that satisfies $\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \le \tilde{\varepsilon}$, and that is computed using data that is independent of the $\boldsymbol{\ell}_t$'s (and hence of the $\boldsymbol{y}_t$'s) for $t \in \mathcal{J}^\alpha$. The goal is to estimate $\text{span}(\boldsymbol{P})$ using $\hat{\boldsymbol{P}}_*$ and the $\boldsymbol{y}_t$'s for $t \in \mathcal{J}^\alpha$.

Projection-SVD / Projection-EVD. Let $\boldsymbol{\Phi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}'$. A natural way to estimate $\boldsymbol{P}$ is to first compute $\hat{\boldsymbol{P}}_{\text{rot}}$ as the top eigenvector of

$$\boldsymbol{D}_{obs} := \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\boldsymbol{y}_t\boldsymbol{y}_t{}'\boldsymbol{\Phi}.$$

and set $\hat{\boldsymbol{P}} = [\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot}}]$. We refer to this strategy as "projection-EVD" or "projection-SVD". In this paper, we are restricting ourselves to only one changed direction and hence we compute only the top eigenvector (or left singular vector) of $\boldsymbol{D}_{obs}$. In general if there were $r_{\text{ch}} > 1$ directions, we would compute all eigenvectors with eigenvalues above a threshold, see [22, 34].

The Guarantee. We can prove the following about projection-SVD.

Theorem 2.7. *Consider the above setting for an $\alpha \ge \alpha_0$ where*

$$\alpha_0 := C\eta \max(f(r\log n), \eta f^2(r + \log n)).$$

Assume that the $\boldsymbol{M}_t$'s can be decomposed as $\boldsymbol{M}_t = \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}$ where $\boldsymbol{M}_{2,t}$ is such that $\|\boldsymbol{M}_{2,t}\| \le 1$ but

$$\left\| \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}{}' \right\| \le b_0 = 0.01. \tag{2.11}$$

Let q_0 denote a bound on $\max_t \|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\|$ and let q_{rot} denote a bound on $\max_t \|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\|$, i.e., we have $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\| \le q_0$ and $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\| \le q_{\text{rot}}$ for all $t \in \mathcal{J}^\alpha$. Assume that

$$q_0 \le 2\tilde{\varepsilon},\ q_{\text{rot}} \le 0.2|\sin\theta|, \tilde{\varepsilon}f \le 0.01|\sin\theta|,\ \textit{and}$$
$$b_z \le C(q_0\sqrt{r\lambda^+} + q_{\text{rot}}\sqrt{\lambda_{\text{ch}}}) \tag{2.12}$$

Define the event $\mathcal{E}_* := \{\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \le \tilde{\varepsilon}\}$. The following hold.

1. *Conditioned on* $\mathcal{E}_*$, *w.p. at least* $1 - 12n^{-12}$,

   $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \tilde{\varepsilon} + \mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}_{\mathrm{rot}})$ *and*

$$\begin{aligned}\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}_{\mathrm{rot}}) &\leq (\tilde{\varepsilon} + |\sin\theta|)\frac{0.39 q_{\mathrm{rot}} + 0.1\tilde{\varepsilon}}{|\sin\theta|} \\ &\leq 1.01|\sin\theta|\frac{0.39 q_{\mathrm{rot}} + 0.1\tilde{\varepsilon}}{|\sin\theta|} \\ &< 0.4 q_{\mathrm{rot}} + 0.11\tilde{\varepsilon}.\end{aligned}$$

2. *Conditioned on* $\mathcal{E}_*$, *w.p. at least* $1 - 12n^{-12}$,

$$\begin{aligned}&\lambda_{\max}(\boldsymbol{D}_{obs}) \\ &\geq (0.97\sin^2\theta - 0.4 q_{\mathrm{rot}}|\sin\theta| - 0.15\tilde{\varepsilon}|\sin\theta|)\lambda_{\mathrm{ch}}.\end{aligned}$$

For large n, r, $r\log n > r + \log n$. *Thus the following simpler expression for* α_0 *suffices:* $\alpha \geq \alpha_0 = C\eta^2 f^2 (r\log n)$.

Remark 2.8. *Theorem 2.7 holds even when* $\mathcal{E}_*$ *is replaced by* $\mathcal{E}_0 := \mathcal{E}_* \cap \tilde{\mathcal{E}}(Z)$ *where* $\tilde{\mathcal{E}}(Z)$ *is an event that depends on a r.v.* Z *that is such that the pair* $\{\hat{\boldsymbol{P}}_*, Z\}$ *is still independent of the* $\boldsymbol{\ell}_t$*'s (and hence of the* $\boldsymbol{y}_t$*'s) for* $t \in \mathcal{J}^\alpha$.

Proof. The proof follows using a careful application of the Davis-Kahan $\sin\theta$ theorem [9] followed by using matrix Bernstein [24] to bound the numerator terms in the $\sin\theta$ theorem bound and Vershynin's sub-Gaussian result [29] to bound the extra terms in its denominator. While the overall approach is similar to that used by [27] for the basic correlated-PCA problem, this proof requires significantly more work. We give the proof in Appendix 2.10. The most important idea in the proof is the use of Cauchy-Schwarz to show that the time-averaged projected-data - noise correlation and time-averaged noise power are both $\sqrt{b_0}$ times their instantaneous values. We explain this next. □

In the result above, the bounds assumed in (2.12) are not critical. They only help to get a simple expression for the subspace error bound. As will be evident from the proof, we can also get

a (more complicated) guarantee without assuming (2.12), and with any value of ρ_{row}. The main assumption needed by Theorem 2.7 is (2.11) on the data-dependency matrices $\boldsymbol{M}_t$. This is required because the noise $\boldsymbol{w}_t$ depends on the true data $\boldsymbol{\ell}_t$ and hence the instantaneous values of both the noise power and of the signal-noise correlation (even after being projected orthogonal to $\hat{\boldsymbol{P}}_*$) can be large if λ_{ch} is large. However, (2.11) helps ensure that the time-averaged noise power and the time-averaged projected-signal-noise correlation are much smaller. Using the definitions of q_0 and q_{rot}, $\|\mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t{}']\| \le q_0^2\lambda^+ + q_{\text{rot}}^2\lambda_{\text{ch}} := c_w$ and $\|\mathbb{E}[\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{w}_t{}']\| \le \tilde{\varepsilon}q_0\lambda^+ + (\tilde{\varepsilon} + |\sin\theta|)q_{\text{rot}}\lambda_{\text{ch}} := c_{wl}$. By appropriately applying Cauchy-Schwarz (Theorem 2.26),

$$\left\|\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t{}']\right\| \le \sqrt{b_0}c_w \text{ and}$$

$$\left\|\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\mathbb{E}[\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{w}_t{}'\boldsymbol{\Phi}]\right\| \le \sqrt{b_0}c_{wl}.$$

Since $b_0 = 0.01$, both bounds are 0.1 times their instantaneous values. See proof of Lemma 2.24 for their derivation.

For our problem, (2.11) holds because we can let $\boldsymbol{M}_{2,t} = \boldsymbol{I}_{\mathcal{T}_t}$ and $\boldsymbol{M}_{1,t}$ as the rest of the matrix multiplying $\boldsymbol{\ell}_t$ in (2.10). Then, using the bound on outlier fractions per row from Theorem 2.2, it is not hard to see that $\left\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}{}'\right\| \le \rho_{\text{row}}$. We state this formally next in Lemma 2.9.

Lemma 2.9. *Assume that the max-outlier-frac-row$^\alpha$ bound of Theorem 2.2 holds. Then, for any α-length interval $\mathcal{J}^\alpha \subseteq [t_1, d]$,*

$$\left\|\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\right\| = \gamma(\mathcal{J}^\alpha) \le \textit{max-outlier-frac-row}^\alpha$$

$$\le \rho_{\text{row}} = 0.01/f^2 \le 0.01 = b_0.$$

Proof of Lemma 2.9. The proof is straightforward. Let $\boldsymbol{C}_t := \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'$. Then,

$$\boldsymbol{C}_t = \text{diag}\left((c_t)_1,\ (c_t)_2,\ \cdots,\ (c_t)_n\right),$$

where

$$(c_t)_i = \begin{cases} 1, & \text{if } i \in \mathcal{T}_t, \\ 0, & \text{if } i \notin \mathcal{T}_t \end{cases} = \mathbb{1}_{\{i\in\mathcal{T}_t\}}.$$

Since each $\boldsymbol{C}_t$ is diagonal, so is $\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{C}_t$. The latter has diagonal entries given by

$$\left(\frac{1}{\alpha}\sum_t \boldsymbol{C}_t\right)_{i,i} = \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}(c_t)_i = \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\mathbb{1}_{\{i\in\mathcal{T}_t\}}$$

Thus,

$$\begin{aligned}\left\|\frac{1}{\alpha}\sum_t \boldsymbol{C}_t\right\| &= \max_{i=1,2,\ldots,n}\left|\left(\frac{1}{\alpha}\sum_t \boldsymbol{C}_t\right)_{i,i}\right| \\ &= \max_{i=1,2,\ldots,n}\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\mathbb{1}_{\{i\in\mathcal{T}_t\}} = \gamma(\mathcal{J}^\alpha)\end{aligned}$$

where $\gamma(\mathcal{J}^\alpha)$ defined in (2.7) is the outlier fraction per row of $\boldsymbol{X}_{\mathcal{J}^\alpha}$. From Theorem 2.2, this is bounded by ρ_{row}. □

2.5.2 Two simple lemmas from [22]

The following two lemmas taken from [22] will be used in proving the main lemmas that together imply Theorem 2.2 with $\hat{t}_j = t_j$.

Lemma 2.10. *[22, Lemma 2.10] Suppose that $\boldsymbol{P}$, $\hat{\boldsymbol{P}}$ and $\boldsymbol{Q}$ are three basis matrices. Also, $\boldsymbol{P}$ and $\hat{\boldsymbol{P}}$ are of the same size, $\boldsymbol{Q}'\boldsymbol{P} = \boldsymbol{0}$ and $\|(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{P}\| = \zeta_*$. Then,*

1. $\|(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{P}\boldsymbol{P}'\| = \|(\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}')\hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'\| = \|(\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}')\hat{\boldsymbol{P}}\| = \|(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{P}\| = \zeta_*$
2. $\|\boldsymbol{P}\boldsymbol{P}' - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'\| \le 2\|(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{P}\| = 2\zeta_*$
3. $\|\hat{\boldsymbol{P}}'\boldsymbol{Q}\| \le \zeta_*$
4. $\sqrt{1-\zeta_*^2} \le \sigma_i\left((\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{Q}\right) \le 1$

Lemma 2.11. *[22, Lemma 3.7] For an $n \times r$ basis matrix $\boldsymbol{P}$,*

1. $\max_{|\mathcal{T}|\le s}\|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}\|^2 \le s\max_{i=1,2,\ldots,r}\|\boldsymbol{I}_i'\boldsymbol{P}\|^2$
2. $\delta_s(\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}') = \max_{|\mathcal{T}|\le s}\|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}\|^2$
3. *If $\boldsymbol{P} = [\boldsymbol{P}_1, \boldsymbol{P}_2]$ then $\|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}\|^2 \le \|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}_1\|^2 + \|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}_2\|^2$. Also, $\|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}_1\|^2 \le \|\boldsymbol{I}_{\mathcal{T}}'\boldsymbol{P}\|^2$.*

Recall that $\delta_s(\boldsymbol{M})$ is the s-restricted isometry constant [4] of a matrix $\boldsymbol{M}$, i.e., it is the smallest real number for which the following holds for all s-sparse vectors $\boldsymbol{z}$: $(1-\delta_s)\|\boldsymbol{z}\|^2 \leq \|\boldsymbol{M}\boldsymbol{z}\|^2 \leq (1+\delta_s)\|\boldsymbol{z}\|^2$.

2.5.3 Definitions and main claim needed for Theorem 2.2 and Corollary 2.3 with $\hat{t}_j = t_j$

Definition 2.12. *We will use the following definitions in our proof.*

1. *Let* $\theta := \theta_j$, $\boldsymbol{P}_* := \boldsymbol{P}_{j-1}$, $\boldsymbol{P}_{\text{new}} := \boldsymbol{P}_{j,\text{new}}$, $\boldsymbol{P}_{\text{ch}} := P_{j-1,\text{ch}}$, $\boldsymbol{P}_{\text{rot}} := \boldsymbol{P}_{j,\text{rot}} := (\boldsymbol{P}_{\text{ch}}\cos\theta + \boldsymbol{P}_{\text{new}}\sin\theta)$, $\boldsymbol{P} := \boldsymbol{P}_j$. *Similarly define* $\hat{\boldsymbol{P}}_* := \hat{\boldsymbol{P}}_{j-1}$, $\hat{\boldsymbol{P}} := \hat{\boldsymbol{P}}_j$, *and let* $\hat{\boldsymbol{P}}_{\text{rot},k} := \hat{\boldsymbol{P}}_{j,\text{rot},k}$ *denote the k-th estimate of* $\boldsymbol{P}_{\text{rot}}$ *with* $\hat{\boldsymbol{P}}_{\text{rot},0} = [.]$.

2. *The scalars*

   $\zeta_0^+ := \tilde{\varepsilon} + |\sin\theta|$,

   $\zeta_1^+ := 0.4 \cdot 1.2((0.1+\tilde{\varepsilon})|\sin\theta| + \tilde{\varepsilon}) + 0.11\tilde{\varepsilon}$ *and*

   $\zeta_k^+ := 0.4 \cdot (1.2\zeta_{k-1}^+) + 0.11\tilde{\varepsilon}$ *for* $k = 2, 3, \ldots, K$.

   We will show that these are high probability bounds on $\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}], \boldsymbol{P}_{\text{rot}})$.

3. *The events*

   $\Gamma_0 := \{\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \leq \tilde{\varepsilon}\}$*: clearly* Γ_0 *implies that* $\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_{\text{rot}}) \leq \zeta_0^+ := \tilde{\varepsilon} + |\sin\theta|$,

   $\Gamma_k := \Gamma_{k-1} \cap \{\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}], \boldsymbol{P}_{\text{rot}}) \leq \zeta_k^+\}$ *for* $k = 1, 2, \ldots, K$, *and*

   $\Gamma_{K+1} := \Gamma_K \cap \{\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \tilde{\varepsilon}\}$.

4. *The time intervals:*

   $\mathcal{J}_k := [t_j + (k-1)\alpha, t_j + k\alpha)$ *for* $k = 1, 2, \ldots, K$*: the projection-SVD intervals,*

   $\mathcal{J}_{K+1} := [t_j + K\alpha, t_j + (K+1)\alpha)$*: the subspace re-estimation interval,*

   $\mathcal{J}_{K+2} := [t_j + K\alpha + \alpha, t_{j+1})$*: the interval when the current subspace update is complete and before the next change.*

We first prove the SE bounds of Theorem 2.2. With these, the other bounds follow easily. To obtain the SE bounds, we will be done if we prove the following claim.

Claim 2.13. *Given* $\mathrm{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \le \tilde{\varepsilon}$, *w.p. at least* $1 - (K+1)12n^{-12}$,

1. $\mathrm{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_{\mathrm{rot}}) \le \zeta_0^+$, $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot},1}], \boldsymbol{P}_{\mathrm{rot}}) \le \zeta_1^+$,
2. *for* $k > 1$, $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot},k}], \boldsymbol{P}_{\mathrm{rot}}) \le \zeta_k^+$,
3. *and so* $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot},K}], \boldsymbol{P}_{\mathrm{rot}}) \le \zeta_K^+ \le \tilde{\varepsilon}$ *(using definition of* K*) and* $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot},K}], \boldsymbol{P}) \le 2\tilde{\varepsilon}$.
4. *Further, after the deletion step,* $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le \tilde{\varepsilon}$.

Proving the above claim is equivalent to showing that $\Pr(\Gamma_{K+1}|\Gamma_0) \ge 1 - (K+1)12n^{-12}$.

This claim is an easy consequence of the three main lemmas and Fact 2.14 given below. Fact 2.14 provides simple upper bounds on ζ_k^+ that will be used at various places. The first lemma, Lemma 2.15, shows that, assuming that the "subspace estimates so far are good enough", the projected CS step "works" for the next α frames, i.e., for all $t \in \mathcal{J}_k$, $\hat{\mathcal{T}}_t = \mathcal{T}_t$; $\boldsymbol{e}_t$ is sparse and supported on $\mathcal{T}_t$ and satisfies (2.10), and $\|\boldsymbol{e}_t\|$ is bounded. The second lemma, Lemma 2.16, uses Lemma 2.15 and Theorem 2.7 to show that, assuming that the "subspace estimates so far are good enough", with high probability (whp), the subspace estimate at the next projection-SVD step is even better than the previous ones. Applying Lemma 2.16 for each $k = 1, 2, \ldots, K$ proves the first two parts of Claim 2.13. The third part follows easily from the first two and the definition of K. The fourth part follows using Lemma 2.17, which shows that, assuming that the K-th projection-SVD step produces a subspace estimate that is within $2\tilde{\varepsilon}$ of the true subspace, the subspace re-estimation step returns an estimate that is within $\tilde{\varepsilon}$ of the true subspace.

Fact 2.14. *Using* $\tilde{\varepsilon} \le \tilde{\varepsilon} f \le 0.01|\sin\theta|$,

1. $\zeta_0^+ := \tilde{\varepsilon} + |\sin\theta| \le 1.01|\sin\theta|$,
2. $\zeta_1^+ := 0.4 \cdot 1.2((0.1 + \tilde{\varepsilon})|\sin\theta| + \tilde{\varepsilon}) + 0.11\tilde{\varepsilon} \le 0.06|\sin\theta|$,
3. $\zeta_k^+ := 0.4 \cdot 1.2\zeta_{k-1}^+ + 0.11\tilde{\varepsilon} \le 0.5^{k-1}\zeta_1^+ + \frac{0.11}{1-0.5}\tilde{\varepsilon} \le 0.5^{k-1}(0.06|\sin\theta|) + 0.11\tilde{\varepsilon} \le 0.03|\sin\theta|$

This claim essentially implies Theorem 2.2 and Corollary 2.3 with $\hat{t}_j = t_j$. We prove these without this assumption in Appendix 2.9.

2.5.4 The three main lemmas needed to prove the main claim and their proofs

Lemma 2.15 (Projected CS)**.** *Recall from Sec. 2.5 that s is an upper bound on $|\mathcal{T}_t|$. Under assumptions of Theorem 2.2 or or Corollary 2.3, the following hold for $k = 1, 2, \ldots, K+2$. Let $\boldsymbol{\Psi}_1 := \boldsymbol{I} - \hat{\boldsymbol{P}}_* \hat{\boldsymbol{P}}_*{}'$, $\boldsymbol{\Psi}_k := \boldsymbol{I} - \hat{\boldsymbol{P}}_* \hat{\boldsymbol{P}}_*{}' - \hat{\boldsymbol{P}}_{\mathrm{rot},k-1} \hat{\boldsymbol{P}}_{\mathrm{rot},k-1}{}'$ for $k = 2, 3, \ldots, K+1$, and $\boldsymbol{\Psi}_{K+2} := \boldsymbol{I} - \hat{\boldsymbol{P}} \hat{\boldsymbol{P}}'$. From Algorithm 1,*

$$\boldsymbol{\Psi} = \boldsymbol{\Psi}_k \quad \text{for} \quad t \in \mathcal{J}_k, \ \ k = 1, 2, \ldots, K+2.$$

Assume that Γ_{k-1} holds. Then,

1. $\max_{|\mathcal{T}| \le 2s} \|\boldsymbol{I}_{\mathcal{T}}{}' \hat{\boldsymbol{P}}_*\| \le 0.3 + \tilde{\varepsilon} \le 0.31.$
2. $\max_{|\mathcal{T}| \le 2s} \|\boldsymbol{I}_{\mathcal{T}}{}' \hat{\boldsymbol{P}}_{\mathrm{rot},k-1}\| \le 0.1 + \tilde{\varepsilon} + \frac{\zeta_{k-1}^+ + \tilde{\varepsilon}}{|\sin\theta|} \le 0.1 + 0.01 + 0.04 < 0.15.$
3. $\delta_{2s}(\boldsymbol{\Psi}_1) \le 0.31^2 < 0.12$, $\delta_{2s}(\boldsymbol{\Psi}_k) \le 0.31^2 + 0.15^2 < 0.12$ *for* $k = 2, 3, \ldots, K+2$,
4. *for all* $t \in \mathcal{J}_k$, $\|(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\| \le 1.2$
5. *for all* $t \in \mathcal{J}_k$, $\hat{\mathcal{T}}_t = \mathcal{T}_t$
6. *for all* $t \in \mathcal{J}_k$, $\boldsymbol{e}_t := \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t + \boldsymbol{v}_t$ *satisfies* (2.10) *with* $\boldsymbol{\Psi} = \boldsymbol{\Psi}_k$ *for* $t \in \mathcal{J}_k$
7. *for* $t \in \mathcal{J}_1$, $\|\boldsymbol{e}_t\| \le 2.4\sqrt{\eta}(\tilde{\varepsilon}\sqrt{r\lambda^+} + 0.11\Delta\sqrt{\lambda_{\mathrm{ch}}})$;
   for $t \in \mathcal{J}_k$, $\|\boldsymbol{e}_t\| \le 2.4\sqrt{\eta}(\tilde{\varepsilon}\sqrt{r\lambda^+} + \zeta_{k-1}^+\sqrt{\lambda_{\mathrm{ch}}})$ *for* $k = 2, 3, \ldots K$;
   for $t \in \mathcal{J}_{K+1}$, $\|\boldsymbol{e}_t\| \le 4.8\sqrt{\eta}\tilde{\varepsilon}\sqrt{r\lambda^+}$.
   for $t \in \mathcal{J}_{K+2}$, $\|\boldsymbol{e}_t\| \le 2.4\sqrt{\eta}\tilde{\varepsilon}\sqrt{r\lambda^+}$.

Proof. Since the noise bound of Theorem 2.2 is much smaller or equal to those assumed by Corollary 2.3, if we can prove the latter, we would have also proved the former. Using the first claim of Lemma 2.11, the max-outlier-frac-col bound of Theorem 2.2 and the incoherence assumption (2.6) imply that, for any set $\mathcal{T}$ with $|\mathcal{T}| \le 2s$,

$$\|\boldsymbol{I}_{\mathcal{T}}{}' \boldsymbol{P}_*\| \le 0.1 < 0.3 \text{ and } \|\boldsymbol{I}_{\mathcal{T}}{}' \boldsymbol{P}_{\mathrm{new}}\| \le 0.1 \tag{2.13}$$

(In order to simplify our assumptions, we have simplified the incoherence/denseness assumption from what it was in the original version of this work; as a result, even the first term above is bounded by 0.1 (not 0.3 as before). To not have to change the rest of the proof given below, we still use the 0.3 bound in the writing below). Using (2.13), for any set $\mathcal{T}$ with $|\mathcal{T}| \leq 2s$,

$$\begin{aligned}
\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_*\| &\leq \|\boldsymbol{I}_{\mathcal{T}}{}'(\boldsymbol{I} - \boldsymbol{P}_*\boldsymbol{P}_*{}')\hat{\boldsymbol{P}}_*\| + \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_*\boldsymbol{P}_*{}'\hat{\boldsymbol{P}}_*\| \\
&\leq \|(\boldsymbol{I} - \boldsymbol{P}_*\boldsymbol{P}_*{}')\hat{\boldsymbol{P}}_*\| + \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_*\| \\
&= \|(\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}')\boldsymbol{P}_*\| + 0.3 \\
&\leq \tilde{\varepsilon} + 0.3 \leq 0.31. \qquad (2.14)
\end{aligned}$$

The second row used (2.13) and the following: since $\hat{\boldsymbol{P}}_*$ and $\boldsymbol{P}_*$ have the same dimension, $\mathrm{SE}(\boldsymbol{P}_*, \hat{\boldsymbol{P}}_*) = \mathrm{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*)$ (follows by Lemma 2.10, item 1). The third row follows using the definition of event Γ_{k-1}. Proceeding similarly for $\hat{\boldsymbol{P}}_{\mathrm{rot},k-1}$ and $\boldsymbol{P}_{\mathrm{new}}$ (both have the same dimension),

$$\begin{aligned}
\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{\mathrm{rot},k-1}\| &\leq \|(\boldsymbol{I} - \boldsymbol{P}_{\mathrm{new}}\boldsymbol{P}_{\mathrm{new}}{}')\hat{\boldsymbol{P}}_{\mathrm{rot},k-1}\| + \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_{\mathrm{new}}\| \\
&= \|(\boldsymbol{I} - \hat{\boldsymbol{P}}_{\mathrm{rot},k-1}\hat{\boldsymbol{P}}_{\mathrm{rot},k-1}{}')\boldsymbol{P}_{\mathrm{new}}\| + 0.1 \\
&\leq \|(\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}' - \hat{\boldsymbol{P}}_{\mathrm{rot},k-1}\hat{\boldsymbol{P}}_{\mathrm{rot},k-1}{}')\boldsymbol{P}_{\mathrm{new}}\| \\
&\quad + \|\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\mathrm{new}}\| + 0.1 \\
&\leq \frac{\zeta_{k-1}^{+} + \tilde{\varepsilon}}{|\sin\theta|} + \tilde{\varepsilon} + 0.1 \leq 0.04 + 0.01 + 0.1 \\
&\leq 0.15.
\end{aligned}$$

The second row used item 1 of Lemma 2.10 and (2.13). The third row used triangle inequality. The last row follows using $\boldsymbol{P}_{\mathrm{new}} = \frac{\boldsymbol{P}_{\mathrm{rot}} - \boldsymbol{P}_{\mathrm{ch}}\cos\theta}{\sin\theta}$ and the definition of event Γ_{k-1}. Using this, triangle inequality and $|\cos\theta| \leq 1$, we bound the first term. Using item 3 of Lemma 2.10, we bound the second term. The final bounds use Fact 2.14.

To get the above bound, we use $\boldsymbol{P}_{\mathrm{new}}$ (and not $\boldsymbol{P}_{\mathrm{rot}}$) because $\|\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\mathrm{new}}\| \leq \tilde{\varepsilon}$ since $\boldsymbol{P}_*$ is orthogonal to $\boldsymbol{P}_{\mathrm{new}}$. But we do not have a small upper bound on $\|\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\mathrm{rot}}\|$.

The third claim follows using the first two claims and Lemma 2.11. The fourth claim follows from the third claim as follows:

$$\left\|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\right\| \leq \frac{1}{1-\delta_s(\boldsymbol{\Psi})} \leq \frac{1}{1-\delta_{2s}(\boldsymbol{\Psi})} \leq \frac{1}{1-0.12} < 1.2.$$

The last three claims follow the approach of the proof of [22, Lemma 6.4]. There are minor differences because we set ξ a little differently now and because we assume $\boldsymbol{v}_t \neq 0$. We provide the proof in Appendix 2.13. □

Lemma 2.16 (Projection-SVD)**.** *Under the assumptions of Theorem 2.2 or Corollary 2.3, the following holds for $k = 1, 2, \ldots K$. Conditioned on Γ_{k-1}, w.p. at least $1 - 12n^{-12}$, $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot},k}], \boldsymbol{P}_{\mathrm{rot}}) \leq \zeta_k^+$, i.e., Γ_k holds.*

Proof. Since the noise bound of Theorem 2.2 is much smaller or equal to those assumed by Corollary 2.3, if we can prove the latter, we would have also proved the former.

Assume that Γ_{k-1} holds. The proof first uses Lemma 2.15 to get an expression for $\boldsymbol{e}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t + \boldsymbol{v}_t$ and then applies Theorem 2.7 with the modification given in Remark 2.8. Using Lemma 2.15, for all $t \in \mathcal{J}_k$,

$$\begin{aligned}\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t - \boldsymbol{e}_t + \boldsymbol{v}_t &= \boldsymbol{\ell}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t) + \boldsymbol{v}_t \\ &:= \boldsymbol{\ell}_t - \boldsymbol{e}_{l,t} - \boldsymbol{e}_{v,t} + \boldsymbol{v}_t\end{aligned}$$

where $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}' - \hat{\boldsymbol{P}}_{\mathrm{rot},k-1}\hat{\boldsymbol{P}}_{\mathrm{rot},k-1}{}'$ with $\hat{\boldsymbol{P}}_{\mathrm{rot},0} = [.]$.

In the k-th projection-SVD step, we use these $\hat{\boldsymbol{\ell}}_t$'s and $\hat{\boldsymbol{P}}_*$ to get a new estimate of $\boldsymbol{P}_{\mathrm{rot}}$ using projection-SVD. To bound $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot},k}], \boldsymbol{P}_{\mathrm{rot}})$, we apply Theorem 2.7 (Remark 2.8)[11] with $\mathcal{E}_0 \equiv \Gamma_{k-1}$, $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv -\boldsymbol{e}_{l,t}$, $\boldsymbol{z}_t \equiv -\boldsymbol{e}_{v,t} + \boldsymbol{v}_t$, $\alpha \geq \alpha_0 \equiv \alpha_*$, and $\mathcal{J}^\alpha \equiv \mathcal{J}_k$. We can let $\boldsymbol{M}_{2,t} = -\boldsymbol{I}_{\mathcal{T}_t}$ which implies $b_0 \equiv \text{max-outlier-frac-row}^\alpha$ and $\boldsymbol{M}_{1,t} = (\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}$. Using the max-outlier-frac-row$^\alpha$ bound of Theorem 2.2 and Lemma 2.9, the main assumption needed by

[11] We use Remark 2.8 with $\mathcal{E}_* \equiv \Gamma_0$, $Z \equiv \{\hat{\boldsymbol{\ell}}_1, \hat{\boldsymbol{\ell}}_2, \ldots, \hat{\boldsymbol{\ell}}_{t_j+(k-1)\alpha-1}\}$, and $\tilde{\mathcal{E}}(Z) = \Gamma_{k-1} \setminus \Gamma_0$.

Theorem 2.7, (2.11), holds. With $\boldsymbol{P} = \boldsymbol{P}_j$ satisfying (2.5), and α_* defined in (2.9), all the key assumptions of Theorem 2.7 hold. The simpler expression of α_* suffices because we treat η as a numerical constant and so $f^2(r\log n) > f^2(r+\log n)$ for large n, r.

We now just need to compute q_0 and q_{rot} for each k, ensure that they satisfy (2.12), and apply the result. The computation for $k=1$ is different from the rest. When $k=1$, $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}'$. Thus, using item 4 of Lemma 2.15 and the definition of event Γ_{k-1}, $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\| \le 1.2\tilde{\varepsilon} = q_0$, $q_0 < 2\tilde{\varepsilon}$, and

$$\begin{aligned}\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\| &\le 1.2(\|\boldsymbol{I}_{\mathcal{T}_t}{}'(\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}')\boldsymbol{P}_{\text{new}}\||\sin\theta| + \tilde{\varepsilon}) \\ &\le 1.2(\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{P}_{\text{new}}\| + \|\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\text{new}}\|)|\sin\theta| + 1.2\tilde{\varepsilon} \\ &\le 1.2((0.1+\tilde{\varepsilon})|\sin\theta| + \tilde{\varepsilon}) = q_{\text{rot}}.\end{aligned}$$

The third row follows using (2.13) and $\|\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\text{new}}\| \le \tilde{\varepsilon}$ (folows by item 3 of Lemma 2.10). Using $\tilde{\varepsilon} \le \tilde{\varepsilon} f \le 0.01|\sin\theta|$, clearly $q_{\text{rot}} < 0.2|\sin\theta|$. Finally, in this interval, the bound on b_z is satisfied since $b_z = b_{v,t}$ and the expression for $b_{v,t}$ in $\mathcal{J}_1$ is equal to the required upper bound on b_z. Applying Theorem 2.7, $\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},1}], \boldsymbol{P}_{\text{rot}}) \le 0.4q_{\text{rot}} + 0.11\tilde{\varepsilon} = 0.4\cdot 1.2((0.1+\tilde{\varepsilon})|\sin\theta| + \tilde{\varepsilon}) + 0.11\tilde{\varepsilon} = \zeta_1^+$.

Consider $k > 1$. Now $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}' - \hat{\boldsymbol{P}}_{\text{rot},k-1}\hat{\boldsymbol{P}}_{\text{rot},k-1}{}'$. With this, we still have $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\| \le 1.2\tilde{\varepsilon} = q_0$ and $q_0 < 2\tilde{\varepsilon}$. But, to bound $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\|$ we cannot use the approach that worked for $k=1$. The reason is that $\|[\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k-1}]'\boldsymbol{P}_{\text{new}}\|$ is not small. However, instead, we can now use the fact that $[\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k-1}]$ is a good estimate of $\boldsymbol{P}_{\text{rot}}$, with $\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k-1}], \boldsymbol{P}_{\text{rot}}) \le \zeta_{k-1}^+$ (from definition of event Γ_{k-1}). Thus,

$$\begin{aligned}\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\| &\le 1.2\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k-1}], \boldsymbol{P}_{\text{rot}}) \\ &\le 1.2\zeta_{k-1}^+ = q_{\text{rot}}.\end{aligned} \tag{2.15}$$

By Fact 2.14, $q_{\text{rot}} < 0.2|\sin\theta|$. Even in this interval, the required bound on b_z holds. Thus, applying Theorem 2.7, $\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}], \boldsymbol{P}_{\text{rot}}) \le 0.4q_{\text{rot}} + 0.11\tilde{\varepsilon} = 0.4\cdot 1.2\zeta_{k-1}^+ + 0.11\tilde{\varepsilon} = \zeta_k^+$. □

Lemma 2.17 (Simple SVD based subspace re-estimation)**.** *Under the assumptions of Theorem 2.2 or Corollary 2.3, the following holds. Conditioned on* Γ_K*, w.p. at least* $1-12n^{-12}$*,* $\mathrm{SE}(\hat{\boldsymbol{P}},\boldsymbol{P}) \le \tilde{\varepsilon}$*, i.e.,* Γ_{K+1} *holds.*

Proof. Assume that Γ_K holds. Using Lemma 2.15, for all $t \in \mathcal{J}_{K+1}$,

$$\begin{aligned}\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t - \boldsymbol{e}_t + \boldsymbol{v}_t &= \boldsymbol{\ell}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t+\boldsymbol{v}_t)+\boldsymbol{v}_t \\ &:= \boldsymbol{\ell}_t - \boldsymbol{e}_{l,t} - \boldsymbol{e}_{v,t} + \boldsymbol{v}_t\end{aligned}$$

where $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}' - \hat{\boldsymbol{P}}_{\mathrm{rot},K}\hat{\boldsymbol{P}}_{\mathrm{rot},K}{}'$. Re-estimating the entire subspace using simple SVD applied to these $\hat{\boldsymbol{\ell}}_t$'s is an instance of correlated-PCA with $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv -\boldsymbol{e}_{l,t}$ and $\boldsymbol{z}_t \equiv -\boldsymbol{e}_{v,t}+\boldsymbol{v}_t$. We can apply the following result for correlated-PCA [27, Theorem 2.13] to bound $\mathrm{SE}(\hat{\boldsymbol{P}},\boldsymbol{P})$. Recall $\hat{\boldsymbol{P}}$ contains the top r eigenvectors of $\sum_{t\in\mathcal{J}_{K+1}}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'$. The following is a simplified version of [27, Theorem 2.13]. It follows by upper bounding $\lambda_{z,\boldsymbol{P},\boldsymbol{P}_\perp}$ and $\lambda^+_{z,rest}$ by λ^+_z and lower bound $\lambda^-_{z,\boldsymbol{P}}$ by zero in [27, Theorem 2.13].

Theorem 2.18. *For* $t \in \mathcal{J}^\alpha$*, we are given data vectors* $\boldsymbol{y}_t := \boldsymbol{\ell}_t+\boldsymbol{w}_t+\boldsymbol{z}_t$ *where* $\boldsymbol{w}_t = \boldsymbol{M}_t\boldsymbol{\ell}_t$*,* $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t$ *and* $\boldsymbol{z}_t$ *is small unstructured noise. Let* $\hat{\boldsymbol{P}}$ *be the matrix of top* r *eigenvectors of* $\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{y}_t\boldsymbol{y}_t{}'$*. Assume that* $\boldsymbol{M}_t$ *can be decomposed as* $\boldsymbol{M}_t = \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}$ *so that* $\|\boldsymbol{M}_{2,t}\| \le 1$ *but* $\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}{}'\| \le b$ *for a* $b<1$*. Let* q *be an upper bound on* $\max_{t\in\mathcal{J}^\alpha}\|\boldsymbol{M}_{1,t}\boldsymbol{P}\|$*. We assume that* $\|\boldsymbol{z}_t\| \le b_z$ *and define* $\|\mathbb{E}[\boldsymbol{z}_t\boldsymbol{z}_t{}']\| \le \lambda^+_z := b_z^2/r$*. For an* $\varepsilon_{\mathrm{SE}} > 0$*, define*

$$\begin{aligned}\alpha_0 := C\eta\max\Bigg(& f^2(r\log n)\frac{q^2}{\varepsilon^2_{\mathrm{SE}}}, \\ & \frac{\lambda^+_z q^2}{\lambda^-\varepsilon^2_{\mathrm{SE}}} fr(\log n), \eta f^2(r\log 9 + 10\log n)\Bigg).\end{aligned}$$

If $\alpha \ge \alpha_0$*, and* $3\sqrt{b}qf + \frac{\lambda^+_z}{\lambda^-} \le 0.46\varepsilon_{\mathrm{SE}}$*, then, w.p. at least* $1-12n^{-12}$*,* $\mathrm{SE}(\hat{\boldsymbol{P}},\boldsymbol{P}) \le \varepsilon_{\mathrm{SE}}$*. .*

Apply the above result with $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv -\boldsymbol{e}_{l,t}$, $\boldsymbol{z}_t \equiv -\boldsymbol{e}_{v,t}+\boldsymbol{v}_t$, $\alpha \ge \alpha_*$, and $\mathcal{J}^\alpha \equiv \mathcal{J}_{K+1}$. From the expression for $\boldsymbol{e}_t$, we can let $\boldsymbol{M}_{2,t} \equiv -\boldsymbol{I}_{\mathcal{T}_t}$, $\boldsymbol{M}_{1,t} \equiv (\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}$. Next we compute q. Since Γ_K holds, $\mathrm{SE}([\hat{\boldsymbol{P}}_*,\hat{\boldsymbol{P}}_{\mathrm{rot},K}],\boldsymbol{P}) \le \tilde{\varepsilon}+\zeta^+_K \le 2\tilde{\varepsilon}$. Thus, $\|\boldsymbol{M}_{1,t}\boldsymbol{P}\| \le 1.2\mathrm{SE}([\hat{\boldsymbol{P}}_*,\hat{\boldsymbol{P}}_{\mathrm{rot},K}],\boldsymbol{P}) \le 1.2{\cdot}2\tilde{\varepsilon} = q$. The final desired error is $\varepsilon_{\mathrm{SE}} = \tilde{\varepsilon}$. Using Lemma 2.9 and the max-outlier-frac-row$^\alpha$ bound from

Theorem 2.2, the bound on the time-average of $\boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}'$ holds with $b \equiv \rho_{\text{row}} = \frac{0.01}{f^2} < \frac{0.5^2}{(3\cdot 2.4f)^2}$. Also, in this interval $b_z^2 = Cr\lambda^+$ and so $\lambda_z = C\lambda^+$ and thus the second term in the α_0 expression equals the first term. The third term can of course be ignored in the large n, r regime. Applying the above result with $\varepsilon_{\text{SE}} = \tilde{\varepsilon}$, and $q = 2.4\tilde{\varepsilon}$, we conclude the following: for $\alpha \geq \alpha_*$, w.p. at least $1 - 12n^{-12}$, $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \tilde{\varepsilon}$. The simpler expression of α_* suffices because η is treated as a numerical constant and so $f^2(r \log n) > f^2(r + \log n)$ for large n, r. Also under the assumption of Corollary 2.3, the second term is dominated by the first term. □

Proof of Claim 2.13. Lemma 2.16 tells us that $\Pr(\Gamma_k|\Gamma_{k-1}) \geq 1 - 12n^{-12}$. Lemma 2.17 tells us that $\Pr(\Gamma_{K+1}|\Gamma_K) \geq 1 - 12n^{-12}$. Thus, $\Pr(\Gamma_{K+1}|\Gamma_0) = \Pr(\Gamma_{K+1}, \Gamma_K, \ldots \Gamma_1|\Gamma_0) = \Pr(\Gamma_1|\Gamma_0)\Pr(\Gamma_2|\Gamma_1)\ldots\Pr(\Gamma_{K+1}|\Gamma_K) \geq (1-12n^{-12})^K(1-12n^{-12})$. since $\Gamma_{K+1} \subseteq \Gamma_K \subseteq \Gamma_{K-1} \cdots \subseteq \Gamma_0$. The result follows since $(1 - 12n^{-12})^K(1 - 12n^{-12}) \geq 1 - (K+1)12n^{-12}$. □

Proof of Theorem 2.2 with $\hat{t}_j = t_j$. Define the events $\Gamma_{1,0} := \{\text{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \leq \tilde{\varepsilon}\}$, $\Gamma_{j,k} := \Gamma_{j,k-1} \cap \{\text{SE}([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j,\text{rot},k}]) \leq \zeta_k^+\}$, for $k = 1, 2, \ldots, K$, $\Gamma_{j,K+1} := \Gamma_{j,K} \cap \{\text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \tilde{\varepsilon}\}$ and $\Gamma_{j+1,0} := \Gamma_{j,K+1}$. We can state and prove Lemmas 2.16 and 2.17 with Γ_k replaced by $\Gamma_{j,k}$. Then Claim 2.13 implies that $\Pr(\Gamma_{j,K+1}|\Gamma_{j,0}) \geq 1 - 12n^{-12}$. Using $\Gamma_{J,K+1} \subseteq \Gamma_{J-1,K+1} \cdots \subseteq \Gamma_{1,K+1} \subseteq \Gamma_{1,0}$ and $\Gamma_{j+1,0} := \Gamma_{j,K+1}$, $\Pr(\Gamma_{J,K+1}|\Gamma_{1,0}) = \Pr(\Gamma_{J,K+1}, \Gamma_{J-1,K+1}, \ldots \Gamma_{1,K+1}|\Gamma_{1,0}) = \Pr(\Gamma_{1,K+1}|\Gamma_{1,0})\Pr(\Gamma_{2,K+1}|\Gamma_{2,0})\ldots\Pr(\Gamma_{J,K+1}|\Gamma_{J,0}) \geq (1-(K+1)12n^{-12})^J \geq 1-J(K+1)12n^{-12} \geq 1 - dn^{-12}$.

Event $\Gamma_{J,K+1}$ implies that $\Gamma_{j,k}$ holds for all j and for all k. Thus, all the SE bounds given in Theorem 2.2 hold. Using Lemma 2.15, $\hat{\mathcal{T}}_t = \mathcal{T}_t$ for all the time intervals of interest, and the bounds on $\|\boldsymbol{e}_t\|$ hold. □

2.6 Empirical Evaluation

In this section we illustrate the superiority of s-ReProCS over existing state of the art methods on synthetic and real data. In particular, we consider the task of background subtraction. All time comparisons are performed on a Desktop Computer with Intel® Xeon E3-1240 8-core

CPU @ 3.50GHz and 32GB RAM. And all experiments with synthetic data are averaged over 100 independent trials. All codes are available at https://github.com/praneethmurthy/ReProCS.

Similar experiments have been shown in the earlier ReProCS works (original-ReProCS) [22, 11, 16, 34]. The purpose of this section is to illustrate that, even though s-ReProCS is much simpler, is provably faster and memory efficient, and provably works under much simpler assumptions, its experimental performance is still similar to that of original-ReProCS. It outperforms existing works for the same classes of videos and simulated data for which original-reprocs outperforms them.

2.6.1 Synthetic Data

Our first simulation experiment is done to illustrate the advantage of s-ReProCS over existing batch and online RPCA techniques. As explained earlier, because s-ReProCS exploits dynamics (slow subspace change), it is provably able to tolerate a much larger fraction of outliers per row than all the existing techniques without needing uniformly randomly generated support sets. When the number of subspace changes, J, is large, it also tolerates a significantly larger fraction of outliers per column. The latter is hard to demonstrate via simulations (making J large will require a very long sequence). Thus we demonstrate only the former. Our second experiment shows results with using an i.i.d. Bernoulli model on support change (which is the model assumed in the other works).

One practical instance where outlier fractions per row can be larger than those per column is in the case of video moving objects that are either occasionally static or slow moving [16, 34]. The outlier support model for our first and second experiments is inspired by this example and the model used in [16, 34]. It models a 1D video consisting of a person/object of length s pacing up and down in a room with frequent stops. The object is static for β frames at a time and then moves down. It keeps moving down for a period of τ frames, after which it turns back up and does the same thing in the other direction. We let $\beta = \lceil c_0 \tau \rceil$ for a $c_0 < 1$. With this model, for any interval of the form $[(k_1 - 1)\tau + 1, k_2 \tau]$ for k_1, k_2 integers, the outlier fraction per row is bounded by c_0. For any general interval of length $\alpha \geq \tau$, this max-outlier-frac-row$^\alpha$ is still bounded by $2c_0$ while max-outlier-frac-col is bounded by s/n.

Model 2.19. *Let $\beta = \lceil c_0\tau \rceil$. Assume that the $\mathcal{T}_t$ satisfies the following. For the first τ frames (downward motion),*

$$\mathcal{T}_t = \begin{cases} [1,\ s], & t \in [1, \beta] \\ [s+1,\ 2s], & t \in [\beta+1, 2\beta] \\ \vdots & \\ [(1/c_0 - 1)s + 1,\ s/c_0], & t \in [\tau - \beta + 1, \tau] \end{cases}$$

for the next τ frames (upward motion), $\mathcal{T}_t =$

$$\begin{cases} [(1/c_0 - 1)s + 1,\ s/c_0], & t \in [\tau + 1,\ \tau + \beta] \\ [(1/c_0 - 2)s + 1,\ (1/c_0 - 1)s], & t \in [\tau + \beta + 1, \tau + 2\beta] \\ \vdots & \\ [1,\ s], & t \in [2\tau - \beta + 1, 2\tau]. \end{cases}$$

Starting at $t = 2\tau + 1$, the above pattern is repeated every 2τ frames until the end, $t = d$.

This model is motivated by the model assumed for the guarantees in older works [16, 34]. The above model is one practically motivated way to simulate data that is not not generated uniformly at random (or as i.i.d. Bernoulli, which is approximately the same as the uniform model for large n). It also provides a way to generate data with a different bounds on outlier fractions per row and per column. The maximum outlier fraction per column is s/n. For any time interval of length $\alpha \geq \tau$, the outlier fraction per row is bounded by $2c_0$. Thus, for Theorem 2.2, with this model, $\rho_{\text{row}} = 2c_0/f^2$. By picking $2c_0$ larger than s/n we can ensure larger outlier fractions per row than per column.

We compare Algorithm 4 and its offline counterpart with three of the batch methods with provably guarantees discussed in Sec. 2.2.3 - PCP [5], AltProj [20] and RPCA-GD [33] - and with two recently proposed online algorithms known to have good experimental performance and for which code was available - ORPCA [10] and GRASTA [12]. The code for all these techniques are cloned from the Low-Rank and Sparse library (`https://github.com/andrewssobral/lrslibrary`).

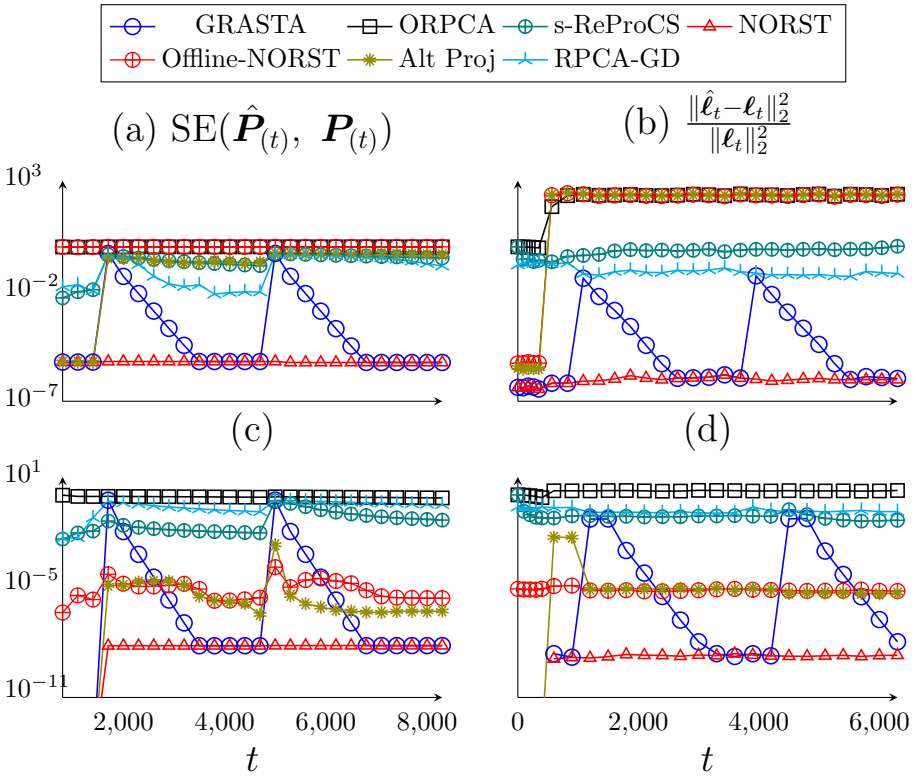


Figure 2.2: First row ((a), (b)): Illustrate the subspace error and the normalized $\boldsymbol{\ell}_t$ error for $n = 5000$ and outlier supports generated using Model 2.19. Both the metrics are plotted every $k\alpha - 1$ time-frames. The results are averaged over 100 iterations. Second row ((c), (d)) illustrate the subspace error and the normalized $\boldsymbol{\ell}_t$ error for $n = 500$ and Bernoulli outlier support model. They are plotted every $k\alpha - 1$ time-frames. The plots clearly corroborates the nearly-exponential decay of the subspace error as well as the error in $\boldsymbol{\ell}_t$.

For generating data we used $d = 8000$, $t_{\text{train}} = 500$, $J = 2$, $r = 5$, $f = 16$ with $t_1 = 1000$, $\theta_1 = 30°$, $t_2 = 4300$, $\theta_2 = 1.01\theta_1$ and varying n. The $\boldsymbol{a}_t$'s are zero mean i.i.d uniform random variables generated exactly as described before and so is $\boldsymbol{P}_{(t)}$. We generated a basis matrix $\boldsymbol{Q}$ by ortho-normalizing the first $r+2$ columns of a $n \times n$ i.i.d. standard Gaussian matrix. For $t \in [1,\ t_1)$, we set $\boldsymbol{P}_{(t)} = \boldsymbol{P}_0$ with $\boldsymbol{P}_0$ being the first r columns of $\boldsymbol{Q}$. We let $\boldsymbol{P}_{1,\text{new}}$ be $(r+1)$-th column of $\boldsymbol{Q}$, and rotated it in using (2.5) with $\boldsymbol{U}_1 = \boldsymbol{I}$ and with angle θ_1 to get $\boldsymbol{P}_1$. We set $\boldsymbol{P}_{(t)} = \boldsymbol{P}_1$ for $t \in [t_1, t_2)$. We set $\boldsymbol{P}_{2,\text{new}}$ to be the last column of $\boldsymbol{Q}$, $\boldsymbol{U}_2 = \boldsymbol{I}$, and rotate using angle θ_2 just as done in the first subspace change and finally for $t \in [t_2, d]$ we set $\boldsymbol{P}_{(t)} = \boldsymbol{P}_2$. At all times t, we let $\boldsymbol{\ell}_t = \boldsymbol{P}_{(t)}\boldsymbol{a}_t$

Table 2.5: Average subspace error SE($\hat{\boldsymbol{P}}_{(t)},\ \boldsymbol{P}_{(t)}$) and time comparison for different values of signal size n. The values in brackets denote average time taken per frame (– indicates that the algorithm does not work).

	ReProCS	GRASTA	ORPCA	**Offline ReProCS**	PCP	AltProj	RPCA-GD
$n = 500$ (in $10^{-4}s$)	**0.066** (**3.1**)	0.996 (2.8)	0.320 (10)	$\mathbf{8.25 \times 10^{-5}}$ (**6.3**)	1.00 (51)	0.176 (104)	0.215 (454)
$n = 500$, Bern. (in $10^{-4}s$)	**0.044** (**4.8**)	0.747 (1.9)	0.078 (1.8)	$\mathbf{3.9 \times 10^{-7}}$ (**9.2**)	1.2×10^{-4} (395)	0.0001 (32)	0.303 (329)
$n = 5000$ (in $10^{-2}s$)	**0.048** (**3.7**)	0.999 (0.11)	0.322 (0.30)	$\mathbf{6.05 \times 10^{-5}}$ (**8.5**)	0.999 (8.9)	0.354 (13.0)	0.223 (47.0)
$n = 10,000$ (in $10^{-2}s$)	**0.090** (**15.6**)	0.999 (0.25)	0.3235 (0.68)	**0.0006** (**36.8**)	–	–	–

with $\boldsymbol{a}_t$ being zero mean, i.i.d uniform random variables such that $(\boldsymbol{a}_t)_i \sim unif\left(-\sqrt{f},\ \sqrt{f}\right)$ for $i = 1,\ \cdots,\ r-2$ and $(\boldsymbol{a}_t)_{[r-1,r]} \sim unif(-1, 1)$. With this the condition number is f, the covariance matrix, $\Lambda = \text{diag}(f, f, \cdots, f, 1, 1)/3$, $\lambda^+ = f/3$, $\lambda_{\text{ch}} = \lambda^- = 1/3$, and $\eta = 3$. We generate $\mathcal{T}_t$ using Model 2.19 as follows. For $t \in [t_{\text{train}}, d]$, we used $s = 0.1n$, $c_0 = 0.2$ and $\tau = 100$. Thus $\rho_{\text{row}} = 0.4/f^2$. For $t \in [1, t_{\text{train}}]$, we used $s = 0.05n$ and $c_0 = 0.02$. This was done to ensure that AltProj (or any other batch technique works well for this period and provides a good initialization). The magnitudes of the nonzero entries of $\boldsymbol{x}_t$ (outliers) were generated s i.i.d uniform r.v.'s between $x_{\min} = 10$ and $x_{\max} = 25$.

We implemented Algorithm 4 for s-ReProCS (with initialization using AltProj) with $\alpha = Cf^2 r \log n = 500$, $K = \lceil -0.8 \log(0.9\tilde{\varepsilon}) \rceil = 5$, $\omega_{supp} = x_{\min}/2$, and $\omega_{evals} = 0.0025\lambda^-$. We initialized using AltProj applied to $\boldsymbol{Y}_{[1,t_{\text{train}}]}$. For the batch methods used in the comparisons – PCP, AltProj and RPCA-GD, we implement the algorithms on $\boldsymbol{Y}_{[1,t]}$ every $t = t_{\text{train}} + k\alpha - 1$ frames. Further, we set the regularization parameter for PCP $\lambda = 1/\sqrt{n}$ in accordance with [5]. The other known parameters, r for Alt-Proj, outlier-fraction for RPCA-GD, are set using the true values. For online methods we implement the algorithms without modifications. The regularization parameter for ORPCA was set as with $\lambda_1 = 1/\sqrt{n}$ and $\lambda_2 = 1/\sqrt{d}$ according to [10]. We plot the subspace error and the normalized error of $\boldsymbol{\ell}_t$ over time in Fig. 2.2(a) and 2.2(b) for $n = 5000$. We display the time-averaged error for other values of n in Table 2.5. This table also contains the time comparisons.

As can be seen, s-ReProCS outperforms all the other methods and offline s-ReProCS significantly outperforms all the other methods for this experiment. The reason is that the outlier fraction per row are quite large, but s-ReProCS exploits slow subspace change. In principle, even GRASTA

exploits slow subspace change, however, it uses approximate methods for computing the SVD and does not use projection-SVD and hence it fails. s-ReProCS and offline s-ReProCS are faster than all the batch methods especially for large n. In fact when $n = 10000$, the batch methods are out of memory and cannot work, while s-ReProCS still can. But s-ReProCS is slower than GRASTA and ORPCA.

Comparison with other algorithms - random outlier support using the i.i.d. Bernoulli model. We generated data exactly as described above with the following change: $\mathcal{T}_t$ was now generated as i.i.d. Bernoulli with probability of any index i being in $\cup_{t\in[1,n]}\mathcal{T}_t$ being $\rho_s = 0.02$ for the first t_{train} frames and $\rho_s = 0.2$ for the subsequent data. Notice that under the Bernoulli model, $\rho_{\text{row}} = \rho_{\text{col}} = \rho_s$. We used $n = 500$. We show the results in Fig. 2.2(c) and 2.2(d). For this experiment, the batch methods PCP and AltProj have good performance, that is better than s-ReProCS at most time instants. Offline s-ReProCS still outperforms all the other methods.

2.6.2 Real Data: Background Subtraction

In this section we provide simulation results for on real videos on three benchmark datasets. For all the sequences, to implement s-ReProCS, we obtained an estimate using the AltProj algorithm. For the initialization we set $r = 40$ and the other parameters for the proposed algorithm, were set as follows. We used $\alpha = 60$, $K = 3$, $\xi_t = \|\boldsymbol{\Psi}\hat{\boldsymbol{\ell}}_{t-1}\|_2$ and $\omega_{evals} = 0.0011\lambda^-$. We found that these parameters work for most videos that we verified our algorithm on. For a more detailed empirical evaluation on real world data-sets, please see [25]. The other state-of-the-art algorithms were implemented using the default setting. In algorithms where we are required to provide an esimate of the rank, we used $r = 40$ consistently. Additionally, for RPCA-GD we set the corruption fraction, $\alpha = 0.2$ as described in the paper. We must also mention that the performance of s-ReProCS w.r.t. original-ReProCS is very similar, but is provably fast, and needs fewer assumptions. Again, a more detailed comparison is presented in [25].

`Meeting Room (MR) dataset`: The meeting room sequence is a set of 1964 images of resolution 64×80. The first 1755 frames consists of outlier-free data and so we only consider the last 1209

frames. Here and below, we use the first 400 "noisy" frames as the training data and the algorithm parameters are set as mentioned before. This is a challenging video sequence because the color of the person and the color of the curtain are hard to distinguish. s-ReProCS algorithm is able to perform the separation at around 43 frames per second. The recovered background images are shown in the first two rows of Fig. 2.3.

`Switch Light (SL) dataset`: This dataset contains 2100 images of resolution 120×160. The first 770 frames are outlier free. This is a challenging sequence because there are drastic changes in the subspace as indicated in the last two rows of Fig. 2.3. This causes all the batch techniques to fail. For this sequence, s-ReProCS achieves a "test" processing rate of 16 frames-per-second. The recovered background images are shown in the middle two rows of Fig. 2.3.

`Lobby (LB) dataset`: This dataset contains 1555 images of resolution 128×160. The first 341 frames are outlier free. This is a challenging sequence, as the background changes often due to illumination changes, and there are multiple objects in the foreground to detect and subtract. For this sequence, s-ReProCS achieves a "test" processing rate of 12 frames-per-second. The images are shown in the last two rows of Fig. 2.3.

2.7 Conclusions and Future Work

We obtained the first complete guarantee for any online, streaming or dynamic RPCA algorithm that holds under weakened versions of standard RPCA assumptions, slow subspace change, and outlier magnitudes are either large or very small. Our guarantee implies that, by exploiting these extra assumptions, one can significantly weaken the required bound on outlier fractions per row. This has many important practical implications especially for video analytics. We analyzed a simple algorithm based on the Recursive Projected Compressive Sensing (ReProCS) framework introduced in [22]. The algorithm itself is simpler than other previously studied ReProCS-based methods, it is provably faster, and has near-optimal memory complexity. Moreover, our guarantee removes all the strong assumptions made by the previous two guarantees for ReProCS-based methods.

As described earlier, our current result still has limitations, some of which can be removed with a little more work. For example, it assumes a very simple model on subspace change in which only one direction can change at any given change time. Of course the changing direction could be different at different change times, and hence over a long period, the entire subspace could change. In follow-up work [17], we have studied another algorithm that removes this limitation. Another issue that we would like to study is whether the lower bound on outlier magnitudes can be relaxed further if we use the stronger assumption on outlier fractions per row (assume they are of order $1/r$). It may be possible to do this by borrowing the AltProj [20] proof idea.

A question of practical and theoretical interest is to develop a streaming version of simple-ReProCS for dynamic RPCA. A preprint that studies a streaming algorithm for standard RPCA but only for the restrictive $r_{_L} = r = 1$ setting is [21]. By streaming we mean that the algorithm makes only one pass through the data and needs storage of order exactly nr. Simple-ReProCS needs only a little more storage than this, however, it makes multiple passes through the data in the SVD steps. Algorithmically, streaming ReProCS is easy to develop: one can replace the projection SVD and SVD steps in the subspace update by their streaming versions, e.g., block stochastic power method. However, in order to prove that this still works (with maybe an extra factor of $\log n$ in the delay), one would need to analyze the block stochastic power method for the problems of PCA in data-dependent noise, and for its extension that assumes availability of partial subspace knowledge. Finally, as explained in [16], any guarantee for dynamic RPCA also provides a guarantee for dynamic Matrix Completion (MC) as an almost direct corollary. The reason is that MC can be interpreted as RPCA with outlier supports $\mathcal{T}_t$ being known.

2.8 References

[1] Adali, T., and Haykin, S., Eds. *Adaptive Signal Processing: Next Generation Solutions.* Wiley & Sons, 2010.

[2] Balzano, L., Recht, B., and Nowak, R. Online Identification and Tracking of Subspaces from Highly Incomplete Information. In *Allerton Conf. Comm., Control, Comput.* (2010).

[3] Balzano, L., and Wright, S. Local convergence of an algorithm for subspace identification from partial data. *Found. Comput. Math. 15*, 5 (2015).

[4] Candes, E. The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris Serie I* (2008).

[5] Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *J. ACM 58*, 3 (2011).

[6] Chandrasekaran, V., Sanghavi, S., Parrilo, P. A., and Willsky, A. S. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization 21* (2011).

[7] Cherapanamjeri, Y., Gupta, K., and Jain, P. Nearly-optimal robust matrix completion. *ICML* (2016).

[8] Chi, Y., Eldar, Y. C., and Calderbank, R. Petrels: Parallel subspace estimation and tracking by recursive least squares from partial observations. *IEEE Transactions on Signal Processing* (December 2013).

[9] Davis, C., and Kahan, W. M. The rotation of eigenvectors by a perturbation. iii. *SIAM J. Numer. Anal. 7* (Mar. 1970), 1–46.

[10] Feng, J., Xu, H., and Yan, S. Online robust pca via stochastic optimization. In *NIPS* (2013).

[11] Guo, H., Qiu, C., and Vaswani, N. An online algorithm for separating sparse and low-dimensional signal sequences from their sum. *IEEE Trans. Sig. Proc. 62*, 16 (2014), 4284–4297.

[12] He, J., Balzano, L., and Szlam, A. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *IEEE Conf. on Comp. Vis. Pat. Rec. (CVPR)* (2012).

[13] Horn, R., and Johnson, C. *Matrix Analysis*. Cambridge Univ. Press, 1985.

[14] Hsu, D., Kakade, S. M., and Zhang, T. Robust matrix decomposition with sparse corruptions. *IEEE Trans. Info. Th.* (Nov. 2011).

[15] Li, Y., Xu, L., Morphett, J., and Jacobs, R. An integrated algorithm of incremental and robust pca. In *IEEE Intl. Conf. Image Proc. (ICIP)* (2003), pp. 245–248.

[16] Lois, B., and Vaswani, N. Online matrix completion and online robust pca. In *IEEE Intl. Symp. Info. Th. (ISIT)* (2015).

[17] Narayanamurthy, P., and Vaswani, N. Nearly optimal robust subspace tracking. *arxiv:1712.06061 under review for IEEE Trans. Info Theory* (2017).

[18] Narayanamurthy, P., and Vaswani, N. A fast and memory-efficient algorithm for robust pca (merop). In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2018), IEEE, pp. 4684–4688.

[19] Narayanamurthy, P., and Vaswani, N. Nearly optimal robust subspace tracking. In *International Conference on Machine Learning* (2018), pp. 3701–3709.

[20] Netrapalli, P., Niranjan, U. N., Sanghavi, S., Anandkumar, A., and Jain, P. Non-convex robust pca. In *NIPS* (2014).

[21] Niranjan, U. N., and Shi, Y. Streaming robust pca. *arxiv* (2016).

[22] Qiu, C., Vaswani, N., Lois, B., and Hogben, L. Recursive robust pca or recursive sparse recovery in large but structured noise. *IEEE Trans. Info. Th.* (August 2014), 5007–5039.

[23] Skocaj, D., and Leonardis, A. Weighted and robust incremental method for subspace learning. In *IEEE Intl. Conf. Comp. Vis. (ICCV)* (Oct 2003), vol. 2, pp. 1494 –1501.

[24] Tropp, J. A. User-friendly tail bounds for sums of random matrices. *Found. Comput. Math. 12*, 4 (2012).

[25] Vaswani, N., Bouwmans, T., Javed, S., and Narayanamurthy, P. Robust subspace learning: Robust pca, robust subspace tracking, and robust subspace recovery. *IEEE signal processing magazine 35*, 4 (2018), 32–55.

[26] Vaswani, N., and Guo, H. Correlated-pca: Principal components' analysis when data and noise are correlated. In *NIPS* (2016).

[27] Vaswani, N., and Narayanamurthy, P. Finite sample guarantees for pca in non-isotropic and data-dependent noise. In *Allerton 2017, long version at arXiv:1709.06255* (2017).

[28] Vaswani, N., and Narayanamurthy, P. Static and dynamic robust pca and matrix completion: A review. *Proceedings of the IEEE 106*, 8 (2018), 1359–1379.

[29] Vershynin, R. Introduction to the non-asymptotic analysis of random matrices. *Compressed sensing* (2012), 210–268.

[30] Xiao, L., and Zhang, T. A proximal-gradient homotopy method for the l1-regularized least-squares problem. In *ICML* (2012).

[31] Yang, B. Projection approximation subspace tracking. *IEEE Trans. Sig. Proc.* (1995), 95–107.

[32] Yang, B. Asymptotic convergence analysis of the projection approximation subspace tracking algorithms. *Signal Processing 50* (1996), 123–136.

[33] Yi, X., Park, D., Chen, Y., and Caramanis, C. Fast algorithms for robust pca via gradient descent. In *NIPS* (2016).

[34] Zhan, J., Lois, B., Guo, H., and Vaswani, N. Online (and Offline) Robust PCA: Novel Algorithms and Performance Guarantees. In *Intnl. Conf. Artif. Intell. Stat. (AISTATS)* (2016).

[35] Zhan, J., and Vaswani, N. Robust pca with partial subspace knowledge. *IEEE Trans. Sig. Proc.* (July 2015).

2.9 Appendix A: Proof of Theorem 2.2 or Corollary 2.3 without assuming t_j known

The key results needed for this and later proofs – Cauchy-Schwarz for sums of matrices, matrix Bernstein, and Vershynin's sub-Gaussian result – are summarized in the last appendix, Appendix 2.15. For a summary of notation used for this and later proofs, please see Table 2.4.

Here we prove Theorem 2.2 in the general case. The main idea is explained in Sec. 2.4.2. Define

$$\hat{t}_{j-1,fin} := \hat{t}_{j-1} + K\alpha + \alpha - 1,$$
$$t_{j,*} = \hat{t}_{j-1,fin} + \left\lceil \frac{t_j - \hat{t}_{j-1,fin}}{\alpha} \right\rceil \alpha$$

Thus, $\hat{t}_{j-1,fin}$ is the time at which the $(j-1)$-th subspace update is complete; whp, this occurs before t_j. Under this assumption, $t_{j,*}$ is such that t_j lies in the interval $[t_{j,*} - \alpha + 1, t_{j,*}]$. Recall from the algorithm that we increment j to $j+1$ at $t = \hat{t}_j + K\alpha + \alpha := \hat{t}_{j,fin}$. Thus, for $t \in [t_j, \hat{t}_{j,fin})$, $\boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_* \hat{\boldsymbol{P}}_*{}'$, while for $t \in [\hat{t}_{j,fin}, t_{j+1})$, $\boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'$.

Definition 2.20. *Define the events*

1. $\text{Det0} := \{\hat{t}_j = t_{j,*}\} = \{\lambda_{\max}(\frac{1}{\alpha}\sum_{t=t_{j,*}-\alpha+1}^{t_{j,*}} (\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}')\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'(\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}')) > \omega_{evals}\}$ *and*
   $\text{Det1} := \{\hat{t}_j = t_{j,*} + \alpha\} = \{\lambda_{\max}(\frac{1}{\alpha}\sum_{t=t_{j,*}+1}^{t_{j,*}+\alpha} (\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}')\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'(\boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}')) > \omega_{evals}\}$,
2. $\text{ProjSVD} := \cap_{k=1}^{K} \text{ProjSVD}_k$ *where* $\text{ProjSVD}_k := \{\text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot},k}]) \le \zeta_k^+\}$,
3. $\text{Del} := \{\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le \tilde{\varepsilon}\}$,
4. $\text{NoFalseDets} := \{$*for all* $\mathcal{J}^\alpha \subseteq [\hat{t}_{j,fin}, t_{j+1})$, $\lambda_{\max}(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} (\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')) \le \omega_{evals}\}$

5. $\Gamma_{0,\text{end}} := \{\text{SE}(\hat{\boldsymbol{P}}_*, \boldsymbol{P}_*) \le \tilde{\varepsilon}\}$,

6. $\Gamma_{j,\text{end}} := \Gamma_{j-1,\text{end}} \cap \big((\text{Det0} \cap \text{ProjSVD} \cap \text{Del} \cap \text{NoFalseDets}) \cup (\overline{\text{Det0}} \cap \text{Det1} \cap \text{ProjSVD} \cap \text{Del} \cap \text{NoFalseDets})\big)$.

Let p_0 denote the probability that, conditioned on $\Gamma_{j-1,\text{end}}$, the change got detected at $t = t_{j,*}$, i.e., let

$$p_0 := \Pr(\text{Det0}|\Gamma_{j-1,\text{end}}).$$

Thus, $\Pr(\overline{\text{Det0}}|\Gamma_{j-1,\text{end}}) = 1 - p_0$. It is not easy to bound p_0. However, as we will see, this will not be needed.

Assume that $\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}}$ holds. Consider the interval $\mathcal{J}^\alpha := [t_{j,*}, t_{j,*} + \alpha)$. This interval starts at or after t_j, so, for all t in this interval, the subspace has changed. For this interval, $\boldsymbol{\Psi} = \boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_* \hat{\boldsymbol{P}}_*{}'$. Applying the last item of Theorem 2.7, w.p. at least $1 - 12n^{-12}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right)$$
$$\ge (0.97\sin^2\theta - 0.4q_{\text{rot}}|\sin\theta| - 0.15\tilde{\varepsilon}|\sin\theta|)\lambda_{\text{ch}}$$

where q_{rot} is the bound $\|(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{I}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}\boldsymbol{P}_{\text{rot}}\|$. Theorem 2.7 is applicable for the reasons given in the proof of Lemma 2.16. Proceeding as in the proof of Lemma 2.16 for $k = 1$, we get that $q_{\text{rot}} = 1.2((0.1 + \tilde{\varepsilon})|\sin\theta| + \tilde{\varepsilon})$. Thus, using the bound on $\tilde{\varepsilon}$, we can conclude that, w.p. at least $1 - 12n^{-12}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right) \ge 0.91\sin^2\theta\lambda_{\text{ch}}$$
$$\ge 0.9\sin^2\theta\lambda^- > \omega_{evals}$$

and thus $\hat{t}_j = t_{j,*} + \alpha$. This follows since $\omega_{evals} = 5\tilde{\varepsilon}^2\lambda^+ = 5\tilde{\varepsilon}^2 f\lambda^- \le 5\tilde{\varepsilon}^2 f^2\lambda^- \le 5(0.01\min_j \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j))^2\lambda^-$ and $\sin\theta = \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$. In other words,

$$\Pr(\text{Det1}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}}) \ge 1 - 12n^{-12}.$$

Conditioned on $\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}$, the first projection-SVD step is done at $t = \hat{t}_j + \alpha = t_{j,*} + 2\alpha$ and so on. We can state and prove Lemma 2.16 with Γ_k replaced by $\Gamma_{j,\text{end}} \cap \overline{\text{Det0}} \cap$

$\text{Det1} \cap \text{ProjSVD}_1 \cap \text{ProjSVD}_2 \ldots \text{ProjSVD}_k$ and with the k-th projection-SVD interval being $\mathcal{J}_k := [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$. We can state and prove a similarly changed version of Lemma 2.17 for the simple SVD based deletion step. Applying Lemma 2.16 for each k, and then apply Lemma 2.17,

$$\Pr(\text{ProjSVD} \cap \text{Del} | \Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}) \geq (1 - 12n^{-12})^{K+1}.$$

We can also do a similar thing for the case when the change is detected at $t_{j,*}$, i.e. when Det0 holds. In this case, we replace Γ_k by $\Gamma_{j,\text{end}} \cap \text{Det0} \cap \text{ProjSVD}_1 \cap \text{ProjSVD}_2 \ldots \text{ProjSVD}_k$ and conclude that

$$\Pr(\text{ProjSVD} \cap \text{Del} | \Gamma_{j-1,\text{end}} \cap \text{Det0}) \geq (1 - 12n^{-12})^{K+1}.$$

Finally consider the NoFalseDets event. First, assume that $\Gamma_{j-1,\text{end}} \cap \text{Det0} \cap \text{ProjSVD} \cap \text{Del}$ holds. Consider any interval $\mathcal{J}^\alpha \subseteq [\hat{t}_{j,fin}, t_{j+1})$. In this interval, $\hat{\boldsymbol{P}}_{(t)} = \hat{\boldsymbol{P}}$, $\boldsymbol{\Psi} = \boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'$ and $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \tilde{\varepsilon}$. Also, using Lemma 2.15, $\boldsymbol{e}_t$ satisfies (2.10) for t in this interval. Thus, defining $\boldsymbol{e}_{l,t} = \boldsymbol{\Phi}\boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{\ell}_t$, $\boldsymbol{e}_{v,t} = \boldsymbol{\Phi}\boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{v}_t$ and $\boldsymbol{z}_t = \boldsymbol{e}_{v,t} + \boldsymbol{\Phi}\boldsymbol{v}_t$

$$\begin{aligned}
\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\boldsymbol{\Phi} &= \frac{1}{\alpha}\boldsymbol{\Phi}\boldsymbol{P}\left(\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{a}_t\boldsymbol{a}_t{}'\right)\boldsymbol{P}'\boldsymbol{\Phi} \\
&+ \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{e}_{l,t}{}' + (.)' + \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t{}' + (.)' \\
&+ \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{e}_{l,t}\boldsymbol{e}_{l,t}{}' + \frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{z}_t\boldsymbol{z}_t{}'
\end{aligned}$$

We can bound the first term using Vershynin's sub-Gaussian result (Theorem 2.28) and the other terms using matrix Bernstein (Theorem 2.27). The approach is similar to that of the proof of Lemma 2.24. The derivation is more straightforward in this case, since for the above interval $\|\boldsymbol{\Psi}\boldsymbol{P}\| = \|\boldsymbol{\Phi}\boldsymbol{P}\| \leq \tilde{\varepsilon}$. The required bounds on α are also the same as those needed for Lemma 2.24 to hold. We conclude that, w.p. at least $1 - 12n^{-12}$,

$$\begin{aligned}
&\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right) \\
&\leq \tilde{\varepsilon}^2(\lambda^+ + 0.01\lambda^+)[1 + 6\sqrt{\rho_{\text{row}}}f(1.2)^2 + 6f\sqrt{\rho_{\text{row}}}1.2] \\
&\leq 2.6\tilde{\varepsilon}^2 f\lambda^- < \omega_{evals}
\end{aligned}$$

This follows since $\omega_{evals} = 5\tilde{\varepsilon}^2\lambda^+ = 5\tilde{\varepsilon}^2 f\lambda^-$. Since Det0 holds, $\hat{t}_j = t_{j,*}$. Thus, we have a total of $\lfloor \frac{t_{j+1}-t_{j,*}-K\alpha-\alpha}{\alpha} \rfloor$ intervals $\mathcal{J}^\alpha$ that are subsets of $[\hat{t}_{j,fin}, t_{j+1})$. Moreover, $\lfloor \frac{t_{j+1}-t_{j,*}-K\alpha-\alpha}{\alpha} \rfloor \leq \lfloor \frac{t_{j+1}-t_j-K\alpha-\alpha}{\alpha} \rfloor \leq \lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+1)$ since $\alpha \leq \alpha$. Thus,

$$\begin{aligned}&\Pr(\text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \text{Det0} \cap \text{ProjSVD} \cap \text{Del}) \\ &\qquad \geq (1-12n^{-12})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+1)}\end{aligned}$$

On the other hand, if we condition on $\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1} \cap \text{ProjSVD} \cap \text{Del}$, then $\hat{t}_j = t_{j,*} + \alpha$. Thus,

$$\begin{aligned}&\Pr(\text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1} \cap \text{ProjSVD} \cap \text{Del}) \\ &\qquad \geq (1-12n^{-12})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+2)}\end{aligned}$$

We can now combine the above facts to bound $\Pr(\Gamma_{j,\text{end}}|\Gamma_{j-1,\text{end}})$. Recall that $p_0 := \Pr(\text{Det0}|\Gamma_{j-1,\text{end}})$. Clearly, the events $(\text{Det0} \cap \text{ProjSVD} \cap \text{Del} \cap \text{NoFalseDets})$ and $(\overline{\text{Det0}} \cap \text{Det1} \cap \text{ProjSVD} \cap \text{Del} \cap \text{NoFalseDets})$ are disjoint. Thus,

$$\begin{aligned}&\Pr(\Gamma_{j,\text{end}}|\Gamma_{j-1,\text{end}}) \\ &= p_0 \Pr(\text{ProjSVD} \cap \text{Del} \cap \text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \text{Det0}) \\ &+ (1-p_0) \Pr(\text{Det1}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}}) \times \\ &\Pr(\text{ProjSVD} \cap \text{Del} \cap \text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}) \\ &\geq p_0(1-12n^{-12})^{K+1}(1-12n^{-12})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+1)} \\ &+ (1-p_0)(1-12n^{-12})(1-12n^{-12})^{K+1} \times \\ &(1-12n^{-12})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+2)} \\ &= (1-12n^{-12})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor} \geq (1-12n^{-12})^{t_{j+1}-t_j}.\end{aligned}$$

Since the events $\Gamma_{j,\text{end}}$ are nested, the above implies that

$$\begin{aligned}\Pr(\Gamma_{J,\text{end}}|\Gamma_{0,\text{end}}) &= \prod_j \Pr(\Gamma_{j,\text{end}}|\Gamma_{j-1,\text{end}}) \\ &\geq \prod_j (1-12n^{-12})^{t_{j+1}-t_j} \\ &= (1-12n^{-12})^d \geq 1-12dn^{-12}.\end{aligned}$$

2.10 Appendix B: Proof of Theorem 2.7: PCA in data-dependent noise with partial subspace knowledge

We prove Theorem 2.7 with the modification given in Remark 2.8. Thus we condition on $\mathcal{E}_0$ defined in the remark. Recall that $\boldsymbol{\Phi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*'$. Let

$$\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}} \overset{\text{QR}}{=} \boldsymbol{E}_{\text{rot}}\boldsymbol{R}_{\text{rot}} \tag{2.16}$$

denote the reduced QR decomposition of $(\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}})$. Here, and in the rest of this proof, we write things in a general fashion to allow $\boldsymbol{P}_{\text{rot}}$ to contain *more* than one direction. This makes it easier to understand how our guarantees extend to the more general case ($\boldsymbol{P}_{\text{rot}}$ being an $n \times r_{\text{ch}}$ basis matrix with $r_{\text{ch}} > 1$) easier. The proof uses the following simple lemma at various places.

Lemma 2.21. *Assume that $\mathcal{E}_0$ holds. Then,*

1. $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{fix}}\| \leq q_0$, $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{ch}}\| \leq q_0$ *and* $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{rot}}\| \leq q_{\text{rot}}$
2. $\|\boldsymbol{\Phi}\boldsymbol{P}_{\text{fix}}\| \leq \tilde{\varepsilon}$, $\|\boldsymbol{\Phi}\boldsymbol{P}_{\text{ch}}\| \leq \tilde{\varepsilon}$, $\|\boldsymbol{\Phi}\boldsymbol{P}_{\text{new}}\| \leq 1$,
3. $\|\boldsymbol{R}_{\text{rot}}\| = \|\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\| \leq \tilde{\varepsilon}|\cos\theta| + |\sin\theta| \leq \tilde{\varepsilon} + |\sin\theta|$
4. $\sigma_{\min}(\boldsymbol{R}_{\text{rot}}) = \sigma_{\min}(\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}) \geq \sqrt{\sin^2\theta(1-\tilde{\varepsilon}^2) - 2\tilde{\varepsilon}|\sin\theta|}$
5. $\|\boldsymbol{\Phi}\boldsymbol{\ell}_t\| \leq 2\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + |\sin\theta|\sqrt{\eta\lambda_{\text{ch}}}$.

Proof of Lemma 2.21. Item 1 follows because $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\| \leq q_0 \leq 2\tilde{\varepsilon}$ and $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\| = \|\boldsymbol{M}_{1,t}[\boldsymbol{P}_{\text{fix}}, \boldsymbol{P}_{\text{ch}}]\| \geq \|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{fix}}\|$. Similarly, $\|\boldsymbol{M}_{1,t}\boldsymbol{P}_*\| \geq \|\boldsymbol{M}_{1,t}\boldsymbol{P}_{\text{ch}}\|$. The first two claims of item 2

follow because $\|\boldsymbol{\Phi}\boldsymbol{P}_*\| \le \tilde{\varepsilon}$ and the bound on item 1. Third claim uses $\|\boldsymbol{\Phi}\boldsymbol{P}_{\text{new}}\| \le \|\boldsymbol{\Phi}\|\|\boldsymbol{P}_{\text{new}}\| = 1$. The fourth claim uses triangle inequality and definition of $\boldsymbol{P}_{\text{rot}}$. For Item 3, recall that $\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}} \overset{\text{QR}}{=} \boldsymbol{E}_{\text{rot}}\boldsymbol{R}_{\text{rot}}$. Thus, $\sigma_i(\boldsymbol{R}_{\text{rot}}) = \sigma_i(\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}})$. Thus $\|\boldsymbol{R}_{\text{rot}}\| = \|\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\| \le \tilde{\varepsilon} + |\sin\theta|$

Item 4: From above, $\sigma_{\min}(\boldsymbol{R}_{\text{rot}}) = \sigma_{\min}(\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}})$. Moreover, $\sigma_{\min}(\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}) = \sqrt{\lambda_{\min}(\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}})} = \sqrt{\lambda_{\min}(\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}})}$. We bound this as follows. Recall that $\boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}'$.

$$
\begin{aligned}
&\lambda_{\min}(\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}) \ge \lambda_{\min}(\cos^2\theta\boldsymbol{P}_{\text{ch}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{ch}}) \\
&+ \lambda_{\min}(\sin^2\theta\boldsymbol{P}_{\text{new}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{new}}) \\
&- 2|\sin\theta||\cos\theta|\ \|\boldsymbol{P}_{\text{ch}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{new}}\| \\
&\ge 0 + \lambda_{\min}(\sin^2\theta\boldsymbol{P}_{\text{new}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{new}}) - 2\tilde{\varepsilon}|\sin\theta| \\
&= \sin^2\theta\lambda_{\min}(\boldsymbol{I} - \boldsymbol{P}_{\text{new}}{}'\hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\text{new}}) - 2\tilde{\varepsilon}|\sin\theta| \\
&= \sin^2\theta(1 - \|\boldsymbol{P}_{\text{new}}{}'\hat{\boldsymbol{P}}_*\|^2) - 2\tilde{\varepsilon}|\sin\theta| \\
&\ge \sin^2\theta(1 - \tilde{\varepsilon}^2) - 2\tilde{\varepsilon}|\sin\theta|
\end{aligned}
$$

The last inequality used Lemma 2.10.

Item 5: Using the previous items and the definition of η,

$$
\begin{aligned}
\|\boldsymbol{\Phi}\boldsymbol{\ell}_t\| &:= \|\boldsymbol{\Phi}(\boldsymbol{P}_{\text{fix}}\boldsymbol{a}_{t,\text{fix}} + \boldsymbol{P}_{\text{rot}}\boldsymbol{a}_{t,\text{ch}})\| \\
&\le \|\boldsymbol{\Phi}\boldsymbol{P}_{\text{fix}}\boldsymbol{a}_{t,\text{fix}}\| + \|\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\boldsymbol{a}_{t,\text{ch}}\| \\
&\le \left(\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + (\tilde{\varepsilon}|\cos\theta| + |\sin\theta|)\sqrt{\eta r_{\text{ch}}\lambda_{\text{ch}}}\right)
\end{aligned}
$$

□

2.11 Appendix C: Proof of Theorem 2.7

Proof of Theorem 2.7. We have

$$
\begin{aligned}
\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) &= \text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot}}], [\boldsymbol{P}_{\text{fix}}, \boldsymbol{P}_{\text{rot}}]) \\
&\le \text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot}}], \boldsymbol{P}_{\text{fix}}) + \text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot}}], \boldsymbol{P}_{\text{rot}}) \\
&\le \tilde{\varepsilon} + \text{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\text{rot}}], \boldsymbol{P}_{\text{rot}})
\end{aligned}
$$

where the last inequality used Lemma 2.21. Consider $\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot}}], \boldsymbol{P}_{\mathrm{rot}})$.

$$\begin{aligned}\mathrm{SE}([\hat{\boldsymbol{P}}_*, \hat{\boldsymbol{P}}_{\mathrm{rot}}], \boldsymbol{P}_{\mathrm{rot}}) &\leq \left\|(\boldsymbol{I} - \hat{\boldsymbol{P}}_{\mathrm{rot}}\hat{\boldsymbol{P}}_{\mathrm{rot}}{}')\boldsymbol{E}_{\mathrm{rot}}\right\| \|\boldsymbol{R}_{\mathrm{rot}}\| \\ &\leq \mathrm{SE}(\hat{\boldsymbol{P}}_{\mathrm{rot}}, \boldsymbol{E}_{\mathrm{rot}})(\tilde{\varepsilon} + |\sin\theta|) \end{aligned} \tag{2.17}$$

The last inequality used Lemma 2.21. To bound $\mathrm{SE}(\hat{\boldsymbol{P}}_{\mathrm{rot}}, \boldsymbol{E}_{\mathrm{rot}})$, we use the Davis-Kahan $\sin\theta$ theorem [9] given below.

Theorem 2.22 (Davis-Kahan $\sin\theta$ theorem)**.** *Consider $n \times n$ Hermitian matrices, $\boldsymbol{D}$ and $\hat{\boldsymbol{D}}$ such that*

$$\boldsymbol{D} = \begin{bmatrix}\boldsymbol{E}\, \boldsymbol{E}_\perp\end{bmatrix} \begin{bmatrix}\boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_{\mathrm{rest}}\end{bmatrix} \begin{bmatrix}\boldsymbol{E}' \\ \boldsymbol{E}_\perp{}'\end{bmatrix}$$

$$\hat{\boldsymbol{D}} = \begin{bmatrix}\boldsymbol{F}\, \boldsymbol{F}_\perp\end{bmatrix} \begin{bmatrix}\boldsymbol{\Lambda} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Lambda}_{\mathrm{rest}}\end{bmatrix} \begin{bmatrix}\boldsymbol{F}' \\ \boldsymbol{F}_\perp{}'\end{bmatrix}$$

where $[\boldsymbol{E}, \boldsymbol{E}_\perp]$ and $[\boldsymbol{F}, \boldsymbol{F}_\perp]$ are orthogonal matrices and $\mathrm{rank}(\boldsymbol{F}) = \mathrm{rank}(\boldsymbol{E})$. Let

$$\boldsymbol{H} = \hat{\boldsymbol{D}} - \boldsymbol{D}.$$

If $\lambda_{\min}(\boldsymbol{A}) - \lambda_{\max}(\boldsymbol{A}_{\mathrm{rest}}) - \|\boldsymbol{H}\| > 0$ and $\mathrm{rank}(\boldsymbol{E}) = \mathrm{rank}(\boldsymbol{F})$, then,

$$\left\|\left(\boldsymbol{I} - \boldsymbol{F}\boldsymbol{F}'\right)\boldsymbol{E}\right\| \leq \frac{\|\boldsymbol{H}\|}{\lambda_{\min}(\boldsymbol{A}) - \lambda_{\max}(\boldsymbol{A}_{\mathrm{rest}}) - \|\boldsymbol{H}\|}. \tag{2.18}$$

To use this result to bound $\mathrm{SE}(\hat{\boldsymbol{P}}_{\mathrm{rot}}, \boldsymbol{E}_{\mathrm{rot}})$, let $\hat{\boldsymbol{D}} := \boldsymbol{D}_{obs} = \frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{y}_t\boldsymbol{y}_t{}'\boldsymbol{\Phi}$. Its top eigenvector is $\hat{\boldsymbol{P}}_{\mathrm{rot}}$. We need to define a matrix $\boldsymbol{D}$ that is such that its top eigenvector is $\boldsymbol{E}_{\mathrm{rot}}$ and the gap between its first and second eigenvalues is more than $\|\boldsymbol{H}\|$. Consider the matrix

$$\begin{aligned}&\boldsymbol{D} := \boldsymbol{E}_{\mathrm{rot}}\boldsymbol{A}\boldsymbol{E}_{\mathrm{rot}}{}' + \boldsymbol{E}_{\mathrm{rot},\perp}\boldsymbol{A}_{\mathrm{rest}}\boldsymbol{E}_{\mathrm{rot},\perp}{}' \text{ where} \\ &\boldsymbol{A} := \boldsymbol{E}_{\mathrm{rot}}{}'\left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right)\boldsymbol{E}_{\mathrm{rot}}, \\ &\boldsymbol{A}_{\mathrm{rest}} := \boldsymbol{E}_{\mathrm{rot},\perp}{}'\left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right)\boldsymbol{E}_{\mathrm{rot},\perp}.\end{aligned}$$

If $\lambda_{\max}(\boldsymbol{A}_{\text{rest}}) < \lambda_{\min}(\boldsymbol{A})$, then $\boldsymbol{E}_{\text{rot}}$ is the top eigenvector of $\boldsymbol{D}$. Moreover, if $\lambda_{\max}(\boldsymbol{A}_{\text{rest}}) < \lambda_{\min}(\boldsymbol{A}) - \|\boldsymbol{H}\|$, then the gap requirement holds too. Thus, by the $\sin\theta$ theorem,

$$\begin{aligned} \text{SE}(\hat{\boldsymbol{P}}_{\text{rot}}, \boldsymbol{E}_{\text{rot}}) &= \left\| \left(\boldsymbol{I} - \hat{\boldsymbol{P}}_{\text{rot}} \hat{\boldsymbol{P}}_{\text{rot}}{}' \right) \boldsymbol{E}_{\text{rot}} \right\| \\ &\leq \frac{\|\boldsymbol{H}\|}{\lambda_{\min}(\boldsymbol{A}) - \lambda_{\max}(\boldsymbol{A}_{\text{rest}}) - \|\boldsymbol{H}\|} \end{aligned} \tag{2.19}$$

Here again, we should point out that, in the simple case that we consider where $\boldsymbol{P}_{\text{rot}}$ is a vector (only one direction changes), $\boldsymbol{A}$ is a non-negative scalar and $\lambda_{\min}(\boldsymbol{A}) = \boldsymbol{A}$. However the above discussion applies even in the general case when $r_{\text{ch}} > 1$. The rest of the proof obtains high probability bounds on the terms in the above expression. $\|\boldsymbol{H}\| = \left\| \boldsymbol{D} - \hat{\boldsymbol{D}} \right\|$ can be bounded as follows.

Lemma 2.23. *Let*

$\text{term11} = \frac{1}{\alpha} \sum_t \boldsymbol{E}_{\text{rot}} \boldsymbol{E}_{\text{rot}}{}' \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{\ell}_t{}' \boldsymbol{\Phi} \boldsymbol{E}_{\text{rot},\perp} \boldsymbol{E}_{\text{rot},\perp}{}'$. *Then,*

$$\begin{aligned} \|\boldsymbol{H}\| \leq\ & 2 \left\| \frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \boldsymbol{\Phi} \right\| + 2 \|\text{term11}\| \\ & + \left\| \frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{w}_t \boldsymbol{w}_t{}' \boldsymbol{\Phi} \right\| + 2 \left\| \frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{z}_t{}' \boldsymbol{\Phi} \right\| \\ & + \left\| \frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{z}_t \boldsymbol{z}_t{}' \boldsymbol{\Phi} \right\| \end{aligned}$$

Proof of Lemma 2.23. Recall that $\boldsymbol{H} = \hat{\boldsymbol{D}} - \boldsymbol{D}$. Thus

$$\begin{aligned}
\boldsymbol{H} &= \left(\hat{\boldsymbol{D}} - \frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right) + \left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi} - \boldsymbol{D}\right) \\
&= \left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{y}_t\boldsymbol{y}_t{}'\boldsymbol{\Phi} - \frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right) \\
&+ \Bigg(\left(\boldsymbol{E}_{\text{rot}}\boldsymbol{E}_{\text{rot}}{}' + \boldsymbol{E}_{\text{rot},\perp}\boldsymbol{E}_{\text{rot},\perp}{}'\right)\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\times \\
&\left(\boldsymbol{E}_{\text{rot}}\boldsymbol{E}_{\text{rot}}{}' + \boldsymbol{E}_{\text{rot},\perp}\boldsymbol{E}_{\text{rot},\perp}{}'\right) - \boldsymbol{D}\Bigg) \\
&= \left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi} + \frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{w}_t{}'\boldsymbol{\Phi}\right) \\
&+ \left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{z}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi} + \frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t{}'\boldsymbol{\Phi}\right) \\
&+ \left(\frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{w}_t{}'\boldsymbol{\Phi} + \frac{1}{\alpha}\sum \boldsymbol{\Phi}\boldsymbol{z}_t\boldsymbol{z}_t{}'\boldsymbol{\Phi}\right) \\
&+ \left(\frac{1}{\alpha}\sum \boldsymbol{E}_{\text{rot}}\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot},\perp}\boldsymbol{E}_{\text{rot},\perp}{}'\right) \\
&+ \left(\frac{1}{\alpha}\sum \boldsymbol{E}_{\text{rot},\perp}\boldsymbol{E}_{\text{rot},\perp}{}'\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}\boldsymbol{E}_{\text{rot}}{}'\right)
\end{aligned}$$

Using triangle inequality the bound follows. □

The next lemma obtains high probability bounds on the above terms and the two other terms from (2.19).

Lemma 2.24. *Assume that the assumptions of Theorem 2.7 with the modification given in Remark 2.8 hold. Let* $\epsilon_0 = 0.01|\sin\theta|(\tilde{\varepsilon} + q_{\text{rot}})$, $\epsilon_1 = 0.01(q_{\text{rot}}^2 + \tilde{\varepsilon}^2)$, *and* $\epsilon_2 = 0.01$. *For an* $\alpha \geq \alpha_0 := C\eta\max\left\{fr\log n,\ \eta f^2(r+\log n)\right\}$, *conditioned on* $\mathcal{E}_0$, *all the following hold w.p. at least* $1-12n^{-12}$*:*

1. $\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{w}_t{}'\boldsymbol{\Phi}\right\| \leq \left[\sqrt{b_0}\left(2\tilde{\varepsilon}^2 f + (\tilde{\varepsilon} + |\sin\theta|)q_{\text{rot}}\right) + \epsilon_0\right]\lambda_{\text{ch}}$,
2. $\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{w}_t{}'\boldsymbol{\Phi}\right\| \leq \left[\sqrt{b_0}\left(4\tilde{\varepsilon}^2 f + q_{\text{rot}}^2\right) + \epsilon_1\right]\lambda_{\text{ch}}$,
3. $\lambda_{\min}(\boldsymbol{A}) \geq (\sin^2\theta(1-\tilde{\varepsilon}^2) - 2\tilde{\varepsilon}|\sin\theta|)(1-\epsilon_2)\lambda_{\text{ch}} - 2\tilde{\varepsilon}(\tilde{\varepsilon} + |\sin\theta|)\epsilon_2\lambda_{\text{ch}}$,
4. $\lambda_{\max}(\boldsymbol{A}_{\text{rest}}) \leq \tilde{\varepsilon}^2\lambda^+ + \tilde{\varepsilon}^2\epsilon_2\lambda_{\text{ch}}$,
5. $\|\text{term11}\| \leq \left[\tilde{\varepsilon}^2 f + 2\tilde{\varepsilon}^2\epsilon_2 + \tilde{\varepsilon}|\sin\theta|\epsilon_2|\right]\lambda_{\text{ch}}$.

6. $\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t{}'\boldsymbol{\Phi}\right\| \le \epsilon_0\lambda_{\text{ch}}$

7. $\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{z}_t\boldsymbol{z}_t{}'\boldsymbol{\Phi}\right\| \le \left[\left(4\tilde{\varepsilon}^2 f + q_{\text{rot}}^2\right) + \epsilon_1\right]\lambda_{\text{ch}}$

Using Lemma 2.24 and substituting for $\epsilon_0, \epsilon_1, \epsilon_2$, we conclude the following. Conditioned on $\mathcal{E}_0$, with probability at least $1 - 12n^{-12}$,

$$\begin{aligned}
&\text{SE}(\hat{\boldsymbol{P}}_{\text{rot}}, \boldsymbol{E}_{\text{rot}}) \le \\
&\frac{\begin{array}{r}2\sqrt{b_0}\left[2\tilde{\varepsilon}^2 f + (\tilde{\varepsilon} + |\sin\theta|)q_{\text{rot}}\right] + \sqrt{b_0}\left[4\tilde{\varepsilon}^2 f + q_{\text{rot}}^2\right] \\ + 2\left[\tilde{\varepsilon}^2 f + 2\tilde{\varepsilon}^2\epsilon_2 + \tilde{\varepsilon}|\sin\theta|\epsilon_2\right] + 4\epsilon_0 + 2\epsilon_1\end{array}}{\begin{array}{r}(\sin^2\theta(1-\tilde{\varepsilon}^2) - 2\tilde{\varepsilon}|\sin\theta|)(1-\epsilon_2) \\ - 2\tilde{\varepsilon}(\tilde{\varepsilon} + |\sin\theta|)\epsilon_2 - (\tilde{\varepsilon}^2 f + \tilde{\varepsilon}^2\epsilon_2) - \text{numer}\end{array}},
\end{aligned}$$

where numer denotes the numerator expression. The numerator, numer, expression can be simplified to

$$\begin{aligned}
\text{numer} &\le q_{\text{rot}}\left[2\sqrt{\rho_{\text{row}}}f(\tilde{\varepsilon} + |\sin\theta|) + 0.04|\sin\theta|\right] \\
&\quad + q_{\text{rot}}^2(\sqrt{b_0} + 0.02) + \tilde{\varepsilon}\big[(8\sqrt{\rho_{\text{row}}}f + 2)\tilde{\varepsilon}f \\
&\quad + (2\tilde{\varepsilon} + |\sin\theta|)0.01 + 0.04|\sin\theta| + 0.02\tilde{\varepsilon}\big].
\end{aligned}$$

Further, using $\tilde{\varepsilon}f \le 0.01|\sin\theta|$, $\sqrt{b_0} \le 0.1$ and $q_{\text{rot}} \le 0.2|\sin\theta|$,

$$\begin{aligned}
\text{numer} &\le |\sin\theta|(0.242q_{\text{rot}} + 0.07\tilde{\varepsilon}) + 0.12q_{\text{rot}}^2 \\
&\le |\sin\theta|(0.27q_{\text{rot}} + 0.07\tilde{\varepsilon})
\end{aligned}$$

This can be loosely upper bounded by $0.26\sin^2\theta$. We use this loose upper bound when this term appears in the denominator. Following a similar approach for the denominator, denoted denom,

$$\begin{aligned}
&\text{denom} \\
&\ge \sin^2\theta\left[1 - \tilde{\varepsilon}^2 - \frac{2\tilde{\varepsilon}}{|\sin\theta|} - \frac{3\tilde{\varepsilon}^2\epsilon_2}{\sin^2\theta} - \frac{\tilde{\varepsilon}^2 f}{\sin^2\theta} - \frac{\text{numer}}{\sin^2\theta}\right] \\
&\ge \sin^2\theta\left[0.95 - \frac{\text{numer}}{\sin^2\theta}\right] \ge 0.69\sin^2\theta
\end{aligned}$$

Thus,

$$\begin{aligned}
\text{SE}(\hat{\boldsymbol{P}}_{\text{rot}}, \boldsymbol{E}_{\text{rot}}) &\le \frac{(0.27q_{\text{rot}} + 0.07\tilde{\varepsilon})|\sin\theta|}{0.69\sin^2\theta} \\
&\le \frac{0.39q_{\text{rot}} + 0.1\tilde{\varepsilon}}{|\sin\theta|},
\end{aligned}$$

Using (2.17) and $\tilde{\varepsilon} \le \tilde{\varepsilon} f \le 0.01|\sin\theta|$,

$$\begin{aligned}\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}_{\mathrm{rot}}) &\le (\tilde{\varepsilon} + |\sin\theta|)\frac{0.39 q_{\mathrm{rot}} + 0.1\tilde{\varepsilon}}{|\sin\theta|} \\ &\le 1.01|\sin\theta|\frac{0.39 q_{\mathrm{rot}} + 0.1\tilde{\varepsilon}}{|\sin\theta|} \\ &\le 0.40 q_{\mathrm{rot}} + 0.11\tilde{\varepsilon}.\end{aligned}$$

Proof of the last claim: lower bound on $\lambda_{\max}(\boldsymbol{D}_{obs})$. Using Weyl's inequality,

$$\begin{aligned}\lambda_{\max}(\boldsymbol{D}_{obs}) \ge \lambda_{\max}(\boldsymbol{D}) - \|\boldsymbol{H}\| &\ge \lambda_{\max}(\boldsymbol{A}) - \|\boldsymbol{H}\| \\ &\ge \lambda_{\min}(\boldsymbol{A}) - \|\boldsymbol{H}\|.\end{aligned}$$

Using the bounds from Lemmas 2.23 and 2.24 and (2.12), we get the lower bound. □

2.12 Appendix D: Proof of Lemma 2.24: high probability bounds on the $\sin\theta$ theorem bound terms

Proof of Lemma 2.24. Recall the definition of the event $\mathcal{E}_0$ from Remark 2.8. To prove this lemma, we first bound the probabilities of all the events conditioned on $\{\hat{\boldsymbol{P}}_*, Z\}$, for values of $\{\hat{\boldsymbol{P}}_*, Z\} \in \mathcal{E}_0$. Then we use the following simple fact.

Fact 2.25. *If* $\Pr(\text{Event}|\{\hat{\boldsymbol{P}}_*, Z\}) \ge p_0$ *for all* $\{\hat{\boldsymbol{P}}_*, Z\} \in \mathcal{E}_0$*, then,*

$$\Pr(\text{Event}|\mathcal{E}_0) \ge p_0.$$

In the discussion below, we condition on $\{\hat{\boldsymbol{P}}_*, Z\}$, for values of $\{\hat{\boldsymbol{P}}_*, Z\}$ in $\mathcal{E}_0$. Conditioned on $\{\hat{\boldsymbol{P}}_*, Z\}$, the matrices $\boldsymbol{E}_{\mathrm{rot}}$, $\boldsymbol{E}_{\mathrm{rot},\perp}$, $\boldsymbol{\Phi}$, etc, are constants (not random). All the terms that we bound in this lemma are either of the form $\sum_{t\in\mathcal{J}^\alpha} g_1(\hat{\boldsymbol{P}}_*, Z)\boldsymbol{\ell}_t\boldsymbol{\ell}_t' g_2(\hat{\boldsymbol{P}}_*, Z)$, for some functions $g_1(.), g_2(.)$, or are sub-matrices of such a term.

Since the pair $\{\hat{\boldsymbol{P}}_*, Z\}$ is independent of the $\boldsymbol{\ell}_t$'s for $t \in \mathcal{J}^\alpha$, and these $\boldsymbol{\ell}_t$'s are mutually independent, hence, even conditioned on $\{\hat{\boldsymbol{P}}_*, Z\}$, the same holds: the $\boldsymbol{\ell}_t$'s for $t \in \mathcal{J}^\alpha$ are mutually independent. Thus, once we condition on $\{\hat{\boldsymbol{P}}_*, Z\}$, the summands in the terms we need to bound are

mutually independent. As a result, matrix Bernstein (Theorem 2.27) or Vershynin's sub-Gaussian result (Theorem 2.28) are applicable.

Item 1: *In the proof of this and later items, we condition on* $\{\hat{\boldsymbol{P}}_*, Z\}$*, for values of* $\{\hat{\boldsymbol{P}}_*, Z\}$ *in* $\mathcal{E}_0$.

Since $\|\boldsymbol{\Phi}\| = 1$,

$$\left\| \frac{1}{\alpha} \sum_t \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \boldsymbol{\Phi} \right\| \leq \left\| \frac{1}{\alpha} \sum_t \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \right\|.$$

To bound the RHS above, we will apply *matrix Bernstein* (Theorem 2.27) with $\boldsymbol{Z}_t = \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}'$. As explained above, conditioned on $\{\hat{\boldsymbol{P}}_*, Z\}$, the $\boldsymbol{Z}_t$'s are mutually independent. We first obtain a bound on the expected value of the time average of the $\boldsymbol{Z}_t$'s and then compute R and σ^2 needed by Theorem 2.27. By Cauchy-Schwarz,

$$\begin{aligned}
&\left\| \mathbb{E}\left[\frac{1}{\alpha} \sum_t \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \right] \right\|^2 = \left\| \frac{1}{\alpha} \sum_t \boldsymbol{\Phi} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \boldsymbol{M}_{2,t}{}' \right\|^2 \\
&\overset{(a)}{\leq} \left\| \frac{1}{\alpha} \sum_t \left(\boldsymbol{\Phi} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \right) \left(\boldsymbol{M}_{1,t} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{\Phi} \right) \right\| \times \\
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{M}_{2,t} \boldsymbol{M}_{2,t}{}' \right\| \\
&\overset{(b)}{\leq} b_0 \left[\max_t \left\| \boldsymbol{\Phi} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \right\|^2 \right] \\
&\leq b_0 \Bigg[\max_t \Big(\left\| \boldsymbol{\Phi} \boldsymbol{P}_{\text{fix}} \boldsymbol{\Lambda}_{\text{fix}} \boldsymbol{P}_{\text{fix}}{}' \boldsymbol{M}_{1,t}{}' \right\| \\
&+ \left\| \boldsymbol{\Phi} \boldsymbol{P}_{\text{rot}} \boldsymbol{\Lambda}_{\text{ch}} \boldsymbol{P}_{\text{rot}}{}' \boldsymbol{M}_{1,t}{}' \right\| \Big)^2 \Bigg] \\
&\leq b_0 \left[\tilde{\varepsilon} q_0 \lambda^+ + (\tilde{\varepsilon} + |\sin\theta|) \, q_{\text{rot}} \lambda_{\text{ch}} \right]^2
\end{aligned} \tag{2.20}$$

where (a) follows by Cauchy-Schwarz (Theorem 2.26) with $\boldsymbol{X}_t = \boldsymbol{\Phi} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}'$ and $\boldsymbol{Y}_t = \boldsymbol{M}_{2,t}$, (b) follows from the assumption on $\boldsymbol{M}_{2,t}$, and the last inequality follows from Lemma 2.21. Using $q_0 \leq 2\tilde{\varepsilon}$,

$$\left\| \mathbb{E}\left[\frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \right] \right\| \leq \sqrt{b_0} \left[2\tilde{\varepsilon}^2 \lambda^+ + (\tilde{\varepsilon} + |\sin\theta|) \, q_{\text{rot}} \lambda_{\text{ch}} \right].$$

To compute R, using Lemma 2.21 and using $q_0 \le 2\tilde{\varepsilon}$ and $q_{\text{rot}} < |\sin\theta|$,

$$\begin{aligned}
\|\boldsymbol{Z}_t\| \le \|\boldsymbol{\Phi}\boldsymbol{\ell}_t\|\,\|\boldsymbol{w}_t\| &\le \left(\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + (\tilde{\varepsilon} + |\sin\theta|)\sqrt{\eta\lambda_{\text{ch}}}\right) \\
&\quad \left(q_0\sqrt{\eta r\lambda^+} + q_{\text{rot}}\sqrt{\eta\lambda_{\text{ch}}}\right) \\
&\le 4\tilde{\varepsilon}^2\eta r\lambda^+ + |\sin\theta| q_{\text{rot}}\eta\lambda_{\text{ch}} \\
&\quad + 2\tilde{\varepsilon}\eta\sqrt{r\lambda^+\lambda_{\text{ch}}}(q_{\text{rot}} + |\sin\theta|) \\
&\le c_1\tilde{\varepsilon}|\sin\theta|\eta r\lambda^+ + c_2|\sin\theta| q_{\text{rot}}\eta\lambda_{\text{ch}} \\
&:= R
\end{aligned}$$

for numerical constants c_1, c_2. Next we compute σ^2. Since $\boldsymbol{w}_t$'s are bounded r.v.'s, we have

$$\begin{aligned}
\left\|\frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{Z}_t\boldsymbol{Z}_t{}']\right\| &= \left\|\frac{1}{\alpha}\sum_t \mathbb{E}\left[\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{w}_t{}'\boldsymbol{w}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right]\right\| \\
&= \left\|\frac{1}{\alpha}\mathbb{E}[\|\boldsymbol{w}_t\|^2\,\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}]\right\| \\
&\le \left(\max_{\boldsymbol{w}_t}\|\boldsymbol{w}_t\|^2\right)\left\|\frac{1}{\alpha}\sum_t \mathbb{E}\left[\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right]\right\| \\
&\le \left(8\tilde{\varepsilon}^2\eta r\lambda^+ + 2q_{\text{rot}}^2\eta\lambda_{\text{ch}}\right) \\
&\quad \left(2\tilde{\varepsilon}^2\lambda^+ + \sin^2\theta\lambda_{\text{ch}}\right) \\
&\le c_1 q_{\text{rot}}^2\sin^2\theta\eta(\lambda_{\text{ch}})^2 + c_2\tilde{\varepsilon}^2\eta r\sin^2\theta\lambda^+\lambda_{\text{ch}} \\
&:= \sigma_1^2
\end{aligned}$$

for numerical constants c_1 and c_2. The above bounds again used $q_0 \le 2\tilde{\varepsilon}$ and $q_{\text{rot}} < |\sin\theta|$. For bounding $\left\|\frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{Z}_t{}'\boldsymbol{Z}_t]\right\|$ we get the same expression except for the values of c_1, c_2. Thus, applying matrix Bernstein (Theorem 2.27) followed by Fact 2.25,

$$\begin{aligned}
&\Pr\left(\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{w}_t{}'\right\|\right. \\
&\left.\le \sqrt{\rho_{\text{row}}}f\left[2\tilde{\varepsilon}^2\lambda^+ + (\tilde{\varepsilon} + |\sin\theta|)q_{\text{rot}}\lambda_{\text{ch}}\right] + \epsilon\,\middle|\,\mathcal{E}_0\right) \\
&\ge 1 - 2n\exp\left(\frac{-\alpha}{4\max\left\{\frac{\sigma_1^2}{\epsilon^2},\ \frac{R}{\epsilon}\right\}}\right).
\end{aligned}$$

Let $\epsilon = \epsilon_0 \lambda_{\text{ch}}$ where $\epsilon_0 = 0.01 \sin\theta(q_{\text{rot}} + \tilde{\varepsilon})$. Then, clearly,

$$\frac{\sigma^2}{\epsilon^2} \leq c\eta \max\{1,\ fr\} = c\eta fr, \quad \text{and}$$
$$\frac{R}{\epsilon} \leq c\eta \max\{1,\ fr\} = c\eta fr.$$

Hence, for the probability to be of the form $1 - 2n^{-12}$ we require that $\alpha \geq \alpha_{(1)}$ where

$$\alpha_{(1)} := C \cdot \eta f (r \log n)$$

Thus, if $\alpha \geq \alpha_{(1)}$, conditioned on $\mathcal{E}_0$, the bound on $\left\| \frac{1}{\alpha} \sum_t \boldsymbol{\Phi} \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \boldsymbol{\Phi} \right\|$ given in Lemma 2.24 holds w.p. at least $1 - 2n^{-12}$.

Item 2: We use Theorem 2.27 (matrix Bernstein) with $\boldsymbol{Z}_t := \boldsymbol{\Phi} \boldsymbol{w}_t \boldsymbol{w}_t{}' \boldsymbol{\Phi}$. The proof approach is similar to that of the proof of item 1. First we bound the norm of the expectation of the time average of $\boldsymbol{Z}_t$:

$$\begin{aligned}
&\left\| \mathbb{E}\left[\frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{w}_t \boldsymbol{w}_t{}' \boldsymbol{\Phi} \right] \right\| \\
&= \left\| \frac{1}{\alpha} \sum \boldsymbol{\Phi} \boldsymbol{M}_{2,t} \boldsymbol{M}_{1,t} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \boldsymbol{M}_{2,t}{}' \boldsymbol{\Phi} \right\| \\
&\leq \left\| \frac{1}{\alpha} \sum \boldsymbol{M}_{2,t} \boldsymbol{M}_{1,t} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \boldsymbol{M}_{2,t}{}' \right\| \\
&\overset{(a)}{\leq} \left(\left\| \frac{1}{\alpha} \sum_t \boldsymbol{M}_{2,t} \boldsymbol{M}_{2,t}{}' \right\| \right. \\
&\left. \left[\max_t \left\| \boldsymbol{M}_{2,t} \boldsymbol{M}_{1,t} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t} (\cdot)' \right\|^2 \right] \right)^{1/2} \\
&\overset{(b)}{\leq} \sqrt{b_0} \left[\max_t \left\| \boldsymbol{M}_{1,t} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \boldsymbol{M}_{2,t}{}' \right\| \right] \\
&\overset{(c)}{\leq} \sqrt{b_0} \left[q_0^2 \lambda^+ + q_{\text{rot}}^2 \lambda_{\text{ch}} \right] \leq \sqrt{b_0} \left[4\tilde{\varepsilon}^2 \lambda^+ + q_{\text{rot}}^2 \lambda_{\text{ch}} \right].
\end{aligned}$$

(a) follows from Cauchy-Schwarz (Theorem 2.26) with $\boldsymbol{X}_t = \boldsymbol{M}_{2,t}$ and $\boldsymbol{Y}_t = \boldsymbol{M}_{1,t} \boldsymbol{P} \boldsymbol{\Lambda} \boldsymbol{P}' \boldsymbol{M}_{1,t}{}' \boldsymbol{M}_{2,t}{}'$, (b) follows from the assumption on $\boldsymbol{M}_{2,t}$, and (c) follows from Lemma 2.21. The last inequality

used $q_0 \le 2\tilde{\varepsilon}$. To obtain R,

$$\begin{aligned}
\|\boldsymbol{Z}_t\| &= \left\|\boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{w}_t'\boldsymbol{\Phi}\right\| \\
&\le 2\left(\|\boldsymbol{\Phi}\boldsymbol{M}_t\boldsymbol{P}_{\text{fix}}\boldsymbol{a}_{t,\text{fix}}\|^2 + \|\boldsymbol{\Phi}\boldsymbol{M}_t\boldsymbol{P}_{\text{rot}}\boldsymbol{a}_{t,\text{ch}}\|^2\right) \\
&\le 2\left(q_0^2\eta r\lambda^+ + q_{\text{rot}}^2\eta\lambda_{\text{ch}}\right) \\
&\le 8\tilde{\varepsilon}^2 r\eta\lambda^+ + 2q_{\text{rot}}^2\eta\lambda_{\text{ch}} := R
\end{aligned}$$

To obtain σ^2,

$$\begin{aligned}
&\left\|\frac{1}{\alpha}\sum_t \mathbb{E}\left[\boldsymbol{\Phi}\boldsymbol{w}_t(\boldsymbol{\Phi}\boldsymbol{w}_t)'(\boldsymbol{\Phi}\boldsymbol{w}_t)\boldsymbol{w}_t'\boldsymbol{\Phi}\right]\right\| \\
&= \left\|\frac{1}{\alpha}\sum_t \mathbb{E}\left[\boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{w}_t'\boldsymbol{\Phi}\,\|\boldsymbol{\Phi}\boldsymbol{w}_t\|^2\right]\right\| \\
&\le \left(\max_{\boldsymbol{w}_t}\|\boldsymbol{\Phi}\boldsymbol{w}_t\|^2\right)\left\|\boldsymbol{\Phi}\boldsymbol{M}_t\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\boldsymbol{M}_t'\boldsymbol{\Phi}\right\| \\
&\le 2\left(q_0^2 r\eta\lambda^+ + q_{\text{rot}}^2\eta\lambda_{\text{ch}}\right)\left(q_0^2\lambda^+ + q_{\text{rot}}^2\lambda_{\text{ch}}\right) \\
&\le c_1 q_{\text{rot}}^4\eta(\lambda_{\text{ch}})^2 + c_2 q_{\text{rot}}^2\tilde{\varepsilon}^2\eta r\lambda^+\lambda_{\text{ch}} := \sigma^2
\end{aligned}$$

Applying matrix Bernstein (Theorem 2.27) followed by Fact 2.25, we have

$$\begin{aligned}
\Pr\left(\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{w}_t'\boldsymbol{\Phi}\right\| \le \sqrt{b_0}\left[4\tilde{\varepsilon}^2\lambda^+ + q_{\text{rot}}^2\lambda_{\text{ch}}\right] + \epsilon \,\middle|\, \mathcal{E}_0\right) \\
\ge 1 - n\exp\left(\frac{-\alpha\epsilon^2}{2(\sigma^2 + R\epsilon)}\right)
\end{aligned}$$

Let $\epsilon = \epsilon_1\lambda_{\text{ch}}$, $\epsilon_1 = 0.01(q_{\text{rot}}^2 + \tilde{\varepsilon}^2)$. Then we get

$$\frac{R}{\epsilon} \le c\eta\max\{1,\ rf\}, \quad \text{and} \quad \frac{\sigma^2}{\epsilon^2} \le c\eta\max\{1,\ rf\}.$$

For the success probability to be of the form $1 - 2n^{-12}$ we require $\alpha \ge \alpha_{(2)}$ where

$$\alpha_{(2)} := C\eta \cdot 13 f(r\log n)$$

Thus, if $\alpha \ge \alpha_{(2)}$,

$\Pr\left(\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{w}_t\boldsymbol{w}_t'\boldsymbol{\Phi}\right\| \le \left[\sqrt{b_0}\left(4\tilde{\varepsilon}^2 f + q_{\text{rot}}^2\right) + \epsilon_1\right]\lambda_{\text{ch}} | \mathcal{E}_0\right) \ge 1 - n^{-12}$.

Item 3: Expanding the expression for $\boldsymbol{A}$,

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{fix}}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{fix}}\boldsymbol{a}_{t,\text{fix}}{}'\right)\boldsymbol{P}_{\text{fix}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}} \\ &\quad + \boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right)\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}} \\ &\quad + \text{term1} + \text{term1}'\end{aligned}$$

where $\text{term1} := \boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{fix}}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{fix}}\boldsymbol{a}_{t,\text{ch}}{}'\right)\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}$. Since the first term on the RHS is positive semi-definite,

$$\begin{aligned}&\lambda_{\min}(\boldsymbol{A}) \\ &\geq \lambda_{\min}\left(\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right)\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}\right) \\ &+ \lambda_{\min}(\text{term1} + \text{term1}') \\ &\geq \lambda_{\min}\left(\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right)\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}\right) \\ &- 2\left\|\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{fix}}{}'\right)\boldsymbol{P}_{\text{fix}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}\right\|\end{aligned} \tag{2.21}$$

Under our current assumptions, the $\boldsymbol{a}_{t,\text{ch}}$'s are scalars, so $\boldsymbol{A}$ and $\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'$ are actually scalars. However, we write things in a general fashion (allowing $\boldsymbol{a}_{t,\text{ch}}$'s to be r_{ch} length vectors), so as to make our later discussion of the $r_{\text{ch}} > 1$ case easier. Using (2.21),

$$\begin{aligned}&\lambda_{\min}(\boldsymbol{A}) \\ &\geq \lambda_{\min}\left(\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\boldsymbol{P}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}\right)\lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right) \\ &- 2\left\|\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\right\|\left\|\boldsymbol{P}_{\text{fix}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot}}\right\|\left\|\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{fix}}{}'\right)\right\| \\ &\geq (\sin^2\theta(1-\tilde{\varepsilon}^2) - 2\tilde{\varepsilon}|\sin\theta|)\lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right) \\ &- 2\tilde{\varepsilon}(\tilde{\varepsilon} + |\sin\theta|)\left\|\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{fix}}{}'\right)\right\|.\end{aligned} \tag{2.22}$$

The second inequality follows using $\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}} = \boldsymbol{E}_{\text{rot}}{}'\boldsymbol{E}_{\text{rot}}\boldsymbol{R}_{\text{rot}} = \boldsymbol{R}_{\text{rot}}$ and Lemma 2.21. The first inequality is straightforward if $\boldsymbol{a}_{t,\text{ch}}$'s are scalars (current setting); it follows using Ostrowski's theorem [13] in the general case.

To bound the remaining terms in the above expression, we use Vershynin's sub-Gaussian result [29, Theorem 5.39] summarized in Theorem 2.28. To apply this, recall that $(\boldsymbol{a}_t)_i$ are bounded random variables satisfying $|(\boldsymbol{a}_t)_i| \leq \sqrt{\eta\lambda_i}$. Hence they are sub-Gaussian with sub-Gaussian norm $\sqrt{\eta\lambda_i}$ [29]. Using [29, Lemma 5.24], the vectors $\boldsymbol{a}_t$ are also sub-Gaussian with sub-Gaussian norm bounded by $\max_i \sqrt{\eta\lambda_i} = \sqrt{\eta\lambda^+}$. Thus, applying Theorem 2.28 with $K \equiv \sqrt{\eta\lambda^+}$, $\epsilon \equiv \epsilon_2\lambda_{\text{ch}}$, $N \equiv \alpha$, $n_w \equiv r$, followed by using Fact 2.25, if $\alpha \geq \alpha_{(3)} := \frac{C(r\log 9 + 10\log n)f^2}{\epsilon_2^2}$, then,

$$\Pr\left(\left\|\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\right\| \leq \epsilon_2\lambda_{\text{ch}}\middle|\mathcal{E}_0\right) \geq 1 - 2n^{-12}. \tag{2.23}$$

We could also have used matrix Bernstein to bound $\|\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}'\|$. However, since the $\boldsymbol{a}_t$'s are r-length vectors and $r \ll n$, the Vershynin result requires a smaller lower bound on α.

If $\boldsymbol{B}_1$ is a sub-matrix of a matrix $\boldsymbol{B}$, then $\|\boldsymbol{B}_1\| \leq \|\boldsymbol{B}\|$. Thus, we can also use (2.23) for bounding the norm of various sub-matrices of $\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\right)$. Doing this, we get

$$\Pr\left(\lambda_{\max}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{fix}}\boldsymbol{a}_{t,\text{fix}}{}'\right) \leq \lambda^+ + \epsilon_2\lambda_{\text{ch}}\middle|\mathcal{E}_0\right)$$
$$\geq 1 - 2n^{-12}, \tag{2.24}$$
$$\Pr\left(\lambda_{\max}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right) \leq \lambda_{\text{ch}} + \epsilon_2\lambda_{\text{ch}}\middle|\mathcal{E}_0\right)$$
$$\geq 1 - 2n^{-12}, \tag{2.25}$$
$$\Pr\left(\lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{ch}}{}'\right) \geq \lambda_{\text{ch}} - \epsilon_2\lambda_{\text{ch}}\middle|\mathcal{E}_0\right)$$
$$\geq 1 - 2n^{-12}, \text{ and} \tag{2.26}$$
$$\Pr\left(\left\|\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{ch}}\boldsymbol{a}_{t,\text{fix}}{}'\right\| \leq \epsilon_2\lambda_{\text{ch}}\middle|\mathcal{E}_0\right)$$
$$\geq 1 - 2n^{-12}. \tag{2.27}$$

Combining (2.22), (2.26) and (2.27), if $\alpha \geq \alpha_{(3)}$,

$$\Pr\Bigg(\lambda_{\min}(\boldsymbol{A}) \geq (\sin^2\theta(1-\tilde{\varepsilon}^2) - 2\tilde{\varepsilon}|\sin\theta|)(1-\epsilon_2)\lambda_{\text{ch}} \\ -2\tilde{\varepsilon}(\tilde{\varepsilon}+|\sin\theta|)\epsilon_2\lambda_{\text{ch}}\Big|\mathcal{E}_0\Bigg) \geq 1-4n^{-12} \tag{2.28}$$

Item 4: Recall that $\boldsymbol{E}_{\text{rot},\perp}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}} = 0$. Thus,

$$\lambda_{\max}(\boldsymbol{A}_{\text{rest}}) \leq \lambda_{\max}\left(\boldsymbol{E}_{\text{rot},\perp}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{fix}}\boldsymbol{P}_{\text{fix}}{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot},\perp}\right) \times \\ \lambda_{\max}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{fix}}\boldsymbol{a}_{t,\text{fix}}{}'\right) \tag{2.29}$$

where the last inequality follows from Ostrowski's theorem [13]. Using this and (2.24), if $\alpha \geq \alpha_{(3)}$,

$$\Pr\left(\lambda_{\max}(\boldsymbol{A}_{\text{rest}}) \leq \tilde{\varepsilon}^2\lambda^+ + \tilde{\varepsilon}^2\epsilon_2\lambda_{\text{ch}}\Big|\mathcal{E}_0\right) \geq 1-2n^{-12}$$

Item 5: Recall that

term11 $= \frac{1}{\alpha}\sum \boldsymbol{E}_{\text{rot}}\boldsymbol{E}_{\text{rot}}{}'\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\boldsymbol{E}_{\text{rot},\perp}\boldsymbol{E}_{\text{rot},\perp}{}'$. As in earlier items, we can expand this into a sum of four terms using $\boldsymbol{\ell}_t = \boldsymbol{P}_{\text{fix}}\boldsymbol{a}_{t,\text{fix}} + \boldsymbol{P}_{\text{rot}}\boldsymbol{a}_{t,\text{ch}}$. Then using $\boldsymbol{E}_{\text{rot},\perp}{}'\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}} = 0$ and $\|\boldsymbol{E}_{\text{rot}}\| = \|\boldsymbol{E}_{\text{rot},\perp}\| = 1$, we get

$$\|\text{term11}\| \leq \|\boldsymbol{\Phi}\boldsymbol{P}_{\text{fix}}\|\left\|\boldsymbol{P}_{\text{fix}}{}'\boldsymbol{\Phi}\right\|\lambda_{\max}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{fix}}\boldsymbol{a}_{t,\text{fix}}{}'\right) \\ + \|\boldsymbol{\Phi}\boldsymbol{P}_{\text{rot}}\|\left\|\boldsymbol{P}_{\text{fix}}{}'\boldsymbol{\Phi}\right\|\left\|\frac{1}{\alpha}\sum_t \boldsymbol{a}_{t,\text{fix}}\boldsymbol{a}_{t,\text{ch}}{}'\right\| \tag{2.30}$$

Using (2.24) and (2.27), if $\alpha \geq \alpha_{(3)}$, w.p. at least $1-4n^{-12}$, conditioned on $\mathcal{E}_0$, $\|\text{term11}\| \leq \tilde{\varepsilon}^2(\lambda^+ + \epsilon_2\lambda_{\text{ch}}) + (\tilde{\varepsilon}(\tilde{\varepsilon}+|\sin\theta|))\epsilon_2\lambda_{\text{ch}}$.

Item 6: Consider $\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t{}'\right\|$. We will apply matrix Bernstein (Theorem 2.27). We have $\left\|\mathbb{E}\left[\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t{}'\right]\right\| = 0$ since $\boldsymbol{\ell}_t$'s are independent of $\boldsymbol{z}_t$'s and both are zero mean. We obtain R as

follows

$$\begin{aligned}
&\left\|\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t'\right\| = \|\boldsymbol{\Phi}\boldsymbol{\ell}_t\|\,\|\boldsymbol{z}_t\| \\
&\leq \left(\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + (\tilde{\varepsilon} + |\sin\theta|)\sqrt{\eta\lambda_{\text{ch}}}\right) b_z \\
&\leq \left(2\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + |\sin\theta|\sqrt{\eta\lambda_{\text{ch}}}\right)\left(q_0\sqrt{r\lambda^+} + q_{\text{rot}}\sqrt{\lambda_{\text{ch}}}\right) \\
&\leq 4\tilde{\varepsilon}^2\sqrt{\eta}r\lambda^+ + |\sin\theta| q_{\text{rot}}\sqrt{\eta}\lambda_{\text{ch}} \\
&+ 2\tilde{\varepsilon}\sqrt{\eta}\sqrt{r\lambda^+\lambda_{\text{ch}}}(q_{\text{rot}} + |\sin\theta|) \\
&\leq c_1\tilde{\varepsilon}|\sin\theta|\sqrt{\eta}r\lambda^+ + c_2|\sin\theta| q_{\text{rot}}\sqrt{\eta}\lambda_{\text{ch}} := R
\end{aligned}$$

for numerical constants c_1, c_2. Next we compute σ^2 as follows. First consider

$$\begin{aligned}
&\left\|\frac{1}{\alpha}\sum_t \mathbb{E}\left[\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{z}_t'\boldsymbol{z}_t\boldsymbol{\ell}_t'\boldsymbol{\Phi}\right]\right\| = \left\|\frac{1}{\alpha}\mathbb{E}[\|\boldsymbol{z}_t\|^2\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t'\boldsymbol{\Phi}]\right\| \\
&\leq \left(\max_{\boldsymbol{z}_t}\|\boldsymbol{z}_t\|^2\right)\left\|\frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t'\boldsymbol{\Phi}]\right\| \\
&\leq (8\tilde{\varepsilon}^2 r\lambda^+ + 2q_{\text{rot}}^2\lambda_{\text{ch}})(2\tilde{\varepsilon}^2\lambda^+ + \sin^2\theta\lambda_{\text{ch}}) \\
&\leq c_1 q_{\text{rot}}^2\sin^2\theta\eta(\lambda_{\text{ch}})^2 + c_2\tilde{\varepsilon}^2\eta r\sin^2\theta\lambda^+\lambda_{\text{ch}} := \sigma_1^2
\end{aligned}$$

we note here that since $b_z^2 = r\lambda_z^+$, the other term in the expression for σ^2 is the same (modulo constants) as σ_1^2. Furthermore, notice that the expressions for both R and σ^2 are the same as the ones obtained in *Item 1.* Thus, we use the same deviation, ϵ_0 here, and hence also obtain the same sample complexity, α; i.e., we let $\epsilon = \epsilon_0\lambda_{\text{ch}}$ where $\epsilon_0 = 0.01\sin\theta(q_{\text{rot}} + \tilde{\varepsilon})$, and obtain $\alpha \geq \alpha_{(1)}$ derived in item 1.

Item 7: This term follows in a similar fashion as Item 2, 6 and $\alpha \geq \alpha_{(2)}$ suffices. □

2.13 Appendix E: Proof of Projected CS Lemma

Proof of Lemma 2.15. The first four claims were already proved below the lemma statement. Consider the fifth claim (exact support recovery). Recall that for any $t \in \mathcal{J}_k$, $\boldsymbol{v}_t$ satisfies $\|\boldsymbol{v}_t\| \leq C(2\tilde{\varepsilon}\sqrt{r\lambda^+} + \zeta_{k-1}^+\sqrt{\lambda_{\text{ch}}}) := b_{v,t}$ (for $t \in \mathcal{J}_1$ and $t \in \mathcal{J}_0$ the bounds are the same) with

$C = \sqrt{\eta}$, and thus $\boldsymbol{b}_t := \boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$ satisfies

$$\begin{aligned}
\|\boldsymbol{b}_t\| = \|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\| &\leq \|\boldsymbol{\Psi}\boldsymbol{\ell}_t\| + \|\boldsymbol{\Psi}\|\,\|\boldsymbol{v}_t\| \\
&\leq \left(\tilde{\varepsilon}\sqrt{r\eta\lambda^+} + \zeta_{k-1}^+\sqrt{\eta\lambda_{\text{ch}}}\right) \\
&+ \sqrt{\eta}\left(2\tilde{\varepsilon}\sqrt{r\lambda^+} + \zeta_{k-1}^+\sqrt{\lambda_{\text{ch}}}\right) \\
&\leq 2\sqrt{\eta}\left(2\tilde{\varepsilon}\sqrt{r\lambda^+} + 0.5^{k-1}\cdot 0.06|\sin\theta|\sqrt{\lambda_{\text{ch}}}\right) \\
&:= b_{b,t} = 2b_{v,t}
\end{aligned}$$

From the lower bound on $x_{\min,t}$ in Theorem 2.2 or that in Corollary 2.3, $b_{b,t} < x_{\min,t}/15$. Also, we set $\xi_t = x_{\min,t}/15$. Using these facts, and $\delta_{2s}(\boldsymbol{\Psi}) \leq 0.12 < 0.15$ (third claim of this lemma), [4, Theorem 1.2] implies that

$$\|\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t\| \leq 7\xi_t = 7x_{\min,t}/15$$

Thus,

$$|(\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t)_i| \leq \|\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t\| \leq 7x_{\min,t}/15 < x_{\min,t}/2$$

We have $\omega_{supp,t} = x_{\min,t}/2$. Consider an index $i \in \mathcal{T}_t$. Since $|(\boldsymbol{x}_t)_i| \geq x_{\min,t}$,

$$\begin{aligned}
x_{\min,t} - |(\hat{\boldsymbol{x}}_{t,cs})_i| &\leq |(\boldsymbol{x}_t)_i| - |(\hat{\boldsymbol{x}}_{t,cs})_i| \\
&\leq |(\boldsymbol{x}_t - \hat{\boldsymbol{x}}_{t,cs})_i| < \frac{x_{\min,t}}{2}
\end{aligned}$$

Thus, $|(\hat{\boldsymbol{x}}_{t,cs})_i| > \frac{x_{\min,t}}{2} = \omega_{supp,t}$ which means $i \in \hat{\mathcal{T}}_t$. Hence $\mathcal{T}_t \subseteq \hat{\mathcal{T}}_t$. Next, consider any $j \notin \mathcal{T}_t$. Then, $(\boldsymbol{x}_t)_j = 0$ and so

$$\begin{aligned}
|(\hat{\boldsymbol{x}}_{t,cs})_j| &= |(\hat{\boldsymbol{x}}_{t,cs})_j)| - |(\boldsymbol{x}_t)_j| \\
&\leq |(\hat{\boldsymbol{x}}_{t,cs})_j - (\boldsymbol{x}_t)_j| \leq b_{b,t} < \frac{x_{\min,t}}{2}
\end{aligned}$$

which implies $j \notin \hat{\mathcal{T}}_t$ and so $\hat{\mathcal{T}}_t \subseteq \mathcal{T}_t$. Thus $\hat{\mathcal{T}}_t = \mathcal{T}_t$.

With $\hat{\mathcal{T}}_t = \mathcal{T}_t$, the sixth claim follows easily. Since $\mathcal{T}_t$ is the support of $\boldsymbol{x}_t$, $\boldsymbol{x}_t = \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{x}_t$, and so

$$\begin{aligned}\hat{\boldsymbol{x}}_t &= \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'(\boldsymbol{\Psi}\boldsymbol{\ell}_t + \boldsymbol{\Psi}\boldsymbol{x}_t) \\ &= \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t) + \boldsymbol{x}_t\end{aligned}$$

since $\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi} = \boldsymbol{I}'_{\mathcal{T}_t}\boldsymbol{\Psi}'\boldsymbol{\Psi} = \boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}$. Thus $\boldsymbol{e}_t = \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t$ satisfies (2.10). Using (2.10) and the earlier claims,

$$\begin{aligned}\|\boldsymbol{e}_t\| &\le \left\|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\right\| \left\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\right\| \\ &\le 1.2\left[\left\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{\ell}_t\right\| + \|\boldsymbol{v}_t\|\right]\end{aligned}$$

When $k = 1$, $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_*\hat{\boldsymbol{P}}_*{}'$. Thus, using (2.13) and $\|\hat{\boldsymbol{P}}_*{}'\boldsymbol{P}_{\text{new}}\| \le \tilde{\varepsilon}$ (follows from Lemma 2.10),

$$\begin{aligned}\left\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{\ell}_t\right\| &\le \|\boldsymbol{\Psi}\boldsymbol{P}_{*,\text{fix}}\|\,\|\boldsymbol{a}_{t,\text{fix}}\| \\ &\quad + \left(\|\boldsymbol{\Psi}\boldsymbol{P}_{*,\text{ch}}\cos\theta\| + \left\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{P}_{\text{new}}\sin\theta\right\|\right)\|\boldsymbol{a}_{t,\text{ch}}\| \\ &\le \tilde{\varepsilon}\sqrt{\eta r\lambda^+} \\ &\quad + \tilde{\varepsilon}|\cos\theta|\sqrt{\eta\lambda_{\text{ch}}} + (0.1 + \tilde{\varepsilon})|\sin\theta|\sqrt{\eta\lambda_{\text{ch}}} \\ &\le 2\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + 0.11|\sin\theta|\sqrt{\eta\lambda_{\text{ch}}}\end{aligned}$$

also, in this interval, $b_{v,t} \le 2\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + 0.11|\sin\theta|\sqrt{\eta\lambda_{\text{ch}}}$ so that $\|\boldsymbol{e}_t\| \le 2.4 \cdot \left(2\tilde{\varepsilon}\sqrt{\eta r\lambda^+} + 0.11|\sin\theta|\sqrt{\eta\lambda_{\text{ch}}}\right)$ When $k > 1$, $\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{\ell}_t\| \le \|\boldsymbol{\Psi}\boldsymbol{\ell}_t\| \le \tilde{\varepsilon}\sqrt{r\eta\lambda^+} + \zeta^+_{k-1}\sqrt{\eta\lambda_{\text{ch}}}$. and the same bound holds on $b_{v,t}$ so that $\|\boldsymbol{e}_t\| \le 2.4 \cdot \left(\tilde{\varepsilon}\sqrt{r\eta\lambda^+} + \zeta^+_{k-1}\sqrt{\eta\lambda_{\text{ch}}}\right)$ □

2.14 Appendix F: Time complexity of s-ReProCS

The time-consuming steps of s-ReProCS are either l_1 minimization or the subspace update steps. Support estimation and LS steps are much faster and hence can be ignored for this discussion. The computational complexity of l_1 minimization (if the best solver were used) [30] is the cost of multiplying the CS matrix or its transpose with a vector times $\log(1/\epsilon)$ if ϵ is the bound on the error w.r.t. the true minimizer of the program. In ReProCS, the CS matrix is of the form $\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'$

where $\hat{\boldsymbol{P}}$ is of size $n \times r$ or $n \times (r+1)$, thus multiplying a vector with it takes time $O(nr)$. Thus, the l_1 minimization complexity per frame is $O(nr \log(1/\epsilon))$, and thus the total cost for $d - t_{\text{train}}$ frames is $O(nr \log(1/\epsilon)(d - t_{\text{train}}))$. The subspace update step consists of $(d - t_{\text{train}} - J\alpha)/\alpha$ rank one SVD's on an $n \times \alpha$ matrix (for either detecting subspace change or for projection-SVD) and J rank r SVD's on an $n \times \alpha$ matrix (for subspace re-estimation). Thus the subspace update complexity is at most $O(n(d - t_{\text{train}})r \log(1/\epsilon))$ and the total ReProCS complexity (without the initialization step) is $O(n(d - t_{\text{train}})r \log(1/\epsilon))$.

If we assume that the initialization uses AltProj, AltProj is applied to a matrix of size $n \times t_{\text{train}}$ with rank r. Thus the initialization complexity is $O(nt_{\text{train}}r^2 \log(1/\epsilon))$. If instead GD [33] is used, then the time complexity is reduced to $O(nt_{\text{train}}rf \log(1/\epsilon))$. Treating f as a constant (our discussion treats condition numbers as constants), the final complexity of s-ReProCS is $O(ndr \log(1/\epsilon))$.

If s-ReProCS is used to only solve the RPCA problem (compute column span of the entire matrix $\boldsymbol{L}$), then the SVD based subspace re-estimation step can be removed. With this change, the complexity of s-ReProCS (without the initialization step) reduces to just $O(nd \log(1/\epsilon))$ since only 1-SVDs are needed. Of course this would mean a slightly tighter bound on max-outlier-frac-col is required – it will need to be less than $c/(r + J)$.

2.15 Appendix G: Preliminaries: Cauchy-Schwarz, matrix Bernstein and Vershynin's sub-Gaussian result

Cauchy-Schwarz for sums of matrices says the following [22].

Theorem 2.26. *For matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ we have*

$$\left\| \frac{1}{\alpha} \sum_t \boldsymbol{X}_t \boldsymbol{Y}_t{}' \right\|^2 \leq \left\| \frac{1}{\alpha} \sum_t \boldsymbol{X}_t \boldsymbol{X}_t{}' \right\| \left\| \frac{1}{\alpha} \sum_t \boldsymbol{Y}_t \boldsymbol{Y}_t{}' \right\| \tag{2.31}$$

Matrix Bernstein [24], conditioned on another r.v. X, says the following.

Theorem 2.27. *Given an α-length sequence of $n_1 \times n_2$ dimensional random matrices and a r.v. X Assume the following. For all $X \in \mathcal{C}$, (i) conditioned on X, the matrices $\boldsymbol{Z}_t$ are mutually*

independent, (i) $\mathbb{P}(\|\boldsymbol{Z}_t\| \leq R|X) = 1$*, and (iii)* $\max\left\{\left\|\frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{Z}_t{}'\boldsymbol{Z}_t|X]\right\|, \left\|\frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{Z}_t\boldsymbol{Z}_t{}'|X]\right\|\right\} \leq \sigma^2$*. Then, for an* $\epsilon > 0$*,*

$$\mathbb{P}\left(\left\|\frac{1}{\alpha}\sum_t \boldsymbol{Z}_t\right\| \leq \left\|\frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{Z}_t|X]\right\| + \epsilon \middle| X\right)$$
$$\geq 1 - (n_1 + n_2)\exp\left(\frac{-\alpha\epsilon^2}{2\left(\sigma^2 + R\epsilon\right)}\right) \text{ for all } X \in \mathcal{C}.$$

Vershynin's result for matrices with independent sub-Gaussian rows [29, Theorem 5.39], conditioned on another r.v. X, says the following.

Theorem 2.28. *Given an* N*-length sequence of sub-Gaussian random vectors* $\boldsymbol{w}_i$ *in* $\mathbb{R}^{n_w}$*, an r.v* X*, and a set* $\mathcal{C}$*. Assume that for all* $X \in \mathcal{C}$*, (i)* $\boldsymbol{w}_i$ *are conditionally independent given* X*; (ii) the sub-Gaussian norm of* $\boldsymbol{w}_i$ *is bounded by* K *for all* i*. Let* $\boldsymbol{W} := [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_N]'$*. Then for an* $0 < \epsilon < 1$ *we have*

$$\mathbb{P}\left(\left\|\frac{1}{N}\boldsymbol{W}'\boldsymbol{W} - \frac{1}{N}\mathbb{E}\big[\boldsymbol{W}'\boldsymbol{W}|X\big]\right\| \leq \epsilon \middle| X\right)$$
$$\geq 1 - 2\exp\left(n_w \log 9 - \frac{c\epsilon^2 N}{4K^4}\right) \text{ for all } X \in \mathcal{C}. \tag{2.32}$$

Figure 2.3: Comparison of background recovery performance is Foreground-Background Separation tasks for `MR` (first two rows), `SL` (middle two rows) and `LB` (last two rows) sequences (first two rows). The recovered background images are shown at $t = t_{\text{train}} + 140, 630$ for MR, $t = t_{\text{train}} + 200, 999$ for SL, and $t = t_{\text{train}} + 260, 610$ for LB. Notice that for the LB sequence, all algorithms work fairly well. In the MR sequence, since the s-ReProCS is able to tolerate larger max-outlier-frac-row, it is able to completely remove the person. Further, only s-ReProCS background does not contain the person or even his shadow. All others do. Finally, in the SL sequence, it is demonstrated that the changing subspace model is much more appropriate for long sequences since only s-ReProCS and GRASTA are able to recognize that the background has changed. GRASTA contains some artifacts, but s-ReProCS is able to clearly isolate the person. The time taken per frame (in milliseconds) is shown in parentheses above the respective video sequence. In all the videos, notice that s-ReProCS is also faster than all algorithms with the exception of GRASTA which only works for the lobby sequence that involves very little background changes.

CHAPTER 3. NEARLY OPTIMAL ROBUST SUBSPACE TRACKING

Praneeth Narayanamurthy and Namrata Vaswani

Dept. of Electrical and Computer Engineering, Iowa State University, Ames, IA, 50010

Modified from a manuscript published in *IEEE Journal of Selected Areas in Information Theory*

Abstract

This work studies the robust subspace tracking (ST) problem. Robust ST can be simply understood as a (slow) time-varying subspace extension of robust PCA. It assumes that the true data lies in a low-dimensional subspace that is either fixed or changes slowly with time. The goal is to track the changing subspaces over time in the presence of additive sparse outliers and to do this quickly (with a short delay). We introduce a "fast" mini-batch robust ST solution that is provably correct under mild assumptions. Here "fast" means two things: (i) the subspace changes can be detected and the subspaces can be tracked with near-optimal delay, and (ii) the time complexity of doing this is the same as that of simple (non-robust) PCA. Our main result assumes piecewise constant subspaces (needed for identifiability), but we also provide a corollary for the case when there is a little change at each time.

A second contribution is a novel non-asymptotic guarantee for PCA in linearly data-dependent noise. An important setting where this is useful is for linearly data dependent noise that is sparse with support that changes enough over time. The analysis of the subspace update step of our proposed robust ST solution uses this result.

3.1 Introduction

Principal Components Analysis (PCA) is one of the most widely used and well studied dimension reduction techniques. It is solved via singular value decomposition (SVD) following by retaining

the top r singular vectors for getting an r-dimensional subspace approximation. Robust PCA (RPCA) refers to PCA in the presence of outliers. According to [3], it can be defined as the problem of decomposing a given data matrix into the sum of a low-rank matrix (true data) and a sparse matrix (outliers). The column space of the low-rank matrix then gives the desired principal subspace (PCA solution). A common application of RPCA is in video analytics in separating a video into a slow-changing background image sequence (modeled as a low-rank matrix) and a foreground image sequence consisting of moving objects or people (modeled as a sparse matrix) [3]. The RPCA problem has been extensively studied in the last decade since [3, 5] introduced the principal components pursuit solution and obtained the first guarantees for it. Follow-up work by Hsu et al [13] studied it further. Later work [25, 38, 6] has developed provable non-convex solutions that are much faster. Alternating Projections or AltProj was the first such approach [25].

Robust Subspace Tracking (ST) can be simply understood as a (slow) time-varying subspace extension of RPCA. It assumes that the true data lies in a low-dimensional subspace that is either fixed or changes slowly with time. We focus on slow changing subspaces because it is not clear how to distinguish the effect of a sudden subspace change from that of an outlier. The goal is to track the changing subspaces over time in the presence of additive sparse outliers and to do this quickly (with a short delay). Time-varying subspaces is a more appropriate model for long data sequences, e.g., long surveillance videos, since if a single subspace model is used, the resulting matrix may not be sufficiently low-rank. Moreover the tracking setting (short tracking delay) is needed for applications where near real-time estimates are needed, e.g., video-based surveillance (object tracking), monitoring seismological activity, or detection of anomalous behavior in dynamic social networks. While many heuristics exist for robust ST, e.g., [27, 28, 11, 9, 8, 15, 42], there has been little work on provably correct solutions [39, 24]. The first result [39] needed many restrictive assumptions (most importantly it required assumptions on intermediate algorithm estimates) and a large tracking delay (the delay was proportional to $1/\varepsilon^2$ to get a ε accurate estimate). The second one [24] significantly improved upon [39], but still required a very specific model on subspace change, needed an ε-accurate initial subspace estimate in order to guarantee ε-accurate recovery at later

time instants, and its tracking delay was r-times sub-optimal. Our work builds on [24] and removes these, and two other more technical, limitations that we explained later.

Contributions. This work has two contributions. (1) First, we introduce a "fast" mini-batch robust ST solution that is provably correct under mild assumptions. Here "fast" means two things: (i) the subspace changes can be detected and the subspaces can be tracked with near-optimal delay (the number of data samples required to track an r-dimensional subspace of $\mathbb{R}^n$ to ε accuracy is within log factors of r); and (ii) the time complexity of doing this is just $O(ndr\log(1/\varepsilon))$, which is, order-wise, the same as that of solving the basic (non-robust) PCA problem for an $n\times d$ matrix. Our main result assumes piecewise constant subspaces (needed for identifiability), but we also provide a corollary for the case when there is a little change at each time. (2) Our second contribution is a novel non-asymptotic guarantee for PCA in data-dependent noise that satisfies certain simple assumptions. An important setting where these hold is for linearly data dependent noise that is sparse with enough support changes over time. This problem occurs in the subspace update step of our proposed robust ST solution. The PCA result is also of independent interest. As an example, it is useful for analyzing PCA and subspace tracking with missing data [10].

Organization. We first summarize our notation and then provide a brief discussion of the significance of our PCA guarantee and how it is used in analyzing our robust ST solution next. In Sec. 3.2, we present the result for PCA in data-dependent noise and its corollary for the sparse data-dependent noise case. In Sec. 3.3, we define the robust ST problem, state the assumptions required to ensure its identifiability, develop the nearly (delay) optimal robust subspace tracker (NORST) algorithm for solving it, and provide and discuss the correctness guarantee for it. Related work is discussed in detail in Sec. 3.4. Two important extensions of our result are provided in Sec. 3.5. We provide a proof of the correctness guarantee for NORST in Sec. 3.6. Empirical evaluation on synthetic and real-world datasets is described in Sec. 3.7. We conclude and discuss future directions in Sec. 3.8.

3.1.1 Notation

We use the interval notation $[a, b]$ to refer to all integers between a and b, inclusive, and we use $[a, b) := [a, b-1]$. $\|.\|$ denotes the l_2 norm for vectors and induced l_2 norm for matrices unless specified otherwise, and $'$ denotes transpose. We use $\boldsymbol{M}_{\mathcal{T}}$ to denote a sub-matrix of $\boldsymbol{M}$ formed by its columns indexed by entries in the set $\mathcal{T}$. In our algorithm statements, we use $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \ldots, \hat{\boldsymbol{\ell}}_t]$ and $SVD_r[\boldsymbol{M}]$ to refer to the matrix of top of r left singular vectors of the matrix $\boldsymbol{M}$. A matrix $\boldsymbol{P}$ with mutually orthonormal columns is referred to as a *basis matrix*; it represents the subspace spanned by its columns. For basis matrices $\boldsymbol{P}_1, \boldsymbol{P}_2$, $\text{SE}(\boldsymbol{P}_1, \boldsymbol{P}_2) := \|(\boldsymbol{I} - \boldsymbol{P}_1 \boldsymbol{P}_1{}')\boldsymbol{P}_2\|$ quantifies the Subspace Error (distance) between their respective subspaces. This is equal to the sine of the largest principal angle between the subspaces. If $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ are of the same dimension, $\text{SE}(\boldsymbol{P}_1, \boldsymbol{P}_2) = \text{SE}(\boldsymbol{P}_2, \boldsymbol{P}_1)$. *We reuse the letters C, c to denote different numerical constants in each use with the convention that $C \geq 1$ and $c < 1$.*

3.1.2 Significance and novelty of our PCA result and its use to analyze Robust Subspace Tracking

There is little existing work that explicitly studies PCA (solved via SVD) in the presence of data-dependent noise (work that exploits knowledge of the data-dependency structure of the noise)[1]. Our work provides a guarantee for one such setting; the setting is motivated by PCA in sparse linearly data-dependent noise (PCA-SDDN). This problem occurs when studying the SVD solution for solving (i) PCA with missing data, (ii) ST with missing data, and (iii) robust ST (with outliers and with and without missing data). We briefly explain the technical novelty of our result here. Let $\boldsymbol{W}$ denote the sparse linearly data-dependent noise matrix corrupting a true low rank r data matrix $\boldsymbol{L}$. We observe $\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{w}_t$, $t = 1, 2, \ldots, \alpha$ with $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t$, $\boldsymbol{P}$ is an $n \times r$ matrix with orthonormal columns and $r \ll n$. Since $\boldsymbol{w}_t$ is linearly data-dependent and sparse, without loss of generality, we can express it as $\boldsymbol{w}_t = \boldsymbol{I}_{\mathcal{T}_t} \boldsymbol{M}_{s,t} \boldsymbol{\ell}_t$ with $\mathcal{T}_t = \text{support}(\boldsymbol{w}_t)$

[1] Of course any work on PCA for an approximately low rank matrix makes no assumptions on true data or noise and thus does implicitly allow data-dependent noise as well. However, this type of work does not exploit knowledge of how the noise depends on the data.

and $\boldsymbol{M}_{s,t}$ being the data-dependency matrix at time/column t. Let b denote the maximum of the fraction of nonzero entries in any row of $\boldsymbol{W}$. We compute the PCA estimate, $\hat{\boldsymbol{P}}$, as the r-SVD of $\boldsymbol{Y} := [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_\alpha] = \boldsymbol{L} + \boldsymbol{W}$.

(1) The sparsity of the noise along with a careful application of the Cauchy-Schwarz inequality implies that $\|\mathbb{E}[\frac{1}{\alpha}\sum_t \boldsymbol{w}_t\boldsymbol{w}_t{}']\| \leq \sqrt{b}\max_t \|\mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t{}']\|$, i.e., the time-averaged noise power is at most $\sqrt{b}$ times its maximum instantaneous value. Thus, if b is small enough (noise support changes sufficiently across columns), the former is much smaller than the latter. (2) Since $\boldsymbol{w}_t$ depends on $\boldsymbol{\ell}_t$, this means that the data-noise correlation $\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{w}_t{}']$ is not zero and, its time-averaged value, $\|\mathbb{E}[\frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{w}_t{}']\|$, is in fact the dominant term in the perturbation $\|\boldsymbol{Y}\boldsymbol{Y}' - \boldsymbol{L}\boldsymbol{L}'\|$ that governs the subspace recovery error, $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P})$. Again using Cauchy-Schwarz and sparsity of $\boldsymbol{w}_t$, we can show that $\|\mathbb{E}[\frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{w}_t{}']\| \leq \sqrt{b}\max_t \|\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{w}_t{}']\|$. Thus, even though signal-noise correlation is not zero (and is, in fact, proportional to signal power), its time-averaged value is $\sqrt{b}$ times smaller. Since $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \lessapprox \frac{\|\boldsymbol{Y}\boldsymbol{Y}' - \boldsymbol{L}\boldsymbol{L}'\|}{\lambda_r(\boldsymbol{L}\boldsymbol{L}')}$ when the numerator is small enough (by Davis-Kahan $\sin\theta$ theorem), the above two facts imply that

$$\begin{aligned}\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) &\lessapprox \sqrt{b}\frac{(2\max_t \|\boldsymbol{M}_{s,t}\boldsymbol{\Sigma}\| + \max_t \|\boldsymbol{M}_{s,t}\boldsymbol{\Sigma}\boldsymbol{M}_{s,t}'\|)}{\lambda_r(\boldsymbol{\Sigma})} \\ &\leq \sqrt{b}(2q + q^2)f\end{aligned}$$

Here $\boldsymbol{\Sigma} := \mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{\ell}_t'] \overset{\text{EVD}}{=} \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'$, $q := \max_t \|\boldsymbol{M}_{s,t}\boldsymbol{P}\|$, and f denotes the condition number of $\boldsymbol{\Lambda}$ [2] q can be understood as the noise-to-signal ratio and is thus a measure of the noise level. (3) Suppose that the $\boldsymbol{a}_t$'s are i.i.d. and bounded, i.e., $\|\boldsymbol{a}_t\|^2 \leq \mu r \lambda_{\max}(\boldsymbol{\Sigma})$. Since the noise is data-dependent, and since we assume that our data $\boldsymbol{\ell}_t$ is generated from a low (r) dimensional subspace, we can use the above facts and matrix-Bernstein [30] to show that $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon$ with high probability, $1 - 3n^{-10}$, if the sample complexity α is $\Omega(\frac{q^2}{\epsilon^2}\kappa^2 r \log n)$. Thus, in order to achieve a recovery error ϵ that is fraction of the noise level, q, the required sample complexity is near optimal (is within log factors of r).

[2] If $\boldsymbol{\ell}_t$ is not stationary, $\boldsymbol{\Sigma} := \frac{1}{\alpha}\sum_t \mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{\ell}_t']$, in this case one needs to redefine $f = \max_t \|\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{\ell}_t']\|/\lambda_r(\boldsymbol{\Sigma})$.

In the above discussion we have assumed zero uncorrelated noise, but our actual result also handles that. This can model the fact that the true data is only approximately low rank. Moreover it provides a guarantee for a more general setting than PCA-SDDN.

Use to analyze Robust ST. For solving the robust ST problem (recover $\boldsymbol{\ell}_t$ and its subspace from $\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{x}_t$ where $\boldsymbol{x}_t$ denotes the sparse outlier at time t), we develop a mini-batch algorithm that (a) processes the observed data to return an estimate of $\boldsymbol{\ell}_t$, $\hat{\boldsymbol{\ell}}_t$, at each time t; and (b) uses α-mini-batches of $\hat{\boldsymbol{\ell}}_t$ to compute a new estimate of the current subspace. This process is repeated K times with K new α-length mini-batches for the current subspace; after this time, the algorithm enters a "subspace change detect" phase. Denote the estimate from the k-th iteration by $\hat{\boldsymbol{P}}_k$. Suppose that the processing is such that (i) $\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t + \boldsymbol{w}_t$ where $\boldsymbol{w}_t$ is sparse and data-dependent noise whose support equals the set of outlier entries at time t; and (ii) $q_k := \max_t \|\boldsymbol{M}_{s,t}\boldsymbol{P}\|$ is proportional to the subspace recovery error from iteration $k-1$, i.e., $q_k = C\ \mathrm{SE}(\hat{\boldsymbol{P}}_{k-1}, \boldsymbol{P})$ and $q_k < 2$. In defining q_k, the max is taken over the mini-batch used in iteration k. We can use our PCA result to show that $\mathrm{SE}(\hat{\boldsymbol{P}}_k, \boldsymbol{P}) \lessapprox \sqrt{b}(2q_k + q_k^2)\kappa \leq \kappa\sqrt{b}6q_k$ and thus $q_{k+1} = C\ \mathrm{SE}(\hat{\boldsymbol{P}}_k, \boldsymbol{P}) \lessapprox 6C\kappa\sqrt{b}q_k = 6C\kappa\sqrt{b}\mathrm{SE}(\hat{\boldsymbol{P}}_{k-1}, \boldsymbol{P})$. Thus, if b is small enough, clearly, q_{k+1}, and hence, $\mathrm{SE}(\hat{\boldsymbol{P}}_k, \boldsymbol{P})$, decreases by a constant fraction in each new iteration (the decay is geometric). Since the error in recovering $\boldsymbol{\ell}_t$ satisfies $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\|/\|\boldsymbol{\ell}_t\| \leq q_k$, this also decays geometrically with each iteration.

3.2 PCA in Data-Dependent Noise

3.2.1 Problem Setting

For $t = 1, 2, \cdots, \alpha$ we are given $\boldsymbol{y}_t \in \mathbb{R}^n$ that satisfies

$$\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{v}_t, \quad \text{where } \ \boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t, \ \ \boldsymbol{w}_t = \boldsymbol{M}_t\boldsymbol{\ell}_t, \tag{3.1}$$

$\boldsymbol{P}$ is an $n \times r$ basis matrix with $r \ll n$; $\boldsymbol{\ell}_t$ is the true data vectors that lies in an r-dimensional subspace of $\boldsymbol{R}^n$, span($\boldsymbol{P}$); $\boldsymbol{a}_t$'s are the projections of $\boldsymbol{\ell}_t$'s onto this subspace; $\boldsymbol{w}_t$ is data-dependent noise with $\boldsymbol{M}_t$ being the data-dependency matrix at time t; and $\boldsymbol{v}_t$ is uncorrelated noise. This means that $\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{v}_t{}'] = 0$ for all times t. Here $\boldsymbol{a}_t$ and $\boldsymbol{v}_t$ are treated as random variables (r.v.), while

everything else is deterministic. The goal is to estimate span($\boldsymbol{P}$) from the observed data stream $\boldsymbol{y}_t$, $t = 1, 2, \ldots, \alpha$.

3.2.2 SVD solution and guarantee for it

SVD Solution. We compute the subspace estimate $\hat{\boldsymbol{P}}$ as the matrix of top r left singular vectors of $\boldsymbol{Y} := [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_\alpha]$. Equivalently it is the matrix of top r eigenvectors of $\frac{1}{\alpha}\sum_t \boldsymbol{y}_t \boldsymbol{y}_t{}'$.

We make the following assumptions on the subspace coefficients, $\boldsymbol{a}_t$, and the uncorrelated noise, $\boldsymbol{v}_t$.

Assumption 3.29 (Statistical Assumption on $\boldsymbol{a}_t$)**.** *Assume that the $\boldsymbol{a}_t$'s are zero mean; mutually independent; have identical diagonal covariance matrix $\boldsymbol{\Lambda}$, i.e., that $\mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t{}'] = \boldsymbol{\Lambda}$; and are bounded:* $\max_t \|\boldsymbol{a}_t\|^2 \leq \mu r \lambda_{\max}(\boldsymbol{\Lambda})$. *Define* $\lambda^+ := \lambda_{\max}(\boldsymbol{\Lambda})$, $\lambda^- := \lambda_{\min}(\boldsymbol{\Lambda})$, $f := \frac{\lambda^+}{\lambda^-}$.

As we explain in Sec. 3.3, this assumption is almost equivalent to assuming μ-incoherence of the right singular vectors of the matrix $\boldsymbol{L} := [\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots, \boldsymbol{\ell}_\alpha]$. We call it μ statistical right incoherence there.

Assumption 3.30 (Statistical Assumption on $\boldsymbol{v}_t$)**.** *Assume that $\boldsymbol{v}_t$ is uncorrelated with $\boldsymbol{\ell}_t$, i.e.,* $\mathbb{E}[\boldsymbol{\ell}_t \boldsymbol{v}_t{}'] = 0$, *and $\boldsymbol{v}_t$'s are zero-mean, independent and identically distributed (i.i.d.) with covariance $\boldsymbol{\Sigma}_v := \mathbb{E}[\boldsymbol{v}_t \boldsymbol{v}_t{}']$, and are bounded. Let $\lambda_v^+ := \|\boldsymbol{\Sigma}_v\|$ be the noise power and let $r_v := \frac{\max_t \|\boldsymbol{v}_t\|_2^2}{\lambda_v^+}$ be the effective noise dimension.*

For a decomposition of the data-dependency matrix $\boldsymbol{M}_t$ as $\boldsymbol{M}_t = \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}$ with $\|\boldsymbol{M}_{2,t}\| = 1$, let

$$q := \max_t \|\boldsymbol{M}_{1,t}\boldsymbol{P}\|, \text{ and} \tag{3.2}$$

$$b := \left\| \frac{1}{\alpha} \sum_{t=1}^{\alpha} \boldsymbol{M}_{2,t} \boldsymbol{M}_{2,t}{}' \right\|. \tag{3.3}$$

Observe that $b \leq \max_t \|\boldsymbol{M}_{2,t}\|^2 = 1$. In many settings, for example, when $\boldsymbol{w}_t$ is sparse with changing support, b is much smaller than one. Our result given below exploits this fact.

Theorem 3.31 (PCA in Data-Dependent Noise)**.** *Consider the data* $\boldsymbol{y}_t$ *defined by* (3.1)*; and assume that Assumptions 3.29 and 3.30 hold. Also assume that* $\boldsymbol{w}_t = \boldsymbol{M}_t\boldsymbol{\ell}_t$ *with the parameters* b, q *satisfying* $b < 1$*,* $q < 2$*, and* $4\sqrt{b}qf + \frac{\lambda_v^+}{\lambda^-} + H(\alpha) + H_{denom}(\alpha) < 1$*. Here,*

$$H(\alpha) := C\sqrt{\eta}qf\sqrt{\frac{r\log n}{\alpha}} + C\sqrt{\eta}\sqrt{\frac{\lambda_v^+}{\lambda^-}}f\sqrt{\frac{r\log n}{\alpha}},$$
$$H_{denom}(\alpha) := C\sqrt{\eta}f\sqrt{\frac{r\log n}{\alpha}}. \tag{3.4}$$

Then, with probability at least $1-10n^{-10}$*, the matrix of top* r *eigenvectors of* $\frac{1}{\alpha}\sum_t \boldsymbol{y}_t\boldsymbol{y}_t'$*,* $\hat{\boldsymbol{P}}$*, satisfies*

$$\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le \frac{4\sqrt{b}qf + \frac{\lambda_v^+}{\lambda^-} + H(\alpha)}{1 - 4\sqrt{b}qf - \frac{\lambda_v^+}{\lambda^-} - H(\alpha) - H_{denom}(\alpha)}$$

Theorem 3.31 is proved in Appendix 3.10. It uses the Davis-Kahan $\sin\Theta$ theorem [7] followed by matrix Bernstein [30] to bound each term. To understand Theorem 3.31 simply, first assume that $\boldsymbol{v}_t = 0$ and $H(\alpha), H_{denom}(\alpha)$ are small enough (α is large enough). From the definition of q, the instantaneous signal-noise correlation $\|\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{w}_t']\| \le q\lambda^+$ and the instantaneous data-dependent noise power $\|\mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t']\| \le q^2\lambda^+$. Thus q^2 is the data-dependent noise-to-signal ratio. Also, λ^+ and λ^- quantify the maximum and the minimum signal power respectively. The PCA subspace recovery error depends on the ratio between the sum of (time-averaged values of) signal-noise correlation and noise power and the minimum signal space eigenvalue λ^-. By Cauchy-Schwarz, it is not hard to see that the time-averaged values of both these quantities satisfies $\|\frac{1}{\alpha}\sum_{t=1}^{\alpha}\mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t']\| \le \sqrt{b}q^2\lambda^+$ and $\|\frac{1}{\alpha}\sum_{t=1}^{\alpha}\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{w}_t']\| \le \sqrt{b}q\lambda^+$. Thus, if $b \ll 1$, the time-averaged values are significantly smaller than the instantaneous ones and this is what helps us get a small bound on the subspace recovery error. For a constant $c_1 < 1$, by assuming $b < (c_1/4f)^2$, we can ensure that $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le c_1 q$, i.e., the subspace recovery error is a fraction of q.

In the general case when $\boldsymbol{v}_t \neq 0$, we can guarantee that $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P})$ is at most $c_1 \max(q, \lambda_v^+/\lambda^-)$.

3.2.3 Application to PCA in Sparse Data-Dependent Noise (PCA-SDDN)

An important application of the above result is for data-dependent noise, $\boldsymbol{w}_t$, that is sparse. In this work we will show how a guarantee for PCA in sparse data-dependent noise (PCA-SDDN)

helps obtain a fast and delay-optimal robust ST algorithm. If we set $\boldsymbol{M}_{2,t} = \boldsymbol{I}_{\mathcal{T}_t}$ then $\boldsymbol{w}_t$ is sparse with support $\mathcal{T}_t$. Thus for $t = 1, 2, \cdots, \alpha$

$$\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{v}_t, \text{ where } \boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t, \ \boldsymbol{w}_t = \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{M}_{s,t}\boldsymbol{\ell}_t, \tag{3.5}$$

The assumption on b is now equivalent to a bound on the maximum fraction of non-zero entries in any row of $\boldsymbol{W} := [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_\alpha]$. To see why this is true, notice that $b = \frac{1}{\alpha}\|\sum_{t=1}^{\alpha} \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\|$. The matrix $\sum_t \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'$ is a diagonal matrix with (i,i)-th entry equal to the number of times t for which $i \in \mathcal{T}_t$. This is the same as the number of nonzero entries in the i-th row of $\boldsymbol{W}$. Using this fact we get the following corollary.

Corollary 3.32 (PCA in Sparse Data-Dependent Noise)**.** *Assume that $\boldsymbol{y}_t$'s satisfy (3.5), Assumptions 3.29, 3.30 hold, and $q := \max_t \|\boldsymbol{M}_{s,t}\boldsymbol{P}\| \leq 2$. Let b denote the maximum fraction of nonzeros in any row of the noise matrix $[\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_\alpha]$, and let $g := \frac{\lambda_v^+}{\lambda^-}$. For an $\epsilon_{\text{SE}} > 0$, if*

$$4\sqrt{b}qf + g < 0.4\epsilon_{\text{SE}},$$

and if

$$\alpha \geq \alpha^* := C \max\left(\frac{q^2 f^2}{\epsilon_{\text{SE}}^2} r \log n, \frac{gf}{\epsilon_{\text{SE}}^2} \max(r_v, r) \log n\right),$$

then w.p. at least $1 - 10n^{-10}$, $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon_{\text{SE}}$.

This corollary follows from Theorem 3.31 by picking α large enough so that $H(\alpha) < \epsilon_{\text{SE}}/10$ and $H_{denom}(\alpha) < 1/10$ (since this term appears in the denominator, we do not need it to be smaller than ϵ_{SE}, just a constant upper bound suffices).

Corollary 3.32 shows that it is possible to achieve recovery error that is a fraction of q, i.e, $\epsilon_{\text{SE}} = c_1 q$, if (i) $4\sqrt{b}f \leq 0.8c_1$ (the data-dependent noise support changes enough over time so that b is small), (ii) $\lambda_v^+ \leq 0.8c_1\epsilon_{\text{SE}}\lambda^-$ (the uncorrelated noise power is small enough), and (iii) $\alpha \geq C\max(f^2 r \log n, f\frac{1}{\epsilon_{\text{SE}}}\max(r_v, r)\log n)$. Notice that the sample complexity α increases with $1/\epsilon_{\text{SE}} = 1/(c_1 q)$. However, if we can make a stronger assumption that $\lambda_v^+ \leq 0.8c_1\epsilon_{\text{SE}}^2\lambda^-$, then we only need $\alpha \geq C\max(f^2 r \log n, f\max(r_v, r)\log n)$. Furthermore if $r_v \leq Cr$, then just $\alpha \geq$

$Cf^2 r \log n$ suffices. Treating f as a numerical constant, observe that this sample complexity is order-wise near-optimal: r is the minimum number of samples needed to even define a subspace.

In particular, in the setting when $\boldsymbol{v}_t = 0$, if the noise support changes enough so that b is small enough, we can estimate the subspace to a fraction of the square root of the noise level, q, using just order $r \log n$ samples. The reason this is possible is because the $\boldsymbol{a}_t$'s are bounded and $\boldsymbol{w}_t = \boldsymbol{M}_t \boldsymbol{P} \boldsymbol{a}_t$ and so the "randomness" in $\boldsymbol{w}_t$ is only r-dimensional (this has implications for what matrix Bernstein returns for the required sample complexity). When $\boldsymbol{v}_t \neq 0$, we have a similar result: if $\boldsymbol{v}_t$ has effective dimension that is of order r, we can still track to $\epsilon_{\text{SE}} = c \max(q, \sqrt{g})$, here g is the square root of uncorrelated noise level.

3.2.4 Generalizations of Theorem 3.31

For notational simplicity, in Theorem 3.31, we have provided a simple result that suffices for the correctness proof of our robust ST algorithm. We state and prove a much more general result in Appendix that relaxes this result in three ways. First, it replaces the identically distributed assumption on $\boldsymbol{a}_t$ and $\boldsymbol{v}_t$ by the following: let $\bar{\boldsymbol{\Lambda}} := \sum_t \boldsymbol{\Lambda}_t / \alpha$, $\lambda^-_{\text{avg}} := \lambda_{\min}(\bar{\boldsymbol{\Lambda}})$, $\lambda^+_{\max} := \max_t \lambda_{\max}(\boldsymbol{\Lambda}_t)$ and $\lambda^+_{v,\max} := \max_t \lambda_{\max}(\boldsymbol{\Sigma}_{v,t})$. It requires that the distributions are "similar" enough so that $f := \lambda^+_{\max} / \lambda^-_{\text{avg}}$ is bounded by a numerical constant and $\lambda^+_{v,\max}$ replaces λ^+_v in $H(\alpha)$ and $H_{denom}(\alpha)$ expressions.

Secondly, it replaces λ^+_v by $\|\boldsymbol{P}'\boldsymbol{\Sigma}_v\boldsymbol{P}_\perp\|$ in the numerator, while $-\lambda^+_v$ in the denominator gets replaced by $-(\lambda_{\max}(\boldsymbol{\Sigma}_v - \boldsymbol{P}\boldsymbol{P}'\boldsymbol{\Sigma}_v\boldsymbol{P}\boldsymbol{P}') - \lambda_{\min}(\boldsymbol{P}'\boldsymbol{\Sigma}_v\boldsymbol{P}))$. Here again, in case of time-varying statistics, the minimum eigenvalues get replaced by the minimum eigenvalue of the average covariance matrix while the maximum ones get replaced by the maximum eigenvalue over all times t. Thirdly, we also provide a guarantee for the case when $\boldsymbol{a}_t$'s and $\boldsymbol{v}_t$'s are sub-Gaussian random vectors. In this case, the required sample complexity increases to order n instead of $\max(r, r_v) \log n$ that we have for the bounded case result given above.

These last two changes allow us to recover the well known result for PCA under the Gaussian spiked covariance model (uncorrelated isotropic noise) [21] as a special case of our most general

result. Spiked covariance means $\boldsymbol{w}_t = 0$ and $\boldsymbol{\Sigma}_v = \lambda_v^+ \boldsymbol{I}$. Thus, $q = 0$, $\|\boldsymbol{P}'\boldsymbol{\Sigma}_v\boldsymbol{P}_\perp\| = 0$ and $\|\boldsymbol{\Sigma}_v - \boldsymbol{P}'\boldsymbol{\Sigma}_v\boldsymbol{P}\| - \|\boldsymbol{P}'\boldsymbol{\Sigma}_v\boldsymbol{P}\| = 0$ and so we get the following corollary.

Corollary 3.33 (Spiked Covariance Model, Gaussian noise [21])**.** *In the setting of Theorem 3.31, if $\boldsymbol{w}_t = 0$ (no data-dependent noise), $\boldsymbol{\Sigma}_v = \lambda_v^+ \boldsymbol{I}$, and $\boldsymbol{a}_t$, $\boldsymbol{v}_t$ are Gaussian, then, w.p. at least $1-5\exp(-cn)$, $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \frac{H(\alpha)}{1-H(\alpha)-H_{denom}(\alpha)}$. with $H(\alpha) = C\sqrt{\eta}\sqrt{gf}\sqrt{\frac{n}{\alpha}}$, $H_{denom}(\alpha) = C\sqrt{\eta}f\sqrt{\frac{n}{\alpha}}$ and $g = \frac{\lambda_v^+}{\lambda^-}$.*

If $\boldsymbol{a}_t$, $\boldsymbol{v}_t$ are bounded then $H(\alpha), H_{denom}(\alpha)$ are as given in Theorem 3.31.

Notice that, under the spiked covariance model, as long as we let the sample complexity α grow with the noise level g, we do not need any bound on noise power. For example, the noise power λ_v^+ could even be larger than λ^-. This is possible because, under this model, $\mathbb{E}[\sum_t \boldsymbol{y}_t\boldsymbol{y}_t'/\alpha] = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}' + \lambda_v^+\boldsymbol{I}$. Thus, its matrix of top r eigenvectors equals $\boldsymbol{P}$. As a result, the error between $\hat{\boldsymbol{P}}$ and $\boldsymbol{P}$ is only due to the fact that we are using a finite α to approximate the expected value. In other words, we only have statistical error. The "bias" terms are zero.

3.3 Nearly Optimal Robust Subspace Tracking (NORST)

In this section, we define the robust ST problem, explain the assumptions needed to make it identifiable, and then explain our proposed mini-batch solution and its guarantee.

3.3.1 Problem setting and algorithm design constraints

At each time t, we observe a data vector $\boldsymbol{y}_t \in \mathbb{R}^n$ that satisfies

$$\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{x}_t + \boldsymbol{v}_t, \text{ for } t = 1, 2, \dots, d \tag{3.6}$$

where $\boldsymbol{v}_t$ is small unstructured noise, $\boldsymbol{x}_t$ is the sparse outlier vector, and $\boldsymbol{\ell}_t$ is the true data vector that lies in a fixed or slowly changing low-dimensional subspace of $\mathbb{R}^n$, i.e.,

$$\boldsymbol{\ell}_t = \boldsymbol{P}_{(t)}\boldsymbol{a}_t$$

where $\boldsymbol{P}_{(t)}$ is an $n \times r$ basis matrix with $r \ll n$ and with $\|(\boldsymbol{I} - \boldsymbol{P}_{(t-1)}\boldsymbol{P}_{(t-1)}')\boldsymbol{P}_{(t)}\|$ small compared to $\|\boldsymbol{P}_{(t)}\| = 1$. We use $\mathcal{T}_t$ to denote the support set of $\boldsymbol{x}_t$. As an example, in the video application, $\boldsymbol{y}_t$ is

the video image at time/frame t, $\boldsymbol{\ell}_t$ is the background at time t, $\mathcal{T}_t$ is the support of the foreground at t, and $\boldsymbol{x}_t$ equals the difference between foreground and background images on $\mathcal{T}_t$ while being zero everywhere else. Slow subspace change is typically a valid assumption for background images of videos taken using a static camera. Given a good initial subspace estimate, $\hat{\boldsymbol{P}}_0$, the goal is to develop a mini-batch algorithm to track $\text{span}(\boldsymbol{P}_{(t)})$ and $\boldsymbol{\ell}_t$ either immediately or within a short delay. A by-product is that $\boldsymbol{x}_t$, and $\mathcal{T}_t$ can also be tracked accurately. The initial subspace estimate, $\hat{\boldsymbol{P}}_0$, can be computed by applying a few iterations of any existing RPCA solutions, e.g., PCP [3] or AltProj [25], on the first order r data points, i.e., on $\boldsymbol{Y}_{[1,t_{\text{train}}]}$, with $t_{\text{train}} = Cr$.

Dynamic RPCA. This is the offline version of the above problem. Define matrices $\boldsymbol{L}, \boldsymbol{X}, \boldsymbol{V}, \boldsymbol{Y}$ with $\boldsymbol{L} = [\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots \boldsymbol{\ell}_d]$ and with $\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{V}$ similarly defined. The goal is to recover $\boldsymbol{L}$ and its column space with accuracy ε. We use $r_{_L}$ to denote the rank of $\boldsymbol{L}$. The maximum fraction of nonzeros in any row (column) of the outlier matrix $\boldsymbol{X}$ is denoted by max-outlier-frac-row (max-outlier-frac-col).

Algorithm constraints. We will develop a nearly real-time tracking algorithm that (i) computes *an* online estimate of $\boldsymbol{x}_t$ and its support $\mathcal{T}_t$, and of $\boldsymbol{\ell}_t$ immediately at each time t using the previous subspace estimate, $\hat{\boldsymbol{P}}_{(t-1)}$, and observed data $\boldsymbol{y}_t$; (ii) it updates the subspace estimates in a mini-batch fashion; and (iii) it provides improved smoothing estimates of all quantities after a delay that is within log factors of r. As we explain in Sec. 3.3.5, recovering $\boldsymbol{x}_t$, $\mathcal{T}_t$, and $\boldsymbol{\ell}_t$ one at a time is the only way to obtain improved row-wise outlier tolerance compared to standard RPCA. However with doing this, correct recovery requires one extra assumption: slow enough subspace change compared to the minimum outlier magnitude.

3.3.2 Nearly Optimal Robust ST (NORST) via Recursive Projected Compressive Sensing (CS): main idea

The algorithm begins with an initial subspace estimate $\hat{\boldsymbol{P}}_0$. At each time t, we use $\hat{\boldsymbol{P}}_{(t-1)}$ and $\boldsymbol{y}_t$ to solve a noisy projected compressive sensing (CS) problem to estimate $\boldsymbol{x}_t$ and its support $\mathcal{T}_t$ from $\tilde{\boldsymbol{y}}_t = \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{b}_t$. Here $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$, $\tilde{\boldsymbol{y}}_t = \boldsymbol{\Psi}\boldsymbol{y}_t$, and $\boldsymbol{b}_t = \boldsymbol{\Psi}\boldsymbol{\ell}_t + \boldsymbol{\Psi}\boldsymbol{v}_t$ (is small under the slow subspace change assumption). This step uses l_1 minimization followed by thresholding to

estimate $\mathcal{T}_t$, and Least Squares (LS) on $\hat{\mathcal{T}}_t$ to get $\hat{\boldsymbol{x}}_t$. We compute $\hat{\boldsymbol{\ell}}_t$ by subtraction as $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$. Every α time instants, we update the subspace estimate by solving the PCA problem using the previous α $\hat{\boldsymbol{\ell}}_t$'s as observed data, i.e., by r-SVD on $\hat{\boldsymbol{L}}_{t;\alpha}$. This is repeated K times, each time with a new set of α $\hat{\boldsymbol{\ell}}_t$'s. At this point, the algorithm enters the subspace change detect phase. The complete algorithm is specified in Algorithm 6, and explained in detail in Sec 3.3.7. Besides α and K, it has two other parameters: ξ (assumed upper bound on $\|\boldsymbol{b}_t\|$) and ω_{supp} (threshold used for support recovery).

3.3.3 Identifiability and other assumptions

For this discussion assume that $\boldsymbol{v}_t = 0$. At each time t we have just one n-length observed data vector $\boldsymbol{y}_t$ but the subspace $\boldsymbol{P}_t$ is specified by nr scalars (it is an r-dimensional subspace of $\mathbb{R}^n$). Thus, even if we had perfect data $\boldsymbol{y}_t = \boldsymbol{\ell}_t$ available, it would be impossible to estimate each different $\boldsymbol{P}_t$. One way to address this is by assuming that the $\boldsymbol{P}_t$'s do not change for at least r time instants.

Assumption 3.34 (Piecewise Constant Subspace Change)**.** *Let $t_1, \dots t_j, \dots t_J$ denote the subspace change times. Let $t_0 = 1$ and $t_{J+1} = d$. Assume that*

$$\boldsymbol{P}_{(t)} = \boldsymbol{P}_j \text{ for all } t \in [t_j, t_{j+1}),\ j = 1, 2, \dots, J,$$

with $t_{j+1} - t_j > r$. Since $\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{x}_t$ (is imperfect), our guarantee needs a larger lower bound than r.

Even with the above assumption, a sparse $\boldsymbol{x}_t$ and its support $\mathcal{T}_t$ cannot be correctly distinguished from $\boldsymbol{\ell}_t = \boldsymbol{P}_j \boldsymbol{a}_t$ without more assumptions. Correct recovery of $\boldsymbol{x}_t$ and $\mathcal{T}_t$ requires that (i) the $\boldsymbol{x}_t$'s are sparse enough (ensured by bounding the maximum allowed outlier fractions per column), (ii) the columns of $\boldsymbol{P}_j$ are not sparse (ensured by the standard incoherence/denseness assumption from the RPCA literature [4, 3, 25]), and (iii) the $\boldsymbol{a}_t$'s are bounded. (iv) Correct support recovery also requires subspace change that is slow enough compared to the minimum nonzero entry of $\boldsymbol{x}_t$ (minimum outlier magnitude), denoted $x_{\min}$. Correct subspace update requires that (v) the $r \times \alpha$

sub-matrices formed by a mini-batch of $\boldsymbol{a}_t$'s are well-conditioned, and (vi) the outlier support $\mathcal{T}_t$ changes enough over time so that there is at least one outlier-free observation of each scalar entry of $\boldsymbol{\ell}_t$ in each mini-batch of $\boldsymbol{y}_t$'s. One way to ensure (v) is to assume that the $\boldsymbol{a}_t$'s are i.i.d. while (vi) can be ensured by bounding the maximum fraction of outliers in any row of any α-mini-batch sub-matrix of $\boldsymbol{X}$. We use max-outlier-frac-row(α) to denote this quantity. We summarize the above assumptions on $\boldsymbol{P}_j$'s and $\boldsymbol{a}_t$'s in Assumption 3.35, those on the outlier fractions in Assumption 3.36, and slow subspace change compared to $x_{\min}$ in Assumption 3.37.

Assumption 3.35 (μ-Incoherence)**.** *Assume the following.*

1. *(Left Incoherence) Assume that $\boldsymbol{P}_j$'s are μ-incoherent with μ being a numerical constant. This means that $\max_{i=1,2,..,n} \|(\boldsymbol{P}_j)^{(i)}\|^2 \leq \mu r/n$. Here $\boldsymbol{P}^{(i)}$ denotes the i-th row of $\boldsymbol{P}$.*

2. *(Statistical Right Incoherence) Assume Assumption 3.29, i.e., the subspace coefficients $\boldsymbol{a}_t$ are zero mean, mutually independent, have identical diagonal covariance matrix $\boldsymbol{\Lambda} := \mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t']$, and are bounded: $\max_t \|\boldsymbol{a}_t\|^2 \leq \mu r \lambda_{\max}(\boldsymbol{\Lambda})$. Let λ^+ (λ^-), $f := \lambda^+/\lambda^-$ denote the maximum (minimum) eigenvalue and condition number of $\boldsymbol{\Lambda}$.*

The second assumption above allows us to obtain high probability upper bounds on the tracking delay of our approach. As we explain later in Sec. 3.3.6, it can be interpreted as a statistical version of right singular vectors' incoherence. The incoherence assumption on $\boldsymbol{P}_j$ is nearly equivalent to left singular vectors' incoherence. It is exactly equivalent if we consider the sub-matrices $\boldsymbol{L}_j := [\boldsymbol{\ell}_{t_j}, \boldsymbol{\ell}_{t_j+1}, \ldots, \boldsymbol{\ell}_{t_{j+1}-1}]$.

Assumption 3.36 (Outliers are spread out)**.** *Let max-outlier-frac-col $:= \max_t |\mathcal{T}_t|/n$; let max-outlier-frac-row(α) be the maximum fraction of nonzeros per row of any sub-matrix of $\boldsymbol{X}_{[t_{\text{train}},d]}$ with α consecutive columns, and let max-outlier-frac-row$_{\text{init}}$ be the maximum fraction of outliers per row of any sub-matrix of $\boldsymbol{X}_{[1,t_{\text{train}}]}$. Assume that max-outlier-frac-col $\leq \frac{c_1}{\mu r}$, max-outlier-frac-row(α) $\leq \frac{c_2}{f^2}$, and max-outlier-frac-row$_{\text{init}} \leq \frac{c_3}{r}$.*

Assumption 3.37 (Slow subspace change)**.** *Let $x_{\min} := \min_t \min_{i \in \mathcal{T}_t} |(\boldsymbol{x}_t)_i|$ and let $\text{SE}_j := \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$. Assume that $\text{SE}_j \leq 0.8$ and $\text{SE}_j \leq \frac{c_4}{\sqrt{r}} \frac{x_{\min}}{\sqrt{\lambda^+}}$.*

The order notation used here and below assumes that f, μ are constants.

3.3.4 Guarantees

Before stating our main result, we define a few terms next.

Definition 3.38. *Let the mini-batch size $\alpha := Cf^2 r \log n$, the number of subspace update iterations needed to get an ε accurate estimate, $K = K(\varepsilon) := C \log(\Delta/\varepsilon)$, where $\Delta := \max_j \mathrm{SE}_j$, noise power $\lambda_v^+ := \max_t \|\mathbb{E}[\boldsymbol{v}_t \boldsymbol{v}_t']\|$, and effective noise dimension, $r_v := \frac{\max_t \|\boldsymbol{v}_t\|^2}{\lambda_v^+}$. Recall from Algorithm 6 that $\hat{t}_j$ denotes the time at which the j-th subspace change is detected.*

We have the following result.

Theorem 3.39. *Assume that Assumptions 3.34, 3.35, 3.36, and 3.37 hold. Assume that the noise $\boldsymbol{v}_t$ is bounded, i.i.d. over time, independent of $\mathcal{T}_t$, uncorrelated with $\boldsymbol{\ell}_t$, i.e., $\mathbb{E}[\boldsymbol{\ell}_t \boldsymbol{v}_t'] = 0$, and with $r_v \le Cr$, and $\sqrt{\lambda_v^+/\lambda^-} < 0.01$. Also, assume that $\boldsymbol{\ell}_t$'s and $\mathcal{T}_t$'s are independent.*

Pick an ε that satisfies $c\sqrt{\lambda_v^+/\lambda^-} \le \varepsilon \le \min\left(c_3 \frac{1}{\sqrt{r}} \frac{x_{\min}}{\sqrt{\lambda^+}}, 0.01\right)$. Consider Algorithm 6 with $K = K(\varepsilon)$ as defined above, $\alpha = Cf^2 r \log n$, $\omega_{evals} = 2\varepsilon^2 \lambda^+$, $\zeta = x_{\min}/15$ and $\omega_{supp} = x_{\min}/2$. If

1. $\max_t \|\boldsymbol{v}_t\| \le c_5 x_{\min}$,
2. $t_{j+1} - t_j > (K+2)\alpha$, *and* $\mathrm{SE}_j > 9\sqrt{f}\varepsilon$
3. *initialization*[3]*:* $\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \le \min\left(c_6 \frac{1}{\sqrt{r}} \frac{x_{\min}}{\sqrt{\lambda^+}}, 0.25\right)$;

then, w.p. at least $1 - 10dn^{-10}$,

1. $t_j \le \hat{t}_j \le t_j + 2\alpha$,

$$\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) \le \begin{cases} (\varepsilon + \mathrm{SE}_j) & \text{if } t \in [t_j, \hat{t}_j + \alpha), \\ (0.3)^{k-1}(\varepsilon + \mathrm{SE}_j) & \text{if } t \in [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha), \\ \varepsilon & \text{if } t \in [\hat{t}_j + K\alpha + \alpha, t_{j+1}), \end{cases}$$

and $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\| \le 1.2\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)})\|\boldsymbol{\ell}_t\| + \|\boldsymbol{v}_t\|$;

[3] *This can be satisfied by using $C \log r$ iterations of AltProj [25] on the first $t_{\mathrm{train}} = Cr$ data samples.*

2. $\hat{\mathcal{T}}_t = \mathcal{T}_t$ *and the bound on* $\|\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t\|$ *is the same as that on* $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\|$.

The time complexity is $O(ndr\log(1/\varepsilon))$ *and memory complexity is* $O(n\alpha) = O(f^2 nr\log n)$.

Proof. We prove this in Sec. 3.6. □

We have the following corollary for the smoothing NORST algorithm (last few lines of Algorithm 6). This is also a mini-batch approach with mini-batch size $(K+2)\alpha$ (instead of α for NORST).

Corollary 3.40. *[Smoothing NORST for dynamic RPCA] Under the assumptions of Theorem 3.39, the following also hold:* $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}^{\text{smoothing}}, \boldsymbol{P}_{(t)}) \le \varepsilon$, $\|\hat{\boldsymbol{\ell}}_t^{\text{smoothing}} - \boldsymbol{\ell}_t\| \le \varepsilon\|\boldsymbol{\ell}_t\| + \|\boldsymbol{v}_t\|$ *at all times* t. *Its time complexity is* $O(ndr\log(1/\varepsilon))$ *and memory complexity is* $O(Kn\alpha) = O(nr\log n\log(1/\varepsilon))$. *All these quantities are computed within a delay of at most* $(K+2)\alpha$.

The above result guarantees that NORST can detect subspace changes in delay at most $\alpha = Cr\log n$ and track them to ε accuracy in delay at most $(K+2)\alpha = Cr\log n\log(\Delta/\varepsilon)$. The corollary for smoothing NORST guarantees that, with this delay, each column of $\boldsymbol{L}$, $\boldsymbol{\ell}_t$, is recovered to ε relative accuracy. The minimum delay needed to compute an r-dimensional subspace even with perfect data $\boldsymbol{y}_t = \boldsymbol{\ell}_t$ is r. Thus, our result guarantees near optimal detection and tracking delay ("near optimal" means that it is within log factors of the minimum delay). Moreover, the required lower bound on the delay between subspace change times is also near optimal. Quick and reliable change detection is an important feature, e.g., this feature has been used in [26] to detect structural changes in a dynamic social network.

When the extra unstructured noise $\boldsymbol{v}_t = 0$, we can track to any $\varepsilon > 0$ otherwise we can track to $\varepsilon \ge \sqrt{\lambda_v^+/\lambda^-}$ (square root of the noise level). It is possible to slightly relax this requirement to $\varepsilon \ge \lambda_v^+/\lambda^-$ by picking a larger α, $\alpha = C(r\log n)(\lambda^-/\lambda_v^+)$, but it cannot be eliminated. The reason is that at each time t, we have an *under-determined* set of equations corrupted by unstructured noise $\boldsymbol{v}_t$. Even assuming the subspace is known or has been perfectly estimated, it is under-determined: we have $n+r$ unknowns at each time t but only n observed scalars. This is also true for any other under-determined problem as well, e.g., standard RPCA or CS[4].

[4]To address a reviewer comment, one cannot get a consistent estimator for our problem, nor for standard RPCA or CS. Consistent estimator means that the recovery error goes to zero as the number of observed data points increases.

Notice also that we have assumed that the "effective noise dimension", $r_v \in O(r)$. This requirement can be eliminated if we set $\alpha = Cf^2 \max(r, r_v) \log n$.

From the perspective of recovering the true data $\boldsymbol{\ell}_t$, both $\boldsymbol{v}_t$ and $\boldsymbol{x}_t$ are noise or perturbations. The difference is that $\boldsymbol{v}_t$ is a vector of small disturbances or modeling errors, while $\boldsymbol{x}_t$ is a sparse outlier vector with few nonzero entries. By definition, an outlier is an infrequent but large disturbance. Our result tolerates what can be called "bi-level perturbations": the small perturbation $\boldsymbol{v}_t$ needs to be small enough and the minimum outlier magnitude $x_{\min}$ needs to be large enough so that $\|\boldsymbol{v}_t\| \leq 0.2 x_{\min}$ (minimum outlier magnitude). Moreover, $x_{\min}$ also needs to be large enough to satisfy Assumption 3.37. The need for both these assumptions is explained in Sec. 3.3.5. Assuming that $x_{\min}^2$ is of order λ^+ (signal power), Assumption 3.37 requires that SE_j be $O(1/\sqrt{r})$. However this is not as restrictive as it may seem. The reason is that SE(.) is only measuring the sine of the largest principal angle. If all principal angles are roughly equal, then, this still allows the chordal subspace distance (l_2 norm of the vector of sines of all r principal angles) [37] to be $O(1)$.

Our result assumes a minor lower bound on SE_j. This is needed to guarantee reliable subspace change detection. Changes that are smaller than order ε cannot be detected when the previous subspace is only tracked to accuracy ε. However, such changes also increase the tracking error only by an extra factor of ε and hence can be treated as noise. If change detection is not important, then, as we explain in Sec. 3.5, we can use a simpler NORST algorithm that does not need the lower bound.

Consider the piecewise constant subspace change assumption. In practice, e.g., in the video application, typically the subspaces change by a little at each time. This can be modeled as piecewise constant subspaces plus modeling error $\boldsymbol{v}_t$. We explain this point in Sec. 3.5 where we also provide a corollary for this setting. This corollary explains why the NORST algorithm "works" (gives good, but not perfect, subspace estimates and estimates of $\boldsymbol{\ell}_t$) for real videos or for simulated

For example for Least Squares estimation, one can show the estimator is consistent. But this is true because number of observed data points increases while the number of unknowns remains constant. In our case, the number of unknowns also increases with time t: at each t, we have $(n + r)$ unknowns even if $\boldsymbol{P}_t$ has been estimated.

data generated so that $\boldsymbol{P}_t$ changes a little at each t; see Sec. 3.7 and more detailed experiments in [32].

To keep the theorem statement simple, we have used tighter bounds than required. Define the intervals $\mathcal{J}_{j,1} = [t_j, \hat{t}_j + \alpha)$, $\mathcal{J}_{j,k} := [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$ for $k = 2, 3, \ldots, K$, and $\mathcal{J}_{j,K+1} = [\hat{t}_j + (K+1)\alpha, t_{j+1})$. For $t \in \mathcal{J}_{j,k}$, for $k = 1, 2, \ldots, K$, we only need $0.3^{k-1}(\varepsilon + \mathrm{SE}_j)\sqrt{r\lambda^+} \leq c \min_{i \in \mathcal{T}_t} |(\boldsymbol{x}_t)_i|$, i.e., the required lower bound on the minimum outlier magnitude at time t decreases as the subspaces get estimated better. For the outliers $\boldsymbol{x}_t$ for $t \in \mathcal{J}_{j,K+1}$, we do not require any lower bound. Secondly, if the outlier vector is such that some entries are very small while the others are large enough, then we can treat the smaller entries as "noise" $\boldsymbol{v}_t$. This will work as long as these small entries are small enough so that the sum of their squares is sufficiently smaller than the square of the magnitude of the larger entries, i.e., for $t \in \mathcal{J}_{j,k}$, we can split $\boldsymbol{x}_t$ as $\boldsymbol{x}_t = (\boldsymbol{x}_t)_{small} + (\boldsymbol{x}_t)_{large}$ with the two components being such that $c\boldsymbol{x}_{t,large,min} \geq \|(\boldsymbol{x}_t)_{small}\|$ and $c\boldsymbol{x}_{t,large,min} \geq 0.3^{k-1}(\varepsilon + \max_j \mathrm{SE}_j)\sqrt{r\lambda^+}$. Finally, if we also state the PCA-SDDN result in its most general form, the subspace error decay rate of 0.3 can be replaced by $(6\sqrt{b_0}f)$ with $b_0 :=$ max-outlier-frac-row, so this requirement becomes $c\boldsymbol{x}_{t,large,min} \geq (6\sqrt{b_0}f)^{k-1}(\varepsilon + \max_j \mathrm{SE}_j)\sqrt{r\lambda^+}$. With this change, the expression for K becomes $K = \left\lceil \frac{\log(\Delta/\varepsilon)}{-\log(6\sqrt{b_0}f)} \right\rceil$. Thus, a smaller b_0 means that the subspace error decays faster. This, in turn, means that a smaller K suffices (faster tracking and a smaller required lower bound on $t_{j+1} - t_j$). It also means a smaller lower bound is needed on the outlier magnitudes at most times.

3.3.5 How slow subspace change (Assumption 3.37) enables improved outlier tolerance

We explain here how the use of Assumption 3.37 enables improved outlier tolerance. Briefly, the reason is we recover each outlier $\boldsymbol{x}_t$ and its support $\mathcal{T}_t$ individually. To understand things simply, assume $\boldsymbol{v}_t = 0$.

Given a good previous subspace estimate, $\hat{\boldsymbol{P}}_{(t-1)}$, slow subspace change implies that $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t-1)}, \boldsymbol{P}_t)$ is small. Consider an α length interval $\mathcal{J}$ during with $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}$ (computed

in the previous α interval). To exploit slow subspace change, we project each $\boldsymbol{y}_t$ orthogonal to $\hat{\boldsymbol{P}}$ to get $\tilde{\boldsymbol{y}}_t := \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{b}_t$ where $\boldsymbol{b}_t := \boldsymbol{\Psi}\boldsymbol{\ell}_t$ is small because of above. Here $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'$. Now $\boldsymbol{b}_t$ itself does not have any structure. But, the matrix $\boldsymbol{B}_{\mathcal{J}}$ formed by the $\boldsymbol{b}_t$'s for $t \in \mathcal{J}$, is low rank with rank r [5]. Accurately recovering $\boldsymbol{X}_{\mathcal{J}}$ from $\tilde{\boldsymbol{Y}}_{\mathcal{J}} := \boldsymbol{\Psi}\boldsymbol{X}_{\mathcal{J}} + \boldsymbol{B}_{\mathcal{J}}$ when $\boldsymbol{B}_{\mathcal{J}}$ has rank r is impossible if the fraction of outliers in any row or in any column of $\boldsymbol{X}_{\mathcal{J}}$ is more than c/r. The reasoning is the same as that used for standard RPCA [25]: we can construct a sparse matrix $\boldsymbol{X}_{\mathcal{J}}$ with rank $1/\max(\text{max-outlier-frac-row}, \text{max-outlier-frac-col})$. Thus if max-outlier-frac-row $= c$, we can construct a sparse $\boldsymbol{X}_{\mathcal{J}}$ with rank $1/c = C \ll r$ [6]. If the rank of $\boldsymbol{X}_{\mathcal{J}}$ is less than r, that of $\boldsymbol{\Psi}\boldsymbol{X}_{\mathcal{J}}$ will also be less than r, making the recovery problem un-identifiable: if we try to find a matrix $\hat{\boldsymbol{B}}_{\mathcal{J}}$ of rank at most r and a matrix $\hat{\boldsymbol{X}}_{\mathcal{J}}$ that is the sparsest and both satisfy $\tilde{\boldsymbol{Y}}_{\mathcal{J}} := \boldsymbol{\Psi}\hat{\boldsymbol{X}}_{\mathcal{J}} + \hat{\boldsymbol{B}}_{\mathcal{J}}$, it is possible that we get the solution $\hat{\boldsymbol{B}}_{\mathcal{J}} = \boldsymbol{B}_{\mathcal{J}} + \boldsymbol{\Psi}\hat{\boldsymbol{X}}_{\mathcal{J}}$ and $\hat{\boldsymbol{X}}_{\mathcal{J}} = \boldsymbol{0}$. Because rank of $\boldsymbol{\Psi}\hat{\boldsymbol{X}}_{\mathcal{J}}$ is less than r and that of $\boldsymbol{B}_{\mathcal{J}}$ is r, it is possible that the sum still has rank r.

Thus, if we would like to improve row-wise outlier tolerance to $O(1)$, we cannot jointly recover all columns of $\boldsymbol{X}_{\mathcal{J}}$ by exploiting the low rank structure of $\boldsymbol{B}_{\mathcal{J}}$. The only other way to proceed is as we do: recover them one $\boldsymbol{x}_t$ at a time from $\tilde{\boldsymbol{y}}_t$. Here we can only use the fact that $\|\boldsymbol{b}_t\|$ is small due to slow subspace change. The problem of recovering a single $\boldsymbol{x}_t$ from $\tilde{\boldsymbol{y}}_t$ is a standard noisy CS problem [2], with small noise $\boldsymbol{b}_t$. To our best knowledge, there are no entry-wise recovery guarantees for CS. One can only bound $\|\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t\|$ by a constant (that depends on the restricted isometry constant of $\boldsymbol{\Psi}$), C, times $\|\boldsymbol{b}_t\|$. Here $\hat{\boldsymbol{x}}_{t,cs}$ is the output of the CS step (line 7 of Algorithm 6). With this, correct support recovery, $\hat{\mathcal{T}}_t = \mathcal{T}_t$, is ensured only if $x_{\min} > 2C\|\boldsymbol{b}_t\|$. The worst case bound on $\|\boldsymbol{b}_t\|$ comes from when the subspace has changed but the change has not been detected so that $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_{j-1}$ and $\boldsymbol{P}_t = \boldsymbol{P}_j$. At this time, $\|\boldsymbol{b}_t\| \leq \max_j \text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j)\sqrt{r}\sqrt{\lambda^+}$. Also, we can show that $\text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j) \leq \text{SE}_j + \varepsilon$. Thus, exact support recovery is guaranteed if Assumption 3.37

[5] The effective (stable) rank, of $\boldsymbol{B}_{\mathcal{J}}$ will be less than r only if we assume more structure on subspace change, e.g., if we assume that only a few subspace directions change. Its exact rank will still be r.

[6] A simple way to do this would be as follows. Let $b_0 =$ max-outlier-frac-row and suppose b_0 is a constant (is more than order $1/r$). Let the support and nonzero entries of $\boldsymbol{X}_{\mathcal{J}}$ be constant for the first $b_0\alpha$ columns; after this, move the nonzero entries down in such a way that there is no overlap of supports; and repeat this every $b_0\alpha$ columns. With this, the max-outlier-frac-row $= b_0$ and the rank of $\boldsymbol{X}_{\mathcal{J}}$ is $\alpha/(b_0\alpha) = 1/b_0$ since there are only $1/b_0$ unique vectors in this matrix construction.

holds and ε is chosen as specified in the theorem. When $\boldsymbol{v}_t \neq 0$, the bound on $\|\boldsymbol{b}_t\|$ contains a $\|\boldsymbol{v}_t\|$ term. In this case, exact support recovery also needs $\|\boldsymbol{v}_t\| \leq c_5 x_{\min}$.

Exact support recovery followed by LS on the recovered support and then subtraction to get $\hat{\boldsymbol{\ell}}_t$ implies that $\hat{\boldsymbol{\ell}}_t$ satisfies $\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t + \boldsymbol{e}_t$ with $\boldsymbol{e}_t := -\boldsymbol{I}_{\mathcal{T}_t} \underbrace{(\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{I}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{\Psi}}_{\boldsymbol{M}_{s,t}} \boldsymbol{\ell}_t$. Notice that $\boldsymbol{e}_t$ is sparse and linearly data-dependent and, conditioned on $\hat{\boldsymbol{P}}$ and the support sets $\mathcal{T}_t$, the matrix $\boldsymbol{M}_{s,t}$ is deterministic. So we can apply the PCA-SDDN result from the previous section. It also needs statistical right incoherence, $q := \max_t \|\boldsymbol{M}_{s,t}\boldsymbol{P}_j\| \leq C\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}_j)$ (holds by left incoherence and max-outlier-frac-col $< c/r$), and max-outlier-frac-row$(\alpha) \leq c$ (constant row-wise outlier fraction bound). If the support recovery were incorrect, the estimated support $\hat{\mathcal{T}}_t$ would depend on $\boldsymbol{b}_t$ and hence on $\boldsymbol{\ell}_t$. This would mean that, even conditioned on $\hat{\boldsymbol{P}}$ and $\mathcal{T}_t$, the matrices $\boldsymbol{M}_{s,t}$ are not deterministic making the PCA-SDDN result inapplicable.

3.3.6 Understanding Statistical Right Incoherence

Let $\boldsymbol{L}_j := \boldsymbol{L}_{[t_j, t_{j+1})}$. From our assumptions, $\boldsymbol{L}_j = \boldsymbol{P}_j\boldsymbol{A}_j$ with $\boldsymbol{A}_j := [\boldsymbol{a}_{t_j}, \boldsymbol{a}_{t_j+1}, \ldots \boldsymbol{a}_{t_{j+1}-1}]$, the columns of $\boldsymbol{A}_j$ are zero mean, mutually independent, have identical covariance $\boldsymbol{\Lambda}$, $\boldsymbol{\Lambda}$ is diagonal, and bounded. Let $d_j := t_{j+1} - t_j$. Define a diagonal matrix Σ with (i,i)-th entry σ_i satisfying $\sigma_i^2 := \sum_t (a_t)_i^2/d_j$. Define a $d_j \times r$ matrix $\tilde{\boldsymbol{V}}$ with the t-th entry of the i-th column being $(\tilde{\boldsymbol{v}}_i)_t := (\boldsymbol{a}_t)_i/(\sigma_i\sqrt{d_j})$. Clearly, $\boldsymbol{L}_j = \boldsymbol{P}_j\Sigma\tilde{\boldsymbol{V}}'$ and each column of $\tilde{\boldsymbol{V}}$ is unit 2-norm. This can be interpreted as an approximation to the SVD of $\boldsymbol{L}_j$; we say approximation because the columns of $\tilde{\boldsymbol{V}}$ are not necessarily exactly mutually orthogonal. However, if d_j is large enough, one can argue using scalar Hoeffding inequality (applicable because $\boldsymbol{a}_t$'s are bounded), that, whp, (i) the columns of $\tilde{\boldsymbol{V}}$ are approximately mutually orthogonal, i.e. $|\tilde{\boldsymbol{v}}_i'\tilde{\boldsymbol{v}}_j| \leq \epsilon$ for all $i \neq j$; and (ii) $0.99\lambda_i \leq \sigma_i^2 \leq 1.01\lambda_i$ for all $i = 1, 2, \ldots, r$. Thus, by the boundedness assumption on the $\boldsymbol{a}_t$'s, the t-th row of $\tilde{\boldsymbol{V}}$ satisfies $\sum_{i=1}^r (\tilde{\boldsymbol{v}}_i)_t^2 \leq (1/d_j)(1/\min_i \sigma_i^2)\|\boldsymbol{a}_t\|^2 \leq (1/d_j)(1/\lambda^-)\mu r\lambda^+ = f\mu r/d_j$. This is the standard incoherence assumption with parameter $f\mu$. Thus, whp, the approximate right singular vectors' matrix $\tilde{\boldsymbol{V}}$ of $\boldsymbol{L}_j$ satisfies the standard incoherence assumption.

3.3.7 Nearly Optimal Robust ST via ReProCS (NORST-ReProCS): details

Algorithm 6 uses the Recursive Projected Compressive Sensing framework introduced in [28]. It starts with a "good" estimate of the initial subspace. This can be obtained by using a few iterations of AltProj applied to $\boldsymbol{Y}_{[1,t_{\text{train}}]}$ with $t_{\text{train}} = Cr$. It then iterates between (a) *Projected Compressive Sensing (CS) / Robust Regression*[7] in order to estimate the sparse outliers, $\boldsymbol{x}_t$'s, and hence the $\boldsymbol{\ell}_t$'s, and (b) *Subspace Update* to update the estimates $\hat{\boldsymbol{P}}_{(t)}$. Projected CS proceeds as follows. At time t, if the previous subspace estimate, $\hat{\boldsymbol{P}}_{(t-1)}$, is accurate enough, because of slow subspace change, projecting $\boldsymbol{y}_t$ onto its orthogonal complement will nullify most of $\boldsymbol{\ell}_t$. We compute $\tilde{\boldsymbol{y}}_t := \boldsymbol{\Psi}\boldsymbol{y}_t$ where $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$. Clearly $\tilde{\boldsymbol{y}}_t = \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$ and $\|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\|$ is small due to slow subspace change and small $\boldsymbol{v}_t$. Recovering $\boldsymbol{x}_t$ from $\tilde{\boldsymbol{y}}_t$ is now a CS / sparse recovery problem in small noise [2]. We compute $\hat{\boldsymbol{x}}_{t,cs}$ using noisy l_1 minimization followed by thresholding based support estimation to obtain $\hat{\mathcal{T}}_t$. A Least Squares (LS) based debiasing step on $\hat{\mathcal{T}}_t$ returns the final $\hat{\boldsymbol{x}}_t$. We then estimate $\boldsymbol{\ell}_t$ as $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$.

The $\hat{\boldsymbol{\ell}}_t$'s are then used for the Subspace Update step which toggles between the "detect" phase and the "update" phase. It starts in the "update" phase with $\hat{t}_0 = t_{\text{train}}$. We then perform K r-SVD steps with the k-th one done at $t = \hat{t}_0 + k\alpha - 1$. Each such step uses the last α estimates, i.e., uses $\hat{\boldsymbol{L}}_{t;\alpha}$. Thus at $t = \hat{t}_0 + K\alpha - 1$, the subspace update of $\boldsymbol{P}_0$ is complete. At this point, the algorithm enters the "detect" phase. For any j, if the j-th subspace change is detected at time t, we set $\hat{t}_j = t$. At this time, the algorithm enters the "update" (subspace update) phase. We then perform K r-SVD steps with the k-th r-SVD step done at $t = \hat{t}_j + k\alpha - 1$ on $\hat{\boldsymbol{L}}_{t;\alpha}$. Thus, at $t = \hat{t}_{j,fin} = \hat{t}_j + K\alpha - 1$, the update is complete. At this t, the algorithm enters the "detect" phase.

To understand the change detection strategy, consider the j-th subspace change. Assume that the previous subspace $\boldsymbol{P}_{j-1}$ has been accurately estimated by $t = \hat{t}_{j-1,fin} = \hat{t}_{j-1} + K\alpha - 1$ and that $\hat{t}_{j-1,fin} < t_j$. Let $\hat{\boldsymbol{P}}_{j-1}$ denote this estimate. At this time, the algorithm enters the

[7] *Robust Regression (with a sparsity model on the outliers) assumes that observed data vector $\boldsymbol{y}$ satisfies $\boldsymbol{y} = \hat{\boldsymbol{P}}\boldsymbol{a} + \boldsymbol{x} + \boldsymbol{b}$ where $\hat{\boldsymbol{P}}$ is a tall matrix (given), $\boldsymbol{a}$ is the vector of (unknown) regression coefficients, $\boldsymbol{x}$ is the (unknown) sparse outliers, $\boldsymbol{b}$ is (unknown) small noise/modeling error. An obvious way to solve this is by solving $\min_{\boldsymbol{a},\boldsymbol{x}} \lambda\|\boldsymbol{x}\|_1 + \|\boldsymbol{y} - \hat{\boldsymbol{P}}\boldsymbol{a} - \boldsymbol{x}\|^2$. In this, one can solve for $\boldsymbol{a}$ in closed form to get $\hat{\boldsymbol{a}} = \hat{\boldsymbol{P}}'(\boldsymbol{y} - \boldsymbol{x})$. Substituting this, the minimization simplifies to $\min_{\boldsymbol{x}} \lambda\|\boldsymbol{x}\|_1 + \|(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')(\boldsymbol{y} - \boldsymbol{x})\|^2$. This is equivalent to the Lagrangian version of the projected CS problem that NORST solves (given in line 7 of Algorithm 6).*

"detect" phase in order to detect the next (j-th) change. Let $\boldsymbol{B}_t := (\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}')\hat{\boldsymbol{L}}_{t;\alpha}$. At every $t = \hat{t}_{j-1,fin} + u\alpha - 1$, $u = 1, 2, \ldots$, we detect change by checking if the maximum singular value of $\boldsymbol{B}_t$ is above a pre-set threshold, $\sqrt{\omega_{evals}\alpha}$, or not. We claim that, with high probability (whp), under assumptions of Theorem 3.39, this strategy has no "false subspace detections" and correctly detects change within a delay of at most 2α samples. The former is true because, for any t for which $[t - \alpha + 1, t] \subseteq [\hat{t}_{j-1,fin}, t_j)$, all singular values of the matrix $\boldsymbol{B}_t$ will be close to zero (will be of order $\varepsilon\sqrt{\lambda^+}$) and hence its maximum singular value will be below $\sqrt{\omega_{evals}\alpha}$. Thus, whp, $\hat{t}_j \geq t_j$. To understand why the change *is* correctly detected within 2α samples, first consider $t = \hat{t}_{j-1,fin} + \lceil \frac{t_j - \hat{t}_{j-1,fin}}{\alpha} \rceil \alpha := t_{j,*}$. Since we assumed that $\hat{t}_{j-1,fin} < t_j$ (the previous subspace update is complete before the next change), t_j lies in the interval $[t_{j,*} - \alpha + 1, t_{j,*}]$. Thus, not all of the $\boldsymbol{\ell}_t$'s in this interval satisfy $\boldsymbol{\ell}_t = \boldsymbol{P}_j\boldsymbol{a}_t$. Depending on where in the interval t_j lies, the algorithm may or may not detect the change at this time. However, in the *next* interval, i.e., for $t \in [t_{j,*} + 1, t_{j,*} + \alpha]$, all of the $\boldsymbol{\ell}_t$'s satisfy $\boldsymbol{\ell}_t = \boldsymbol{P}_j\boldsymbol{a}_t$. We can prove that, whp, $\boldsymbol{B}_t$ for this time t *will* have maximum singular value that is above the threshold. Thus, if the change is not detected at $t_{j,*}$, whp, it *will* get detected at $t_{j,*} + \alpha$. Hence, whp, either $\hat{t}_j = t_{j,*}$, or $\hat{t}_j = t_{j,*} + \alpha$, i.e., $t_j \leq \hat{t}_j \leq t_j + 2\alpha$.

Algorithm parameters. Algorithm 6 assumes knowledge of 4 model parameters: r, λ^+, λ^- and $x_{\min}$ to set the algorithm parameters. The initial dataset used for estimating $\hat{\boldsymbol{P}}_0$ (using AltProj) can be used to get an accurate estimate of r, λ^- and λ^+ using standard techniques. Thus one really only needs to set $x_{\min}$. If continuity over time is assumed, we can let it be time-varying and set it as $\min_{i \in \hat{\mathcal{T}}_{t-1}} |(\hat{\boldsymbol{x}}_{t-1})_i|$ at t.

Time complexity. The time complexity is $O(ndr \log(1/\epsilon))$. We explain this in Supplement Appendix 3.12.1.

3.4 Related Work

We first briefly discuss related work on PCA and then discuss robust PCA and subspace tracking papers. While there has been a large amount of work in the last decade on finite-sample guarantees

for PCA [21, 17] and related problems, such as sparse PCA [35, 1] and kernel PCA [29, 43] most of these assume either the spiked covariance model (noise is modeled as being isotropic) [21, 43] or that the observed data $\boldsymbol{y}_t$ is i.i.d. [21, 17] or consider noiseless settings [35, 1] (typical in sparse PCA). The setting that we study involves linearly data dependent noise $\boldsymbol{w}_t = \boldsymbol{M}_t \boldsymbol{\ell}_t$ with the dependency matrix $\boldsymbol{M}_t$ being time-varying. Thus, the noise is clearly not isotropic. Moreover, this also means that the observed data $\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{v}_t$ cannot be identically distributed over time. In fact, our guarantee is interesting only in the setting where $\boldsymbol{M}_t$ changes enough over time so that the time-averaged expected value of signal-noise correlation and of noise power is sufficiently smaller than their respective instantaneous values.

We should mention also that, while many sophisticated eigenvector perturbation bounds exist in the literature [16, 18, 14], these are designed for different settings than the one we are interested in. For our setting, only the classical Davis-Kahan sin theta theorem [7] applies. In our analysis, we need to bound the sine of the largest principal angle between the true and estimated subspaces, because this helps us get a bound on the "noise"/error seen by the projected compressed sensing step at the next time instant. Thus, [16], which only provides coordinate-wise bounds, cannot be used. The perturbation seen by our sample covariance matrix is additive and our observed data $\boldsymbol{y}_t$ is not identically distributed, and thus the results of [18, 14] do not apply either.

The robust PCA (RPCA) problem has been extensively studied since the first two papers by Candes et al and Chandrasekharan at al [3, 5] and follow-up work by Hsu et al [13] all of which studied a convex optimization solution, called Principal Components Pursuit or PCP. A faster non-convex solution, called Alternating Projections or AltProj, was introduced in [25]. Later work has studied a projected gradient descent based approach, RPCA-GD [38]. The problem of RPCA with partial support knowledge was studied in [40]. All RPCA guarantees assume μ-incoherence of left and right singular vectors of $\boldsymbol{L}$ (needed to ensure that $\boldsymbol{L}$ is not sparse). One way to ensure that $\boldsymbol{X}$ is not low rank is to assume that an entry of $\boldsymbol{X}$ is nonzero with probability ρ independent of all others (Bernoulli model) and to assume a bound on ρ. This was assumed in [3]. This can sometimes be a strong assumption, e.g., in the video setting, it requires that foreground objects are

one pixel wide and jump around completely randomly over time. But, if it holds, and if another stronger left-right incoherence assumption holds[8], then $\rho \in O(1)$ (linear sparsity) can be tolerated while also allowing the rank of $\boldsymbol{L}$, r_L to be grow nearly linearly with $\min(n, d)$ [3]. The other approach to ensure that $\boldsymbol{X}$ is low rank is to assume a bound of $O(1/r_L)$ on the maximum fraction of nonzeros (outliers) in any row or in any column of $\boldsymbol{X}$. This is assumed in most of the later works [5, 13, 25, 38, 6].

Our work provides a fast mini-batch solution to the related problem of robust subspace tracking (RPCA with explicitly assuming slowly changing subspaces). Because we replace right incoherence by its statistical version, we are able to obtain guarantees on detection and tracking delay of our approach and show that both are nearly optimal (are within log factors of the minimum required delay r). This also means that the memory complexity of NORST is also near optimal: we only need to store α n-length vectors in memory with $\alpha = Cr \log n$. Of course, any RPCA approach could also be applied in a mini-batch fashion on α-consecutive column sub-matrices, and then it will also have the same memory complexity. We assume this here in our discussion. With this assumption, max-outlier-frac-row gets replaced by max-outlier-frac-row(α) and r_L gets replaced by r for the RPCA guarantees as well.

Because we assume a lower bound on the minimum outlier magnitudes that is proportional to SE_j, we obtain the following improvement in outlier tolerance (explained in Sec. 3.3.5). Treating f as a constant, *for any mini-batch after* t_{train}, we only need max-outlier-frac-row(α) $\in O(1)$. For standard RPCA, unless a random model on outlier support is assumed, $\max(\text{max-outlier-frac-col}, \text{max-outlier-frac-row}(\alpha)) \in O(1/r)$ is needed [25]. For the video application, this implies that NORST tolerates slow moving and occasionally static foreground objects much better than standard RPCA methods that do not assume slow subspace change. This is also corroborated by our experiments on real videos, e.g., see Fig 3.4 in Sec. 3.7 and also see a more detailed and quantitative evaluation on real data provided in [32]. Since our algorithm needs to be initialized with a standard batch RPCA approach such as AltProj [25] applied to the

[8] $\max_{i,j} |\boldsymbol{U}\boldsymbol{V}')_{i,j}| \leq \sqrt{\frac{\mu r}{nd}}$ where $\boldsymbol{U}, \boldsymbol{V}$ are the matrices of left and right singular vectors of $\boldsymbol{L}$

first $t_{\text{train}} = Cr$ data points, for this initial short batch, we do need AltProj assumptions to hold and this is why we need max-outlier-frac-row$_{\text{init}} \leq \frac{c_3}{r}$. For the per column fraction, we also need max-outlier-frac-col $\in O(1/r)$. Thus, the overall fraction of outliers allowed in a given matrix is still $O(1/r)$, which is the same as standard RPCA, but these can be less spread out row-wise (some rows could have many more outliers than others).

Moreover, we are able to guarantee that each column of $\boldsymbol{L}$, $\boldsymbol{\ell}_t$, is recovered to ε relative accuracy and that the support of outliers can be recovered exactly. Neither is guaranteed by existing RPCA results, these only guarantee $\|\hat{\boldsymbol{L}} - \boldsymbol{L}\|_F \leq \varepsilon$.

Finally, in terms of time complexity, the NORST complexity of $O(n\alpha r \log(1/\varepsilon))$ per mini-batch is comparable to that of simple (non-robust) PCA. In comparison to RPCA solutions, this is much faster than PCP [3, 5, 13] which needs $O(n\alpha^2 \frac{1}{\varepsilon})$ and r-times faster than AltProj [25] which needs $O(n\alpha r^2 \log(1/\varepsilon))$. RPCA-GD [38] is as fast as NORST but requires an even tighter outlier fractions' bound than other RPCA solutions: max(max-outlier-frac-row, max-outlier-frac-col) $\in O(1/r^{3/2})$.

Our work builds upon the simple-ReProCS (s-ReProCS) solution and guarantee [24] and removes many of its limitations. S-ReProCS assumes a specific model of slow subspace change: *only one* subspace direction can change at each change time, and the amount of change needs to be bounded. Even with this assumption, its tracking delay is of order $r \log n \log(1/\varepsilon)$. Since only one direction is changing, this delay is r-times sub-optimal. The same is true for its required lower bound on subspace change times. A second limitation of s-ReProCS is that, in order to track subspaces to ε accuracy, it requires the initial subspace estimate to also be ε accurate. This, in turn, implies that one needs to run the AltProj or PCP algorithm on the initial mini-batch to convergence. Instead, our approach only requires the initial subspace error to be $O(1/\sqrt{r})$. Thus, only order $\log r$ iterations of AltProj suffice to initialize our algorithm. Thirdly, the s-ReProCS guarantee needs a stronger statistical right incoherence assumption than ours: it needs an entry-wise bound of $\max_t \max_{i=1,2,\dots,r} |(\boldsymbol{a}_t)_i|^2 \leq \eta\lambda^+$. Lastly, we develop important extensions of our main result for (i) only tracking subspace changes (without detecting the change), and (ii) for subspaces changing by a little at each time t.

An earlier version of Theorem 3.39 appeared in ICML 2018 [23], but that was a conference paper and the proof of that result is only provided in an unpublished supplement on ArXiV. The results of the current manuscript improve upon the ICML result in various ways: we need a weaker statistical right incoherence assumption, a weaker lower bound on SE_j, and we develop two important extensions of our main result for subspace changes at each time and for applications not requiring change detection. Moreover, [23] did not prove the result for PCA in data-dependent noise, but only used the result proved in our older ISIT paper [34]. The problem of ST with missing data is a special case of robust ST, while ST with missing data and outliers is a simple generalization of robust ST. Interesting guarantees for both of these follow as easy corollaries of either our current result or of its earlier version from [23]. A corollary of the result of [23] for ST-miss is presented in [22]. In comparison to the result of [22], a similarly derived ST-miss corollary of our current result has all the advantages mentioned earlier in this paragraph.

3.5 Extensions: subspace change at each time, subspace tracking without detection

3.5.1 Subspace changing at each time

Suppose $\boldsymbol{y}_t = \tilde{\boldsymbol{\ell}}_t + \boldsymbol{x}_t$ where $\tilde{\boldsymbol{\ell}}_t = \boldsymbol{P}_{(t)}\tilde{\boldsymbol{a}}_t$, $\boldsymbol{P}_t$ changes by a little at each time t, but has more significant changes at certain times t_j. We show here how this case can be handled by treating the error generated by changes at each time t as extra unstructured noise $\boldsymbol{v}_t$. Assume that $\tilde{\boldsymbol{a}}_t$'s are zero mean, bounded, and i.i.d. with diagonal covariance matrix $\tilde{\boldsymbol{\Lambda}}$. Let $\tilde{\lambda}^+$ be its maximum eigenvalue and $\tilde{f}$ the condition number. Define $\boldsymbol{P}_j$ as the matrix of top r left singular vectors of the matrix $\tilde{\boldsymbol{L}}_j := [\tilde{\boldsymbol{\ell}}_{t_j}, \tilde{\boldsymbol{\ell}}_{t_j+1}, \ldots, \tilde{\boldsymbol{\ell}}_{t_{j+1}-1}]$, or equivalently of $[\boldsymbol{P}_{(t_j)}, \boldsymbol{P}_{(t_j+1)}, \ldots, \boldsymbol{P}_{(t_{j+1}-1)}]$. Let $\boldsymbol{a}_t := \boldsymbol{P}_j'\tilde{\boldsymbol{\ell}}_t$, $\boldsymbol{\ell}_t := \boldsymbol{P}_j\boldsymbol{a}_t$ and $\boldsymbol{v}_t := \tilde{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t = \boldsymbol{P}_{j,\perp}\tilde{\boldsymbol{\ell}}_t$.

Another way to understand the above is that we are expressing $\tilde{\boldsymbol{L}}_j = \boldsymbol{L}_j + \boldsymbol{V}_j$ where $\boldsymbol{L}_j$ is the rank-r SVD of $\tilde{\boldsymbol{L}}$, while $\boldsymbol{V}_j$ is the rest. While $\boldsymbol{L}_j\boldsymbol{V}_j' = 0$, we cannot say anything about individual vectors $\boldsymbol{\ell}_t\boldsymbol{v}_t'$ or their expected value. In general, $\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{v}_t{}'] \neq 0$. But even then, we can always use Cauchy-Schwarz to get the bound $\|\mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{v}_t']\|| \leq \sqrt{\lambda^+\lambda_v^+}$. Thus, to analyze this case, we need to

modify Corollary 3.32 for PCA-SDDN as follows: we now need $4\sqrt{b}qf + \frac{\lambda_v^+}{\lambda^-} + \sqrt{\frac{\lambda_v^+}{\lambda^-} f} < 0.4\epsilon_{\text{SE}}$. There is no change to the required lower bound on α. From our definition of $\boldsymbol{v}_t$, $\lambda_v^+ \leq \text{SE}(\boldsymbol{P}_j, \boldsymbol{P}_t)^2 \tilde{\lambda}^+$. Using $\lambda^+ \leq \tilde{\lambda}^+$, $\tilde{\lambda}^- < \lambda^-$, a simple sufficient condition to ensure that the third term is small ($\lambda_v^+/\lambda^- \leq 0.01\varepsilon^2/f$) is $\text{SE}(\boldsymbol{P}_j, \boldsymbol{P}_t)^2 \leq 0.01\varepsilon^2/\tilde{f}^2$.

Corollary 3.41 (Subspace changing at each t)**.** *Consider the setting defined in the first paragraph above. If* $\text{SE}(\boldsymbol{P}_j, \boldsymbol{P}_t)^2 < 0.01\varepsilon^2/\tilde{f}^2$*, Theorem 3.39 applies with* $\boldsymbol{P}_j$*,* $\boldsymbol{\ell}_t$*, and* $\boldsymbol{v}_t$ *as defined above.*

3.5.2 NORST-NoDet: NORST without subspace change detection

A simpler version of the NORST algorithm that does not detect change is as follows. The robust regression (projected CS) step is exactly as explained earlier. The subspace update step is much simpler: it just updates $\hat{\boldsymbol{P}}_{(t)}$ as the top r left singular vectors of $\hat{\boldsymbol{L}}_{t;\alpha}$ once every α frames. We refer to it as NORST-NoDet. We have the following guarantee for it.

Theorem 3.42. *Consider Algorithm 7 with parameters set as* $\alpha = Cf^2\mu r \log n$*,* $\zeta = x_{\min}/15$ *and* $\omega_{supp} = x_{\min}/2$*. Assume everything stated in Theorem 3.39 except the lower bound on* SE_j*. Then, w.p. at least* $1 - 10dn^{-10}$*,*

$$\text{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) \leq \begin{cases} \min(4f\text{SE}_j, 1) & \text{if } t \in \mathcal{J}_1, \\ (0.3)^{k-1} \min(4f\text{SE}_j, 1) & \text{if } t \in \mathcal{J}_k, \\ \varepsilon := c\sqrt{\lambda_v^+/\lambda^-} & \text{if } t \in \mathcal{J}_K, \end{cases}$$

where $\mathcal{J}_1 = [\lfloor t_j/\alpha \rfloor \alpha, (\lfloor t_j/\alpha \rfloor + 1)\alpha)$*,* $J_k = [\lfloor t_j/\alpha \rfloor + 1)\alpha) + (k-1)\alpha, (\lfloor t_j/\alpha) + (k+1))\alpha \rfloor$ *for* $k = 2, 3, \cdots, K-1$ *and* $\mathcal{J}_K = [(\lfloor t_j/\alpha \rfloor + (K+1))\alpha, \lfloor t_{j+1}/\alpha \rfloor \alpha)$*.*

The time complexity is $O(ndr \log(1/\varepsilon))$ *and memory complexity is* $O(n\alpha) = O(f^2 nr \log n \log(1/\varepsilon))$*.*

The advantage of NORST-NoDet is that it does not require a lower bound on the amount of change, SE_j, and it needs fewer algorithm parameters (does not need K or ω_{evals}). The disadvantage is it does not detect subspace change, we cannot obtain a "smoothing" version of it that solves the dynamic RPCA problem to ε accuracy at all times, and its subspace error bound is larger for the

intervals during which the subspace changes, $[\lfloor t_j/\alpha \rfloor \alpha, (\lfloor t_j/\alpha \rfloor + 1)\alpha)$. For times t in this interval, the bound is $\min(4f\mathrm{SE}_j, 1)$. Assuming small enough ε, this is larger than $(\varepsilon + \mathrm{SE}_j)$ which is the NORST bound for this interval. The reason is NORST stops tracking after the current subspace has been estimated accurately enough and until the next change is detected. During this period, it uses $\hat{\boldsymbol{P}}_{j-1}$ as the estimate. But NORST-NoDet updates the subspace in every interval. For the change interval, the rank of $\boldsymbol{L}_{t;\alpha}$ is more than r. It can be $2r$ in general. This is why it is not possible to guarantee a better bound for the r-SVD estimate. At the same time, without extra assumptions, it is not possible to obtain a guarantee for $2r$-SVD estimate either.

For analyzing the change interval we use the following modification of PCA-SDDN. Its proof is in Appendix 3.10. The proof of Theorem 3.42 is given in the Supplement Appendix 3.12.

Corollary 3.43. *Assume that* $\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{v}_t$ *with* $\boldsymbol{w}_t = \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}\boldsymbol{\ell}_t$, *with* $\boldsymbol{\ell}_t = \boldsymbol{P}_0\boldsymbol{a}_t$ *for* $t \in [1, \alpha_0]$ *and* $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t$ *for* $t \in [\alpha_0 + 1, \alpha]$, *and* $\mathrm{SE}(\boldsymbol{P}_0, \boldsymbol{P}) \le \Delta$. *Assume also that Assumptions 3.29, 3.30 hold,* $\max_t \max(\|\boldsymbol{M}_{1,t}\boldsymbol{P}_0\|, \|\boldsymbol{M}_{1,t}\boldsymbol{P}\|) \le q < 1$, *and the fraction of nonzeros in any row of the noise matrix* $[\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_\alpha]$ *is equal to* b. *Let* $g := \frac{\lambda_v^+}{\lambda^-}$. *If* $\Delta < c/f$, *and if* $\alpha \ge \alpha^* = C \max\left(\frac{q^2 f^2}{\epsilon_{\mathrm{SE}}^2} r \log n, \frac{gf}{\epsilon_{\mathrm{SE}}^2} \max(r_v, r) \log n\right)$ *then w.p. at least* $1 - 10n^{-10}$,

$$\begin{aligned}\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) &\le 1.1\left(3((\alpha_0/\alpha)\Delta + 4\sqrt{b}q)f + \frac{\lambda_v^+}{\lambda^-}\right)\\ &\le 3.3\Delta f + 4.4\sqrt{b}qf + 1.1\frac{\lambda_v^+}{\lambda^-}.\end{aligned}$$

3.6 Proof of correctness of the NORST algorithm

In this section we state the three main lemmas and explain how they help prove Theorem 3.39. After this, we prove the three lemmas.

3.6.1 Main Lemmas

We define or recall a few things first.

1. Recall $\Delta = \max_j \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$, let $\Delta_0 = \mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0)$; recall $c\sqrt{\lambda_v^+/\lambda^-} < \varepsilon \le 0.01 < 0.2$

2. Let $\hat{\boldsymbol{P}}_{j,0} = \hat{\boldsymbol{P}}_{j-1}$ and recall (from Algorithm) that $\hat{\boldsymbol{P}}_{j-1} = \hat{\boldsymbol{P}}_{j-1,K}$:

3. Constants for Theorem 3.39: $c_1 = c_2 = 0.001$ (bounds on max-outlier-frac-col, max-outlier-frac-row(α)), and $c_3 = 1/(30\sqrt{\mu})$. We use $b_0 = c_2/f^2$ to denote the bound on max-outlier-frac-row(α).

4. Let $q_0 := 1.2(\varepsilon + \mathrm{SE}_j)$, $q_k = 1.2\max(q_{k-1}/4, \varepsilon)$. Clearly $q_k = \max(0.3^k q_0, 1.2\varepsilon)$.

First consider the simpler case when t_j's are known. In this case $\hat{t}_j = t_j$. Define the events

- $\Gamma_{0,0} := \{\text{assumed bound on } \mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0)\}$,
- $\Gamma_{0,k} := \Gamma_{0,k-1} \cap \{\mathrm{SE}(\hat{\boldsymbol{P}}_{0,k}, \boldsymbol{P}_0) \le \mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0)\}$,
- $\Gamma_{j,0} := \Gamma_{j-1,K}$, $\Gamma_{j,k} := \Gamma_{j,k-1} \cap \{\mathrm{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) \le q_{k-1}/4\}$ for $j = 1, 2, \ldots, J$ and $k = 1, 2, \ldots, K$.
- Using the expression for K given in the theorem, and since $\hat{\boldsymbol{P}}_j = \hat{\boldsymbol{P}}_{j,k}$ (from the Algorithm), it follows that $\Gamma_{j,K}$ implies $\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) = \mathrm{SE}(\hat{\boldsymbol{P}}_{j,K}, \boldsymbol{P}_j) \le \varepsilon$.

Observe that, if we can show that $\Pr(\Gamma_{J,K}|\Gamma_{0,0}) \ge 1 - dn^{-10}$ we will have obtained all the subspace recovery bounds of Theorem 3.39. The next two lemmas, Lemmas 3.44 and 3.45, applied sequentially help show that this is true. The first one proves that $\Pr(\Gamma_{j,1}|\Gamma_{j,0}) \ge 1 - 10n^{-10}$, the second one proves that $\Pr(\Gamma_{j,k}|\Gamma_{j,k-1}) \ge 1 - 10n^{-10}$ for $k = 1, 2, \ldots, K$. The bounds on $\|\boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t\|$ follow easily.

To prove the actual result with t_j unknown, we also need Corollary 3.47 and Lemma 3.48 which proves that the change detection step works as desired. Moreover, we will need a different definition of $\Gamma_{j,0}$; we cannot set it equal to $\Gamma_{j-1,K}$. The proof is given in Appendix 3.11.

Lemma 3.44 (first update interval)**.** *Under the conditions of Theorem 3.39, conditioned on* $\Gamma_{j,0}$*,*

1. *for all* $t \in [\hat{t}_j, \hat{t}_j + \alpha)$, $\|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\| \le (\varepsilon + \Delta)\sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+} < x_{\min}/15$, $\|\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t\| \le 7x_{\min}/15 < x_{\min}/2$, $\hat{\mathcal{T}}_t = \mathcal{T}_t$, *the error* $\boldsymbol{e}_t := \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t$ *satisfies*

$$\boldsymbol{e}_t = \boldsymbol{I}_{\mathcal{T}_t} \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1} \boldsymbol{I}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t) \tag{3.7}$$

and $\|\boldsymbol{e}_t\| \le 1.2[(\varepsilon + \Delta)\sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+}]$. *Here* $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j,0}\hat{\boldsymbol{P}}_{j,0}{}'$. *Recall we let* $\hat{\boldsymbol{P}}_{j,0} = \hat{\boldsymbol{P}}_{j-1}$.

2. *w.p. at least* $1 - 10n^{-10}$, $\hat{\boldsymbol{P}}_{j,1}$ *satisfies* $\mathrm{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \leq \max(q_0/4, \varepsilon)$, *i.e.,* $\Gamma_{j,1}$ *holds.*

Lemma 3.45 (k-th update interval)**.** *Under the conditions of Theorem 3.39, conditioned on* $\Gamma_{j,k-1}$,

1. *for all* $t \in [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha - 1)$, *all claims of the first part of Lemma 3.44 holds,* $\|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\| \leq \max(0.3^{k-1}(\varepsilon + \Delta), \varepsilon)\sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+}$, $\boldsymbol{e}_t$ *satisfies* (3.7), *and* $\|\boldsymbol{e}_t\| \leq \max((0.3)^{k-1} \cdot 1.2(\varepsilon + \Delta), \varepsilon)\sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+}$. *Here* $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j,k-1}\hat{\boldsymbol{P}}_{j,k-1}{}'$.

2. *w.p. at least* $1 - 10n^{-10}$, $\hat{\boldsymbol{P}}_{j,k}$ *satisfies* $\mathrm{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) \leq \max(q_{k-1}/4, \varepsilon)$, *i.e.,* $\Gamma_{j,k}$ *holds.*

Remark 3.46. *For the case of* $j = 0$, *in both the lemmas above,* Δ *gets replaced by* $\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0)$.

Corollary 3.47. *Under the conditions of Theorem 3.39, the following also hold.*

1. *For all* $t \in [t_j, \hat{t}_j)$, *conditioned on* $\Gamma_{j-1,K}$, *all claims of the first item of Lemma 3.44 hold.*

2. *For all* $t \in [\hat{t}_j + K\alpha, t_{j+1})$, *conditioned on* $\Gamma_{j,K}$, *the first item of Lemma 3.45 holds with* $k = K$.

Thus, for all times t, *under appropriate conditioning,* $\boldsymbol{e}_t$ *satisfies* (3.7).

The following lemma shows that, whp, we can detect subspace change within 2α time instants without any false detections. Recall that the detection threshold $\omega_{evals} = 2\varepsilon^2\lambda^+$.

Lemma 3.48 (Subspace Change Detection)**.** *Assume that the conditions of Theorem 3.39 hold.*

1. *Consider an* α*-length time interval* $\mathcal{J}^\alpha \subset [t_j, t_{j+1}]$ *during which* $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_{j-1}$ *so that* $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}'$. *Let* $\boldsymbol{\Phi} = \boldsymbol{\Psi}$. *Assume that* $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \varepsilon$ *and* $\boldsymbol{e}_t$ *satisfies* (3.7). *Then, w.p. at least* $1 - 10n^{-10}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t \in \mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\boldsymbol{\Phi}\right) \geq 0.59\lambda^- \mathrm{SE}_j(\mathrm{SE}_j - 8\varepsilon) > \omega_{evals}$$

since $\mathrm{SE}_j > 9\sqrt{f}\varepsilon$.

2. *Consider an* α*-length time interval* $\mathcal{J}^\alpha \subset [t_j, t_{j+1}]$ *during which* $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_j$ *so that* $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_j\hat{\boldsymbol{P}}_j{}'$. *Let* $\boldsymbol{\Phi} = \boldsymbol{\Psi}$. *Assume that* $\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \varepsilon$ *and* $\boldsymbol{e}_t$ *satisfies* (3.7). *Then, w.p. at*

least $1 - 10n^{-10}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^{\alpha}}\mathbf{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\mathbf{\Phi}\right) \leq 1.37\varepsilon^2\lambda^+ < \omega_{evals}$$

3.6.2 Proof of the first two lemmas

The projected CS proof (item one of the first two lemmas) uses the following lemma from [28] that relates the s-Restricted Isometry Constant (RIC), $\delta_s(.)$ [2] of $\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}'$ to incoherence of $\boldsymbol{P}$.

Lemma 3.49. *[[28]] For an $n \times r$ basis matrix $\boldsymbol{P}$, (1) $\delta_s(\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}') = \max_{|\mathcal{T}|\leq s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}\|^2$; and (2) $\max_{|\mathcal{T}|\leq s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}\|^2 \leq s \max_{i=1,2,\ldots,n} \|\boldsymbol{I}_i{}'\boldsymbol{P}\|^2 \leq s\mu r/n$.*

The last bound of the above lemma used the definition of the incoherence parameter μ. We will apply this lemma with $s = \text{max-outlier-frac-col} \cdot n$. The subspace update step proof (item 2 of the first two lemmas) uses Corollary 3.32 for PCA-SDDN and the following simple lemma proved in the Appendix.

Lemma 3.50. *Let $\boldsymbol{Q}_1$, $\boldsymbol{Q}_2$ and $\boldsymbol{Q}_3$ be r-dimensional subspaces in $\mathbb{R}^n$ such that $\text{SE}(\boldsymbol{Q}_1, \boldsymbol{Q}_2) = \Delta_1$ and $\text{SE}(\boldsymbol{Q}_2, \boldsymbol{Q}_3) = \Delta_2$. Then, $\Delta_1 - 2\Delta_2 \leq \text{SE}(\boldsymbol{Q}_1, \boldsymbol{Q}_3) \leq \Delta_1 + \Delta_2$.*

Proof of Lemma 3.44. Proof of item 1. First consider $j > 0$. We have conditioned on the event $\Gamma_{j,0} := \Gamma_{j-1,K}$. This implies that $\text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \varepsilon$.

For the interval $t \in [\hat{t}_j, \hat{t}_j + \alpha)$, $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_{j-1}$ and thus $\mathbf{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}'$ (from Algorithm). Let $s := \text{max-outlier-frac-col} \cdot n$. For the sparse recovery step, we need to bound the $2s$-RIC of $\mathbf{\Psi}$. To do this, we obtain bound on $\max_{|\mathcal{T}|\leq 2s} \|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{j-1}\|$ as follows. Consider any set $\mathcal{T}$ such that $|\mathcal{T}| \leq 2s$. Then,

$$\begin{aligned}
\left\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{j-1}\right\| &\leq \left\|\boldsymbol{I}_{\mathcal{T}}{}'(\boldsymbol{I} - \boldsymbol{P}_{j-1}\boldsymbol{P}_{j-1}{}')\hat{\boldsymbol{P}}_{j-1}\right\| + \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_{j-1}\boldsymbol{P}_{j-1}{}'\hat{\boldsymbol{P}}_{j-1}\right\| \\
&\leq \text{SE}(\boldsymbol{P}_{j-1}, \hat{\boldsymbol{P}}_{j-1}) + \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_{j-1}\right\| \\
&= \text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) + \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_{j-1}\right\|
\end{aligned}$$

Using Lemma 3.49, and the bound on max-outlier-frac-col from Theorem 3.39,

$$\max_{|\mathcal{T}|\leq 2s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_{j-1}\|^2 \leq 2s \max_i \|\boldsymbol{I}_i{}'\boldsymbol{P}_{j-1}\|^2 \leq \frac{2s\mu r}{n} \leq 0.01 \tag{3.8}$$

Thus, using $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \le \varepsilon$, (where $c\sqrt{\lambda_v^+/\lambda^-} \le \varepsilon \le 0.01$),

$$\max_{|\mathcal{T}|\le 2s} \left\| \boldsymbol{I}_{\mathcal{T}}{}' \hat{\boldsymbol{P}}_{j-1} \right\| \le \varepsilon + \max_{|\mathcal{T}|\le 2s} \left\| \boldsymbol{I}_{\mathcal{T}}{}' \boldsymbol{P}_{j-1} \right\| \le \varepsilon + 0.1$$

Finally, using Lemma 3.49, $\delta_{2s}(\boldsymbol{\Psi}) \le 0.11^2 < 0.15$. Hence

$$\left\| \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t} \right)^{-1} \right\| \le \frac{1}{1-\delta_s(\boldsymbol{\Psi})} \le \frac{1}{1-\delta_{2s}(\boldsymbol{\Psi})} \le \frac{1}{1-0.15} < 1.2.$$

When $j = 0$, there are some minor changes. From the initialization assumption, we have $\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \le 0.25$. Thus, $\max_{|\mathcal{T}|\le 2s} \left\| \boldsymbol{I}_{\mathcal{T}}{}' \hat{\boldsymbol{P}}_0 \right\| \le 0.25 + 0.1 = 0.35$. Thus, using Lemma 3.49, $\delta_{2s}(\boldsymbol{\Psi}_0) \le 0.35^2 < 0.15$. The rest of the proof given below is the same for $j = 0$ and $j > 0$.

Next we bound norm of $\boldsymbol{b}_t := \boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)$.

$$\begin{aligned} \|\boldsymbol{b}_t\| = \|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\| &\le \left\| (\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}') \boldsymbol{P}_j \boldsymbol{a}_t \right\| + \|\boldsymbol{v}_t\| \le \mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j)\, \|\boldsymbol{a}_t\| + \sqrt{r_v \lambda_v^+} \\ &\overset{(a)}{\le} (\varepsilon + \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)) \sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+} \end{aligned}$$

where (a) follows from Lemma 3.50 with $\boldsymbol{Q}_1 = \hat{\boldsymbol{P}}_{j-1}$, $\boldsymbol{Q}_2 = \boldsymbol{P}_{j-1}$ and $\boldsymbol{Q}_3 = \boldsymbol{P}_j$. Under the assumptions of Theorem 3.39, the RHS of (a) is bounded by $x_{\min}/15$. This is why we have set $\xi = x_{\min}/15$ in the Algorithm. Using these facts, and $\delta_{2s}(\boldsymbol{\Psi}) \le 0.15$, the CS guarantee from [2, Theorem 1.3] implies that

$$\|\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t\| \le 7\xi = 7x_{\min}/15 < x_{\min}/2$$

Consider support recovery. From above,

$$|(\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t)_i| \le \|\hat{\boldsymbol{x}}_{t,cs} - \boldsymbol{x}_t\| \le 7x_{\min}/15 < x_{\min}/2$$

The Algorithm sets $\omega_{supp} = x_{\min}/2$. Consider an index $i \in \mathcal{T}_t$. Since $|(\boldsymbol{x}_t)_i| \ge x_{\min}$,

$$\begin{aligned} x_{\min} - |(\hat{\boldsymbol{x}}_{t,cs})_i| &\le |(\boldsymbol{x}_t)_i| - |(\hat{\boldsymbol{x}}_{t,cs})_i| \\ &\le |(\boldsymbol{x}_t - \hat{\boldsymbol{x}}_{t,cs})_i| < \frac{x_{\min}}{2} \end{aligned}$$

Thus, $|(\hat{\boldsymbol{x}}_{t,cs})_i| > \frac{x_{\min}}{2} = \omega_{supp}$ which means $i \in \hat{\mathcal{T}}_t$. Hence $\mathcal{T}_t \subseteq \hat{\mathcal{T}}_t$. Next, consider any $j \notin \mathcal{T}_t$. Then, $(\boldsymbol{x}_t)_j = 0$ and so

$$|(\hat{\boldsymbol{x}}_{t,cs})_j| = |(\hat{\boldsymbol{x}}_{t,cs})_j)| - |(\boldsymbol{x}_t)_j| \le |(\hat{\boldsymbol{x}}_{t,cs})_j - (\boldsymbol{x}_t)_j| < \frac{x_{\min}}{2}$$

which implies $j \notin \hat{\mathcal{T}}_t$ and $\hat{\mathcal{T}}_t \subseteq \mathcal{T}_t$ implying that $\hat{\mathcal{T}}_t = \mathcal{T}_t$.

With $\hat{\mathcal{T}}_t = \mathcal{T}_t$ and since $\mathcal{T}_t$ is the support of $\boldsymbol{x}_t$, $\boldsymbol{x}_t = \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{x}_t$, and so

$$\begin{aligned}\hat{\boldsymbol{x}}_t &= \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'(\boldsymbol{\Psi}\boldsymbol{\ell}_t + \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{\Psi}\boldsymbol{v}_t)\\ &= \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t) + \boldsymbol{x}_t\end{aligned}$$

since $\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi} = \boldsymbol{I}'_{\mathcal{T}_t}\boldsymbol{\Psi}'\boldsymbol{\Psi} = \boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}$. Thus $\boldsymbol{e}_t = \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t$ satisfies

$$\begin{aligned}\boldsymbol{e}_t &= \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t) := (\boldsymbol{e}_{\boldsymbol{\ell}})_t + (\boldsymbol{e}_{\boldsymbol{v}})_t,\\ \|\boldsymbol{e}_t\| &\le \left\|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\right\|\left\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\right\|\\ &\le 1.2\left\|\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\right\|\end{aligned}$$

Proof of Item 2: Recall that $q_0 := 1.2(\varepsilon + \mathrm{SE}_j)$, $q_k = 1.2\max(q_{k-1}/4, \varepsilon) = \max(0.3^k q_0, 1.2\varepsilon)$. From our definition of K, $0.3^K q_0 = \varepsilon$. Thus, for $k \le K$, $\max(q_{k-1}/4, \varepsilon) = q_{k-1}/4$.

We are considering the interval $[\hat{t}_j, \hat{t}_j + \alpha)$. Since $\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t - \boldsymbol{e}_t + \boldsymbol{v}_t$ with $\boldsymbol{e}_t$ satisfying (3.7), updating $\hat{\boldsymbol{P}}_{(t)}$ from the $\hat{\boldsymbol{\ell}}_t$'s is a problem of PCA in sparse data-dependent noise (SDDN). To analyze this, we use Corollary 3.32. Define $(\boldsymbol{e}_{\boldsymbol{\ell}})_t = \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{\ell}_t$ and $(\boldsymbol{e}_{\boldsymbol{v}})_t = \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{v}_t$. Recall from the Algorithm that we compute $\hat{\boldsymbol{P}}_{j,1}$ as the top r eigenvectors of $\frac{1}{\alpha}\sum_{t=\hat{t}_j}^{\hat{t}_j+\alpha-1}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'$. In the notation of Corollary 3.32, $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv (\boldsymbol{e}_{\boldsymbol{\ell}})_t$, $\boldsymbol{v}_t \equiv (\boldsymbol{e}_{\boldsymbol{v}})_t + \boldsymbol{v}_t$, $\boldsymbol{\ell}_t \equiv \boldsymbol{\ell}_t$, $\boldsymbol{M}_{s,t} = -\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'$, $\hat{\boldsymbol{P}} = \hat{\boldsymbol{P}}_{j,1}$, $\boldsymbol{P} = \boldsymbol{P}_j$, and so $\|\boldsymbol{M}_{s,t}\boldsymbol{P}\| = \|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{P}_j\| \le 1.2(\varepsilon + \mathrm{SE}_j) := q_0$. Also, $\lambda_v^+ \equiv 2.2\lambda_v^+$ since $\mathbb{E}[(\boldsymbol{e}_{\boldsymbol{v}})_t(\boldsymbol{e}_{\boldsymbol{v}})_t{}'] \le (1.2)^2\lambda_v^+$. And $b \equiv b_0$ which is the upper bound on max-outlier-frac-row(α). Applying Corollary 3.32 with $q \equiv q_0$, $b \equiv b_0$ and using $\varepsilon_{\mathrm{SE}} = \max(q_0/4, \varepsilon)$, observe that we require

$$4\sqrt{b_0}q_0 f + (2.2)^2\lambda_v^+/\lambda^- \le 0.4\max(q_0/4, \varepsilon).$$

From above, $\max(q_0/4, \varepsilon) = q_0/4$ (if the max is ε we stop the tracking). The required bound holds since $q_0/4 \ge \varepsilon > c\sqrt{\lambda_v^+/\lambda^-}$ (from Theorem) and $\sqrt{b_0} = 0.01/f$. Corollary 3.32 also requires $\alpha \ge \alpha_*$ which is defined in it. Our choice of $\alpha = Cf^2\mu r\log n$ satisfies this since $q_0^2/\varepsilon_{\mathrm{SE}}^2 = 4^2$ and $(\lambda_v^+/\lambda^-)/\varepsilon_{\mathrm{SE}}^2 < C$. Thus, by Corollary 3.32, with probability at least $1 - 10n^{-10}$, $\mathrm{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \le \max(q_0/4, \varepsilon)$. □

Remark 3.51 (Clarification about conditioning)**.** *In the proof above we have used Corollary 3.32 for $\hat{\boldsymbol{\ell}}_t$'s for $t \in \mathcal{J}^\alpha := [\hat{t}_j, \hat{t}_j + \alpha)$. This corollary assumes that, for $t \in \mathcal{J}^\alpha$, $\boldsymbol{a}_t$'s are mutually independent and $\boldsymbol{M}_{s,t}$'s are deterministic matrices. Let $\boldsymbol{y}_{\text{old}} := \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{\hat{t}_j - 1}\}$. We apply Corollary 3.32 conditioned on $\boldsymbol{y}_{\text{old}}$, for a $\boldsymbol{y}_{\text{old}} \in \Gamma_{j,0}$. Conditioned on $\boldsymbol{y}_{\text{old}}$, clearly, the matrices $\boldsymbol{M}_{s,t}$ used in the proof above are deterministic. Also $\boldsymbol{y}_{\text{old}}$ is independent of the $\boldsymbol{a}_t$'s for $t \in \mathcal{J}^\alpha$ and thus, even conditioned on $\boldsymbol{y}_{\text{old}}$, the $\boldsymbol{a}_t$'s for $t \in \mathcal{J}^\alpha$ are mutually independent. Corollary 3.32 tells us that, for any $\boldsymbol{y}_{\text{old}} \in \Gamma_{j,0}$, conditioned on $\boldsymbol{y}_{\text{old}}$, w.p. at least $1 - 10n^{-10}$, $\text{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \leq \max(q_0/4, \varepsilon)$. Since this holds with the same probability for all $\boldsymbol{y}_{\text{old}} \in \Gamma_{j,0}$, it also holds with the same probability when we condition on $\Gamma_{j,0}$. Thus, conditioned on $\Gamma_{j,0}$, with this probability, $\Gamma_{j,1}$ holds. An analogous argument also applies for the next proof.*

Proof of Lemma 3.45. We first present the proof for the $k = 2$ case and then generalize it for an arbitrary k. Consider $k = 2$. We have conditioned on $\Gamma_{j,1}$. This implies that $\text{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \leq q_0/4$. We consider the interval $t \in [\hat{t}_j + \alpha, \hat{t}_j + 2\alpha)$. For this interval, $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_{j,1}$ and thus $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j,1}\hat{\boldsymbol{P}}_{j,1}{}'$. Consider any set $\mathcal{T}$ such that $|\mathcal{T}| \leq 2s$. We have

$$\begin{aligned}\left\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{j,1}\right\| &\leq \left\|\boldsymbol{I}_{\mathcal{T}}{}'(\boldsymbol{I} - \boldsymbol{P}_j\boldsymbol{P}_j{}')\hat{\boldsymbol{P}}_{j,1}\right\| + \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_j\boldsymbol{P}_j{}'\hat{\boldsymbol{P}}_{j,1}\right\| \\ &\leq \text{SE}(\boldsymbol{P}_j, \hat{\boldsymbol{P}}_{j,1}) + \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_j\right\| = \text{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) + \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_j\right\|\end{aligned}$$

The equality holds since SE is symmetric for subspaces of the same dimension. Using $\text{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \leq \max(q_0/4, \varepsilon)$, (3.8),

$$\begin{aligned}\max_{|\mathcal{T}| \leq 2s} \left\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{j,1}\right\| &\leq \max(q_0/4, \varepsilon) + \max_{|\mathcal{T}| \leq 2s} \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_j\right\| \\ &\leq \max(q_0/4, \varepsilon) + 0.1.\end{aligned}$$

By the assumptions of Theorem 3.39, $q_0 \leq 0.96$ and $\varepsilon \leq 0.2$. Using this and Lemma 3.49,

$$\begin{aligned}\delta_{2s}(\boldsymbol{\Psi}_j) = \max_{|\mathcal{T}| \leq 2s} \left\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{j,1}\right\|^2 \leq 0.35^2 < 0.15 \\ \implies \left\|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\right\| \leq 1.2.\end{aligned}$$

Finally,

$$\|\boldsymbol{b}_t\| = \|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\| \leq \left\|(\boldsymbol{I} - \hat{\boldsymbol{P}}_{j,1}\hat{\boldsymbol{P}}_{j,1}{}')\boldsymbol{P}_j\boldsymbol{a}_t\right\| + \|\boldsymbol{v}_t\|$$
$$\leq \max(q_0/4, \varepsilon)\sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+}$$

The rest of the proof is the same[9] and this ensures exact support recovery and the expression for $\boldsymbol{e}_t$.

Proof of Item 2: Again, updating $\hat{\boldsymbol{P}}_{(t)}$ using $\hat{\boldsymbol{\ell}}_t$'s is a PCA-SDDN problem. We use Corollary 3.32. We compute $\hat{\boldsymbol{P}}_{j,2}$ as the top r eigenvectors of $\frac{1}{\alpha}\sum_{t=\hat{t}_j+\alpha}^{\hat{t}_j+2\alpha-1}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'$. From item 1, $\boldsymbol{e}_t$ satisfies (3.7) for this interval. In the notation of Corollary 3.32, $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv (\boldsymbol{e}_{\boldsymbol{\ell}})_t$, $\boldsymbol{\ell}_t \equiv \boldsymbol{\ell}_t$, $\boldsymbol{v}_t \equiv (\boldsymbol{e}_{\boldsymbol{v}})_t + \boldsymbol{v}_t$, $\boldsymbol{P} \equiv \boldsymbol{P}_j$, $\hat{\boldsymbol{P}} \equiv \hat{\boldsymbol{P}}_{j,2}$, and $\boldsymbol{M}_{s,t} = -\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'$. So $\|\boldsymbol{M}_{s,t}\boldsymbol{P}_j\| = \|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{P}_j\| \leq 1.2\max(q_0/4, \varepsilon) := q_1$. Applying Corollary 3.32 with $q \equiv q_1$, $b \equiv b_0$ (b_0 bounds max-outlier-frac-row(α)), and setting $\varepsilon_{\text{SE}} = \max(q_1/4, \varepsilon)$, observe that we require

$$4\sqrt{b_0}q_1 f + (2.2)^2\lambda_v^+/\lambda^- \leq 0.4\max(q_1/4, \varepsilon)$$

Once again recall that the max is $q_1/4$. The above bound holds since $\sqrt{b_0}f \leq 0.01$ and $q_1/4 > \varepsilon > \sqrt{\lambda_v^+/\lambda^-}$. Corollary 3.32 also requires $\alpha \geq \alpha_*$. Our choice of $\alpha = Cf^2\mu r \log n$ satisfies this requirement since $q_1^2/\varepsilon_{\text{SE}}^2 = 4^2$ and $(\lambda_v^+/\lambda^-)/\varepsilon_{\text{SE}}^2 < C$. Thus, from Corollary 3.32, with probability at least $1 - 10n^{-10}$, $\text{SE}(\hat{\boldsymbol{P}}_{j,2}, \boldsymbol{P}_j) \leq \max(q_1/4, \varepsilon)$.

(B) General k: We have conditioned on $\Gamma_{j,k-1}$. This implies that $\text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j) \leq \max(q_{k-1}/4, \varepsilon)$. Consider the interval $[\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$. In this interval, $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_{j,k-1}$ and thus $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j,k-1}\hat{\boldsymbol{P}}_{j,k-1}{}'$. Using the same idea as for the $k = 2$ case, we have that for the

[9]Notice here that, we could have loosened the required lower bound on $x_{\min}$ for this interval in the case when there is no noise

k-th interval, $q_{k-1} = \max(0.3^{k-1}q_0, \varepsilon)$. Pick $\varepsilon_{\text{SE}} = \max(q_{k-1}/4, \varepsilon)$. From this it is easy to see that

$$\begin{aligned}
\delta_{2s}(\boldsymbol{\Psi}) &\leq \left(\max_{|\mathcal{T}|\leq 2s} \left\|\boldsymbol{I}_{\mathcal{T}}{}'\hat{\boldsymbol{P}}_{j,k-1}\right\|\right)^2 \\
&\leq (\text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j) + \max_{|\mathcal{T}|\leq 2s} \left\|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}_j\right\|)^2 \\
&\overset{(a)}{\leq} (\text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j) + 0.1)^2 \\
&\leq \left[\max\left(0.3^{k-1}(\varepsilon + \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j), \varepsilon\right) + 0.1\right]^2 < 0.15
\end{aligned}$$

where (a) follows from (3.8). Also, as before,

$$\begin{aligned}
&\|\boldsymbol{\Psi}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\| \leq \text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j)\, \|\boldsymbol{a}_t\| + \|\boldsymbol{v}_t\| \\
&\leq \max\left(0.3^{k-1}(\varepsilon + \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)), \varepsilon\right) \sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+} \\
&\overset{(a)}{\leq} \max\left(0.3^{k-1}(\varepsilon + \Delta), \varepsilon\right) \sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+}
\end{aligned}$$

Proof of Item 2: Again, updating $\hat{\boldsymbol{P}}_{(t)}$ from $\hat{\boldsymbol{\ell}}_t$'s is a problem of PCA in sparse data-dependent noise given in Corollary 3.32. From *Item 1* of this lemma we know that, for $t \in [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha]$, $\boldsymbol{e}_t$ satisfies (3.7). We update the subspace, $\hat{\boldsymbol{P}}_{j,k}$ as the top r eigenvectors of $\frac{1}{\alpha}\sum_{t=\hat{t}_j+(k-1)\alpha}^{\hat{t}_j+k\alpha-1} \hat{\boldsymbol{\ell}}_t \hat{\boldsymbol{\ell}}_t{}'$. In the setting above $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv (\boldsymbol{e}_{\boldsymbol{\ell}})_t$, $\boldsymbol{\ell}_t \equiv \boldsymbol{\ell}_t$, $\boldsymbol{v}_t \equiv (\boldsymbol{e}_{\boldsymbol{v}})_t + \boldsymbol{v}_t$, and $\boldsymbol{M}_{s,t} = -\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'$, and so $\|\boldsymbol{M}_{s,t}\boldsymbol{P}_j\| = \|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{P}_j\| \leq 1.2\max(q_{k-2}/4, \varepsilon) := q_{k-1}$. Applying Corollary 3.32 with $q \equiv q_{k-1}$, $b \equiv b_0$ (b_0 bounds max-outlier-frac-row(α)), and setting $\varepsilon_{\text{SE}} = \max(q_{k-1}/4, \varepsilon)$, we require$4\sqrt{b_0}q_{k-1}f + \lambda_v^+/\lambda^- \leq 0.4\max(q_{k-1}/4, \varepsilon)$. This holds as explained earlier and hence, by Corollary 3.32, the result follows. □

3.6.3 Proof of Lemma 3.48

Proof. Proof of Item 1: We are considering an α-consecutive frames interval $\mathcal{J}^\alpha$ in $[t_j, t_{j+1})$ during which $\hat{\boldsymbol{P}}_{(t-1)} = \hat{\boldsymbol{P}}_{j-1}$. Thus $\boldsymbol{\Psi} = \boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}'$. Recall from earlier that at all times t, $\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t - \boldsymbol{e}_t + \boldsymbol{v}_t$, where $\boldsymbol{e}_t = (\boldsymbol{e}_{\boldsymbol{\ell}})_t + (\boldsymbol{e}_{\boldsymbol{v}})_t$, $\boldsymbol{w}_t \equiv (\boldsymbol{e}_{\boldsymbol{\ell}})_t = \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}\boldsymbol{\ell}_t$ is sparse and data-dependent noise, and $\boldsymbol{v}_t \equiv (\boldsymbol{e}_{\boldsymbol{v}})_t + \boldsymbol{v}_t$ is small unstructured noise. As in the earlier proofs, $\boldsymbol{w}_t = (\boldsymbol{e}_{\boldsymbol{\ell}})_t$ can be expressed as $\boldsymbol{w}_t = \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{M}_{s,t}\boldsymbol{\ell}_t$ where $\boldsymbol{M}_{s,t} = \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}$. Thus,

$q = q_0 = 1.2\text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j) \leq 1.2(\varepsilon + \text{SE}_j)$ and $b = b_0$. Let

$$\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\boldsymbol{\Phi} = \frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}' + \boldsymbol{\Phi}\text{noise}\boldsymbol{\Phi} + \boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}$$

where noise $= \frac{1}{\alpha}\sum_t \boldsymbol{w}_t\boldsymbol{w}_t' + \frac{1}{\alpha}\sum_t \boldsymbol{v}_t\boldsymbol{v}_t'$ and cross contains the cross terms. By Weyl's inequality,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\boldsymbol{\Phi}\right) \geq \lambda_{\max}\left(\frac{1}{\alpha}\sum_t \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'\boldsymbol{\Phi}\right) - \|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}\| \tag{3.9}$$

Using Corollary 3.54 from Appendix 3.10, w.p. at least $1 - 10n^{-10}$, if α is as given in our Theorem,

$$\|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}'\| \leq 2.02\sqrt{b}\|\boldsymbol{\Phi}\boldsymbol{P}_j\|q_0\lambda^+ \tag{3.10}$$

Since $\|\boldsymbol{\Phi}\boldsymbol{P}_j\| \leq q = 1.2(\varepsilon + \text{SE}_j)$, using the above two inequalities,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\boldsymbol{\Phi}\right) \geq$$

$$\lambda_{\max}\left(\underbrace{\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\boldsymbol{P}_j\boldsymbol{a}_t\boldsymbol{a}_t{}'\boldsymbol{P}_j{}'\boldsymbol{\Phi}}_{\text{Term1}}\right) - 2.02\sqrt{b}(1.2(\varepsilon + \text{SE}_j)^2)\lambda^+ \tag{3.11}$$

We bound the first term of (3.11), Term1, as follows. Let $\boldsymbol{\Phi}\boldsymbol{P}_j \overset{QR}{=} \boldsymbol{E}_j\boldsymbol{R}_j$ be its reduced QR decomposition. Thus $\boldsymbol{E}_j$ is an $n \times r$ matrix with orthonormal columns and $\boldsymbol{R}_j$ is an $r \times r$ upper triangular matrix. Let

$$\boldsymbol{A} := \boldsymbol{R}_j\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{a}_t\boldsymbol{a}_t{}'\right)\boldsymbol{R}_j{}'.$$

Observe that Term1 can also be written as

$$\text{Term1} = \begin{bmatrix}\boldsymbol{E}_j\, \boldsymbol{E}_{j,\perp}\end{bmatrix}\begin{bmatrix}\boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{E}_j{}' \\ \boldsymbol{E}_{j,\perp}{}'\end{bmatrix} \tag{3.12}$$

and thus $\lambda_{\max}(\boldsymbol{A}) = \lambda_{\max}(\text{Term1})$. We work with $\lambda_{\max}(\boldsymbol{A})$ in the sequel. We will use the following simple claim.

Claim 3.52. *If $\boldsymbol{X} \succeq 0$ (i.e., $\boldsymbol{X}$ is a p.s.d matrix), where $\boldsymbol{X} \in \mathbb{R}^{r\times r}$, then $\boldsymbol{R}\boldsymbol{X}\boldsymbol{R}' \succeq 0$ for all $\boldsymbol{R} \in \mathbb{R}^{r\times r}$.*

Proof. Since $\boldsymbol{X}$ is p.s.d., $\boldsymbol{y}'\boldsymbol{X}\boldsymbol{y} \geq 0$ for any vector $\boldsymbol{y}$. Use this with $\boldsymbol{y} = \boldsymbol{R}'\boldsymbol{z}$ for any $\boldsymbol{z} \in \mathbb{R}^r$. We get $\boldsymbol{z}'\boldsymbol{R}\boldsymbol{X}\boldsymbol{R}'\boldsymbol{z} \geq 0$. Since this holds for all $\boldsymbol{z}$, $\boldsymbol{R}\boldsymbol{X}\boldsymbol{R}' \succeq 0$. □

By Lemma 3.55 from Appendix 3.10, with $\epsilon_0 = 0.01\lambda^-$,

$$\Pr\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - (\lambda^- - \epsilon_0)\boldsymbol{I} \succeq 0\right) \geq 1 - 2n^{-10}$$

By Claim 3.52 from above, with probability $1 - 2n^{-10}$,

$$\boldsymbol{R}_j\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - (\lambda^- - \epsilon_0)\boldsymbol{I}\right)\boldsymbol{R}_j{}' \succeq 0$$
$$\implies \lambda_{\min}\left(\boldsymbol{R}_j\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - (\lambda^- - \epsilon_0)\boldsymbol{I}\right)\boldsymbol{R}_j{}'\right) \geq 0$$

Using Weyl's inequality, with the same probability,

$$\lambda_{\min}\left(\boldsymbol{R}_j\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - (\lambda^- - \epsilon_0)\boldsymbol{I}\right)\boldsymbol{R}_j{}'\right)$$
$$\leq \lambda_{\max}\left(\boldsymbol{R}_j\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}'\right)\boldsymbol{R}_j{}'\right) - (\lambda^- - \epsilon_0)\lambda_{\max}\left(\boldsymbol{R}_j\boldsymbol{R}_j{}'\right)$$

and so,

$$\lambda_{\max}(\boldsymbol{A}) \geq (\lambda^- - \epsilon_0)\lambda_{\max}(\boldsymbol{R}_j\boldsymbol{R}_j{}'). \tag{3.13}$$

Using Lemma 3.50 and since $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \varepsilon$ we get

$$\lambda_{\max}(\boldsymbol{R}_j\boldsymbol{R}_j{}') = \|\boldsymbol{R}_j\|^2 = \mathrm{SE}^2(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j) \geq (\mathrm{SE}_j - 2\varepsilon)^2 \tag{3.14}$$

Thus, combining (3.11), (3.12), (3.13), (3.14), w.p. at least $1 - 10n^{-10}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t{}'\boldsymbol{\Phi}\right)$$
$$\geq 0.99\lambda^-(\mathrm{SE}_j - 2\varepsilon)^2 - 2.02\sqrt{b_0}(1.2(\varepsilon + \mathrm{SE}_j)^2)\lambda^+$$
$$\geq 0.99\lambda^-\mathrm{SE}_j(0.6\mathrm{SE}_j - 4.8\varepsilon) = 0.59\lambda^-\mathrm{SE}_j(\mathrm{SE}_j - 8\varepsilon)$$

In the above, we used $\sqrt{b_0}f = 0.1$. Since $\mathrm{SE}_j > 9\sqrt{f}\varepsilon$, $0.59\lambda^-\mathrm{SE}_j(\mathrm{SE}_j - 8\varepsilon) > 5\lambda^+\varepsilon^2 > \omega_{evals}$.

Proof of Item 2: We proceed as in the proof of item 1 except that now $\boldsymbol{\Phi} = \boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_j \hat{\boldsymbol{P}}_j'$. Thus, $q = q_K = \varepsilon$ and $\|\boldsymbol{\Phi}\boldsymbol{P}_j\| \leq q_K$. Using Weyl's inequality and Corollary 3.54 from Appendix 3.10, w.p. at least $1 - 10n^{-10}$,

$$\begin{aligned}
&\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right) \\
&\leq \lambda_{\max}\left(\frac{1}{\alpha}\sum_{t} \boldsymbol{\Phi}\boldsymbol{\ell}_t\boldsymbol{\ell}_t'\boldsymbol{\Phi}\right) + \|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}\| + \lambda_{\max}(\boldsymbol{\Phi}\text{noise}\boldsymbol{\Phi}) \\
&\leq \lambda_{\max}\left(\underbrace{\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\boldsymbol{P}_j\boldsymbol{a}_t\boldsymbol{a}_t'\boldsymbol{P}_j'\boldsymbol{\Phi}}_{\text{Term1}}\right) \\
&+ 2.02\sqrt{b}\|\boldsymbol{\Phi}\boldsymbol{P}_j\|q_K\lambda^+ + 1.01\sqrt{b}q_K^2\lambda^+ + \varepsilon^2\lambda^-
\end{aligned}$$

Proceeding as before to bound $\lambda_{\max}(\text{Term1})$, define $\boldsymbol{\Phi}\boldsymbol{P}_j \overset{QR}{=} \boldsymbol{E}_j\boldsymbol{R}_j$, define $\boldsymbol{A}$ as before, we know $\lambda_{\max}(\text{Term1}) = \lambda_{\max}(\boldsymbol{E}_j'(\text{Term1})\boldsymbol{E}_j) = \lambda_{\max}(\boldsymbol{A})$. Further,

$$\begin{aligned}
\lambda_{\max}(\boldsymbol{A}) &= \lambda_{\max}\left(\boldsymbol{R}_j\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{a}_t\boldsymbol{a}_t'\right)\boldsymbol{R}_j'\right) \\
&\overset{(a)}{\leq} \lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{a}_t\boldsymbol{a}_t\right)\lambda_{\max}(\boldsymbol{R}_j\boldsymbol{R}_j')
\end{aligned}$$

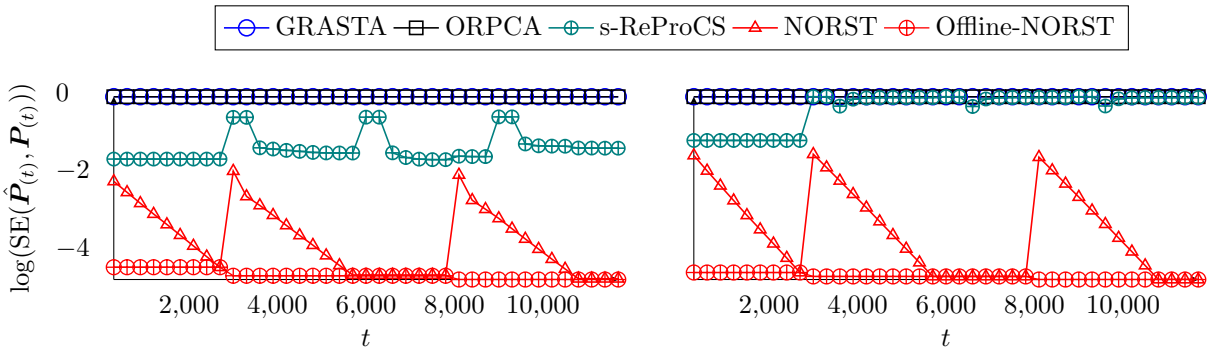
where (a) uses Ostrowski's theorem [12, Theorem 5.4.9]. We have

$$\lambda_{\max}(\boldsymbol{R}_j\boldsymbol{R}_j') = \sigma^2_{\max}(\boldsymbol{R}_j) = \|\boldsymbol{\Phi}\boldsymbol{P}_j\|^2 \leq \varepsilon^2$$

and we can bound $\lambda_{\max}(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{a}_t\boldsymbol{a}_t')$ using the first item of Lemma 3.55. Combining all of the above, and using $\|\boldsymbol{\Phi}\boldsymbol{P}_j\| \leq q_K \leq \varepsilon$ and $b_0 f^2 = 0.01$, w.p. at least $1 - 10n^{-10}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha}\boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right) \leq 1.37\varepsilon^2\lambda^+$$

Recall that $\omega_{evals} = 2\varepsilon^2\lambda^+$ and thus, $1.37\varepsilon^2\lambda^+ < \omega_{evals}$. □



Outlier Model	GRASTA (0.02 ms)	s-ReProCS (0.9 ms)	ORPCA (1.2ms)	**NORST (0.9 ms)**	RPCA-GD (7.8ms)	AltProj (4.6ms)	**smoothing-NORST (1.7ms)**
Moving Object	0.630	0.598	0.861	$\mathbf{4.23 \times 10^{-4}}$	4.283	4.441	$\mathbf{8.2 \times 10^{-6}}$
Bernoulli	6.163	2.805	1.072	**0.002**	0.092	0.072	$\mathbf{2.3 \times 10^{-4}}$

Figure 3.1: **Top:** Left plot illustrates the $\boldsymbol{\ell}_t$ error for outlier supports generated using Moving Object Model and right plot illustrates the error under the Bernoulli model. The values are plotted every $k\alpha - 1$ time-frames. **Bottom:** Comparison of $\|\hat{\boldsymbol{L}} - \boldsymbol{L}\|_F / \|\boldsymbol{L}\|_F$ for Online and offline RPCA methods. Average time for the Moving Object model is given in parentheses. The offline (batch) methods are performed once on the complete dataset.

3.7 Empirical Evaluation

In this section we present numerical experiments on synthetic and real data to validate our theoretical claims. Extra experimental details are presented in the Supplementary Material.

Synthetic Data. First we compare the results of NORST and smoothing-NORST with RST, Online RPCA, and static RPCA methods. We generate the changing subspaces, $\boldsymbol{P}_j = e^{\gamma_j \boldsymbol{B}_j} \boldsymbol{P}_{j-1}$ as done in [11] where γ_j controls the amount of subspace change and $\boldsymbol{B}_j$'s are skew-symmetric matrices. In the first experiment we used the following parameters. $n = 1000$, $d = 12000$, $J = 2$, $t_1 = 3000$, $t_2 = 8000$, $r = 30$, $\gamma_1 = 0.001$, $\gamma_2 = \gamma_1$. We set $\alpha = 300$. Next, we generate the coefficients $\boldsymbol{a}_t \in \mathbb{R}^r$ as independent zero-mean, bounded random variables. They are $(\boldsymbol{a}_t)_i \overset{i.i.d}{\sim} unif[-q_i, q_i]$ where $q_i = \sqrt{f} - \sqrt{f}(i-1)/2r$ for $i = 1, 2, \cdots, r-1$ and $q_r = 1$. thus the condition number is f and we selected $f = 50$. For the sparse supports, we considered two models according to which the supports are generated. First we use Model G.24 [24] which simulates a moving object pacing in the video. For the first $t_{\text{train}} = 100$ frames, we used a smaller fraction of outliers with

parameters $s/n = 0.01$, $b_0 = 0.01$. For $t > t_{\text{train}}$ we used $s/n = 0.05$ and $b_0 = 0.3$. Secondly, we used the Bernoulli model to simulate sampling uniformly at random, i.e., each entry of the matrix, is independently selected with probability $\rho = 0.01$ for the first t_{train} frames and with probability $\rho = 0.3$ for subsequent frames. The sparse outlier magnitudes for both support models are generated uniformly at random from the interval $[x_{\min}, x_{\max}]$ with $x_{\min} = 10$ and $x_{\max} = 20$.

We initialized the s-ReProCS and NORST algorithms using AltProj applied to $\boldsymbol{Y}_{[1,t_{\text{train}}]}$ with $t_{\text{train}} = 100$. For the parameters to AltProj we used used the true value of r, 15 iterations and a threshold of 0.01. This, and the choice of γ_1 and γ_2 ensure that $\text{SE}(\hat{\boldsymbol{P}}_{\text{init}}, \boldsymbol{P}_0) \approx \text{SE}(\boldsymbol{P}_1, \boldsymbol{P}_0) \approx \text{SE}(\boldsymbol{P}_2, \boldsymbol{P}_1) \approx 0.01$. The other algorithm parameters are set as mentioned in the theorem, i.e., $K = \lceil \log(c/\varepsilon) \rceil = 8$, $\alpha = Cr \log n = 300$, $\omega = x_{\min}/2 = 5$ and $\xi = 7x_{\min}/15 = 0.67$, $\omega_{evals} = 2\varepsilon^2\lambda^+ = 7.5 \times 10^{-4}$. For the other online methods we implement the algorithms without modifications. The regularization parameter for ORPCA was set as with $\lambda_1 = 1/\sqrt{n}$ and $\lambda_2 = 1/\sqrt{d}$ according to [9]. Wherever possible we set the tolerance as 10^{-6} and 100 iterations to match that of our algorithm. As shown in Fig. 3.1, NORST is significantly better than all the RST methods - s-ReProCS [24], and two popular heuristics - ORPCA [9] and GRASTA [11].

We also provide a comparison of smoothing techniques in Fig 3.1. To ensure a valid time comparison, we implement the static RPCA methods on the entire data matrix $\boldsymbol{Y}$. Although, we could implement the static techniques on disjoint batches of size α, we observed that this did not yield significant improvement in terms of reconstruction accuracy, while being considerably slower, and thus we report only the latter setting. As can be seen, smoothing NORST outperforms all static RPCA methods, both for the moving object and the Bernoulli models. For the batch comparison we used PCP, AltProj and RPCA-GD. We set the regularization parameter for PCP $1/\sqrt{n}$ in accordance with [3]. The other known parameters, r for Alt-Proj, outlier-fraction for RPCA-GD, are set the ground truth. For all algorithms we set the threshold as 10^{-6} and the number of iterations to 100. All results are averaged over 100 independent runs.

In Fig. 3.2 we validate our claim of NORST admitting a higher fraction of outliers per row. We only compare with AltProj since it is has the highest tolerance among other methods. We chose 10

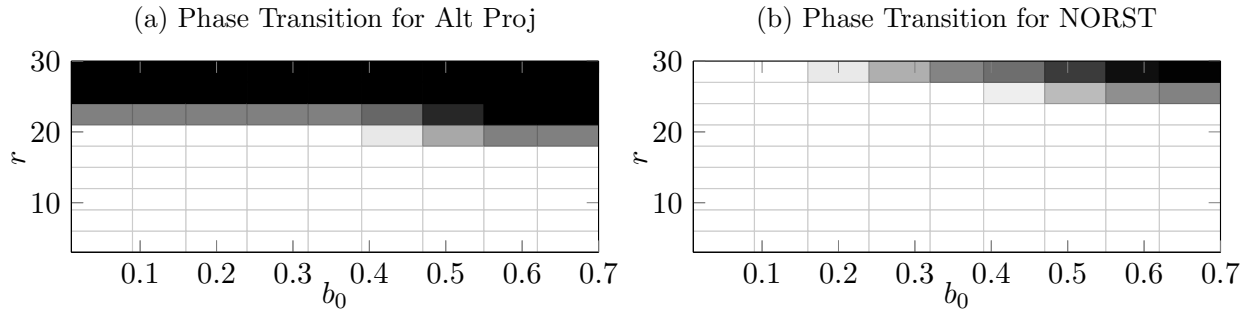


Figure 3.2: Empirical probability that $\|\hat{\boldsymbol{L}} - \boldsymbol{L}\|_F / \|\boldsymbol{L}\|_F < 0.5$ for AltProj and for smoothing NORST. Note that NORST indeed has a much higher tolerance to outlier fraction per row as compared to AltProj. Black denotes 0 and white denotes 1.

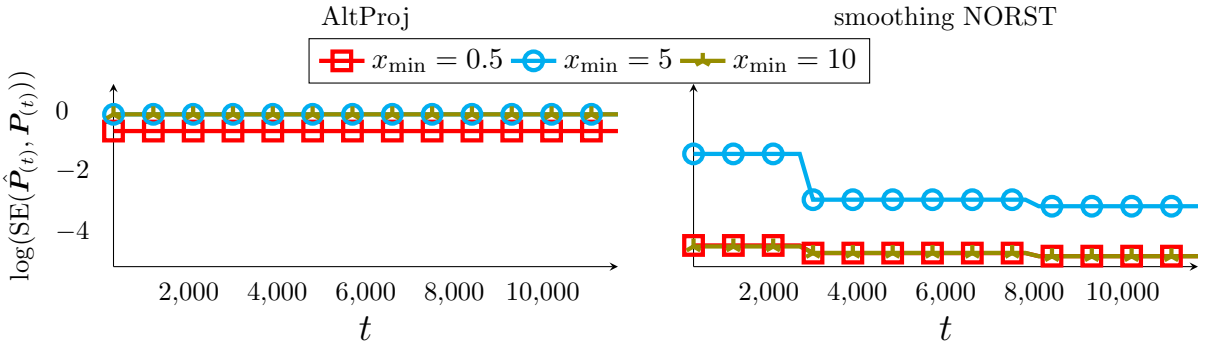


Figure 3.3: In the above plots we show the variation of the subspace errors for varying $x_{\min}$. In particular, we set all the non-zero outlier values to $x_{\min}$. The results are averaged over 100 independent trials.

different values of each of r and b_0 (we slightly misuse notation here to let $b_0 :=$ max-outlier-frac-row for this section only). For each pair of b_0 and r we implemented NORST and ALtProj over 100 independent trials and computed the relative error, $\|\hat{\boldsymbol{L}} - \boldsymbol{L}\|_F / \|L\|_F$ for each run. We illustrate the fraction of times the error seen by each algorithm is less than a threshold, 0.5. We chose this threshold since for smaller values, AltProj consistently failed. As can be seen, NORST is able to tolerate a much larger fraction of outlier-per-row as compared to AltProj.

In the third experiment we analyze the effect of the lower bound on the outlier magnitude $x_{\min}$ with the performance of NORST and AltProj. We show the results in Fig. 3.3. The only change in data generation is that we now choose three different values of $x_{\min} = \{0.5, 5, 10\}$, and

we set all the non-zero entries of the sparse matrix to be equal to $x_{\min}$. This is actually harder than allowing the sparse outliers to take on any value since for a moderately low value of $x_{\min}$ the outlier-lower magnitude bound of Theorem 3.39 is violated. This is indeed confirmed by the numerical results presented in Fig. 3.3. (i) When $x_{\min} = 0.5$, NORST works well since now all the outliers get classified as the small unstructured noise $\boldsymbol{v}_t$. (ii) When $x_{\min} = 10$, NORST still works well because now $x_{\min}$ is large enough so that the outlier support is mostly correctly recovered. (iii) But when $x_{\min} = 5$ the NORST reconstruction error stagnates around 10^{-3}. All AltProj errors are much worse than those of NORST because the outlier fraction per row is the same as in the first experiment and thus the effect of varying $x_{\min}$ is not pronounced.

Real Data. We also evaluate our algorithm for the task of Background Subtraction. For the AltProj algorithm we set $r = 40$. The remaining parameters were used with default setting. For NORST, we set $\alpha = 60$, $K = 3$, $\xi_t = \|\boldsymbol{\Psi}\hat{\boldsymbol{\ell}}_{t-1}\|_2$. We found that these parameters work for most videos that we verified our algorithm on. For RPCA-GD we set the "corruption fraction" $\alpha = 0.2$ as described in their paper.

We use two standard datasets, the Meeting Room (MR) and the Lobby (LB) sequences. LB is a relatively easy sequence since the background is static for the most part, and the foreground occlusions are small in size. As can be seen from Fig. 3.4 (first two rows), most algorithms perform well on this dataset. MR is a challenging data set since the color of the foreground (person) is very similar to the background curtains, and the size of the object is very large. Thus, NORST is able to outperform all methods, while being fast.

3.8 Conclusions and Future Directions

In this work we developed a fast and (nearly) delay optimal robust subspace tracking solution that we called NORST. NORST is a mini-batch algorithm with memory complexity that is also nearly optimal. It detects subspace changes and tracks them to ε accuracy with a delay that is more than the subspace dimension r by only log factors: the delay is order $r \log n \log(1/\varepsilon)$. The memory complexity is n times this number while nr is the amount of memory required to store the

Figure 3.4: Comparison of visual performance in Foreground Background separation. The first two rows are for the LB dataset and the last two rows are for the MR dataset. The time taken by each algorithm (per frame) in milliseconds is provided in parenthesis.

output subspace estimate. Our guarantee for NORST needs assumptions similar to those needed by standard robust PCA solutions. Different from standard robust PCA, we need slow subspace change, we replace right singular vectors' incoherence by a statistical version of it, but we need a weaker bound on outlier fractions per row.

Slow subspace change is a natural assumption for background images of static camera videos (with no sudden scene changes). Our statistical assumptions on $\boldsymbol{a}_t$ are mild and can be relaxed further. As already explained, the identically distributed requirement can be relaxed. In the video application, the zero mean assumption can be approximately satisfied if we estimate the mean background image by computing the empirical average of the first t_{train} frames, $\hat{\boldsymbol{L}}_{[1:t_{\text{train}}]}$, obtained using AltProj. Mutual independence of $\boldsymbol{a}_t$'s models the fact that the changes in each background image w.r.t. a "mean" background are independent, when conditioned on their subspace. This is valid, for example, if the background changes are due to illumination variations or due to moving

curtains (see Fig. 3.4). Mutual independence can be relaxed to instead assuming an autoregressive model on the $\boldsymbol{a}_t$'s: this will require using the matrix Azuma inequality [30] to replace matrix Bernstein. We believe the zero mean requirement can also be eliminated.

Acknowledgments

The authors would like to thank Praneeth Netrapalli and Prateek Jain of Microsoft Research India for fruitful discussions on strengthening the guarantee by removing assumptions on subspace change model.

3.9 References

[1] Cai, T. T., Ma, Z., and Wu, Y. Sparse pca: Optimal rates and adaptive estimation. *The Annals of Statistics 41*, 6 (2013), 3074–3110.

[2] Candes, E. The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris Serie I* (2008).

[3] Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *J. ACM 58*, 3 (2011).

[4] Candes, E. J., and Recht, B. Exact matrix completion via convex optimization. *Found. of Comput. Math*, 9 (2008), 717–772.

[5] Chandrasekaran, V., Sanghavi, S., Parrilo, P. A., and Willsky, A. S. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization 21* (2011).

[6] Cherapanamjeri, Y., Gupta, K., and Jain, P. Nearly-optimal robust matrix completion. *ICML* (2016).

[7] Davis, C., and Kahan, W. M. The rotation of eigenvectors by a perturbation. iii. *SIAM J. Numer. Anal. 7* (Mar. 1970), 1–46.

[8] Dung, N. V., Trung, N. L., Abed-Meraim, K., et al. Robust subspace tracking with missing data and outliers via admm. In *2019 27th European Signal Processing Conference (EUSIPCO)* (2019), IEEE, pp. 1–5.

[9] Feng, J., Xu, H., and Yan, S. Online robust pca via stochastic optimization. In *NIPS* (2013).

[10] Guo, H., Qiu, C., and Vaswani, N. An online algorithm for separating sparse and low-dimensional signal sequences from their sum. *IEEE Trans. Sig. Proc. 62*, 16 (2014), 4284–4297.

[11] He, J., Balzano, L., and Szlam, A. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *IEEE Conf. on Comp. Vis. Pat. Rec. (CVPR)* (2012).

[12] Horn, R., and Johnson, C. *Matrix Analysis*. Cambridge Univ. Press, 1985.

[13] Hsu, D., Kakade, S. M., and Zhang, T. Robust matrix decomposition with sparse corruptions. *IEEE Trans. Info. Th.* (Nov. 2011).

[14] Ipsen, I. C. An overview of relative sin θ theorems for invariant subspaces of complex matrices. *Journal of computational and applied mathematics 123*, 1-2 (2000), 131–153.

[15] Javed, S., Mahmood, A., Dias, J., and Werghi, N. Robust structural low-rank tracking. *IEEE Transactions on Image Processing 29* (2020), 4390–4405.

[16] Ke, Z. T., and Wang, M. A new svd approach to optimal topic estimation. *arXiv preprint arXiv:1704.07016* (2017).

[17] Koltchinskii, V., Lounici, K., et al. Normal approximation and concentration of spectral projectors of sample covariance. *The Annals of Statistics 45*, 1 (2017), 121–157.

[18] Li, R.-C. Relative perturbation theory: Ii. eigenspace and singular subspace variations. *SIAM J. Matrix Anal. Appl. 20*, 2 (1998), 471–492.

[19] Lois, B., and Vaswani, N. Online matrix completion and online robust pca. In *IEEE Intl. Symp. Info. Th. (ISIT)* (2015).

[20] Musco, C., and Musco, C. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *Advances in Neural Information Processing Systems* (2015), pp. 1396–1404.

[21] Nadler, B. Finite sample approximation results for principal component analysis: A matrix perturbation approach. *Ann. Statist.* (2008).

[22] Narayanamurthy, P., Daneshpajooh, V., and Vaswani, N. Provable subspace tracking from missing data and matrix completion. *IEEE Transactions on Signal Processing* (2019), 4245–4260.

[23] Narayanamurthy, P., and Vaswani, N. Nearly optimal robust subspace tracking. In *International Conference on Machine Learning* (2018), pp. 3701–3709.

[24] Narayanamurthy, P., and Vaswani, N. Provable dynamic robust pca or robust subspace tracking. *IEEE Transactions on Information Theory 65*, 3 (2019), 1547–1577.

[25] Netrapalli, P., Niranjan, U. N., Sanghavi, S., Anandkumar, A., and Jain, P. Non-convex robust pca. In *NIPS* (2014).

[26] Ozdemir, A., Bernat, E. M., and Aviyente, S. Recursive tensor subspace tracking for dynamic brain network analysis. *IEEE Transactions on Signal and Information Processing over Networks* (2017).

[27] Qiu, C., and Vaswani, N. Real-time robust principal components' pursuit. In *Allerton Conf. on Communication, Control, and Computing* (2010).

[28] Qiu, C., Vaswani, N., Lois, B., and Hogben, L. Recursive robust pca or recursive sparse recovery in large but structured noise. *IEEE Trans. Info. Th.* (August 2014), 5007–5039.

[29] Schölkopf, B., Smola, A., and Müller, K.-R. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation 10*, 5 (1998), 1299–1319.

[30] Tropp, J. A. Just relax: Convex programming methods for identifying sparse signals. *IEEE Trans. Info. Th.* (March 2006), 1030–1051.

[31] Tropp, J. A. User-friendly tail bounds for sums of random matrices. *Found. Comput. Math. 12*, 4 (2012).

[32] Vaswani, N., Bouwmans, T., Javed, S., and Narayanamurthy, P. Robust subspace learning: Robust pca, robust subspace tracking, and robust subspace recovery. *IEEE signal processing magazine 35*, 4 (2018), 32–55.

[33] Vaswani, N., and Lu, W. Modified-CS: Modifying compressive sensing for problems with partially known support. *IEEE Trans. Signal Processing* (September 2010).

[34] Vaswani, N., and Narayanamurthy, P. Pca in sparse data-dependent noise. In *ISIT* (2018), pp. 641–645.

[35] Vu, V. Q., and Lei, J. Minimax sparse principal subspace estimation in high dimensions. *Annals of Statistics* (2013).

[36] Xiao, L., and Zhang, T. A proximal-gradient homotopy method for the l1-regularized least-squares problem. In *ICML* (2012).

[37] Ye, K., and Lim, L. H. Schubert varieties and distances between subspaces of different dimensions. *SIAM Journal on Matrix Analysis and Applications 37*, 3 (2016), 1176–1197.

[38] Yi, X., Park, D., Chen, Y., and Caramanis, C. Fast algorithms for robust pca via gradient descent. In *NIPS* (2016).

[39] Zhan, J., Lois, B., Guo, H., and Vaswani, N. Online (and Offline) Robust PCA: Novel Algorithms and Performance Guarantees. In *Intnl. Conf. Artif. Intell. Stat. (AISTATS)* (2016).

[40] Zhan, J., and Vaswani, N. Robust pca with partial subspace knowledge. *IEEE Trans. Sig. Proc.* (July 2015).

[41] Zhan, J., and Vaswani, N. Time invariant error bounds for modified-CS based sparse signal sequence recovery. *IEEE Trans. Info. Th. 61*, 3 (2015), 1389–1409.

[42] Zhang, T., Xu, C., and Yang, M.-H. Robust structural sparse tracking. *IEEE transactions on pattern analysis and machine intelligence 41*, 2 (2018), 473–486.

[43] Zwald, L., and Blanchard, G. On the convergence of eigenspaces in kernel principal component analysis. In *Advances in neural information processing systems* (2006), pp. 1649–1656.

3.10 Appendix A: Proofs for Sec. 3.2

3.10.1 Proof of Theorem 3.31

Proof of Theorem 3.31. This uses the Davis-Kahan sin theta theorem [7]:

Lemma 3.53 (Davis-Kahan $\sin\theta$ theorem)**.** *Let $\boldsymbol{D}_0$ be a Hermitian matrix whose span of top r eigenvectors equals* $\text{span}(\boldsymbol{P})$*. Let $\boldsymbol{D}$ be the Hermitian matrix with top r eigenvectors $\hat{\boldsymbol{P}}$. Then,*

$$\begin{aligned} \text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) &\leq \frac{\|(\boldsymbol{D}-\boldsymbol{D}_0)\boldsymbol{P}\|}{\lambda_r(\boldsymbol{D}_0) - \lambda_{r+1}(\boldsymbol{D})} \\ &\leq \frac{\|\boldsymbol{D}-\boldsymbol{D}_0\|}{\lambda_r(\boldsymbol{D}_0) - \lambda_{r+1}(\boldsymbol{D}_0) - \lambda_{\max}(\boldsymbol{D}-\boldsymbol{D}_0)} \end{aligned} \tag{3.15}$$

as long as the denominator is positive. The second inequality follows from the first using Weyl's inequality.

For our proof, set $\boldsymbol{D}_0 = \frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{\ell}_t{}'$. Notice that this is a Hermitian matrix with $\boldsymbol{P}$ as the top r eigenvectors. Let $\boldsymbol{D} = \frac{1}{\alpha}\sum_t \boldsymbol{y}_t\boldsymbol{y}_t{}'$. Recall that $\hat{\boldsymbol{P}}$ is its matrix of top r eigenvectors. Observe

$$\begin{aligned}
\boldsymbol{D} - \boldsymbol{D}_0 &= \frac{1}{\alpha}\sum_t (\boldsymbol{y}_t\boldsymbol{y}_t{}' - \boldsymbol{\ell}_t\boldsymbol{\ell}_t{}') \\
&= \frac{1}{\alpha}\sum_t (\boldsymbol{w}_t\boldsymbol{w}_t{}' + \boldsymbol{v}_t\boldsymbol{v}_t{}' + \boldsymbol{\ell}_t\boldsymbol{w}_t{}' + \boldsymbol{v}_t\boldsymbol{w}_t{}' \\
&+ \boldsymbol{\ell}_t\boldsymbol{v}_t{}' + \boldsymbol{w}_t\boldsymbol{\ell}_t{}' + \boldsymbol{w}_t\boldsymbol{v}_t{}' + \boldsymbol{v}_t\boldsymbol{\ell}_t{}') \\
&:= \text{noise}_{\boldsymbol{w}} + \text{noise}_{\boldsymbol{v}_t} + \text{cross}_{\boldsymbol{\ell},\boldsymbol{w}} + \text{cross}_{\boldsymbol{\ell},\boldsymbol{v}_t} \\
&+ \text{cross}_{\boldsymbol{v}_t,\boldsymbol{w}} + \text{cross}_{\boldsymbol{\ell},\boldsymbol{w}}{}' + \text{cross}_{\boldsymbol{\ell},\boldsymbol{v}_t}{}' + \text{cross}_{\boldsymbol{v}_t,\boldsymbol{w}}{}' \\
&:= \text{noise} + \text{cross} + \text{cross}'
\end{aligned}$$

Also notice that $\lambda_{r+1}(\boldsymbol{D}_0) = 0$, $\lambda_r(\boldsymbol{D}_0) = \lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}'\right)$. Now, applying Theorem 3.53,

$$\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \frac{2\|\text{cross}\| + \|\text{noise}\|}{\lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}'\right) - \text{numerator}}$$

Now, we can bound $\|\text{cross}\| \leq \|\mathbb{E}[\text{cross}]\| + \|\text{cross} - \mathbb{E}[\text{cross}]\|$ and similarly for the noise term. We use the Cauchy-Schwartz inequality for bounding the expected values of both.

Recall that $\boldsymbol{M}_t = \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}$ with $b := \|\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}{}'\|$ and $q := \max_t \|\boldsymbol{M}_{1,t}\boldsymbol{P}\|$ with $q < 2$. Thus,

$$\begin{aligned}
\|\mathbb{E}[\text{noise}]\| &\leq \left\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_t\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\boldsymbol{M}_{1,t}{}'\boldsymbol{M}_{2,t}{}'\right\| + \|\Sigma_{\boldsymbol{v}_t}\| \\
&\leq \sqrt{\left\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_t\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\boldsymbol{M}_{1,t}{}'(\cdot)'\right\| \left\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}{}'\right\|} + \lambda_v^+ \\
&\leq \sqrt{\max_t \|\boldsymbol{M}_t\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\boldsymbol{M}_{1,t}{}'\|^2 \; b} + \lambda_v^+ \leq \sqrt{b}q^2\lambda^+ + \lambda_v^+
\end{aligned} \tag{3.16}$$

Similarly,

$$\begin{aligned}
\|\mathbb{E}[\text{cross}_{\boldsymbol{\ell},\boldsymbol{w}_t}]\| &= \left\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{1,t}\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\right\| \\
&\leq \sqrt{\left\|\frac{1}{\alpha}\sum_t \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\boldsymbol{M}_{1,t}{}'\boldsymbol{M}_{1,t}\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\right\| \left\|\frac{1}{\alpha}\sum_t \boldsymbol{M}_{2,t}\boldsymbol{M}_{2,t}{}'\right\|} \\
&\leq \sqrt{\max_t \|\boldsymbol{M}_{1,t}\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}'\|^2 \; b} \leq \sqrt{b}q\lambda^+
\end{aligned} \tag{3.17}$$

Since $\boldsymbol{v}_t$ is uncorrelated noise, $\mathbb{E}[\text{cross}_{\boldsymbol{\ell},\boldsymbol{v}_t}] = 0$ and $\mathbb{E}[\text{cross}_{\boldsymbol{w}_t,\boldsymbol{v}_t}] = 0$. We now lower bound $\lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}'\right)$ as

$$\begin{aligned}\lambda_{\min}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}'\right) &= \lambda_{\min}\left(\boldsymbol{\Lambda} - \left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\right)\right)\\ &\geq \lambda_{\min}(\boldsymbol{\Lambda}) - \lambda_{\max}\left(\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\right)\\ &\geq \lambda^- - \left\|\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\right\|\end{aligned}$$

and thus we have

$$\begin{aligned}&\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P})\\ &\leq \frac{4\sqrt{b}q\lambda^+ + \lambda_v^+ + 2\|\text{cross} - \mathbb{E}[\text{cross}]\| + \|\text{noise} - \mathbb{E}[\text{noise}]\|}{\lambda^- - \left\|\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\right\| - \text{numerator}}\end{aligned}$$

Concentration bounds. Now we only need to bound $\|\text{noise} - \mathbb{E}[\text{noise}]\|$ and $\|\text{cross} - \mathbb{E}[\text{cross}]\|$. These are often referred to as "statistical error", while the error due to nonzero $\|\mathbb{E}[\text{cross}]\|$ or $\|\mathbb{E}[\text{noise}]\|$ is called the "bias". We use concentration bounds from Lemma 3.55.

$$\begin{aligned}&\|\text{noise} - \mathbb{E}[\text{noise}]\| + 2\|\text{cross} - \mathbb{E}[\text{cross}]\|\\ \leq&\left\|\frac{1}{\alpha}\sum_t(\boldsymbol{w}_t\boldsymbol{w}_t{}' - \mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t{}'])\right\| + \left\|\frac{1}{\alpha}\sum_t(\boldsymbol{v}_t\boldsymbol{v}_t{}' - \mathbb{E}[\boldsymbol{v}_t\boldsymbol{v}_t{}'])\right\|\\ &+ 2\left\|\frac{1}{\alpha}\sum_t(\boldsymbol{\ell}_t\boldsymbol{w}_t{}' - \mathbb{E}[\boldsymbol{\ell}_t\boldsymbol{w}_t{}'])\right\|\\ &+ 2\left\|\frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{v}_t{}'\right\| + 2\left\|\frac{1}{\alpha}\sum_t \boldsymbol{w}_t\boldsymbol{v}_t{}'\right\|\\ \leq&C\sqrt{\eta}q^2 f\sqrt{\frac{r\log n}{\alpha}}\lambda^- + C\sqrt{\eta}\frac{\lambda_v^+}{\lambda^-}\sqrt{\frac{r\log n}{\alpha}}\lambda^-\\ &+ C\sqrt{\eta}qf\sqrt{\frac{r\log n}{\alpha}}\lambda^-\\ &+ C\sqrt{\eta}\sqrt{\frac{\lambda_v^+}{\lambda^-}}f\sqrt{\frac{r\log n}{\alpha}}\lambda^- + C\sqrt{\eta}q\sqrt{\frac{\lambda_v^+}{\lambda^-}}f\sqrt{\frac{r\log n}{\alpha}}\lambda^-\\ \leq&C\sqrt{\eta}qf\sqrt{\frac{r\log n}{\alpha}}\lambda^- + C\sqrt{\eta}\sqrt{\frac{\lambda_v^+}{\lambda^-}}f\sqrt{\frac{r\log n}{\alpha}}\lambda^- := H(\alpha)\lambda^-\end{aligned}$$

where the last line follows from using $q^2 < 2q$ and $\lambda_v^+ \leq \lambda^+$.

In case we only need to bound $\|\text{noise} - \mathbb{E}[\text{noise}]\|$, we can get a tighter bound that contains only the first two terms and not all five. Clearly, we have

$$\begin{aligned}
&\|\text{noise} - \mathbb{E}[\text{noise}]\| \\
&\leq C\sqrt{\eta}q^2 f\sqrt{\frac{r\log n}{\alpha}}\lambda^- + C\sqrt{\eta}\frac{\lambda_v^+}{\lambda^-}\sqrt{\frac{r\log n}{\alpha}}\lambda^- + \\
&:= H_{noise}(\alpha)
\end{aligned}$$

The bound on $\|\frac{1}{\alpha}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t{}' - \boldsymbol{\Lambda}\|_2$ follows directly from the first item of Lemma 3.55. □

3.10.2 A useful corollary that follows from above proof

From the above proof, we can write out a bound for $\|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}'\|$ for a projection matrix $\boldsymbol{\Phi}$ by noticing that each term of cross is of the form $\sum_t \boldsymbol{\ell}_t(.)' = \boldsymbol{P}\sum_t \boldsymbol{a}_t(.)'$. Thus $\|\boldsymbol{P}'\text{cross}\| = \|\text{cross}\|$. Thus, $\|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}'\| \leq \|\boldsymbol{\Phi}\boldsymbol{P}\|\|\text{cross}\| \leq \|\boldsymbol{\Phi}\boldsymbol{P}\|(\|\mathbb{E}[\text{cross}]\| + \|\text{cross} - \mathbb{E}[\text{cross}]\|)$. Similarly, we can also get a bound on $\lambda_{\max}(\text{noise}) = \|\text{noise}\|$.

Assume $b = 0.01/f^2$, $q > \varepsilon > \sqrt{g}$. Consider cross. If $\alpha \geq C\max\left(\frac{q^2f^2}{\epsilon_1^2}r\log n, \frac{gf}{\epsilon_1^2}\max(r_v, r)\log n\right)$, then $H(\alpha) \leq \epsilon_1\lambda^-$. If we set $\epsilon_1 = 0.002\max(\sqrt{b}q, \sqrt{b}\varepsilon)$ and $b = 0.01/f^2$ (bound on max-outlier-frac-row(α)), then, since $\varepsilon > \sqrt{g}$, $\alpha = Cf^2\max(r_v, r)\log n$ suffices. Since $q \geq \varepsilon$, then, $\epsilon_1 = 0.002\sqrt{b}q$. Thus,

$$\|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}'\| \leq \|\boldsymbol{\Phi}\boldsymbol{P}\|(2\sqrt{b}q\lambda^+ + H(\alpha)\lambda^-) \leq 2.02\sqrt{b}\|\boldsymbol{\Phi}\boldsymbol{P}\|q\lambda^+$$

Consider noise. We will use $H_{noise}(\alpha)$ for this. If $\alpha > C\max\left(\frac{q^4f^2}{\epsilon_2^2}r\log n, \frac{g^2}{\epsilon_2^2}\max(r_v, r)\log n\right)$, then $H_{noise}(\alpha) \leq \epsilon_2\lambda^-$. If we set $\epsilon_2 = 0.002\sqrt{b}\max(q^2, \varepsilon^2)$, then since $\varepsilon^4 > g^2$, thus, $\alpha = Cf^2\max(r_v, r)\log n$ suffices. Since $q > \varepsilon$, $\epsilon_2 = 0.002\sqrt{b}q^2$. We have the following corollary.

Corollary 3.54. *If* $\alpha = Cf^2 \max(r_v, r) \log n$, *and if* $q \geq \varepsilon > \sqrt{g}$, *then, w.p.* $1 - 10n^{-10}$,

$$\begin{aligned}
\|\boldsymbol{\Phi}\text{cross}\boldsymbol{\Phi}'\| &\leq \|\boldsymbol{\Phi}\boldsymbol{P}\|(2\sqrt{b}q\lambda^+ + H(\alpha)\lambda^-) \\
&\leq 2.02\sqrt{b}\|\boldsymbol{\Phi}\boldsymbol{P}\|q\lambda^+, \\
\lambda_{\max}(\boldsymbol{\Phi}\text{noise}\boldsymbol{\Phi}) \leq \|\text{noise}\| &\leq \sqrt{b}q^2\lambda^+ + \lambda_v^+ + H(\alpha)\lambda^- \\
&\leq 1.01\sqrt{b}q^2\lambda^+ + \lambda_v^+ \\
&\leq 1.01\sqrt{b}q^2\lambda^+ + \varepsilon^2\lambda^-
\end{aligned}$$

3.10.3 Main idea of the proof of Corollary 3.43

The key difference in this proof is our choice of $\boldsymbol{D}_0$. Since we want to bound $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P})$, we need to pick it in such a way that its matrix of top r singular vectors equals $\text{span}(\boldsymbol{P})$. We pick

$$\boldsymbol{D}_0 = \frac{1}{\alpha}\boldsymbol{P}\left((\alpha - \alpha_0)\boldsymbol{\Lambda} + \alpha_0\boldsymbol{P}'\boldsymbol{P}_0\boldsymbol{\Lambda}\boldsymbol{P}_0'\boldsymbol{P}\right)\boldsymbol{P}'$$

Clearly, $\lambda_{r+1}(\boldsymbol{D}_0) = 0$. With this choice of $\boldsymbol{D}_0$,

$$\begin{aligned}
\boldsymbol{D} - \boldsymbol{D}_0 = \text{cross} + \text{cross}' + \text{noise} \\
+ \left(\frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{\ell}_t' - \mathbb{E}[\frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{\ell}_t']\right) + \left(\mathbb{E}[\frac{1}{\alpha}\sum_t \boldsymbol{\ell}_t\boldsymbol{\ell}_t'] - \boldsymbol{D}_0\right)
\end{aligned}$$

where $\text{cross}, \text{noise}$ are as defined earlier with the change that $\boldsymbol{\ell}_t$ is now defined differently. Thus, the only thing that changes when bounding these is our definition of q. The last term in the expression above equals $c_0\boldsymbol{P}_\perp\boldsymbol{P}_\perp'\boldsymbol{P}_0\boldsymbol{\Lambda}\boldsymbol{P}_0'\boldsymbol{P}_\perp\boldsymbol{P}_\perp{}' + c_0\boldsymbol{P}_\perp\boldsymbol{P}_\perp'\boldsymbol{P}_0\boldsymbol{\Lambda}\boldsymbol{P}_0'\boldsymbol{P}\boldsymbol{P}' + (.)'$ with $c_0 := \frac{\alpha_0}{\alpha}$. This is what generates the extra $4\Delta f$ term in our SE bound. A complete proof is provided in the Appendix.

3.10.4 Concentration Bounds

We state the lemma below so that it can also be used in proving the most general PCA result given in the Supplement. Let $\boldsymbol{\Lambda}_t = \mathbb{E}[\boldsymbol{a}_t\boldsymbol{a}_t{}']$, $\bar{\boldsymbol{\Lambda}} = \frac{1}{\alpha}\sum_t \boldsymbol{\Lambda}_t$, $\lambda^+_{\text{max}} := \max_t \|\boldsymbol{\Lambda}_t\|$, $\lambda^-_{\text{avg}} = \lambda_{\min}(\bar{\boldsymbol{\Lambda}})$, $f = \lambda^+_{max}/\lambda^-_{\text{avg}}$, $\lambda^+_{v,\max} := \max_t \|\mathbb{E}[\boldsymbol{v}_t\boldsymbol{v}_t{}']\|$ and $g = \lambda^+_{v,\max}/\lambda^-_{\text{avg}}$.

To use the lemma under the simpler i.i.d. assumption used in the main paper, remove the $_{\text{max,avg}}$ subscripts from all terms, e.g., replace λ^+_{max} by λ^+, λ^-_{avg} by λ^- and so on.

Lemma 3.55. *With probability at least* $1 - 10n^{-10}$,

$$\begin{aligned}
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{a}_t \boldsymbol{a}_t{}' - \bar{\boldsymbol{\Lambda}} \right\| \le C\sqrt{\eta} f \sqrt{\frac{r \log n}{\alpha}} \lambda^-_{\text{avg}}, \\
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{\ell}_t \boldsymbol{w}_t{}' - \frac{1}{\alpha} \mathbb{E}\left[\sum_t \boldsymbol{\ell}_t \boldsymbol{w}_t{}' \right] \right\|_2 \le C\sqrt{\eta} q f \sqrt{\frac{r \log n}{\alpha}} \lambda^-_{\text{avg}}, \\
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{w}_t \boldsymbol{w}_t{}' - \frac{1}{\alpha} \mathbb{E}\left[\sum_t \boldsymbol{w}_t \boldsymbol{w}_t{}' \right] \right\|_2 \le C\sqrt{\eta} q^2 f \sqrt{\frac{r \log n}{\alpha}} \lambda^-_{\text{avg}}, \\
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{\ell}_t \boldsymbol{v}_t{}' \right\|_2 \le C\sqrt{\eta} \sqrt{gf} \sqrt{\frac{\max(r_v, r) \log n}{\alpha}} \lambda^-_{\text{avg}}, \\
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{w}_t \boldsymbol{v}_t{}' \right\|_2 \le C\sqrt{\eta} q \sqrt{gf} \sqrt{\frac{\max(r_v, r) \log n}{\alpha}} \lambda^-_{\text{avg}}, \\
&\left\| \frac{1}{\alpha} \sum_t \boldsymbol{v}_t \boldsymbol{v}_t{}' - \frac{1}{\alpha} \mathbb{E}\left[\sum_t \boldsymbol{v}_t \boldsymbol{v}_t{}' \right] \right\|_2 \le C\sqrt{\eta} g \sqrt{\frac{r_v \log n}{\alpha}} \lambda^-_{\text{avg}}.
\end{aligned}$$

Proof of Lemma 3.55. $\boldsymbol{a}_t \boldsymbol{a}_t{}'$ *term.* This and all other items use Matrix Bernstein for rectangular matrices, Theorem 1.6 of [31]. This says the following. For a finite sequence of $d_1 \times d_2$ zero mean independent matrices $\boldsymbol{Z}_k$ with

$$\begin{aligned}
&\|\boldsymbol{Z}_k\|_2 \le R, \text{ and} \\
&\max(\|\sum_k \mathbb{E}[\boldsymbol{Z}_k{}' \boldsymbol{Z}_k]\|_2, \|\sum_k \mathbb{E}[\boldsymbol{Z}_k \boldsymbol{Z}_k{}']\|_2) \le \sigma^2,
\end{aligned}$$

we have $\Pr(\|\sum_k \boldsymbol{Z}_k\|_2 \ge s) \le (d_1 + d_2) \exp\left(-\frac{s^2/2}{\sigma^2 + Rs/3}\right) \le (d_1 + d_2) \exp\left(-c \min\left(\frac{s^2}{2\sigma^2}, \frac{3s}{2R}\right)\right)$. Let $\tilde{\boldsymbol{Z}}_t := \boldsymbol{a}_t \boldsymbol{a}_t{}'$ and we apply the above result to $\boldsymbol{Z}_t = \tilde{\boldsymbol{Z}}_t - \mathbb{E}[\tilde{\boldsymbol{Z}}_t]$. with $s = \epsilon\alpha$. Now it is easy to see that $\|\boldsymbol{Z}_t\| \le 2\|\boldsymbol{a}_t \boldsymbol{a}_t{}'\| \le 2\|\boldsymbol{a}_t\|_2^2 \le 2\eta r \lambda^+_{\max} := R$ and similarly, $\|\mathbb{E}[\boldsymbol{Z}_t^2]\| = \|\mathbb{E}[\|\boldsymbol{a}_t\|_2^2 \boldsymbol{a}_t \boldsymbol{a}_t{}']\| \le \alpha \cdot \max_{\boldsymbol{a}_t} \|\boldsymbol{a}_t\|_2^2 \cdot \max_t \mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t{}'] \le \alpha\eta r (\lambda^+_{\max})^2 := \sigma^2$ and thus, w.p. at most $2r \exp\left(-c \min\left(\frac{\epsilon^2 \alpha}{r(\lambda^+_{\max})^2}, \frac{\epsilon^2 d}{r\lambda^+_{\max}\epsilon}\right)\right)$. Now we set $\epsilon = \epsilon_5 \lambda^-_{\min}$ with $\epsilon_5 = C\sqrt{\eta} f \sqrt{\frac{r \log n}{\alpha}}$ to get our result.

$\boldsymbol{\ell}_t \boldsymbol{w}_t'$ *term.* Let $\boldsymbol{Z}_t := \boldsymbol{\ell}_t \boldsymbol{w}_t{}'$. We apply this result to $\tilde{\boldsymbol{Z}}_t := \boldsymbol{Z}_t - \mathbb{E}[\boldsymbol{Z}_t]$ with $s = \epsilon\alpha$. To get the values of R and σ^2 in a simple fashion, we use the facts that (i) if $\|\boldsymbol{Z}_t\|_2 \le R_1$, then $\|\tilde{\boldsymbol{Z}}_t\| \le 2R_1$;

and (ii) $\sum_t \mathbb{E}[\tilde{\boldsymbol{Z}}_t \tilde{\boldsymbol{Z}}_t{}'] \preccurlyeq \sum_t \mathbb{E}[\boldsymbol{Z}_t \boldsymbol{Z}_t{}']$. Thus, we can set R to two times the bound on $\|\boldsymbol{Z}_t\|_2$ and we can set σ^2 as the maximum of the bounds on $\|\sum_t \mathbb{E}[\boldsymbol{Z}_t \boldsymbol{Z}_t{}']\|_2$ and $\|\sum_t \mathbb{E}[\boldsymbol{Z}_t{}' \boldsymbol{Z}_t]\|_2$.

It is easy to see that $R = 2\sqrt{\eta r \lambda^+_{\max}}\sqrt{\eta r q^2 \lambda^+_{\max}} = 2\eta r q \lambda^+_{\max}$. To get σ^2, observe that

$$\begin{aligned}\left\|\sum_t \mathbb{E}[\boldsymbol{w}_t \boldsymbol{\ell}_t{}' \boldsymbol{\ell}_t \boldsymbol{w}_t{}']\right\|_2 &\leq \alpha(\max_{\boldsymbol{\ell}_t} \|\boldsymbol{\ell}_t\|^2) \cdot \max_t \|\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t{}']\| \\ &\leq \alpha\eta r \lambda^+_{\max} \cdot q^2 \lambda^+ = \alpha \eta r q^2 (\lambda^+_{\max})^2.\end{aligned}$$

Repeating the above steps, we get the same bound on $\|\sum_t \mathbb{E}[\boldsymbol{Z}_t \boldsymbol{Z}_t{}']\|_2$. Thus, $\sigma^2 = \alpha\eta r q^2 (\lambda^+_{\max})^2$.

Thus, we conclude that, $\|\sum_t \boldsymbol{\ell}_t \boldsymbol{w}_t{}' - \mathbb{E}[\sum_t \boldsymbol{\ell}_t \boldsymbol{w}_t{}']\|_2 \geq \epsilon\alpha$ w.p. at most $2n \exp\left(-c \min\left(\frac{\epsilon^2 \alpha}{\eta r q^2 (\lambda^+_{\max})^2}, \frac{\epsilon\alpha}{\eta r q \lambda^+_{\max}}\right)\right)$ Set $\epsilon = \epsilon_0 \lambda^-$ with $\epsilon_0 = cqf\sqrt{\frac{r \log n}{\alpha}}$ so that our bound holds w.p. at most $2n^{-10}$. This follows because $\alpha \geq C f^2 r \log n$.

$\boldsymbol{w}_t \boldsymbol{w}_t{}'$, $\boldsymbol{\ell}_t \boldsymbol{v}_t'$, $\boldsymbol{w}_t \boldsymbol{v}_t'$ and $\boldsymbol{v}_t \boldsymbol{v}_t'$ terms. Apply matrix Bernstein as done above. □

3.11 Appendix B: Proof of Theorem 3.39 and Corollary 3.40

Proof of Theorem 3.39. The overall structure of this proof is similar to that in [19, 39]. Define

$$\hat{t}_{j-1,fin} := \hat{t}_{j-1} + K\alpha,\ t_{j,*} = \hat{t}_{j-1,fin} + \left\lceil \frac{t_j - \hat{t}_{j-1,fin}}{\alpha} \right\rceil \alpha$$

Thus, $\hat{t}_{j-1,fin}$ is the time at which the $(j-1)$-th subspace update is complete; w.h.p., this occurs before t_j. With this assumption, $t_{j,*}$ is such that t_j lies in the interval $[t_{j,*} - \alpha + 1, t_{j,*}]$.

Recall from the algorithm that we increment j to $j+1$ at $t = \hat{t}_j + K\alpha := \hat{t}_{j,fin}$. Define the events

1. $\text{Det0} := \{\hat{t}_j = t_{j,*}\} = \{\lambda_{\max}(\frac{1}{\alpha}\sum_{t=t_{j,*}-\alpha+1}^{t_{j,*}} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t \hat{\boldsymbol{\ell}}_t' \boldsymbol{\Phi}) > \omega_{evals}\}$ and
   $\text{Det1} := \{\hat{t}_j = t_{j,*} + \alpha\} = \{\lambda_{\max}(\frac{1}{\alpha}\sum_{t=t_{j,*}+1}^{t_{j,*}+\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t \hat{\boldsymbol{\ell}}_t' \boldsymbol{\Phi}) > \omega_{evals}\}$,
2. $\text{SubUpd} := \cap_{k=1}^K \text{SubUpd}_k$ where $\text{SubUpd}_k := \{\text{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) \leq q_k\}$,
3. $\text{NoFalseDets} := \{\text{for all } \mathcal{J}^\alpha \subseteq [\hat{t}_{j,fin}, t_{j+1}),\ \lambda_{\max}(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t \hat{\boldsymbol{\ell}}_t' \boldsymbol{\Phi}) \leq \omega_{evals}\}$
4. $\Gamma_{0,\text{end}} := \{\text{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \leq 0.25\}$,
5. $\Gamma_{j,\text{end}} := \Gamma_{j-1,\text{end}} \cap \big((\text{Det0} \cap \text{SubUpd} \cap \text{NoFalseDets}) \cup (\overline{\text{Det0}} \cap \text{Det1} \cap \text{SubUpd} \cap \text{NoFalseDets})\big)$.

Let p_0 denote the probability that, conditioned on $\Gamma_{j-1,\text{end}}$, the change got detected at $t = t_{j,*}$, i.e., let

$$p_0 := \Pr(\text{Det0}|\Gamma_{j-1,\text{end}}).$$

Thus, $\Pr(\overline{\text{Det0}}|\Gamma_{j-1,\text{end}}) = 1 - p_0$. It is not easy to bound p_0. However, as we will see, this will not be needed. Assume that $\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}}$ holds. Consider the interval $\mathcal{J}^\alpha := [t_{j,*}, t_{j,*} + \alpha)$. This interval starts at or after t_j, so, for all t in this interval, the subspace has changed. For this interval, $\boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}'$. Applying the first item of Lemma 3.48, w.p. at least $1 - 10n^{-10}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right) \geq \omega_{evals}$$

and thus $\hat{t}_j = t_{j,*} + \alpha$. In other words,

$$\Pr(\text{Det1}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}}) \geq 1 - 10n^{-10}.$$

Conditioned on $\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}$, the first SVD step is done at $t = \hat{t}_j + \alpha = t_{j,*} + 2\alpha$ and the subsequent steps are done every α samples. We can prove Lemma 3.44 with $\Gamma_{j,0}$ replaced by $\Gamma_{j,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}$ and Lemma 3.45 with $\Gamma_{j,k-1}$ replaced by $\Gamma_{j,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1} \cap \text{SubUpd}_1 \cap \cdots \cap \text{SubUpd}_{k-1}$ and with the k-th SVD interval being $\mathcal{J}_k := [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$. Applying Lemmas 3.44, and 3.45 for each k, we get

$$\Pr(\text{SubUpd}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}) \geq (1 - 10n^{-10})^{K+1}.$$

We can also do a similar thing for the case when the change is detected at $t_{j,*}$, i.e. when Det0 holds. In this case, we replace $\Gamma_{j,0}$ by $\Gamma_{j,\text{end}} \cap \text{Det0}$ and $\Gamma_{j,k}$ by $\Gamma_{j,\text{end}} \cap \text{Det0} \cap \text{SubUpd}_1 \cap \cdots \cap \text{SubUpd}_{k-1}$ and conclude that

$$\Pr(\text{SubUpd}|\Gamma_{j-1,\text{end}} \cap \text{Det0}) \geq (1 - 10n^{-10})^{K}.$$

Finally consider the NoFalseDets event. First, assume that $\Gamma_{j-1,\text{end}} \cap \text{Det0} \cap \text{SubUpd}$ holds. Consider any interval $\mathcal{J}^\alpha \subseteq [\hat{t}_{j,fin}, t_{j+1})$. In this interval, $\hat{\boldsymbol{P}}_{(t)} = \hat{\boldsymbol{P}}_j$, $\boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_j\hat{\boldsymbol{P}}_j'$ and $\text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \varepsilon$. Using the second part of Lemma 3.48 we conclude that w.p. at least $1 - 10n^{-10}$,

$$\lambda_{\max}\left(\frac{1}{\alpha}\sum_{t\in\mathcal{J}^\alpha} \boldsymbol{\Phi}\hat{\boldsymbol{\ell}}_t\hat{\boldsymbol{\ell}}_t'\boldsymbol{\Phi}\right) < \omega_{evals}$$

Since Det0 holds, $\hat{t}_j = t_{j,*}$. Thus, we have a total of $\lfloor \frac{t_{j+1}-t_{j,*}-K\alpha-\alpha}{\alpha} \rfloor$ intervals $\mathcal{J}^\alpha$ that are subsets of $[\hat{t}_{j,fin}, t_{j+1})$. Moreover, $\lfloor \frac{t_{j+1}-t_{j,*}-K\alpha-\alpha}{\alpha} \rfloor \leq \lfloor \frac{t_{j+1}-t_j-K\alpha-\alpha}{\alpha} \rfloor \leq \lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+1)$ since $\alpha \leq \alpha$. Thus,

$$\begin{aligned}&\Pr(\text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \text{Det0} \cap \text{SubUpd}) \\ &\qquad \geq (1-10n^{-10})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K)}\end{aligned}$$

On the other hand, if we condition on $\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1} \cap \text{SubUpd}$, then $\hat{t}_j = t_{j,*} + \alpha$. Thus,

$$\begin{aligned}&\Pr(\text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1} \cap \text{SubUpd}) \\ &\qquad \geq (1-10n^{-10})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+1)}\end{aligned}$$

We can now combine the above facts to bound $\Pr(\Gamma_{j,\text{end}}|\Gamma_{j-1,\text{end}})$. Recall that $p_0 := \Pr(\text{Det0}|\Gamma_{j-1,\text{end}})$. Clearly, the events (Det0$\cap$SubUpd$\cap$NoFalseDets) and ($\overline{\text{Det0}}\cap$Det1$\cap$SubUpd$\cap$ NoFalseDets) are disjoint. Thus,

$$\begin{aligned}&\Pr(\Gamma_{j,\text{end}}|\Gamma_{j-1,\text{end}}) \\ &= p_0 \Pr(\text{SubUpd} \cap \text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \text{Det0}) \\ &+ (1-p_0) \Pr(\text{Det1}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}}) \cdot \\ &\Pr(\text{SubUpd} \cap \text{NoFalseDets}|\Gamma_{j-1,\text{end}} \cap \overline{\text{Det0}} \cap \text{Det1}) \\ &\geq p_0(1-10n^{-10})^K (1-10n^{-10})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K)} \\ &+ (1-p_0)(1-10n^{-10}) \cdot \\ &(1-10n^{-10})^K (1-10n^{-10})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor - (K+1)} \\ &= (1-10n^{-10})^{\lfloor \frac{t_{j+1}-t_j}{\alpha} \rfloor} \geq (1-10n^{-10})^{t_{j+1}-t_j}.\end{aligned}$$

Thus, since the events $\Gamma_{j,\text{end}}$ are nested, $\Pr(\Gamma_{J,\text{end}}|\Gamma_{0,\text{end}}) = \prod_j \Pr(\Gamma_{j,\text{end}}|\Gamma_{j-1,\text{end}}) \geq \prod_j (1-10n^{-10})^{t_{j+1}-t_j} = (1-10n^{-10})^d \geq 1-10dn^{-10}$. □

Proof of Corollary 3.40. It should be noted that $basis(\boldsymbol{M})$ is not a unique matrix, it refers to any matrix $\boldsymbol{P}$ that has orthonormal columns and whose span equals the span of $\boldsymbol{M}$. Thus

$basis([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_j]) \equiv basis([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_{j-1,\perp}\hat{\boldsymbol{P}}_j]) \equiv basis([\hat{\boldsymbol{P}}_j, \hat{\boldsymbol{P}}_{j,\perp}\hat{\boldsymbol{P}}_{j-1}])$. Let us denote any of these matrices by $\hat{\boldsymbol{P}}_{j-1,j}$.

For $t \in [\hat{t}_{j-1} + K\alpha, t_j)$, $\boldsymbol{P}_t = \boldsymbol{P}_{j-1}$ while for $t \in [t_j, \hat{t}_j + K\alpha - 1)$, $\boldsymbol{P}_t = \boldsymbol{P}_j$. For all t in these two intervals $\hat{\boldsymbol{P}}_{(t)} = \hat{\boldsymbol{P}}_{j-1,j}$. The proof of this corollary is an easy consequence of this fact and the fact that, for two basis matrices $\boldsymbol{P}_1, \boldsymbol{P}_2$ that are mutually orthonormal, i.e., for which $\boldsymbol{P}_1{}'\boldsymbol{P}_2 = 0$,

$$(\boldsymbol{I} - \boldsymbol{P}_1\boldsymbol{P}_1' - \boldsymbol{P}_2\boldsymbol{P}_2') = (\boldsymbol{I} - \boldsymbol{P}_1\boldsymbol{P}_1')(\boldsymbol{I} - \boldsymbol{P}_2\boldsymbol{P}_2').$$

Thus, $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1,j}, \boldsymbol{P}_{j-1}) \leq \mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \varepsilon$ and $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1,j}, \boldsymbol{P}_j) \leq \mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \varepsilon$. □

3.12 Appendix C: Proofs for Section 3.3: Time complexity derivation and Proof of Theorem 3.42

3.12.1 Time complexity derivation

Consider initialization. To ensure that $\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \in O(1/\sqrt{r})$, we need to use $C \log r$ iterations of AltProj. Since there is no lower bound in the AltProj guarantee on the required number of matrix columns (except the trivial lower bound of rank) [25], we can use $t_{\text{train}} = Cr$ frames for initialization. Thus the initialization complexity is $O(nt_{\text{train}}r^2 \log(\sqrt{r}) = O(nr^3 \log r)$ [25]. The projected-CS step complexity is equal to the cost of a matrix vector multiplication with the measurement matrix times negative logarithm of the desired accuracy in solving the l_1 minimization problem. Since the measurement matrix for the CS step is $\boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$, the cost per CS step (per frame) is $O(nr \log(1/\epsilon))$ [36] and so the total cost is $O((d - t_{\text{train}})nr \log(1/\epsilon))$. The subspace update involves at most $((d - t_{\text{train}})/\alpha)$ rank r-SVD's on $n \times \alpha$ matrices all of which have constant eigen-gap (this is indirectly proved in the proofs of the second item of Lemmas 3.44 and 3.45). Thus the total time for subspace update steps is at most $((d - t_{\text{train}})/\alpha) * O(n\alpha r \log(1/\epsilon)) = O((d - t_{\text{train}})nr \log(1/\epsilon))$ [20]. Thus the running time of the complete algorithm is $O(ndr \log(1/\epsilon) + nr^3 \log r)$. As long as $r^2 \log r \leq d \log(1/\epsilon)$, the time complexity of the entire algorithm is $O(ndr \log(1/\epsilon))$.

3.12.2 Proof of Theorem 3.42 for NORST-NoDet

In this algorithm we do not detect change. We just keep updating the subspace by r-SVD applied every α time instants on the last α $\hat{\boldsymbol{\ell}}_t$'s, $\hat{\boldsymbol{L}}_{t;\alpha}$. For α-intervals $\mathcal{J}$ for which $\boldsymbol{P}_t = \boldsymbol{P}_j$ for all $t \in \mathcal{J}$, there is no change to the analysis. We start at $t = t_0 = \hat{t}_0 = 1$ with initial subspace estimate $\hat{\boldsymbol{P}}_0$ available. Let $\Delta_0 = \mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0)$. The first subspace update is done at $t = \alpha$, the second at $t = 2\alpha$, and so on. By Lemma 3.44 with $\hat{\boldsymbol{P}}_{j,0} = \hat{\boldsymbol{P}}_0$, we can show that after one update, the error reduces to $1.2\max(\Delta_0/4, \varepsilon)$. After this, by applying Lemma 3.45 $K-1$ times, we can show that, after at most K steps with $K = \log(\Delta_0/\varepsilon)$, the error reduces to 1.2ε. Beyond this time, the error does not decrease further. We know that $\boldsymbol{P}_t = \boldsymbol{P}_0$ for $t \in [t_0, (K+2)\alpha]$, but can change after that.

Consider the α-interval $\mathcal{J}$ that contains the change time t_1. The projected CS analysis for this interval remains exactly the same as above. But to analyze the subspace update for this interval we need to use Corollary 3.43. More generally consider the j-th change, and the interval $\mathcal{J} = [\lfloor t_j/\alpha \rfloor + 1, \lfloor t_j/\alpha \rfloor + \alpha]$, which is the α-frame interval that contains t_j.

For $t \in \mathcal{J}$, we have $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t = \boldsymbol{\ell}_t + \boldsymbol{e}_t + \boldsymbol{v}_t$ where

$$\boldsymbol{e}_t = \boldsymbol{I}_{\mathcal{T}_t} \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1} \boldsymbol{I}_{\mathcal{T}_t}{}' \boldsymbol{\Psi} (\boldsymbol{\ell}_t + \boldsymbol{v}_t) := (\boldsymbol{e}_{\boldsymbol{\ell}})_t + (\boldsymbol{e}_{\boldsymbol{v}})_t$$

$\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1} \hat{\boldsymbol{P}}_{j-1}{}'$, $\boldsymbol{\ell}_t = \boldsymbol{P}_{j-1} \boldsymbol{a}_t$ for $t \in [\lfloor t_j/\alpha \rfloor, t_j)$ and $\boldsymbol{\ell}_t = \boldsymbol{P}_j \boldsymbol{a}_t$ for $t \in [t_j, \lfloor t_j/\alpha \rfloor + \alpha)$.

Let $\hat{\boldsymbol{P}}_{j,0}$ denote the subspace estimate $\hat{\boldsymbol{P}}_{(t)}$ computed for this interval. We apply Corollary 3.43 with $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv (\boldsymbol{e}_{\boldsymbol{\ell}})_t$, $\boldsymbol{v}_t \equiv (\boldsymbol{e}_{\boldsymbol{v}})_t + \boldsymbol{v}_t$, $\boldsymbol{\ell}_t \equiv \boldsymbol{\ell}_t$, $\boldsymbol{M}_{1,t} = -\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}{}'$, $\hat{\boldsymbol{P}} = \hat{\boldsymbol{P}}_{j,0}$, $\boldsymbol{P} = \boldsymbol{P}_j$, $\boldsymbol{P}_0 = \boldsymbol{P}_{j-1}$. Since $\|\boldsymbol{M}_{1,t} \boldsymbol{P}_0\| = \|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{P}_j\| \le 1.2\varepsilon$, $\|\boldsymbol{M}_{1,t} \boldsymbol{P}\| = \|\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{P}_j\| \le 1.2(\varepsilon + \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j))$, thus $q_{00} = 1.2(\varepsilon + \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j))$. Also, $b \equiv b_0 = 0.01/f^2$ which is the upper bound on max-outlier-frac-row(α), $\|\mathbb{E}[(\boldsymbol{e}_{\boldsymbol{v}})_t (\boldsymbol{e}_{\boldsymbol{v}})_t{}']\| \le (1.2)^2 \lambda_v^+$. Thus, with probability at least $1 - 10n^{-10}$,

$$\mathrm{SE}(\hat{\boldsymbol{P}}_{j,0}, \boldsymbol{P}_j) \le 2.5(3(\Delta f + 4 \cdot 0.1 \cdot 1.2(\varepsilon + \Delta) + \frac{\lambda_v^+}{\lambda^-}) \le 10\Delta$$

Here we used $\frac{\lambda_v^+}{\lambda^-} = \varepsilon^2 < \Delta$.

Redefine $\hat{t}_j = \lfloor t_j/\alpha \rfloor + \alpha$ and $\hat{\boldsymbol{P}}_{j,0}$ to denote the estimate from the change interval. To analyze the next α-interval for new-NORST, we apply Lemma 3.44 with above re-definitions. Thus, $q_0 =$

$1.2 \cdot 10\Delta$. We can conclude that $\mathrm{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \leq \max(0.3q_0, \varepsilon) = q_1$. For the next $K-1$ intervals, we apply Lemma 3.45 $K-1$ times with $q_k = 1.2\max(0.25q_{k-1}, \varepsilon)$.

Algorithm 6 NORST Algorithm. We obtain $\hat{\boldsymbol{P}}_0$ by $C(\log r)$ iterations of AltProj on $\boldsymbol{Y}_{[1,t_{\text{train}}]}$, $t_{\text{train}} = Cr$.

Input: $\hat{\boldsymbol{P}}_0$, $\boldsymbol{y}_t$; **Output**: $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}_{(t)}$; **Parameters:** ω_{supp}, ξ, α, K, ω_{evals}
$\hat{\boldsymbol{P}}_{(t_{\text{train}})} \leftarrow \hat{\boldsymbol{P}}_0$; $j \leftarrow 1$, $k \leftarrow 1$
phase $\leftarrow$ update; $\hat{t}_0 \leftarrow t_{\text{train}}$;
for $t > t_{\text{train}}$ **do**
  $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$
  $\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$.
  $\hat{\boldsymbol{x}}_{t,cs} \leftarrow \arg\min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}}\|_1$ s.t. $\|\tilde{\boldsymbol{y}}_t - \boldsymbol{\Psi}\tilde{\boldsymbol{x}}\| \le \xi$.
  $\hat{\mathcal{T}}_t \leftarrow \{i : |\hat{\boldsymbol{x}}_{t,cs}| > \omega_{supp}\}$.
  $\hat{\boldsymbol{x}}_t \leftarrow \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\tilde{\boldsymbol{y}}_t$.
  $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$
  if phase = detect and $t = \hat{t}_{j-1,fin} + u\alpha$ **then**
    $\boldsymbol{\Phi} \leftarrow (\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}')$.
    $\boldsymbol{B} \leftarrow \boldsymbol{\Phi}\hat{\boldsymbol{L}}_{t,\alpha}$ with $\hat{\boldsymbol{L}}_{t,\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \dots \hat{\boldsymbol{\ell}}_t]$.
    if $\lambda_{\max}(\boldsymbol{B}\boldsymbol{B}') \ge \alpha\omega_{evals}$ **then**
      phase $\leftarrow$ update, $\hat{t}_j \leftarrow t$,
    end if
  end if
  if phase = update **then**
    if $t = \hat{t}_j + u\alpha - 1$ for $u = 1,\ 2,\ \cdots$, **then**
      $\hat{\boldsymbol{P}}_{j,k} \leftarrow SVD_r[\hat{\boldsymbol{L}}_{t;\alpha}]$, $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{j,k}$, $k \leftarrow k+1$.
    else
      $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$
    end if
    if $t = \hat{t}_j + K\alpha - 1$ **then**
      $\hat{t}_{j,fin} \leftarrow t$, $\hat{\boldsymbol{P}}_j \leftarrow \hat{\boldsymbol{P}}_{(t)}$
      $k \leftarrow 1$, $j \leftarrow j+1$, phase $\leftarrow$ detect.
    end if
  end if
end for
Smoothing NORST: At $t = \hat{t}_j + K\alpha$, for all $t \in [\hat{t}_{j-1} + K\alpha, \hat{t}_j + K\alpha - 1]$,
$\hat{\boldsymbol{P}}_{(t)}^{\text{smoothing}} \leftarrow basis([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_j])$, where $basis(\boldsymbol{M})$ refers to a basis matrix that has span equal to span$(\boldsymbol{M})$.
$\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t)}^{\text{smoothing}}\hat{\boldsymbol{P}}_{(t)}^{\text{smoothing}\prime}$; $\hat{\boldsymbol{x}}_t^{\text{smoothing}} \leftarrow \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{y}_t$; $\hat{\boldsymbol{\ell}}_t^{\text{smoothing}} \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t^{\text{smoothing}}$.

Algorithm 7 NORST-NoDet

Input: $\hat{\boldsymbol{P}}_0$, $\boldsymbol{y}_t$; **Output**: $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}_{(t)}$; **Parameters:** ω_{supp}, ξ, α
$\hat{\boldsymbol{P}}_{(t_{\text{train}})} \leftarrow \hat{\boldsymbol{P}}_0$;
for $t > t_{\text{train}}$ **do**
    Lines 6-11 of Algorithm 6
    if $t = t_{\text{train}} + u\alpha - 1$ for $u = 1,\ 2,\ \cdots$, **then**
        $\hat{\boldsymbol{P}}_u \leftarrow SVD_r[\hat{\boldsymbol{L}}_{t;\alpha}]$, $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_u$
    else
        $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$
    end if
end for

CHAPTER 4. SUBSPACE TRACKING FROM INCOMPLETE DATA IN THE PRESENCE OF OUTLIERS

Praneeth Narayanamurthy, Vahid Daneshpajooh, and Namrata Vaswani

Dept. of Electrical and Computer Engineering, Iowa State University, Ames, IA, 50010

Modified from a manuscript published in *IEEE Transactions on Signal Processing*[1]

Abstract

We study the problem of subspace tracking in the presence of missing data (ST-miss). In recent work, we studied a related problem called robust ST. In this work, we show that a simple modification of our robust ST solution also provably solves ST-miss and robust ST-miss. To our knowledge, our result is the first "complete" guarantee for ST-miss. This means that we can prove that under assumptions on only the algorithm inputs, the output subspace estimates are close to the true data subspaces at all times. Our guarantees hold under mild and easily interpretable assumptions, and allow the underlying subspace to change with time in a piecewise constant fashion. In contrast, all existing guarantees for ST are partial results and assume a fixed unknown subspace. Extensive numerical experiments are shown to back up our theoretical claims. Finally, our solution can be interpreted as a provably correct mini-batch and memory-efficient solution to low rank Matrix Completion (MC).

4.1 Introduction

Subspace tracking from missing data (ST-miss) is the problem of tracking the (fixed or time-varying) low-dimensional subspace in which a given data sequence approximately lies when some of the data entries are not observed. The assumption here is that consecutive subsets of the data are

[1] I performed the literature survey, writing, and designing the experiments, V.D. performed the experiments, N.V. helped with all components.

well-approximated as lying in a subspace that is significantly lower-dimensional than the ambient dimension. Time-varying subspaces is a more appropriate model for long data sequences (e.g. long surveillance videos). For such data, if a fixed subspace model is used, the required subspace dimension may be too large. As is common in time-series analysis, the simplest model for time-varying quantities is to assume that they are piecewise constant with time. We adopt this model here. If the goal is to provably track the subspaces to any desired accuracy, $\varepsilon > 0$, then, as we explain later in Sec. 4.1.3, this assumption is, in fact, necessary. Of course, experimentally, our proposed algorithm, and all existing ones, "work" (return good but not perfect estimates) even without this assumption, as long as the amount of change at each time is small enough. The reason is one can interpret subspace changes at each time as a "piecewise constant subspace" plus noise. The algorithms are actually tracking the "piecewise constant subspace" up to the noise level. We explain this point further in Sec. 4.1.3.

ST-miss can be interpreted as an easier special case of robust ST (ST in the presence of additive sparse outliers) [33]. We also study robust ST-miss which is a generalization of both ST-miss and robust ST. Finally, our solutions for ST-miss and robust ST-miss also provide novel mini-batch solutions for low-rank matrix completion (MC) and robust MC respectively.

Example applications where these problems occur include recommendation system design and video analytics. In video analytics, foreground occlusions are often the source of both missing and corrupted data: if the occlusion is easy to detect by simple means, e.g., color-based thresholding, then the occluding pixel can be labeled as "missing"; while if this cannot be detected easily, it is labeled as an outlier pixel. Missing data also occurs due to detectable video transmission errors (typically called "erasures"). In recommendation systems, data is missing because all users do not label all items. In this setting, time-varying subspaces model the fact that, as different types of users enter the system, the factors governing user preferences change.

Brief review of related work. ST has been extensively studied in both the controls' and the signal processing literature, see [14, 1, 19, 46] for comprehensive overviews of both classical and modern approaches. Best known existing algorithms for ST and ST-miss include Projection Ap-

proximate Subspace Tracking (PAST) [48, 49], Parallel Estimation and Tracking by Recursive Least Squares (PETRELS) [12] and Grassmannian Rank-One Update Subspace Estimation (GROUSE) [3, 4, 53, 38]. Of these, PETRELS is known to have the best experimental performance. There have been some attempts to obtain guarantees for GROUSE and PETRELS for ST-miss [4, 53, 47], however all of these results assume the statistically stationary setting of a *fixed unknown subspace* and all of them provide only *partial guarantees*. This means that the result does not tell us what assumptions the algorithm inputs (input data and/or initialization) need to satisfy in order to ensure that the algorithm output(s) are close to the true value(s) of the quantity of interest, either at all times or at least at certain times. For example, [4] requires that the intermediate algorithm estimates of GROUSE need to satisfy certain properties (see Theorem 4.64 given later). It does not tell us what assumptions on algorithm inputs will ensure that these properties hold. On the other hand, [47] guarantees closeness of the PETRELS output to a quantity other than the true value of the "quantity of interest" (here, the true data subspace); see Theorem 4.65. Of course, the advantage of GROUSE and PETRELS is that they are streaming solutions (require a single-pass through the data). This may also be the reason that a complete guarantee is harder to obtain for these. Other related work includes streaming PCA with missing data [32, 20]. A provable algorithmic framework for robust ST is Recursive Projected Compressive Sensing (ReProCS) [40, 41, 51, 34, 33]. Robust ST-miss has not received much attention in the literature.

Provable MC has been extensively studied, e.g., [8, 35, 11]. We discuss these works in detail in Sec. 4.3.

Contributions. (1) We show that a simple modification of a ReProCS-based algorithm called Nearly Optimal Robust ST via ReProCS (NORST for short) [33] also provably solves the ST-miss problem while being fast and memory-efficient. An extension for robust ST-miss is also presented. Unlike all previous work on ST-miss, our guarantee is a *complete guarantee (correctness result)*: we show that, with high probability (whp), under simple assumptions on only the algorithm inputs, the output subspace estimates are close to the true data subspaces and get to within ε accuracy of the current subspace within a "near-optimal" delay. Moreover, unlike past work, our result

allows time-varying subspaces (modeled as piecewise-constant with time) and shows that NORST-miss can provably detect and track each changed subspace quickly. Here and below, *near-optimal* means that our bound is within logarithmic factors of the minimum required. For r-dimensional subspace tracking, the minimum required delay is r; thus our delay of order $r \log n \log(1/\varepsilon)$ is *near-optimal*. Moreover, since ST-miss is an easier problem than robust ST, our guarantee for ST-miss is significantly better than the original one [33] that it follows from. It does not assume a good first subspace initialization and does not require slow subspace change.

(2) Our algorithm and result can also be interpreted as a novel provably correct mini-batch and memory-efficient solution to low rank MC. We explain in Sec. 4.2.2 that our guarantee is particularly interesting in the regime when subspace changes frequently enough, e.g., if it changes every order $r \log n \log(1/\varepsilon)$ time instants.

Organization. We explain the algorithm and provide the guarantees for it in Sec. 4.2; first for the noise-free case and then for the noisy case. A detailed discussion is also given that explains why our result is an interesting solution for MC. In this section, we also develop simple heuristics that improve the experimental performance of NORST-miss. We provide a detailed discussion of existing guarantees and how our work relates to the existing body of work in Sec. 4.3. Robust ST-miss is discussed in Sec. 4.4. Exhaustive experimental comparisons for simulated and partly real data (videos with simulated missing entries) are provided in Sec. 4.5. These show that as long as the fraction of missing entries is not too large, (i) basic NORST-miss is nearly as good as the best existing ST-miss approach (PETRELS), while being faster and having a *complete guarantee*; (ii) its extensions have better performance than PETRELS and are also faster than PETRELS; (iii) the performance of NORST-miss is worse than convex MC solutions, but much better than non-convex ones (for which code is available); however, NORST-miss is much faster than the convex MC methods. We conclude in Sec. 4.6.

4.1.1 Notation

We use the interval notation $[a, b]$ to refer to all integers between a and b, inclusive, and we use $[a, b) := [a, b-1]$. $\|.\|$ denotes the l_2 norm for vectors and induced l_2 norm for matrices unless specified otherwise, and $'$ denotes transpose. We use $\boldsymbol{M}_{\mathcal{T}}$ to denote a sub-matrix of $\boldsymbol{M}$ formed by its columns indexed by entries in the set $\mathcal{T}$. For a matrix $\boldsymbol{P}$ we use $\boldsymbol{P}^{(i)}$ to denote its i-th row.

A matrix $\boldsymbol{P}$ with mutually orthonormal columns is referred to as a *basis matrix* and is used to represent the subspace spanned by its columns. For basis matrices $\boldsymbol{P}_1, \boldsymbol{P}_2$, we use $\mathrm{SE}(\boldsymbol{P}_1, \boldsymbol{P}_2) := \|(\boldsymbol{I} - \boldsymbol{P}_1\boldsymbol{P}_1{}')\boldsymbol{P}_2\|$ as a measure of Subspace Error (distance) between their respective subspaces. This is equal to the sine of the largest principal angle between the subspaces. If $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ are of the same dimension, $\mathrm{SE}(\boldsymbol{P}_1, \boldsymbol{P}_2) = \mathrm{SE}(\boldsymbol{P}_2, \boldsymbol{P}_1)$.

We use $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \cdots, \hat{\boldsymbol{\ell}}_t]$ to denote the matrix formed by $\hat{\boldsymbol{\ell}}_t$ and $(\alpha-1)$ previous estimates. Also, r-SVD$[\boldsymbol{M}]$ refers to the matrix of top r left singular vectors of $\boldsymbol{M}$.

A set Ω that is randomly sampled from a larger set (universe), $\mathcal{U}$, is said be *"i.i.d. Bernoulli with parameter ρ"* if each entry of $\mathcal{U}$ has probability ρ of being selected to belong to Ω independent of all others.

We reuse C, c to denote different numerical constants in each use; C is for constants greater than one and c for those less than one.

Definition 4.56 (μ-incoherence)**.** *An $n \times r_P$ basis matrix $\boldsymbol{P}$ is μ-incoherent if* $\max_i \|\boldsymbol{P}^{(i)}\|_2^2 \leq \mu \frac{r_P}{n}$ *($\boldsymbol{P}^{(i)}$ is i-th row of $\boldsymbol{P}$). Clearly, $\mu \geq 1$.*

Throughout this paper, we assume that f, which is the condition number of the population covariance of $\boldsymbol{\ell}_t$, and the parameter, μ, are constants. This is assumed when the $\mathcal{O}(\cdot)$ notation is used.

4.1.2 Problem Statement

ST-miss is precisely defined as follows. At each time t, we observe a data vector $\boldsymbol{y}_t \in \mathbb{R}^n$ that satisfies

$$\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{\ell}_t) + \boldsymbol{v}_t, \text{ for } t = 1, 2, \ldots, d \tag{4.1}$$

where $\mathcal{P}_{\Omega_t}(\boldsymbol{z}_i) = \boldsymbol{z}_i$ if $i \in \Omega_t$ and 0 otherwise. Here $\boldsymbol{v}_t$ is small unstructured noise, Ω_t is the set of observed entries at time t, and $\boldsymbol{\ell}_t$ is the true data vector that lies in a fixed or changing low (r) dimensional subspace of $\mathbb{R}^n$, i.e., $\boldsymbol{\ell}_t = \boldsymbol{P}_{(t)}\boldsymbol{a}_t$ where $\boldsymbol{P}_{(t)}$ is an $n \times r$ basis matrix with $r \ll n$. The goal is to track span($\boldsymbol{P}_{(t)}$) and $\boldsymbol{\ell}_t$ either immediately or within a short delay. Denoting the set of missing entries at time t as $\mathcal{T}_t$, (4.1) can also be written as

$$\boldsymbol{y}_t := \boldsymbol{\ell}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\ell}_t + \boldsymbol{v}_t. \tag{4.2}$$

We use $\boldsymbol{z}_t := -\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\ell}_t$ to denote the missing entries. Clearly, $\mathcal{T}_t = (\Omega_t)^c$ (here ^{c} denotes the complement set w.r.t. $\{1, 2, \ldots, n\}$). Writing $\boldsymbol{y}_t$ as above allows us to tap into the solution framework from earlier work [41, 33]. This was developed originally for solving robust ST which involves tracking $\boldsymbol{\ell}_t$ and $\boldsymbol{P}_{(t)}$ from $\boldsymbol{y}_t := \boldsymbol{\ell}_t + \boldsymbol{v}_t + \boldsymbol{x}_t$ where $\boldsymbol{x}_t$ is a sparse vector with the outliers as its nonzero entries. ST-miss can be interpreted as its (simpler) special case if we let $\boldsymbol{x}_t = -\boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\ell}_t$. It is simpler because the support of $\boldsymbol{x}_t$, $\mathcal{T}_t$, is known.

Defining the $n \times d$ matrix $\boldsymbol{L} := [\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots \boldsymbol{\ell}_d]$, the above is also a matrix completion (MC) problem; with the difference that for MC the estimates are needed only in the end (not on-the-fly). We use $r_{_L}$ to denote the rank of $\boldsymbol{L}$.

4.1.3 Identifiability assumptions

The above problem definition does not ensure identifiability. If $\boldsymbol{L}$ is sparse, it is impossible to recover it from a subset of its entries. Moreover, even if it is dense, it is impossible to complete it if all the missing entries are from a few rows or columns. Finally, if the subspace changes at every time t, the number of unknowns (nr) is more than the amount of available data at time t (n) making it impossible to recover all of them.

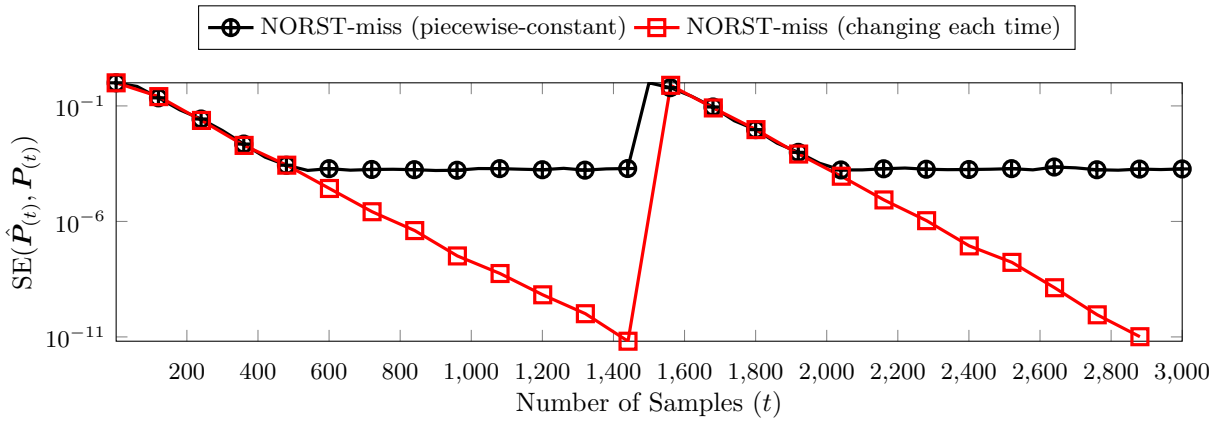


Figure 4.1: Demonstrating the need for the piecewise constant subspace change model. The black circles plot is for subspace changing at each time t, while the red squares one is for piecewise constant subspace change, with change occurring at $t = t_1$. The data is generated so that, in both experiments, $\mathrm{SE}(\boldsymbol{P}_{(t_1)}, \boldsymbol{P}_{(0)})$ is the same. In the piecewise constant case (red squares), we can achieve near perfect subspace recovery. But this is not possible in the "changing at each time" (black circles) case. For details, see Sec. 4.5 and Fig. 4.3(c).

One way to ensure subspaces' identifiability is to assume that they are piecewise constant with time, i.e., that

$$\boldsymbol{P}_{(t)} = \boldsymbol{P}_j \text{ for all } t \in [t_j, t_{j+1}),\ j = 1, 2, \ldots, J.$$

with $t_{j+1} - t_j \geq r$. Let $t_0 = 1$ and $t_{J+1} = d$. This ensures that at least r n-dimensional data vectors $\boldsymbol{y}_t$ are available (this is the minimum needed to compute the subspace even if perfect data $\boldsymbol{y}_t = \boldsymbol{\ell}_t$ were available). The t_j's are the subspace change times. With this model, $r_L \leq rJ$. When the above model is not assumed, one cannot track to any desired accuracy, see the black circles plot in Fig. 4.1. This is because the subspace change at each time can be interpreted as a r-dimensional piecewise constant subspace change plus noise. To understand this precisely, consider the first α frames, for any $\alpha \geq r$. Let $\boldsymbol{P}$ be the matrix of top r left singular vectors of $[\boldsymbol{P}_{(0)}, \boldsymbol{P}_{(1)}, \ldots, \boldsymbol{P}_{(\alpha-1)}]$. Then, in this interval, $\boldsymbol{y}_t := \mathcal{P}_{\Omega_t}(\boldsymbol{P}_{(t)}\boldsymbol{a}_t)$ can be rewritten as $\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{P}(\boldsymbol{P}'\boldsymbol{P}_{(t)}\boldsymbol{a}_t)) + \boldsymbol{v}_t$ where $\boldsymbol{v}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{P}_{(t)}\boldsymbol{a}_t - \boldsymbol{P}(\boldsymbol{P}'\boldsymbol{P}_{(t)})\boldsymbol{a}_t)$. A similar argument can be extended to any set of α frames.

As explained in earlier work on MC [15, 8, 42], one way to ensure that $\boldsymbol{L}$ is not sparse is to assume that its left and right singular vectors are dense. This is the well-known incoherence or denseness assumption. Left singular vectors incoherent is nearly equivalent to imposing μ-incoherence of the

$\boldsymbol{P}_j$'s with μ being a numerical constant. As explained in [33, Remark 2.4], the following assumption on $\boldsymbol{a}_t$'s is similar to right incoherence, and hence we call it "statistical right incoherence".

Definition 4.57 (Statistical Right Incoherence)**.** *We assume that the $\boldsymbol{a}_t$'s are zero mean, i.e., $\mathbb{E}[\boldsymbol{a}_t] = 0$; are mutually independent over time; have identical diagonal covariance matrix $\boldsymbol{\Lambda}$, i.e., that $\mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t{}'] = \boldsymbol{\Lambda}$ with $\boldsymbol{\Lambda}$ diagonal; and are element-wise bounded. Element-wise bounded means that there exists a numerical constant $\mu \geq 1$, such that $\max_i \max_t (\boldsymbol{a}_t)_i^2 \leq \mu \max_t \lambda_{\max}(\mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t{}'])$. This implies that the $\boldsymbol{a}_t$'s are sub-Gaussian with sub-Gaussian norm bounded by $\mu \max_t \lambda_{\max}(\mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t{}']) = \mu \lambda_{\max}(\boldsymbol{\Lambda})$. A simple example of element-wise bounded random vectors (r.v) is uniform r.v.s.*

Motivated by the Robust PCA literature [36], one way to ensure that the missing entries are spread out is to bound the maximum fraction of missing entries in any row and in any column. We use max-miss-frac-row and max-miss-frac-col to denote these. Since NORST-miss is a mini-batch approach that works on batches of α frames, we actually need to bound the maximum fraction of missing entries in any sub-matrix of $\boldsymbol{L}$ with α consecutive columns. We denote this by max-miss-frac-row$_\alpha$. We precisely define these below.

Definition 4.58 (max-miss-frac-col, max-miss-frac-row$_\alpha$)**.** *For a discrete time interval, $\mathcal{J}$, let*

$$\gamma(\mathcal{J}) := \max_{i=1,2,\dots,n} \frac{1}{|\mathcal{J}|} \sum_{t \in \mathcal{J}} \mathbb{1}_{\{i \in \mathcal{T}_t\}}$$

where $\mathbb{1}_S$ is the indicator function for statement S. Thus, $\sum_{t \in \mathcal{J}} \mathbb{1}_{\{i \in \mathcal{T}_t\}}$ counts the maximum number of missing entries in row i of the sub-matix $\boldsymbol{L}_{\mathcal{J}}$ of the data matrix $\boldsymbol{L} := [\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots, \boldsymbol{\ell}_d]$. So, $\gamma(\mathcal{J})$ is the maximum fraction of missing entries in any row of $\boldsymbol{L}_{\mathcal{J}}$. Let $\mathcal{J}^\alpha$ denote a time interval of duration α. Then, max-miss-frac-row$_\alpha := \max_{\mathcal{J}^\alpha \subseteq [1,d]} \gamma(\mathcal{J}^\alpha)$. Also, max-miss-frac-col $:= \max_t |\mathcal{T}_t|/n$.

4.2 The NORST-miss algorithm and guarantees

We explain the basic algorithm next. We give and discuss the guarantee for the noise-free $\boldsymbol{v}_t = 0$ case in Sec. 4.2.2. The corollary for the noisy case is given in Sec. 4.2.3. Extensions of basic NORST-miss are given in Sec. 4.2.4.

4.2.1 NORST-miss algorithm

The complete psedo-code for our algorithm is provided in Algorithm 8. After initialization, the algorithm iterates between a *projected Least Squares (LS)* step and a *Subspace Update (including Change Detect)* step. Broadly, projected LS estimates the missing entries of $\boldsymbol{\ell}_t$ at each time t. Subspace update toggles between the "update" phase and the change "detect" phase. In the update phase, it improves the estimate of the current subspace using a short mini-batch of "filled in" versions of $\boldsymbol{\ell}_t$. In the detect phase, it uses these to detect subspace change.

Initialization: The algorithm starts in the "update" phase and with zero initialization: $\hat{\boldsymbol{P}}_0 \leftarrow \boldsymbol{0}_{n\times r}$. For the first α frames, the projected LS step (explained below) simply returns $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t$. Thus, a simpler way to understand the initialization is as follows: wait until $t = \alpha$ and then compute the first estimate of $\text{span}(\boldsymbol{P}_0)$ as the r-SVD (matrix of top r left singular vectors) of $[\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_\alpha]$. This step is solving a PCA with missing data problem which, as explained in [45], can be interpreted as a problem of PCA in sparse data-dependent noise. Because we assume that the number of missing entries at any time t is small enough, and the set of missing entries changes sufficiently over time[2], we can prove that this step gives a good first estimate of the subspace.

Projected LS: Recall that NORST-miss is a modification of NORST for robust ST from [33]. In robust ST, sudden subspace changes cannot be detected because these are confused for outliers. Its projected-LS step is thus deigned using a slow (small) subspace change assumption. However, as we will explain later, for the current missing data setting, it also works in case of sudden changes. Suppose that the previous subspace estimate, $\hat{\boldsymbol{P}}_{(t-1)}$, is a "good enough" estimate of the previous subspace $\boldsymbol{P}_{(t-1)}$. Under slow subspace change, it is valid to assume that $\text{span}(\boldsymbol{P}_{(t-1)})$ is either equal to or close to $\text{span}(\boldsymbol{P}_{(t)})$. Thus, under this assumption, it is a good idea to project $\boldsymbol{y}_t$ onto the orthogonal complement of $\hat{\boldsymbol{P}}_{(t-1)}$ because this will nullify most of $\boldsymbol{\ell}_t$, i.e., the not-nullified part of $\boldsymbol{\ell}_t$, $\boldsymbol{b}_t := \boldsymbol{\Psi}\boldsymbol{\ell}_t$, will be small. Here $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$. Using this idea, we compute $\tilde{\boldsymbol{y}}_t := \boldsymbol{\Psi}\boldsymbol{y}_t = \boldsymbol{\Psi}_{\mathcal{T}_t}\boldsymbol{z}_t + \boldsymbol{b}_t + \boldsymbol{\Psi}\boldsymbol{v}_t$. Estimating $\boldsymbol{z}_t$ can be interpreted as a LS problem $\min_{\boldsymbol{z}} \|\tilde{\boldsymbol{y}}_t - \boldsymbol{\Psi}_{\mathcal{T}_t}\boldsymbol{z}\|^2$.

[2] Equivalently, we bound the maximum number of missing entries in any column and in any row of the data matrix

Solving this gives

$$\hat{\boldsymbol{z}}_t = \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\tilde{\boldsymbol{y}}_t. \tag{4.3}$$

Next, we use to this to compute $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}\hat{\boldsymbol{z}}_t$. Observe that the missing entries $\boldsymbol{z}_t$ are recoverable as long as $\boldsymbol{\Psi}_{\mathcal{T}_t}$ is well-conditioned. A necessary condition for this is $(n-r) > |\mathcal{T}_t|$. As we will see later, a sufficient condition is $|\mathcal{T}_t| < cn/r$ because this ensures that the restricted isometry constant (RIC) [6] of $\boldsymbol{\Psi}$ of level $|\mathcal{T}_t|$ is small.

In settings where $\text{span}(\boldsymbol{P}_{(t-1)})$ is *not* close to $\text{span}(\boldsymbol{P}_{(t)})$ (sudden subspace change), the above approach still works. Of course, in this case, it is not any better (or worse) than re-initialization to zero, because, in this case, $\|\boldsymbol{\Psi}\boldsymbol{\ell}_t\|$ is of the same order as $\|\boldsymbol{\ell}_t\|$. We can use the same arguments as those used for the initialization step to argue that the first subspace update works even in this case.

Subspace Update: The $\hat{\boldsymbol{\ell}}_t$'s are used for subspace update. In its simplest (and provably correct) form, this is done once every α frames by r-SVD on the matrix formed by the previous α $\hat{\boldsymbol{\ell}}_t$'s. Let $\hat{t}_j$ be the time at which the j-th subspace change is detected (let $\hat{t}_0 := 0$). For each $k = 1, 2, \ldots, K$, at $t = \hat{t}_j + k\alpha - 1$, we compute the r-SVD of $\hat{\boldsymbol{L}}_{t;\alpha}$ to get $\hat{\boldsymbol{P}}_{j,k}$ (k-th estimate of subspace $\boldsymbol{P}_j$). After K such updates, i.e., at $t = \hat{t}_j + K\alpha - 1 := \hat{t}_{j,fin}$ the update is complete and the algorithm enters the "detect" phase. Each update step is a PCA in sparse data-dependent noise problem. This allows us to use the result from [45] to show that, as long as the missing entries' set changes enough over time (max-miss-frac-row$_\alpha$ is bounded for each interval), each update step reduces the subspace recovery error to 0.3 times its previous value. Thus, by setting $K = C\log(1/\varepsilon)$, one can show that, after K updates, the subspace is recovered to ε accuracy.

Subspace change detect: To simply understand the detection strategy, assume that the previous subspace $\boldsymbol{P}_{j-1}$ has been estimated to ε accuracy by $t = \hat{t}_{j-1,fin} = \hat{t}_{j-1} + K\alpha - 1$ and denote it by $\hat{\boldsymbol{P}}_{j-1} := \hat{\boldsymbol{P}}_{j-1,K}$. Also assume that $\boldsymbol{v}_t = 0$. At every $t = \hat{t}_{j-1,fin} + u\alpha - 1$, $u = 1, 2, \ldots,$ we detect change by checking if the maximum singular value of the matrix $(\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}')\hat{\boldsymbol{L}}_{t;\alpha}$ is above a pre-set threshold, $\sqrt{\omega_{evals}\alpha}$, or not. This works because, if the subspace has not changed, this matrix will have all singular values of order $\varepsilon\sqrt{\lambda^+}$. If it has changed, its largest singular value

will be at least $\mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)\sqrt{\lambda^-}$. By picking ε small enough, one can ensure that, whp, all changes are detected.

NORST-miss-smoothing for MC: The above is the tracking/online/filtering mode of NORST-miss. It outputs an estimate of $\boldsymbol{\ell}_t$ as soon as a new measurement vector $\boldsymbol{y}_t$ arrives and an estimate of the subspace every α frames. Notice that, order-wise, α is only a little more than r which is the minimum delay needed to compute the subspace even if perfect data $\boldsymbol{y}_t = \boldsymbol{\ell}_t$ were available. Once an ε-accurate estimate of the current subspace is available, one can improve all past estimates of $\boldsymbol{\ell}_t$ to ensure that all estimates are ε-accurate. This is called the smoothing mode of operation. To be precise, this is done as given in line 25 of Algorithm 8. This allows us to get a completed matrix $\hat{\boldsymbol{L}}$ with all columns being ε-accurate.

Memory Complexity: In online or filtering mode, NORST-miss needs $\alpha = O(r \log n)$ frames of storage. In smoothing mode, it needs $\mathcal{O}((K+2)\alpha) = \mathcal{O}(r \log n \log(1/\epsilon))$ frames of memory. Therefore its memory complexity, even in the smoothing mode, is just $\mathcal{O}(nr \log n \log(1/\epsilon))$. Thus, it provides a nearly memory-optimal mini-batch solution for MC.

Algorithm parameters: The algorithm has 4 parameters: r, K, α, and ω_{evals}. Theoretically these are set as follows: assume that r, λ^+, λ^- are known and pick a desired recovery error ε. Set $\alpha = C_1 f^2 r \log n$ with $f = \lambda^+/\lambda^-$, $K = C_2 \log(1/\varepsilon)$ and $\omega_{evals} = c\lambda^-$ with c a small constant. We explain practical approaches in Sec 4.5.

4.2.2 Main Result: noise-free ST-miss and MC

First, for simplicity, consider the noise-free case, i.e., assume $\boldsymbol{v}_t = 0$. Let $\Delta_j := \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$.

Theorem 4.59 (NORST-miss, $\boldsymbol{v}_t = 0$ case)**.** *Consider Algorithm 8. Let* $\alpha := Cf^2 r \log n$, $\boldsymbol{\Lambda} := \mathbb{E}[\boldsymbol{a}_1 \boldsymbol{a}_1{}']$, $\lambda^+ := \lambda_{\max}(\boldsymbol{\Lambda})$, $\lambda^- := \lambda_{\min}(\boldsymbol{\Lambda})$, $f := \lambda^+/\lambda^-$.

Pick an $\varepsilon \le \min(0.01, 0.03 \min_j \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)^2/f)$. *Let* $K := C \log(1/\varepsilon)$. *If*

1. *left and statistical right incoherence:* $\boldsymbol{P}_j$*'s are* μ*-incoherent and* $\boldsymbol{a}_t$*'s satisfy statistical right incoherence (Definition 4.57);*

2. *max-miss-frac-col* $\le \frac{c_1}{\mu r}$, *max-miss-frac-row*$_\alpha \le \frac{c_2}{f^2}$*;*

3. *subspace change: assume* $t_{j+1} - t_j > Cr \log n \log(1/\varepsilon)$*;*

4. $\boldsymbol{a}_t$*'s are independent of the set of missing entries* $\mathcal{T}_t$*;*

then, with probability (w.p.) at least $1 - 10dn^{-10}$*,*

1. *subspace change is detected quickly:* $t_j \le \hat{t}_j \le t_j + 2\alpha$*,*

2. *the subspace recovery error satisfies*

$$\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) \le \begin{cases} (\varepsilon + \Delta_j) & \text{if } t \in \mathcal{J}_1, \\ (0.3)^{k-1}(\varepsilon + \Delta_j) & \text{if } t \in \mathcal{J}_k, \\ \varepsilon & \text{if } t \in \mathcal{J}_{K+1}. \end{cases}$$

3. *and* $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\| \le 1.2(\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)}) + \varepsilon)\|\boldsymbol{\ell}_t\|$.

Here $\mathcal{J}_0 := [t_j, \hat{t}_j + \alpha)$*,* $\mathcal{J}_k := [\hat{t}_j + k\alpha, \hat{t}_j + (k+1)\alpha)$ *and* $\mathcal{J}_{K+1} := [\hat{t}_j + (K+1)\alpha, t_{j+1})$ *and* $\Delta_j := \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$.

The memory complexity is $\mathcal{O}(nr \log n \log(1/\varepsilon))$ *and the time complexity is* $\mathcal{O}(ndr \log(1/\varepsilon))$.

Corollary 4.60 (NORST-miss for MC)**.** *Under the assumptions of Theorem 4.59, NORST-miss-smoothing (line 25 of Algorithm 8) satisfies* $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t\| \le \varepsilon \|\boldsymbol{\ell}_t\|$ *for all* t*. Thus,* $\|\hat{\boldsymbol{L}} - \boldsymbol{L}\|_F \le \varepsilon \|\boldsymbol{L}\|_F$.

The proof is similar to that given in [33] for the correctness of NORST for robust ST. Please see the Appendix for the changes.

For the purpose of this discussion, we treat the condition number f and the incoherence parameter μ as constants. The above result proves that NORST-miss tracks piecewise constant subspaces to ϵ accuracy, within a delay that is near-optimal, under the following assumptions: left and "statistical" right incoherence holds; the fraction of missing entries in any column of $\boldsymbol{L}$ is $\mathcal{O}(1/r)$ while that in any row (of α-consecutive column sub-matrices of it) is $\mathcal{O}(1)$. Moreover, "smoothing mode" NORST-miss returns ε-accurate estimates of each $\boldsymbol{\ell}_t$ and thus also solves the MC problem. Even in this mode, it has *near-optimal memory complexity and is order-wise as fast as vanilla PCA*. The above result is the *first complete guarantee* for ST-miss. Also, unlike past work, it can *deal*

with piecewise constant subspaces while also automatically reliably detecting subspace change with a near-optimal delay.

Consider the total number of times a subspace can change, J. Since we need the subspace to be constant for at least $(K+3)\alpha$ frames, J needs to satisfy $J(K+3)\alpha \leq d$. Since we need $(K+3)\alpha$ to be at least $Cr\log n\log(1/\varepsilon)$, this means that J must satisfy

$$J \leq c\frac{d}{r\log n\log(1/\varepsilon)}.$$

This, in turn, implies that the rank of the entire matrix, $r_{\scriptscriptstyle L}$, can be at most‘

$$r_{\scriptscriptstyle L} = rJ \leq c\frac{d}{\log n\log(1/\varepsilon)}.$$

Observe that this upper bound is nearly linear in d. This is what makes our corollary for MC interesting. It implies that we can recover $\boldsymbol{L}$ to ε accuracy even in this *nearly linearly growing rank regime*, of course only if the subspace changes are piecewise constant with time and frequent enough so that J is close to its upper bound. In contrast, existing MC guarantees, these require left and right incoherence of $\boldsymbol{L}$ and a Bernoulli model on observed entries with observation probability m/nd where m is the required number of observed entries on average. The convex solution [42] needs $m = Cnr_{\scriptscriptstyle L}\log^2 n$ while the best non-convex solution [11] needs $m = Cnr_{\scriptscriptstyle L}^2\log^2 n$ observed entries. The non-convex approach is much faster, but its required m depends on $r_{\scriptscriptstyle L}^2$ instead of $r_{\scriptscriptstyle L}$ in the convex case. See Sec. 4.3 for a detailed discussion, and Table 4.3 for a summary of it. On the other hand, our missing fraction bounds imply that the total number missing entries needs to at most $\min(nd\cdot\text{max-miss-frac-row}, dn\cdot\text{max-miss-frac-col}) = c\frac{nd}{r}$, or that we need at least $m = (1-c/r)nd$ observed entries.

If subspace changes are infrequent (J is small) so that $r_{\scriptscriptstyle L} \approx r \ll d$, our requirement on observed entries is much stronger than what existing MC approaches need. However, suppose that J equals its allowed upper bound so that $r_{\scriptscriptstyle L} = c\frac{d}{\log n\log(1/\varepsilon)}$; but r is small, say $r = \log n$. In this setting, we need $nd(1-c/\log n)$ while the convex MC solution needs $cn\frac{d}{\log n\log(1/\varepsilon)}\log^2 n = cnd\frac{\log n}{\log(1/\varepsilon)}$. If $\varepsilon = 1/n$, this is $c\cdot nd$, if ε is larger, this is even larger than $c\cdot nd$. Thus, in this regime, our requirement on m is only a little more stringent. Our advantage is that we do not require a

Bernoulli (or any probability) model on the observed entries' set *and* our approach is much faster, memory-efficient, and nearly delay-optimal. This is true both theoretically and in practice; see Tables 4.3 and 4.6. If we consider non-convex MC solutions, they are much faster, but they cannot work in this nearly linear rank regime at all because they will need $Cnd^2/\log^2 n$ observed entries, which is not possible.

A possible counter-argument to the above can be: what if one feeds smaller batches of data to an MC algorithm. Since the subspace change times are not known, it is not clear how to do this. One could feed in batches of size $K\alpha$ which is the memory size used by NORST-miss-smoothing. Even in this case the discussion is the same as above. To simplify writing suppose that $\varepsilon = 1/n$. The convex solution will need $m = cn(Cr\log^2 n)$ observed entries for a matrix of size $n \times (Cr\log^2 n)$. Thus m required is again linear in the matrix size. NORST-miss-smoothing will need this number to be $(1 - c/r)n(Cr\log^2 n)$ which is again only slightly worse when r is small. The non-convex methods will again not work.

The Bernoulli model on the observed entries' set can often be an impractical requirement. For example, erasures due to transmission errors or image/video degradation often come in bursts. Similarly video occlusions by foreground objects are often slow moving or occasionally static, rather than being totally random. Our guarantee does not require the Bernoulli model but the tradeoff is that, in general, it needs more observed entries. A similar tradeoff is observed in the robust PCA literature. The guarantee of [7] required a uniform random or Bernoulli model on the outlier supports, but tolerated a constant fraction of corrupted entries. In other words it needed the number of uncorrupted entries to be at least $c \cdot nd$. Later algorithms such as AltProj [36] did not require any random model on outlier support but needed the number of un-corrupted entries to be at least $(1 - c/r)nd$ which is a little more stringent requirement.

4.2.3 Main Result – ST-miss and MC with noise

So far we gave a result for ST-miss and MC in the noise-free case. A more practical model is one that allows for small unstructured noise (modeling error). Our result also extends to this

case with one extra assumption. In the noise-free case, there is no real lower bound on the amount of subspace change required for reliable detection. Any nonzero subspace change can be detected (and hence tracked) as long as the previous subspace is recovered to ε accuracy with ε small enough compared to the amount of change. If the noise $\boldsymbol{v}_t$ is such that its maximum covariance in any direction is smaller than $\varepsilon^2\lambda^-$, then Theorem 4.59 and Corollary 4.60 hold with almost no changes. If the noise is larger, as we will explain next, we will need the amount of subspace change to be larger than the noise-level. Also, we will be able to track the subspaces only up to accuracy equal to the noise level.

Suppose that the noise $\boldsymbol{v}_t$ is bounded. Let $\lambda_v^+ := \|\mathbb{E}[\boldsymbol{v}_t\boldsymbol{v}_t{}']\|$ be the noise power and let $r_v := \max_t \|\boldsymbol{v}_t\|^2/\lambda_v^+$ be the effective noise dimension. Trivially, $r_v \leq n$. To understand things simply, first suppose that the subspace is fixed. If the noise is isotropic (noise covariance is a multiple of identity), then, as correctly pointed out by an anonymous reviewer, one can achieve noise-averaging in the PCA step by picking α large enough: it needs to grow as [3] $n(\lambda_v^+/\lambda^-)/\varepsilon^2$. Isotropic noise is the most commonly studied setting for PCA, but it is not the most practical. In the more practical non-isotropic noise case, it is not even possible to achieve noise-averaging by increasing α. In this setting, with any choice of α, the subspace can be recovered only up to the noise level, i.e., we can only achieve recovery accuracy $c\lambda_v^+/\lambda^-$. If we are satisfied with slightly less accurate estimates, i.e., if we set $\varepsilon = c\sqrt{\frac{\lambda_v^+}{\lambda^-}}$, and if the effective noise dimension $r_v = Cr$, then the required value of α does not change from what it is in Theorem 4.59. Now consider the changing subspace setting. We can still show that we can detect subspace changes that satisfy $0.03\min_j \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)^2/f \geq \varepsilon$, but now $\varepsilon = c\sqrt{\frac{\lambda_v^+}{\lambda^-}}$. This imposes a non-trivial lower bound on the amount of change that can be detected. The above discussion is summarized in the following corollary.

Corollary 4.61 (ST-miss and MC with $\boldsymbol{v}_t \neq 0$)**.** *Suppose that $\boldsymbol{v}_t$ is bounded, mutually independent and identically distributed (iid) over time, and is independent of the $\boldsymbol{\ell}_t$'s. Define $\lambda_v^+ := \|\mathbb{E}[\boldsymbol{v}_t\boldsymbol{v}_t{}']\|$ and $r_v := \frac{\max_t \|\boldsymbol{v}_t\|^2}{\lambda_v^+}$.*

[3] α needs to grow as $C\min(r_v \log n, n)(\lambda_v^+/\lambda^-)/\epsilon^2$; for the isotropic case, $r_v = n$ and thus the discussion follows.

- *If $r_v = Cr$ and $\lambda_v^+ \leq c\varepsilon^2\lambda^-$, then the results of Theorem 4.59 and Corollary 4.60 hold without any changes.*

- *For a general λ_v^+, we have the following modified result. Suppose that $r_v = Cr$, $\min_j \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)^2 \geq Cf\sqrt{\frac{\lambda_v^+}{\lambda^-}}$, and conditions 1, 2, 3 of Theorem 4.59 hold. Then all conclusions of Theorem 4.59 and Corollary 4.60 hold with $\varepsilon = c\sqrt{\frac{\lambda_v^+}{\lambda^-}}$.*

- *For a general r_v, if we set $\alpha = Cf^2 \max(r \log n, \min(n, r_v \log n))$ then the above conclusions hold.*

If the noise is *isotropic*, the next corollary shows that we can track to any accuracy ε by increasing the value of α. It is not interesting from a tracking perspective because its required value of α is much larger. However, it provides a result that is comparable to the result for streaming PCA with missing data from [32] that we discuss later.

Corollary 4.62 (ST-miss and MC, isotropic noise case)**.** *If the noise $\boldsymbol{v}_t$ is isotropic (so that $r_v = n$), then, for any desired recovery error level ε, if $\alpha = Cn\frac{\frac{\lambda_v^+}{\lambda^-}}{\varepsilon^2}$, and all other conditions of Theorem 4.59 hold, then all conclusions of Theorem 4.59 and Corollary 4.60 hold.*

We should mention here that the above discussion and results assume that PCA is solved via a simple SVD step (compute top r left singular vectors). In the non-isotropic noise case, if its covariance matrix were known (or could be estimated), then one can replace simple SVD by pre-whitening techniques followed by SVD, in order to get results similar to the isotropic noise case, e.g., see [29].

4.2.4 Extensions of basic NORST-miss

Sample-Efficient-NORST-miss. This is a simple modification of NORST-miss that will reduce its sample complexity. The reason that NORST-miss needs many more observed entries is because of the projected LS step which solves for the missing entries vector, $\boldsymbol{z}_t$, after projecting $\boldsymbol{y}_t$ orthogonal to $\hat{\boldsymbol{P}}_{(t-1)}$. This step is computing the pseudo-inverse of $(\boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}')_{\mathcal{T}_t}$. Our bound on max-miss-frac-col helps ensure that this matrix is well conditioned for any set

$\mathcal{T}_t$ of size at most max-miss-frac-col $\cdot\, n$. Notice however that we prove that NORST-miss recovers $\boldsymbol{P}_j$ to ϵ accuracy with a delay of just $(K+2)\alpha = Cr\log n\log(1/\epsilon)$. Once the subspace has been recovered to ε accuracy, there is no need to use projected LS to recover $\boldsymbol{z}_t$. One just needs to recover $\boldsymbol{a}_t$ given a nearly perfect subspace estimate and the observed entries. This can be done more easily as follows (borrows PETRELS idea): let $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$, solve for $\boldsymbol{a}_t$ as $\hat{\boldsymbol{a}}_t := (\boldsymbol{I}_{\Omega_t}{}'\hat{\boldsymbol{P}}_{(t)})^{\dagger}\boldsymbol{I}_{\Omega_t}{}'\boldsymbol{y}_t$, and set $\hat{\boldsymbol{\ell}}_t \leftarrow \hat{\boldsymbol{P}}_{(t)}\hat{\boldsymbol{a}}_t$. Recall here that $\Omega_t = \mathcal{T}_t^c$. If the set of observed or missing entries was i.i.d. Bernoulli for just the later time instants, this approach will only need $\Omega(r\log r\log^2 n)$ samples at each time t, whp. This follows from [2, Lemma 3]. Suppose that $\varepsilon = 1/n$, then $K\alpha = Cr\log^2 n$. Let $d_j := t_{j+1} - t_j$ denote the duration for which the subspace is $\boldsymbol{P}_j$. Thus $\sum_j d_j = d$. Also recall that $r_{\scriptscriptstyle L} \le rJ$. Thus, with this approach, the number of observed entries needed is $m = \Omega\left(\sum_{j=1}^{J}\left(n(1-c/r)K\alpha + Cr\log r\log^2 n(d_j - K\alpha)\right)\right) = \Omega\left(\sum_j [n(1-c/r)r\log^2 n + d_j r\log r\log^2 n]\right) = \Omega(\max(n,d)r_{\scriptscriptstyle L}\log^2 n(\log r - c/r))$ as long as the observed entries follow the i.i.d. Bernoulli model for the time after the first $K\alpha$ time instants after a subspace change. Or, we need the observed entries to be i.i.d. Bernoulli$(1-c/r)$ for first $K\alpha$ frames and i.i.d. Bernoulli$(r\log^2 n\log r/n)$ afterwards. Observe that the m needed by sample-efficient-NORST-miss is only $(\log r - c/r)$ times larger than the best sample complexity needed by any MC technique - this is the convex methods (nuclear norm min). However sample-efficient-NORST-miss is much faster and memory-efficient compared to nuclear norm min.

NORST-sliding-window. In the basic NORST approach we use a different set of estimates $\hat{\boldsymbol{\ell}}_t$ for each subspace update step. So, for example, the first subspace estimate is computed at $\hat{t}_j + \alpha - 1$ using $\hat{\boldsymbol{L}}_{\hat{t}_j+\alpha-1;\alpha}$; the second is computed at $\hat{t}_j + 2\alpha - 1$ using $\hat{\boldsymbol{L}}_{\hat{t}_j+2\alpha-1;\alpha}$; and so on. This is done primarily to ensure mutual independence of the set of $\boldsymbol{\ell}_t$'s in each interval because this is what makes the proof easier (allows use of matrix Bernstein for example). However, in practice, we can get faster convergence to an ϵ-accurate estimate of $\boldsymbol{P}_j$, by removing this restriction. This approach is of course motivated by the sliding window idea that is ubiquitous in signal processing. For any sliding-window method, there is the window length which we keep as α and the hop-length which we denote by β.

Thus, NORST-sliding-window (β) is Algorithm 8 with the following change: compute $\hat{\boldsymbol{P}}_{j,1}$ using $\hat{\boldsymbol{L}}_{\hat{t}_j+\alpha-1;\alpha}$; compute $\hat{\boldsymbol{P}}_{j,2}$ using $\hat{\boldsymbol{L}}_{\hat{t}_j+\alpha+\beta-1;\alpha}$; compute $\hat{\boldsymbol{P}}_{j,3}$ using $\hat{\boldsymbol{L}}_{\hat{t}_j+\alpha+2\beta-1;\alpha}$; and so on. Clearly $\beta < \alpha$ and $\beta = \alpha$ returns the basic NORST-miss.

NORST-buffer. Another question if we worry only about practical performance is whether re-using the same α data samples $\boldsymbol{y}_t$ in the following way helps: At $t = \hat{t}_j + k\alpha - 1$, the k-th estimate is improved R times as follows. First we obtain $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \dots \hat{\boldsymbol{\ell}}_t]$ which are used to compute $\hat{\boldsymbol{P}}_{j,k}$ via r-SVD. Let us denote this by $\hat{\boldsymbol{P}}^0_{j,k}$. Now, we use this estimate to obtain a second, and slightly more refined estimate of the same $\boldsymbol{L}_{t;\alpha}$. We denote these as $\hat{\boldsymbol{L}}^{(1)}_{t;\alpha}$ and use this estimate to get $\hat{\boldsymbol{P}}^{(1)}_{j,k}$. This process is repeated for a total of $R+1$ (reuse) times. We noticed that using $R = 4$ suffices in most synthetic data experiments and for real data, $R = 0$ (which reduces to the basic NORST algorithm) suffices. This variant has the same memory requirement as NORST-original. The time complexity, however, increases by a factor of $R + 1$.

4.3 Detailed discussion of prior art

Streaming PCA with missing data, complete guarantee. The problem of streaming PCA with missing data was studied and a provable approach called modified block power method (MBPM) was introduced in [32]. A similar problem called "subspace learning with partial information" is studied in [20]. These give the following complete guarantee.

Theorem 4.63 (streaming PCA, missing data [32, 20])**.** *Consider a data stream, for all* $t = 1, \cdots, d$, $\boldsymbol{\ell}_t = A\boldsymbol{z}_t + \boldsymbol{w}_t$ *where* $\boldsymbol{z}_t$ *are* r *length vectors generated i.i.d from a distribution* $\mathcal{D}$ *s.t.* $\mathbb{E}[(\boldsymbol{z}_t)_i] = 0$ *and* $\mathbb{E}[(\boldsymbol{z}_t)_i^2] = 1$ *and* A *is an* $n \times r$ *matrix with SVD* $A = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}'$ *with* $\lambda_1 = 1 \geq \lambda_2 \geq \cdots \lambda_r = \lambda^- > 0$. *The noise* $\boldsymbol{w}_t$ *is bounded:* $|(\boldsymbol{w}_t)_i| \leq M_\infty$, *and* $\mathbb{E}[(\boldsymbol{w}_t)_i^2] = \sigma^2$. *Assume that (i)* A *is* μ*-incoherent; and (ii) we observe each entry of* $\boldsymbol{\ell}_t$ *independently and uniformly at random with probability* ρ*; this is the Bernoulli*(ρ) *model. If* $d \geq \alpha$ *with* $\alpha :=$

$$\Omega\left(\frac{M_\infty^2(r\mu^2/n + \sigma^2 + nr^2(\mu^2/n+\sigma^2)^2)(\log n)^2\log(1/\varepsilon)}{\log\left(\frac{\sigma^2+0.75\lambda^-}{\sigma^2+0.5\lambda^-}\right)(\lambda^-)^2\epsilon^2\rho^2}\right)$$

then, $\mathrm{SE}(\hat{\boldsymbol{P}}_{(d)}, \boldsymbol{U}) \leq \epsilon$ *w.p. at least 0.99.*

There are many differences between this guarantee and ours: (i) it only recovers a single unknown subspace (since it is solving a PCA problem), and is unable to detect or track changes in the subspace; (ii) it requires the missing entries to follow the i.i.d. Bernoulli model; and (iii) it only provides a guarantee that the final subspace estimate, $\hat{\boldsymbol{P}}_{(d)}$, is ϵ-accurate (it does not say anything about the earlier estimates). (iv) Finally, even with setting $\sigma^2 = \epsilon^2\lambda^-$ in the above (to simply compare its noise bound with ours), the required lower bound on d implied by it is $d \geq Cr^2 \log^2 n \log(1/\epsilon)/\rho^2$. This is $r \log n$ times larger than what our result requires. The lower bound on d can be interpreted as the tracking delay in the setting of ST-miss. The Bernoulli model on missing entries is impractical in many settings as discussed earlier in Sec. 4.2.2. On the other hand, MBPM is streaming as well as memory-optimal while our approach is not streaming and only nearly memory optimal. For a summary, see Table 4.2. Here *"streaming"* means that it needs only one pass over the data. Our approach uses SVD which requires multiple passes over short batches of data of size of order $r \log n$.

ST-miss, partial guarantees. In the ST literature, there are three well-known algorithms for ST-miss: PAST [48, 49], PETRELS [12] and GROUSE [3, 4, 53, 38]. All are motivated by stochastic gradient descent (SGD) to solve the PCA problem and the Oja algorithm [37]. These and many others are described in detail in a review article on subspace tracking [1]. GROUSE can be understood as an extension of Oja's algorithm on the Grassmanian. It is a very fast algorithm since it only involves first order updates. It has been studied in [3, 4, 53]. The best partial guarantee for GROUSE rewritten in our notation is as follows.

Theorem 4.64 (GROUSE [4] (Theorem 2.14))**.** *Assume that the subspace is fixed, i.e., that* $\boldsymbol{P}_{(t)} = \boldsymbol{P}$ *for all* t*. Denote the unknown subspace by* $\boldsymbol{P}$*. Let* $\epsilon_t := \sum_{i=1}^{r} \sin^2 \theta_i(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P})$ *where* θ_i *is the* i*-th largest principal angle between the two subspaces. Also, for a vector* $\boldsymbol{z} \in \mathbb{R}^n$*, let* $\mu(\boldsymbol{z}) := \frac{n\|\boldsymbol{z}\|_\infty^2}{\|\boldsymbol{z}\|_2^2}$ *quantify its denseness. Assume that (i)* $\boldsymbol{P}$ *is* μ*-incoherent; (ii) the coefficients vector* $\boldsymbol{a}_t$ *is drawn independently from a standard Gaussian distribution, i.e.,* $(\boldsymbol{a}_t)_i \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$*; (iii) the size of the set of observed entries at time* t*,* Ω_t*, satisfies* $|\Omega_t| \geq (64/3)r(\log^2 n)\mu \log(20r)$*; and the following assumptions on intermediate algorithm estimates hold:*

- $\epsilon_t \leq \min(\frac{r\mu}{16n}, \frac{q^2}{128n^2r})$;
- *the residual at each time,* $\boldsymbol{r}_t := \boldsymbol{\ell}_t - \hat{\boldsymbol{P}}_{(t)}\hat{\boldsymbol{P}}'_{(t)}\boldsymbol{\ell}_t$ *is "dense", i.e., it satisfies* $\mu(\boldsymbol{r}_t) \leq \min\{\log n[\frac{0.045}{\log 10}C_1 r\mu \log(20r)]^{0.5}, \log^2 n \frac{0.05}{8\log 10}C_1 \log(20r)\}$ *with probability at least* $1-\bar{\delta}$ *where* $\bar{\delta} \leq 0.6$.

Then, $\mathbb{E}[\epsilon_{t+1}|\epsilon_t] \leq \epsilon_t - .32(.6-\bar{\delta})\frac{q}{nr}\epsilon_t + 55\sqrt{\frac{n}{q}}\epsilon_t^{1.5}$.

Observe that the above result makes a denseness assumption on the residual $\boldsymbol{r}_t$ and the residual is a function of $\hat{\boldsymbol{P}}_{(t)}$. Thus it is making assumptions on intermediate algorithm estimates and hence is a partial guarantee.

In follow-up work, the PETRELS [12] approach was introduced. It is slower than GROUSE, but has much better performance in numerical experiments. To understand the main idea of PETRELS, let us ignore the small noise $\boldsymbol{v}_t$. Then, $\boldsymbol{y}_t$ can be expressed as $\boldsymbol{y}_t = \boldsymbol{I}_{\Omega_t}\boldsymbol{I}_{\Omega_t}{}'\boldsymbol{\ell}_t = \boldsymbol{I}_{\Omega_t}\boldsymbol{I}_{\Omega_t}{}'\boldsymbol{P}_{(t)}\boldsymbol{a}_t$. Let $\tilde{\boldsymbol{P}} := \boldsymbol{P}_{(t)}$. If $\tilde{\boldsymbol{P}}$ were known, one could compute $\boldsymbol{a}_t$ by solving a LS problem to get $\hat{\boldsymbol{a}}_t := (\boldsymbol{I}_{\Omega_t}{}'\tilde{\boldsymbol{P}})^{\dagger}\boldsymbol{I}_{\Omega_t}{}'\boldsymbol{y}_t$. This of course implicitly assumes that $\boldsymbol{I}_{\Omega_t}{}'\tilde{\boldsymbol{P}}$ is well-conditioned. This matrix is of size $(n-|\mathcal{T}_t|)\times r$, thus a necessary condition for it to be well conditioned is the same as the one for NORST-miss: it also needs $n-|\mathcal{T}_t| \geq r$ although the required sufficient condition is different[4]. Of course $\tilde{\boldsymbol{P}}$ is actually unknown. PETRELS thus solves for $\tilde{\boldsymbol{P}}$ by solving the following

$$\min_{\tilde{\boldsymbol{P}}} \sum_{m=1}^{t} \lambda^{t-m} \|\boldsymbol{y}_m - \boldsymbol{I}_{\Omega_m}\boldsymbol{I}_{\Omega_m}{}'\tilde{\boldsymbol{P}}(\boldsymbol{I}_{\Omega_m}{}'\tilde{\boldsymbol{P}})^{\dagger}\boldsymbol{I}_{\Omega_m}{}'\boldsymbol{y}_m\|^2.$$

Here $\boldsymbol{M}^{\dagger} := (\boldsymbol{M}'\boldsymbol{M})^{-1}\boldsymbol{M}'$ and λ is the discount factor (set to 0.98 in their code). To solve this efficiently, PETRELS first decomposes it into updating each row of $\tilde{\boldsymbol{P}}$, and then parallely solves the n smaller problems by second-order SGD.

The best guarantee for PETRELS from [47] is summarized next.

Theorem 4.65 (PETRELS [47](Theorem 2))**.** *Assume that the subspace is fixed, i.e., that* $\boldsymbol{P}_{(t)} = \boldsymbol{P}$ *for all* t. *Assume that (i) the set of observed entries are drawn from the i.i.d. Bernoulli model*

[4] If Ω_t follows an i.i.d. Bernoulli model, a sufficient condition would be $n-|\mathcal{T}_t| \geq Cr\log r\log^2 n$ [2], or equivalently, max-miss-frac-col $\leq 1-(Cr\log r\log^2 n)/n$.

with parameter ρ; (ii) the coefficients $(\boldsymbol{a}_t)$'s are zero-mean random vectors with diagonal covariance $\boldsymbol{\Lambda}$ and all higher-order moments finite; (iii) the noise, $\boldsymbol{v}_t$ are i.i.d and independent of $\boldsymbol{a}_t$; (iv) the subspace $\boldsymbol{P}$ and the initial estimate $\hat{\boldsymbol{P}}_0$ satisfies the following incoherence assumption $\sum_{i=1}^{n}\sum_{j=1}^{r}(\boldsymbol{P})_{ij}^4 \leq \frac{C}{n}$, and $\sum_{i=1}^{n}\sum_{j=1}^{r}(\hat{\boldsymbol{P}}_0)_{ij}^4 \leq \frac{C}{n}$; (v) the step-size is appropriately chosen; and (v) the initialization satisfies $\mathbb{E}\left[\|\boldsymbol{Q}_0^{(n)} - \boldsymbol{Q}(0)\|_2\right] \leq \frac{C}{\sqrt{n}}$. Here $\boldsymbol{Q}_0^{(n)} := \hat{\boldsymbol{P}}_0{}'\boldsymbol{P}$ denotes the matrix of initial cosine similarities and $\boldsymbol{Q}(\tau)$ is the "scaling limit" which is defined as the solution of the following coupled ordinary differential equations:

$$\begin{aligned}\frac{d}{d\tau}\boldsymbol{Q}(\tau) =&[\rho\boldsymbol{\Lambda}^2\boldsymbol{Q}(\tau) - 1/2\boldsymbol{Q}(t)\boldsymbol{G}(\tau) - \\ &\boldsymbol{Q}(\tau)(\boldsymbol{I} - 1/2\boldsymbol{G}(\tau))\boldsymbol{Q}'(\tau)\rho\boldsymbol{\Lambda}^2\boldsymbol{Q}(\tau)]\boldsymbol{G}(\tau) \\ \frac{d}{d\tau}\boldsymbol{G}(\tau) =&\boldsymbol{G}(\tau)[\mu - \boldsymbol{G}(\tau)(\boldsymbol{G}(\tau)+\boldsymbol{I})(\boldsymbol{Q}'(\tau)\rho\boldsymbol{\Lambda}^2\boldsymbol{Q}(\tau)+\boldsymbol{I})]\end{aligned}$$

where ρ is the subsampling ratio and $\mu = n(1-\lambda)$ where λ is the discount parameter defined earlier. Then, for any fixed $d > 0$, the time-varying cosine similarity matrix $\boldsymbol{Q}_{\lfloor n\tau \rfloor}^{(n)} = \hat{\boldsymbol{P}}_{(\lfloor n\tau \rfloor)}{}'\boldsymbol{P}$ satisfies $\sup_{n\geq 1}\mathbb{E}\left[\|\boldsymbol{Q}_{\lfloor n\tau \rfloor}^{(n)} - \boldsymbol{Q}(\tau)\|\right] \leq \frac{C_d}{\sqrt{n}}$.

For further details, please refer to [47, Eq's 29, 33, 34]. The above is a difficult result to further simplify since, even for $r = 1$, it is not possible to obtain a closed form solution of the above differential equation. This is why it is impossible to say what this result says about $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P})$ or any other error measure. Hence the above is also a *partial guarantee*. [47] also provides a guarantee for GROUSE that has a similar flavor to the above result.

Online MC, different model. There are a few works with the term *online MC* in their title and a reader may wrongly confuse these as being solutions to our problem. All of them study very different "online" settings than ours, e.g., [25] assumes one matrix entry comes in at a time. The work of [27] considers a problem of designing matrix sampling schemes based on current estimates of the matrix columns. This is useful only in settings where one is allowed to choose which samples to observe. This is often not possible in applications such as video analytics.

MC. There has been a very large amount of work on provable MC. We do not discuss everything here since MC is not the main focus of this work. The first guarantee for MC was provided in

[15]. This studied the nuclear norm minimization (NNM) solution. After NNM, there has been much later work on non-convex, and hence faster, provable solutions: alternating-minimization, e.g., [26, 35, 43, 21], and projected gradient descent (proj GD), e.g., [23, 18, 17] and alternating-projection [24, 28]. All these works assume a uniform random or i.i.d. Bernoulli model on the set of missing entries (both are nearly equivalent for large n, d). There has been some later work that relaxes this assumption. This includes [9, 16] which assumes independent but not identical probability of the (i,j)-th entry being missed. The authors allow this probability to be inversely proportional to row and column "leverage scores" (quantifies denseness of a row or a column of $\boldsymbol{L}$) and hence allows the relaxing of the incoherence requirement on $\boldsymbol{L}$. If leverage scores were known, one could sample more frequently from rows or columns that are less dense (more sparse). Of course it is not clear how one could know or approximate these scores. There is also work that assumes a completely different probabilistic models on the set of observed entries, e.g., [5]. In summary, all existing MC works need a probabilistic model on the set of observed (equivalently, missed) entries, typically i.i.d. Bernoulli. As noted earlier this can be an impractical requirement in some applications. Our work does not make any such assumption but needs more observed entries, a detailed discussion of this is provided earlier.

NORST for robust ST [33]. While both the NORST-miss algorithm and guarantee are simple modifications of those for NORST for robust ST, our current result has two important advantages because it solves a simpler problem than robust ST. Since there are no outliers, there is no need for the amount of subspace change or the initial estimate's accuracy to be smaller than the outlier magnitude lower bound. This was needed in the robust ST case to obtain an estimate of the outlier support $\mathcal{T}_t$. Here, this support is known. This is why NORST-miss has the following two advantages. (i) It works with a zero initialization where as NORST (for robust ST) required a good enough initialization for which AltProj or PCP needed to be applied on an initial short batch of observed data. (ii) It does not need an upper bound on the amount of subspace change at each t_j, it allows both slow and sudden changes.

4.4 Robust ST with missing entries

Robust ST with missing entries (RST-miss) is a generalization of robust ST and of ST-miss. In this case, we observe n-dimensional data vectors that satisfy

$$\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{\ell}_t + \boldsymbol{g}_t) + \boldsymbol{v}_t, \text{ for } t = 1, 2, \dots, d. \tag{4.4}$$

where $\boldsymbol{g}_t$'s are the sparse outliers. Let $\boldsymbol{x}_t := \mathcal{P}_{\Omega_t}(\boldsymbol{g}_t)$. We use $\mathcal{T}_{\text{sparse},t}$ to denote the support of $\boldsymbol{x}_t$. This is the part of the outliers that actually corrupt our measurements, thus in the sequel we will only work with $\boldsymbol{x}_t$. With $\boldsymbol{x}_t$ defined as above, $\boldsymbol{y}_t$ can be expressed as

$$\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{\ell}_t) + \boldsymbol{x}_t + \boldsymbol{v}_t \tag{4.5}$$

Observe that, by definition, $\boldsymbol{x}_t$ is supported outside of $\mathcal{T}_t$ and hence $\mathcal{T}_t$ and $\mathcal{T}_{\text{sparse},t}$ are disjoint. Defining the $n \times d$ matrix $\boldsymbol{L} := [\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots \boldsymbol{\ell}_d]$, the above is a robust MC problem.

The main modification needed in this case is outlier support recovery. The original NORST for robust ST [33] used l_1 minimization followed by thresholding based support recovery for this purpose. In this case, the combined sparse vector is $\tilde{\boldsymbol{x}}_t := \boldsymbol{x}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\ell}_t$. Support recovery in this case is thus a problem of sparse recovery with partial support knowledge $\mathcal{T}_t$. In this case, we can still use l_1 minimization followed by thresholding. However a better approach is to use noisy modified-CS [44, 52] which was introduced to exactly solve this problem. We use the latter. The second modification needed is that, just like in case of robust ST, we need an accurate subspace initialization. To get this, we can use the approach used in robust ST [33]: for the initial $Cr \log n \log(1/\varepsilon)$ samples, use the AltProj algorithm for robust PCA (while ignoring the knowledge of $\mathcal{T}_t$ for this initial period). We summarize the approach in Algorithm 9.

We have the following guarantee for NORST-miss-robust. Let max-outlier-frac-row$_\alpha$ be the maximum fraction of outliers per row of any sub-matrix of $\boldsymbol{X}$ with α consecutive columns; max-outlier-frac-col be the maximum fraction of outlier per column of $\boldsymbol{X}$. Also define $x_{\min} := \min_t \min_{i \in \mathcal{T}_{\text{sparse},t}} |(\boldsymbol{x}_t)_i|$ to denote the minimum outlier magnitude and let $\Delta := \max_j \Delta_j = \max_j \text{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$.

Corollary 4.66. *Consider Algorithm 9. Assume all conditions of Theorem 4.59 hold and*

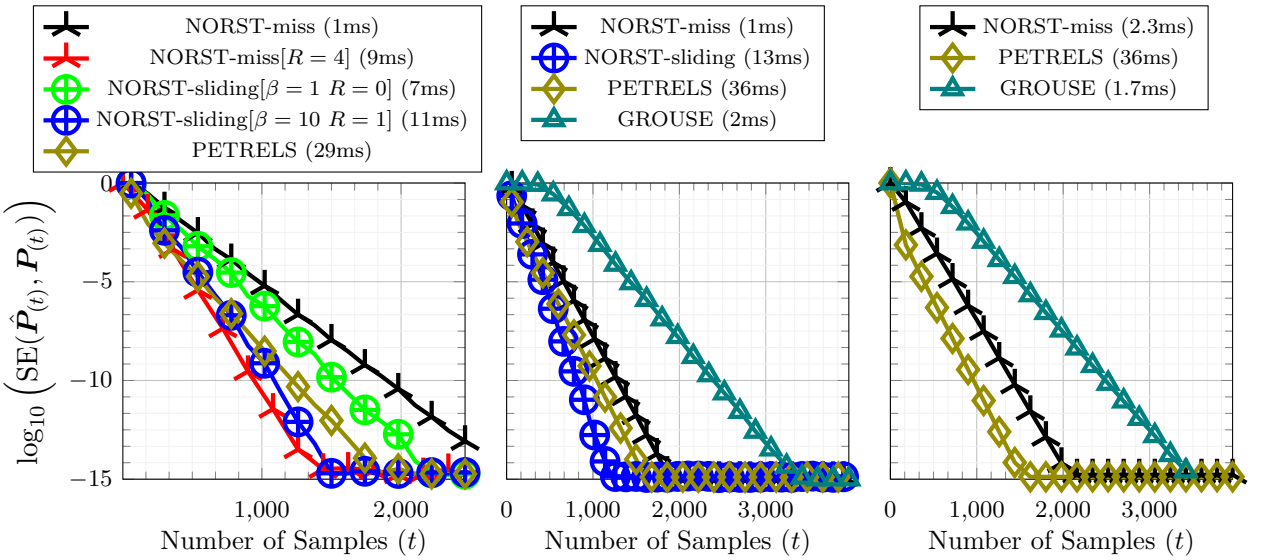
1. *max-miss-frac-col* $+ \ 2 \cdot$ *max-outlier-frac-col* $\leq \frac{c_1}{\mu r}$*; and* *max-miss-frac-row*$_\alpha$ $+$ *max-outlier-frac-row*$_\alpha \leq \frac{c_2}{f^2}$*;*
2. *subspace change:*
   (a) $t_{j+1} - t_j > (K+2)\alpha$*, and*
   (b) $\Delta \leq 0.8$ *and* $C_1\sqrt{r\lambda^+}(\Delta + 2\varepsilon) \leq x_{\min}$
3. *initialization satisfies* $\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \leq 0.25$ *and* $C_1\sqrt{r\lambda^+}\mathrm{SE}(\hat{\boldsymbol{P}}_0, \boldsymbol{P}_0) \leq x_{\min}$*;*

then, all guarantees of Theorem 4.59 and Corollary 4.60 hold.

Remark 4.67 (Relaxing outlier magnitudes lower bound)**.** *As also explained in [33], the outlier magnitude lower bound can be significantly relaxed. First, without any changes, if we look at the proof, our required lower bound on outlier magnitudes is actually* $0.3^{k-1}\sqrt{r\lambda^+}(\Delta + 2\varepsilon)$ *in interval* k *of subspace update. To be precise, we only need* $\min_{t\in\mathcal{J}_k} \min_{i\in\mathcal{T}_{\text{sparse},t}} |(\boldsymbol{x}_t)_i| \geq 0.3^{k-1}\sqrt{r\lambda^+}(\Delta + 2\varepsilon)$*. Here* $\mathcal{J}_k$ *is the interval defined in Theorem 4.59. Thus, for* $t \in \mathcal{J}_{K+1}$ *(after the update step is complete but the subspace has not changed), we only need* $\min_{i\in\mathcal{T}_{\text{sparse},t}} |(\boldsymbol{x}_t)_i| \geq \varepsilon\sqrt{r\lambda^+}$*. Moreover, this can be relaxed even more as explained in Remark 2.4 of [33].*

The proof is similar to that given in [33]. Please see the Appendix for an explanation of the differences. The advantage of using modified-CS to replace l_1 min when recovering the outlier support is that it weakens the required upper bound on max-miss-frac-col by a factor of two. If we used l_1 min, we would need $2 \cdot$ (max-miss-frac-col + max-outlier-frac-col) to satisfy the upper bound given in the first condition.

Comparison with existing work. Existing solutions for robust ST-miss include GRASTA [22], APSM [13] and ROSETA [31]. APSM comes with a partial guarantee, while GRASTA and ROSETA do not have a guarantee. The first few provable guarantees for robust MC were [7, 10]. Both studied the convex optimization solution which was slow. Recently, there have been two other works [50, 11] which are projected-GD based approaches and hence are much faster. These assume



(a) Moving Object Model ($\rho = 0.8$) (b) Bernoulli Model ($\rho = 0.9$) (c) Bernoulli(0.9) and $\boldsymbol{\Lambda}_t$ time-varying

Figure 4.2: We compare NORST-miss and its extensions with PETRELS and GROUSE. We plot the logarithm of the subspace error between the true subspace $\boldsymbol{P}_{(t)}$ and the algorithm estimates, $\hat{\boldsymbol{P}}_{(t)}$ on the y-axis and the number of samples (t) on the x-axis. As can be seen, in the first two cases, NORST-buffer and NORST-sliding have the best performance (while also being faster than PETRELS), followed by PETRELS, basic NORST and then GROUSE. PETRELS performs best in the scenario of time varying $\boldsymbol{\Lambda}_t$. The computational time per sample (in milliseconds) for each algorithm is mentioned in the legend.

an $\mathcal{O}(1/r)$ bound on outlier fractions per row and per column. All these assume that the set of observed entries is i.i.d. Bernoulli.

Compared with these, our result needs slow subspace change and a lower bound on outlier magnitudes; but it does not need a probabilistic model on the set of missing or outlier entries, and improves the required upper bound on outlier fractions per row by a factor of r. Also, our result needs more observed entries in the setting of $r_L \approx r$, but not when r_L is significantly larger than r, for example not when r_L is nearly linear in d. A summary of this discussion is given in Table 4.4.

4.5 Experimental Comparisons

We present the results of numerical experiments on synthetic and real data[5]. All the codes for our experiments are available at https://github.com/vdaneshpajooh/NORST-rmc. *In this section, we refer to NORST-miss as just NORST.* All time comparisons are performed on a Desktop Computer with Intel Xeon E3-1200 CPU, and 8GB RAM.

4.5.1 Parameter Setting for NORST

The algorithm parameters required for NORST are r, K, α and ω_{evals}. For our theory, we assume r, λ^+, λ^-, are known, and we pick a desired accuracy, ϵ. We set $K = C\log(1/\epsilon)$, $\alpha = Cf^2 r\log n$, and $\omega_{evals} = 2\epsilon^2\lambda^-$ with C being a numerical constant more than one. Experimentally, the value of r needs to be set from model knowledge, however, overestimating it by a little does not significantly affect the results. In most of our experiments, we set $\alpha = 2r$ (ideally it should grow as $r\log n$ but since $\log n$ is very small for practical values of n it can be ignored). α should be a larger multiple of r when either the data is quite noisy or when few entries are observed. We set K based on how accurately we would like to estimate the subspace. The parameter ω_{evals} needs to be set as a small fraction of the minimum signal space eigenvalue. In all synthetic data experiments, we set $\omega_{evals} = 0.0008\lambda^-$. Another way to set ω_{evals} is as follows. After $K\alpha$ frames, we can estimate $\hat{\lambda}^-$ as the r-th eigenvalue of $\sum_{\tau=t-\alpha+1}^{t}\hat{\boldsymbol{\ell}}_\tau\hat{\boldsymbol{\ell}}_\tau'/\alpha$ and set $\omega_{evals} = c\hat{\lambda}^-$ as mentioned before. We use the Conjugate Gradient Least Squares (CGLS) method [39] for the LS step with tolerance as 10^{-16}, and maximum iterations as 20.

For the video experiments, we estimated r using training data from a few videos and fixed it as $r = 30$. We let λ^- be the r-th eigenvalue of the training dataset. We used $\omega_{evals} = 1.6\times 10^{-6}\lambda^- = 0.002$, $\alpha = 2r$ and $K = 3$ for the video data. The reason that we use a smaller fraction of λ^- as ω_{evals} is because videos are only approximately low-rank.

[5]We downloaded the PETRELS' and GROUSE code from the authors' website and all other algorithms from https://github.com/andrewssobral/lrslibrary.

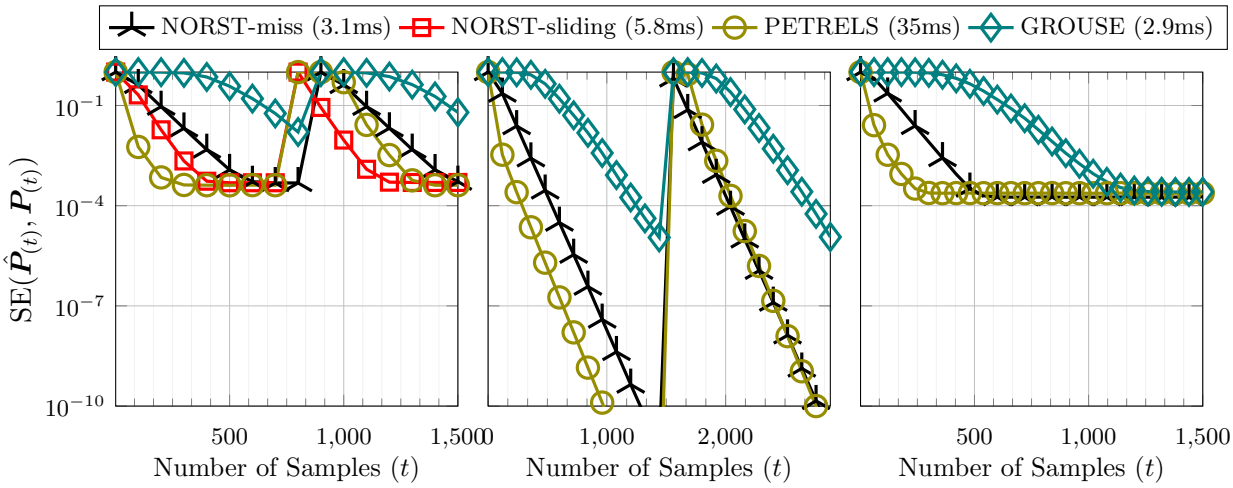
4.5.2 Fixed Subspace, Noise-free data

We generated the data according to (4.1) and set $\boldsymbol{v}_t = 0$. We assume a fixed subspace i.e. $J = 1$. We generate the subspace basis matrix $\boldsymbol{P} \in \mathbb{R}^{n \times r}$ by ortho-normalizing the columns of a random Gaussian matrix with $n = 1000$ and $r = 30$. The $\boldsymbol{a}_t$'s (for $t = 1, \cdots, d$ and $d = 4000$) are generated independently as $(\boldsymbol{a}_t)_i \overset{\text{i.i.d}}{\sim} \text{unif}[-q_i, q_i]$ where $q_i = \sqrt{f} - \sqrt{f}(i-1)/2r \quad \text{for} \quad i = 1, 2, \cdots, r-1$ and $q_r = 1$. Thus, the condition number of $\boldsymbol{\Lambda}$ is f and we set $f = 100$.

For our first experiment, the observed entries' set was i.i.d. Bernoulli with fraction of observed entries $\rho = 0.7$. We compared all NORST extensions and PETRELS. We set the algorithm parameters for NORST and extensions as mentioned before and used $K = 33$ to see how low the NORST error can go. For PETRELS we set `max_cycles` $= 1$, forgetting parameter $\lambda = 0.98$ as specified in the paper. We display the results in Table 4.5 (top). Notice that NORST-miss and its extensions are significantly faster than PETRELS. Also, the $\beta = 10, R = 1$ is the best of all the NORST extensions and is as good as PETRELS.

In our second set of experiments, we compared NORST (and a few extensions) with PETRELS and GROUSE for three settings of missing data. For GROUSE, we set maximum cycles as 1 as specified in the documentation and set the step size, $\eta = 0.1$ and the step-size is udpated according to [53]. The first was for missing generated from the Moving Object model [34, Model 6.19] with $s = 200$, and $b_0 = 0.05$. This translates to $\rho = 0.8$ fraction of observed entries. This is an example of a deterministic model on missing entries. We plot the subspace recovery error versus time for this case in Fig. 4.2(a) As can be seen, NORST-buffer (R=4) and NORST-sliding-window ($\beta = 10, R = 4$) have the best performance, followed by PETRELS, basic NORST, and then GROUSE. PETRELS is the slowest in terms of time taken. In Fig. 4.2(b), we plot the results for Bernoulli observed entries' set with $\rho = 0.9$. Here again, NORST-sliding has the best performance. Basic NORST is only slightly worse than PETRELS. As can be seen from the time taken (displayed in the legend), NORST and its extensions are much faster than PETRELS.

In Fig. 4.2(c), as suggested by an anonymous reviewer, we evaluate the same case but with the covariance matrix of $\boldsymbol{\ell}_t$ being time-varying. We generate the $\boldsymbol{a}_t$'s as described earlier but with



(a) Piecewise Constant (Noisy) (b) Piecewise Constant (Noise-Free)(c) Subspace change at each time

Figure 4.3: Subspace error versus time plot for changing subspaces. We plot the $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)})$ on the y-axis and the number of samples (t) on the x-axis. The entries are observed under Bernoulli model with $\rho = 0.9$. The computational time taken per sample (in milliseconds) is provided in the legend parenthesis. **(a) Piecewise constant subspace change and noise-sensitivity:** Observe that after the first subspace change, NORST-sliding adapts to subspace change using the least number of samples and is also $\approx$ 6x faster than PETRELS whereas GROUSE requires more samples than our approach and thus is unable to converge to the noise-level ($\approx 10^{-4}$); **(b) Piecewise Constant and noise-free:** All algorithms perform significantly better since the data is noise-free. We clip the y-axis at 10^{-10} for the sake of presentation but NORST and PETRELS attain a recovery error of 10^{-14}. **(c) Subspace changes a little at each time:** All algorithms are able to track the span of top-r singular vectors of $[\boldsymbol{P}_{(t-\alpha+1)}, \cdots, \boldsymbol{P}_{(t)}]$ to an accuracy of 10^{-4}. As explained, the subspace change at each time can be thought of as noise. GROUSE needs almost 2x number of samples to obtain the same accuracy as NORST while PETRELS is approximately 10x slower than both NORST and GROUSE.

$q_{t,i} = \sqrt{f} - \sqrt{f}(i-1)/2r - \lambda^-/2$ for $t = 2, 4, 6, \cdots$ and $q_{t,i} = \sqrt{f} - \sqrt{f}(i-1)/2r + \lambda^-/2$ for $t = 1, 3, 5, \cdots$ and $q_{t,r} = 1$. As can be seen all approaches still work in this case. PETRELS converges with the fewest samples but is almost $18x$ slower.

4.5.3 Changing Subspaces, Noisy and Noise-free Measurements

Piecewise constant subspace change, noisy and noise-free: We generate the changing subspaces using $\boldsymbol{P}_j = e^{\gamma_j \boldsymbol{B}_j} \boldsymbol{P}_{j-1}$ as done in [1] where γ_j controls the amount subspace change and $\boldsymbol{B}_j$'s are skew-symmetric matrices. We used the following parameters: $n = 1000$, $d = 10000$, $J = 6$, and the subspace changes after every 800 frames. The other parameters are $r = 30$, $\gamma_j = 100$ and the

matrices $\boldsymbol{B}_i$ are generated as $\boldsymbol{B}_i = (\tilde{\boldsymbol{B}}_i - \tilde{\boldsymbol{B}}_i{}')$ where the entries of $\tilde{\boldsymbol{B}}_i$ are generated independently from a standard normal distribution and $\boldsymbol{a}_t$'s are generated as in the fixed subspace case. For the missing entries supports, we consider the Bernoulli Model with $\rho = 0.9$. The noise $\boldsymbol{v}_t$'s are generated as i.i.d. Gaussian r.v.'s with $\sqrt{\lambda_v^+} = 3 \times 10^{-3}\sqrt{\lambda^-}$. The results are summarized in Fig. 4.3(a). For NORST we set $\alpha = 100$ and $K = 7$. We observe that all algorithms except GROUSE are able to attain final accuracy approximately equal to the noise-level, 10^{-3} within a short delay of the subspace change. We also observe that NORST-sliding-window adapts to subspace change using the fewest samples possible. Moreoever it is much faster than PETRELS.

In Fig. 4.3(b), we plot results for the above setting but with noise $\nu_t = 0$. In this case, the underlying subspace is recovered to accuracy lower than 10^{-12} by NORST and PETRELS but GROUSE only tracks to error 10^{-7}.

Subspace change at each time: Here we generate the data using the approach of [3]: $\boldsymbol{P}_{(1)}$ is generated by ortho-normalizing the columns of a i.i.d. Gaussian matrix and let $\boldsymbol{P}_{(t)} = e^{\gamma \boldsymbol{B}} \boldsymbol{P}_{(t-1)}$. We set $\gamma = 10^{-7}$. No extra noise $\boldsymbol{v}_t$ was added, i.e., $\boldsymbol{v}_t = 0$, in this experiment. We plot $\mathrm{SE}(\hat{\boldsymbol{P}}_{(t)}, \boldsymbol{P}_{(t)})$ in Fig. 4.3(c). Notice that, even without added noise $\boldsymbol{v}_t$, all algorithms are only able to track the subspaces to accuracy at most 10^{-3} in this case. The reason is, as explained earlier in Sec. 4.1.3, subspace change at each time can be interpreted as r dimensional piecewise constant subspace change plus noise.

4.5.4 Matrix Completion

In Table 4.6, we compare NORST-smoothing with existing MC solutions (for which code is available). This table displays the Monte-Carlo mean of the normalized Frobenius norm error along with time-taken per column displayed in parentheses. We compare two solvers for nuclear norm min (NNM) – (i) Singular Value Thresholding (SVT) with maximum iterations as 500, tolerance as 10^{-8}, $\delta = 1.2/\rho$, and $\tau = 5\sqrt{nd}$ and (ii) Inexact Augmented Lagrangian Multiplier (IALM) [30] with maximum iterations 500 and tolerance 10^{-16}. We also evaluate the projected Gradient Descent (projected-GD) algorithm of [11], this is a non-convex and hence fast approach, with the

Figure 4.4: Background Recovery under Moving Object Model missing entries ($\rho = 0.98$). We show the original, observed, and recovered frames at $t = \{980, 1000, 1020\}$. NORST and SVT are the only algorithms that work although NORST is almost 3 orders of magnitude faster than SVT. PETRELS(10) exhibits artifacts, while IALM and GROUSE do not capture the movements in the curtain. The time taken per sample for each algorithm is shown in parenthesis.

best sample complexity among non-convex approaches. This seems to be the only provable non-convex MC approach for which code is available. NORST-smoothing used $K = 33$ and $\alpha = 2r$.

The matrix $\boldsymbol{L}$ was generated as described in Sec. 4.5.2 for the "fixed" subspace rows and as in Sec. 4.5.3 (piecewise constant subspace change) for the "Noisy, Changing" subspace row. The observed entries set followed the Bernoulli model with different values of ρ in the different rows. The table demonstrates our discussion from Sec. 4.2.2. (1) In all cases, NORST-smoothing is much faster than both the solvers for convex MC (NNM), but is slower than the best non-convex MC approach (projected-GD). (2) NORST-smoothing is always better than projected-GD (implemented using default code, it is not easy to change the code parameters). It is nearly as good as IALM (one of the two solvers for NNM) when ρ is large, but is worse than IALM when ρ is small.

4.5.5 Real Video Data

Here we consider the task of Background Recovery for missing data. We use the `Meeting Room` video which is a benchmark dataset in Background Recovery. It contains 1755 images of size 64x80 in which a curtain is moving in the wind. Subsequently, there are 1209 frames in which a person

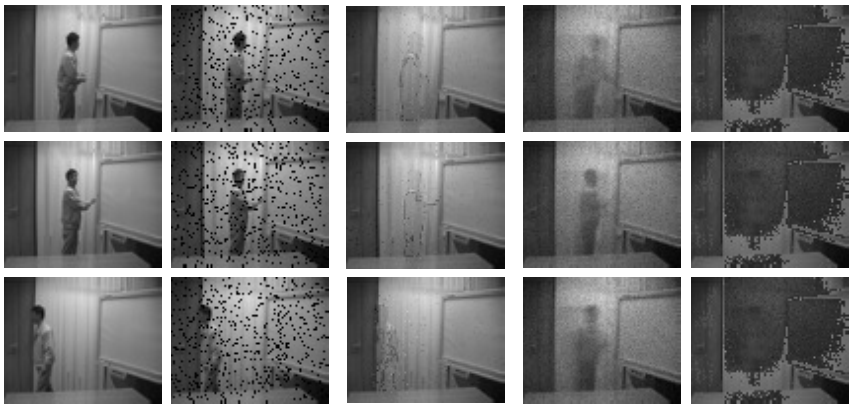


Figure 4.5: Background Recovery with foreground layer, and Bernoulli missing entries ($\rho = 0.9$). We show the original, observed and recovered frames at $t = 1755 + \{1059, 1078, 1157\}$. NORST-miss-rob exhibits artifacts, but is able to capture most of the background information, whereas, GRASTA-RMC and projected-GD fail to obtain meaningful estimates. The time taken per sample for each algorithm is shown in parenthesis.

walks into the room, writes on a blackboard, and exits the room. The first 1755 frames are used for ST-miss while the subsequent frames are used for RST-miss (since we can model the person as a sparse outlier [7]).

We generate the set of observed entries using the Bernoulli model with $\rho = 0.9$. In all experiments, we use the estimate of rank as $r = 30$. The parameters of NORST-miss are $\alpha = 60$, $K = 3$, and $\omega_{evals} = 2 \times 10^{-3}$. We noticed that PETRELS failed to retrieve the background with default parameters so we increased `max_cycles`= 10 and refer to this as PETRELS(10) in the sequel. Furthermore, we also ensured that the input data matrix has more columns than rows by transposing the matrix when necessary. All other algorithms are implemented as done in the previous experiments. We observed that NORST-miss and SVT provide a good estimate of the background and NORST is $\approx$ 150x faster. The relative Frobenius error is provided in the last row of Table. 4.6. Notice that, in this case, SVT outperforms IALM and NORST, but NORST is the fastest one. These results are averaged over 10 independent trials.

Moving Object Missing Entries: In our second video experiment, we generated the set of missing entries using the moving object model with $\rho = 0.98$. All algorithms are implemented as in

the previous experiment. Interestingly, even though we observe 98% of the entries, the performance of all algorithms degrade compared to the Bern(0.9). This is possibly because the support sets are highly correlated over time and thus the assumptions of other algorithms break down. The results are shown in Fig. 4.4. Observe that NORST-miss and SVT provide the best visual comparison and NORST-miss is faster than SVT by $\approx$ 400x. PETRELS(10) contains significant artifacts in the recovered background and IALM provides a *static* output in which the movements of the curtain are not discernible.

4.5.6 RST-miss and RMC

In this experiment, we consider the RST-miss problem, i.e., we generate data according to (4.4). We generate the low rank matrix, $\boldsymbol{L}$, as done in experiment 1 (single subspace). We generate the sparse matrix, $\boldsymbol{X}$ as follows: we use the Moving Object Model to generate the support sets such that $s/n = 0.05$ and $b_0 = 0.05$ which translates to $\rho_{\text{sparse}} = 0.05$ *fraction of sparse outliers*. The non-zero magnitudes of $\boldsymbol{X}$ are generated uniformly at random between $[x_{\min}, x_{\max}]$ with $x_{\min} = 10$ and $x_{\max} = 25$. We generated the support of observed entries using Bernoulli Model with probability $\rho_{\text{obs}} = 0.9$.

For initialization step of NORST-miss-robust (Algorithm 2), for the first $t_{\text{train}} = 400$ data samples, we set $(\boldsymbol{y}_t)_i = 10$ for all $i \in \mathcal{T}_t$. We do this to allow us to use AltProj [36], which is an RPCA solution, for obtaining the initial subspace estimate. The parameters for this step are set as 500 maximum iterations of AltProj, and tolerance 10^{-3}. The other algorithm parameters for NORST-miss-robust are $\alpha = 60$, $K = 33$, $\omega_{evals} = 7.8 \times 10^{-4}$, $\xi = x_{\min}/15$, and $\omega_{supp} = \boldsymbol{x}_{\min}/2 = 5$. We compare[6] GRASTA-RMC [22] and projected-GD [11]. For GRASTA-RMC we used the tolerance 10^{-8}, and `max_cycles`= 1. For projected-GD, we use the default tolerance 10^{-1} and max. iterations 70. The results are given in Table. 4.7. Observe that NORST-miss-robust obtains the best estimate among the RMC algorithms.

[6] we do not compare it with NNM based methods for which code is not available online

Real video data: In this experiment, we consider Background recovery applied on the second part of the dataset (last 1209 frames). In addition to the person who enters the room and writes on the board (sparse component), we generate missing entries from the Bernoulli model with $\rho = 0.9$. We initialize using AltProj with tolerance 10^{-2} and 100 iterations. We set $\omega_{supp,t} = 0.9\|\boldsymbol{y}_t\|/\sqrt{n}$ using the approach of [33]. The comparison results are provided in Fig. 4.5. Notice that both GRASTA-RMC and projected-GD fail to accurately recover the background. Although NORST-miss-robust exhibits certain artifacts around the edges of the sparse object, it is able to capture most of the information in the background.

4.6 Conclusions and Open Questions

This work studied the related problems of subspace tracking in missing data (ST-miss) and its robust version. We show that our proposed approaches are provably accurate under simple assumptions on only the observed data (in case of ST-miss), and on the observed data and initialization (in case of robust ST-miss). Thus, in both cases, the required assumptions are only on the algorithm inputs, making both results *complete guarantees*. Moreover, our guarantees show that our algorithms need near-optimal memory; are as fast as vanilla PCA; and can detect and track subspace changes quickly. We provided a detailed discussion of related work on (R)ST-miss, (R)MC, and streaming PCA with missing data, that help place our work in the context of what already exists. We also show that NORST-miss and NORST-miss-robust have good experimental performance as long as the fraction of missing entries is not too large.

Our guarantee for ST-miss is particularly interesting because it does not require slow subspace change and good initialization. Thus, it can be understood as a novel mini-batch and nearly memory-optimal solution for low-rank Matrix Completion, that works under similar assumptions to standard MC, but needs more numbers of observed entries in general (except in the regime of frequently changing subspaces).

While our approaches have near-optimal memory complexity, they are not streaming. This is because they use SVD and hence need multiple passes over short batches of stored data. A key open

question is whether a fully streaming provably correct solution can be developed without assuming the i.i.d. Bernoulli model on the set of missing entries? Two other important open questions include: (i) can the required number of observed entries be reduced (the limiting bound here is the bound on missing fractions per column); and (ii) in case of robust ST-miss, can the lower bound on outlier magnitudes be removed? Another question is whether we can use the tracked estimates for "control"? For example, can we use the current estimate of the subspace and of the true data vectors to decide how to sample the set of observed entries at the next time instant or later (in applications where one can design this set)?

4.7 Appendix A: Proof of Theorem 4.59 and Corollary 4.61

Much of the proof is a simplification of the proof for NORST for RST [33, Sections 4, 5 and Appendix A]. The analysis of subspace change detection is exactly the same as done there (see Lemma 4.8 and Appendix A of [33]) and hence we do not repeat it here. We explain the main ideas of the rest of the proof. To understand it simply, assume that $\hat{t}_j = t_j$, i.e, that t_j is known. We use the following simplification of [45, Remark 2.3] to analyze the subspace update step.

Corollary 4.68 (PCA in sparse data-dependent noise (Remark 2.3 of [45]))**.** *For $t = 1, \cdots, \alpha$, suppose that $\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{w}_t + \boldsymbol{v}_t$ with $\boldsymbol{w}_t = \boldsymbol{I}_{\mathcal{T}_t} \boldsymbol{M}_{s,t} \boldsymbol{\ell}_t$ being sparse noise with support $\mathcal{T}_t$, and $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t$ where $\boldsymbol{P}$ is a $n \times r$ basis matrix and $\boldsymbol{a}_t$'s satisfy the statistical right-incoherence assumption given in the theorem. Let $\hat{\boldsymbol{P}}$ be the matrix of top r eigenvectors of $\frac{1}{\alpha}\sum_t \boldsymbol{y}_t \boldsymbol{y}_t'$. Assume that $\max_t \|\boldsymbol{M}_{s,t}\boldsymbol{P}\| \le q$ for a $q \le 3$ and that the fraction of non-zeros in any row of the matrix $[\boldsymbol{w}_1, \cdots, \boldsymbol{w}_\alpha]$ is bounded by b. Pick an $\epsilon_{\text{SE}} > 0$. If $6\sqrt{b}qf + \lambda_v^+/\lambda^- < 0.4\epsilon_{\text{SE}}$ and if $\alpha \ge \alpha^*$ where*

$$\alpha^* := C \max\left(\frac{q^2 f^2}{\epsilon_{\text{SE}}^2} r \log n, \frac{\frac{\lambda_v^+}{\lambda^-} f}{\epsilon_{\text{SE}}^2} r_v \log n \right),$$

then, w.p. at least $1 - 10n^{-10}$, $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \le \epsilon_{\text{SE}}$.

First assume that $\boldsymbol{v}_t = 0$ so that $\lambda_v^+ = 0$ and $r_v = 0$. Also, let $b_0 := \frac{c_2}{f^2}$ denote the bound on max-miss-frac-row$_\alpha$ assumed in the Theorem.

Using the expression for $\hat{\boldsymbol{z}}_t$ given in (4.3), it is easy to see that the error $\boldsymbol{e}_t := \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t$ satisfies

$$\boldsymbol{e}_t = \boldsymbol{I}_{\mathcal{T}_t} \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t} \right)^{-1} \boldsymbol{I}_{\mathcal{T}_t}{}' \boldsymbol{\Psi} \boldsymbol{\ell}_t, \tag{4.6}$$

with $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)} \hat{\boldsymbol{P}}_{(t-1)}{}'$. For the first α frames, $\hat{\boldsymbol{P}}_{(t-1)} = \boldsymbol{0}$ (zero initialization) and so, during this time, $\boldsymbol{\Psi} = \boldsymbol{I}$.

We need to analyze the subspace update steps one at a time. We first explain the main ideas of how we do this for $j > 0$ and then explain the different approach needed for $j = 0$ (because of zero initialization). Consider a general $j > 0$ and $k = 1$, i.e., the first subspace update interval of estimating $\boldsymbol{P}_j$. In this interval $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1} \hat{\boldsymbol{P}}_{j-1}{}'$ and recall that $\hat{\boldsymbol{P}}_{j-1} = \hat{\boldsymbol{P}}_{j-1,K}$. Assume that $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \varepsilon$.

Using the μ-incoherence assumption, the bound on max-miss-frac-col $:= \max_t |\mathcal{T}_t|/n$, $\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \varepsilon$ (assumed above), and recalling from the algorithm that $\hat{\boldsymbol{P}}_j := \hat{\boldsymbol{P}}_{j,K}$, it is not hard to see that[7], for all j,

$\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j) \leq \mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) + \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$

$\| \boldsymbol{I}_{\mathcal{T}_t}{}' \boldsymbol{P}_j \| \leq 0.1,$

$\| \boldsymbol{I}_{\mathcal{T}_t}{}' \hat{\boldsymbol{P}}_{j,k} \| \leq \mathrm{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) + 0.1,$

$\| \boldsymbol{I}_{\mathcal{T}_t}{}' \hat{\boldsymbol{P}}_{j-1} \| \leq \varepsilon + 0.1,$

$\| \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t} \right)^{-1} \| \leq 1.2$ with $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j,k} \hat{\boldsymbol{P}}_{j,k}{}'$.

Next we apply Corollary 4.68 to the $\hat{\boldsymbol{\ell}}_t$'s. This bounds the subspace recovery error for PCA in sparse data-dependent noise. Since $\hat{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t + \boldsymbol{e}_t$ with $\boldsymbol{e}_t$ satisfying (4.6), clearly, $\boldsymbol{e}_t$ is sparse and dependent on $\boldsymbol{\ell}_t$ (true data). In the notation of Corollary 4.68, $\boldsymbol{y}_t \equiv \hat{\boldsymbol{\ell}}_t$, $\boldsymbol{w}_t \equiv \boldsymbol{e}_t$, $\boldsymbol{v}_t = 0$, $\mathcal{T}_t \equiv \mathcal{T}_t$, $\boldsymbol{\ell}_t \equiv \boldsymbol{\ell}_t$, $\hat{\boldsymbol{P}} = \hat{\boldsymbol{P}}_{j,1}$, $\boldsymbol{P} = \boldsymbol{P}_j$, and $\boldsymbol{M}_{s,t} = -\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t} \right)^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}{}'$ with $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1} \hat{\boldsymbol{P}}_{j-1}{}'$. Thus, using bounds from above, $\| \boldsymbol{M}_{s,t} \boldsymbol{P} \| = \| \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t} \right)^{-1} \boldsymbol{I}_{\mathcal{T}_t}{}' \boldsymbol{\Psi} \boldsymbol{P}_j \| \leq \| \left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}' \boldsymbol{\Psi}_{\mathcal{T}_t} \right)^{-1} \| \| \boldsymbol{I}_{\mathcal{T}_t}{}' \| \| \boldsymbol{\Psi} \boldsymbol{P}_j \| \leq 1.2(\varepsilon + \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)) \equiv q$. Also, $b \equiv b_0 := \frac{c_2}{f^2}$ ($c_2 = 0.001$) which is the upper bound on max-miss-frac-row$_\alpha$ and so $1.2(\varepsilon + \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)) < 1.2(0.01 + \Delta) < 1.3$ since $\Delta \leq 1$. Thus $q < 3$. We apply Corollary 4.68 with $\varepsilon_{\mathrm{SE}} = q/4$. All its assumptions hold because we have set $\alpha = C f^2 r \log n$ and because we

[7] Use the RIP-denseness lemma from [41] and some simple linear algebra which includes a triangle inequality type bound for SE. See the proof of item 1 of Lemma 4.7 of [33]

have let $b_0 = 0.001/f^2$ and so the required condition $3\sqrt{b}fq \leq 0.9\varepsilon_{\text{SE}}/(1+\varepsilon_{\text{SE}})$ holds. We conclude that $\text{SE}(\hat{\boldsymbol{P}}_{j,1}, \boldsymbol{P}_j) \leq 1.2(0.01+\Delta)/4 = 0.3(0.01+\Delta) := q_1$ whp.

The above is the base case for an induction proof. For the k-th subspace update interval, with $k > 1$, we use a similar approach to the one above. Assume that at the end of the $(k-1)$-th interval, we have $\text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j) \leq q_{k-1} := 0.3^{k-1}(0.01+\Delta)$ whp In this interval, $\|\boldsymbol{M}_{s,t}\boldsymbol{P}\| \leq 1.2\|\boldsymbol{I}_{\mathcal{T}_t}{}'\|\|\boldsymbol{\Psi}\boldsymbol{P}_j\| \leq 1.2\text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j) \leq q_{k-1} = 1.2 \cdot 0.3^{k-1}(0.01+\Delta) \equiv q$. We apply Corollary 4.68 with $\varepsilon_{\text{SE}} = q/4$. This is possible because we have let $b_0 = 0.001/f^2$ and so the required condition $3\sqrt{b}fq \leq 0.9(q/4)/(1+q/4)$ holds. Thus we can conclude that $\text{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) \leq 1.2 \cdot 0.3^{k-1}(0.01+\Delta)/4 = 0.3^k(0.01+\Delta) := q_k$ whp Thus starting from $\text{SE}(\hat{\boldsymbol{P}}_{j,k-1}, \boldsymbol{P}_j) \leq q_{k-1} := 0.3^{k-1}(0.01+\Delta)$, we have shown that $\text{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) \leq 0.3^k(0.01+\Delta)$. This along with the base case, implies that we get $\text{SE}(\hat{\boldsymbol{P}}_{j,k}, \boldsymbol{P}_j) \leq 0.3^k(0.01+\Delta)$ for all $k = 1, 2, \ldots, K$. The choice of K thus implies that $\text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) = \text{SE}(\hat{\boldsymbol{P}}_{j,K}, \boldsymbol{P}_j) \leq \varepsilon$.

For $j = 0$ and first subspace interval ($k = 1$), the proof is a little different from that of [33] summarized above. The reason is we use zero initialization. Thus, in the first update interval for estimating $\boldsymbol{P}_0$, we have $\boldsymbol{\Psi} = \boldsymbol{I}$. In applying the PCA in sparse data-dependent noise result of Corollary 4.68, everything is the same as above except that we now have $\boldsymbol{M}_{s,t} = \boldsymbol{I}_{\mathcal{T}_t}{}'$ and so we get $\|\boldsymbol{M}_{s,t}\boldsymbol{P}\| \leq 0.1$. Thus in this case $q = 0.1 < 3$. The rest of the argument is the same as above.

Now consider $\boldsymbol{v}_t \neq 0$. Recall that the effective noise dimension of $\boldsymbol{v}_t$ is $r_v = \max_t \|\boldsymbol{v}_t\|^2/\lambda_v^+$ where $\lambda_v^+ = \|\mathbb{E}[\boldsymbol{v}_t\boldsymbol{v}_t{}']\|$. Furthermore, recall that $\epsilon_{\text{SE}} = q/4$. Thus, in order to obtain ε-accurate estimate in the noisy case, we will require that $\alpha = \mathcal{O}\left(\max\left(f^2 r \log n, \frac{\frac{\lambda_v^+}{\lambda^-} f r_v \log n}{\epsilon_{\text{SE}}^2}\right)\right)$. Thus, we set $\epsilon_{\text{SE}} = c\sqrt{\lambda_v^+/\lambda^-}$ to ensure that the dependence on ε is on logarithmic (that comes from expression for K).

The above provides the basic proof idea in a condensed fashion but does not define events that one conditions on for each interval, and also does not specify the probabilities. For all these details, please refer to Sections IV and V and Appendix A of [33].

4.8 Appendix B: Proof of Corollary 4.66

This proof is also similar to that of NORST for RST [33]. The difference is NORST-miss-robust uses noisy modified CS [44, 52] to replace l_1 min. In comparison to the ST-miss proof summarized above, we also have to deal with arbitrary outliers, in addition to missing data. This uses requires sparse support recovery with partial subspace knowledge. This is solved by modified-CS followed by thresholding based support recovery. To bound the modified-CS error, we apply Lemma 2.7 of [52]. This uses a bound on $\|\boldsymbol{b}_t\| = \|\boldsymbol{\Psi}\boldsymbol{\ell}_t\|$ and a bound on the (max-miss-frac-col $\cdot$ n + 2max-outlier-frac-col $\cdot$ n)-RIC of $\boldsymbol{\Psi}$. We obtain both these exactly as done for [33, Lemma 4.7, Item 1]: the former uses the slow subspace change bound and the boundedness of $\boldsymbol{a}_t$; for the latter we use the μ-incoherence/denseness assumption and bounds on max-outlier-frac-col and max-miss-frac-col, and the RIP-denseness lemma of [41]. With the modified-CS error bound, we prove exact support recovery using the lower bound on $x_{\min}$. algorithm parameter values of ξ and ω_{supp}.

4.9 References

[1] Balzano, L., Chi, Y., and Lu, Y. M. Streaming pca and subspace tracking: The missing data case. *Proceedings of IEEE* (2018).

[2] Balzano, L., Recht, B., and Nowak, R. High-dimensional matched subspace detection when data are missing. In *ISIT* (2010), pp. 1638–1642.

[3] Balzano, L., Recht, B., and Nowak, R. Online Identification and Tracking of Subspaces from Highly Incomplete Information. In *Allerton Conf. Comm., Control, Comput.* (2010).

[4] Balzano, L., and Wright, S. Local convergence of an algorithm for subspace identification from partial data. *Found. Comput. Math. 15*, 5 (2015).

[5] Bhojanapalli, S., and Jain, P. Universal matrix completion. In *International Conference on Machine Learning* (2014), pp. 1881–1889.

[6] Candes, E. The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris Serie I* (2008).

[7] Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *J. ACM 58*, 3 (2011).

[8] Candes, E. J., and Recht, B. Exact matrix completion via convex optimization. *Found. of Comput. Math*, 9 (2008), 717–772.

[9] Chen, Y., Bhojanapalli, S., Sanghavi, S., and Ward, R. Coherent matrix completion. In *International Conference on Machine Learning* (2014), pp. 674–682.

[10] Chen, Y., Jalali, A., Sanghavi, S., and Caramanis, C. Low-rank matrix recovery from errors and erasures. *IEEE Trans. Inform. Theory 59(7)* (2013), 4324–4337.

[11] Cherapanamjeri, Y., Gupta, K., and Jain, P. Nearly-optimal robust matrix completion. *ICML* (2016).

[12] Chi, Y., Eldar, Y. C., and Calderbank, R. Petrels: Parallel subspace estimation and tracking by recursive least squares from partial observations. *IEEE Transactions on Signal Processing* (December 2013).

[13] Chouvardas, S., Kopsinis, Y., and Theodoridis, S. Robust subspace tracking with missing entries: a set–theoretic approach. *IEEE Trans. Sig. Proc. 63*, 19 (2015), 5060–5070.

[14] Comon, P., and Golub, G. H. Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE 78*, 8 (1990), 1327–1343.

[15] Fazel, M. Matrix rank minimization with applications. *PhD thesis, Stanford Univ* (2002).

[16] Foucart, S., Needell, D., Plan, Y., and Wootters, M. De-biasing low-rank projection for matrix completion. In *Wavelets and Sparsity XVII* (2017), vol. 10394, International Society for Optics and Photonics, p. 1039417.

[17] Ge, R., Jin, C., and Zheng, Y. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. *arXiv preprint arXiv:1704.00708* (2017).

[18] Ge, R., Lee, J. D., and Ma, T. Matrix completion has no spurious local minimum. In *NIPS* (2016), pp. 2973–2981.

[19] Gonen, A., Rosenbaum, D., Eldar, Y. C., and Shalev-Shwartz, S. Subspace learning with partial information. *The Journal of Machine Learning Research 17*, 1 (2016), 1821–1841.

[20] Gonen, A., Rosenbaum, D., Eldar, Y. C., and Shalev-Shwartz, S. Subspace learning with partial information. *Journal of Machine Learning Research 17*, 52 (2016), 1–21.

[21] Hardt, M., and Wootters, M. Fast matrix completion without the condition number. In *COLT* (2014).

[22] He, J., Balzano, L., and Szlam, A. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *IEEE Conf. on Comp. Vis. Pat. Rec. (CVPR)* (2012).

[23] Jain, P., and Netrapalli, P. Fast exact matrix completion with finite samples. In *Conference on Learning Theory* (2015), pp. 1007–1034.

[24] Jiang, X., Zhong, Z., Liu, X., and So, H. C. Robust matrix completion via alternating projection. *IEEE Signal Processing Letters 24*, 5 (2017), 579–583.

[25] Jin, C., Kakade, S. M., and Netrapalli, P. Provable efficient online matrix completion via non-convex stochastic gradient descent. In *NIPS* (2016), pp. 4520–4528.

[26] Keshavan, R., Montanari, A., and Oh, S. Matrix completion from a few entries. *IEEE Trans. Info. Th. 56*, 6 (2010), 2980–2998.

[27] Krishnamurthy, A., and Singh, A. Low-rank matrix and tensor completion via adaptive sampling. In *NIPS* (2013), pp. 836–844.

[28] Lai, M., and Varghese, A. On convergence of the alternating projection method for matrix completion and sparse recovery problems. *arXiv preprint arXiv:1711.02151* (2017).

[29] Leeb, W., and Romanov, E. Optimal spectral shrinkage and pca with heteroscedastic noise. *arXiv:1811.02201* (2018).

[30] Lin, Z., Chen, M., and Ma, Y. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055* (2010).

[31] Mansour, H., and Jiang, X. A robust online subspace estimation and tracking algorithm. In *ICASSP* (2015), pp. 4065–4069.

[32] Mitliagkas, I., Caramanis, C., and Jain, P. Streaming pca with many missing entries. *Preprint* (2014).

[33] Narayanamurthy, P., and Vaswani, N. Nearly optimal robust subspace tracking. In *International Conference on Machine Learning* (2018), pp. 3701–3709.

[34] Narayanamurthy, P., and Vaswani, N. Provable dynamic robust pca or robust subspace tracking. *IEEE Transactions on Information Theory 65*, 3 (2019), 1547–1577.

[35] Netrapalli, P., Jain, P., and Sanghavi, S. Low-rank matrix completion using alternating minimization. In *STOC* (2013).

[36] Netrapalli, P., Niranjan, U. N., Sanghavi, S., Anandkumar, A., and Jain, P. Non-convex robust pca. In *NIPS* (2014).

[37] Oja, E. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology 15*, 3 (1982), 267–273.

[38] Ongie, G., Hong, D., Zhang, D., and Balzano, L. Enhanced online subspace estimation via adaptive sensing. In *Asilomar* (2018).

[39] Paige, C. C., and Saunders, M. A. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS) 8*, 1 (1982), 43–71.

[40] Qiu, C., and Vaswani, N. Real-time robust principal components' pursuit. In *Allerton Conf. on Communication, Control, and Computing* (2010).

[41] Qiu, C., Vaswani, N., Lois, B., and Hogben, L. Recursive robust pca or recursive sparse recovery in large but structured noise. *IEEE Trans. Info. Th.* (August 2014), 5007–5039.

[42] Recht, B. A simpler approach to matrix completion. *Journal of Machine Learning Research 12*, Dec (2011), 3413–3430.

[43] Sun, R., and Luo, Z.-Q. Guaranteed matrix completion via non-convex factorization. *IEEE Trans. Info. Th. 62*, 11 (2016), 6535–6579.

[44] Vaswani, N., and Lu, W. Modified-CS: Modifying compressive sensing for problems with partially known support. *IEEE Trans. Signal Processing* (September 2010).

[45] Vaswani, N., and Narayanamurthy, P. Pca in sparse data-dependent noise. In *ISIT* (2018), pp. 641–645.

[46] Vaswani, N., and Narayanamurthy, P. Static and dynamic robust pca and matrix completion: A review. *Proceedings of the IEEE 106*, 8 (2018), 1359–1379.

[47] Wang, C., Eldar, Y. C., and Lu, Y. M. Subspace estimation from incomplete observations: A high-dimensional analysis. *JSTSP* (2018).

[48] Yang, B. Projection approximation subspace tracking. *IEEE Trans. Sig. Proc.* (1995), 95–107.

[49] Yang, B. Asymptotic convergence analysis of the projection approximation subspace tracking algorithms. *Signal Processing 50* (1996), 123–136.

[50] Yi, X., Park, D., Chen, Y., and Caramanis, C. Fast algorithms for robust pca via gradient descent. In *NIPS* (2016).

[51] Zhan, J., Lois, B., Guo, H., and Vaswani, N. Online (and Offline) Robust PCA: Novel Algorithms and Performance Guarantees. In *Intnl. Conf. Artif. Intell. Stat. (AISTATS)* (2016).

[52] Zhan, J., and Vaswani, N. Time invariant error bounds for modified-CS based sparse signal sequence recovery. *IEEE Trans. Info. Th. 61*, 3 (2015), 1389–1409.

[53] Zhang, D., and Balzano, L. Global convergence of a grassmannian gradient descent algorithm for subspace estimation. In *AISTATS* (2016).

Table 4.1: List of Symbols and Assumptions used in Theorem 4.59.

Observations: $\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{\ell}_t) + \boldsymbol{v}_t = \mathcal{P}_{\Omega_t}(\boldsymbol{P}_{(t)}\boldsymbol{a}_t) + \boldsymbol{v}_t$	
Symbol	Meaning
t_j	j-th subspace change time
for $t \in [t_j, t_{j+1})$, $\boldsymbol{P}_{(t)} = \boldsymbol{P}_j$	Subspace at time t
$\mathcal{P}_{\Omega_t}(\cdot)$	mask to select elements present in Ω_t
Ω_t	Support set of observed entries
$\mathcal{T}_t (= \Omega_t^c)$	Support set of missing entries
$\boldsymbol{v}_t$	dense, unstructured noise
Principal Subspace Coefficients ($\boldsymbol{a}_t$'s)	
element-wise bounded, zero mean, mutually independent with identical and diagonal covariance $\mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t{}'] := \boldsymbol{\Lambda}$	
$\lambda_{\max}(\boldsymbol{\Lambda}) = \lambda^+ (\lambda_{\min}(\boldsymbol{\Lambda}) = \lambda^-)$	Max. (min.) eigenvalue of $\boldsymbol{\Lambda}$
$f := \lambda^+ / \lambda^-$	Condition Number of $\boldsymbol{\Lambda}$
Missing Entries ($\boldsymbol{z}_t = -\boldsymbol{I}_{\mathcal{T}_t}{}' \boldsymbol{\ell}_t$)	
Row-Missing Entries	max-miss-frac-row$_\alpha \leq 0.001/f^2$
Column-Missing Entries	max-miss-frac-col $\leq 0.01/\mu r$
Intervals for j-th subspace change and tracking	
$\hat{t}_j$	j-th subspace change detection time
$\hat{t}_{j,fin}$	j-th subspace update complete
$\mathcal{J}_0 := [t_j, \hat{t}_j)$	interval before j-th subspace change detected
$\mathcal{J}_k := [\hat{t}_j + (k-1)\alpha, \hat{t}_j + k\alpha)$	k-th subspace update interval
$\mathcal{J}_{K+1} := [\hat{t}_j + K\alpha, t_{j+1})$	subspace update completed
Algorithm 8 Parameters	
α	# frames used for subspace update
K	# of subspace updates for each j
ω_{evals}	threshold for subspace detection

Table 4.2: Comparing guarantees for ST-miss. We treat the condition number and incoherence parameters as constants for this discussion.

Algorithm	**Tracking delay**	**Memory**	**Time**	**Allows changing subspaces?**	**Observed Entries**
GROUSE [3]	Partial Guarantee	$\mathcal{O}(nr)$	$\mathcal{O}(nd\rho r^2)$	No	i.i.d. Bernoulli(ρ)
PETRELS [47]	Partial Guarantee	$\mathcal{O}(nr^2)$	$\mathcal{O}(nd\rho r^2)$	No	i.i.d. Bernoulli(ρ)
MBPM [32, 20]	$d \gtrsim \frac{r^2 \log^2 n \log(1/\varepsilon)}{\rho^2}$	$\mathcal{O}(nr)$	$\mathcal{O}(ndr)$	No	i.i.d. Bernoulli(ρ)
NORST-miss (this work)	$d \geq r \log n \log(1/\epsilon)$	$\mathcal{O}\left(nr \log n \log \frac{1}{\epsilon}\right)$	$\mathcal{O}\left(ndr \log \frac{1}{\epsilon}\right)$	Yes	bounded fraction, c/r per column, c per row

Table 4.3: Comparing MC guarantees. Recall $r_L := \text{rank}(\boldsymbol{L}) \leq rJ$. In the regime when the subspace changes frequently so that J equals its upper bound and $r_L \approx d/\log^2 n$, NORST-miss is better than the non-convex methods (AltMin, projGD, SGD) and only slightly worse than the convex ones (NNM). In general, the sample complexity for NORST-miss is significantly worse than all the MC methods.

Algorithm	**Sample complexity (# obs. entries, m)**	**Memory**	**Time**	**Observed entries**
nuc norm min (NNM) [15]	$\Omega(nr_L \log^2 n)$	$\mathcal{O}(nd)$	$\mathcal{O}(n^3/\sqrt{\epsilon})$	i.i.d. Bernoulli (m/nd)
weighted NNM [9]	$\Omega(nr_L \log^2 n)$	$\mathcal{O}(nd)$	$\mathcal{O}(n^3/\sqrt{\epsilon})$	indep. Bernoulli
AltMin [26]	$\Omega(nr_L^{4.5} \log \frac{1}{\epsilon})$	$\mathcal{O}(nd)$	$\mathcal{O}(nr_L \log \frac{1}{\epsilon})$	i.i.d. Bernoulli (m/nd)
projected-GD [11]	$\Omega(nr_L^2 \log^2 n)$	$\mathcal{O}(nd)$	$\mathcal{O}(nr_L^3 \log^2 n \log \frac{1}{\epsilon})$	i.i.d. Bernoulli (m/nd)
online SGD [25]	$\Omega\left(nr_L^2 \log n \left(r_L + \log \frac{1}{\epsilon}\right)\right)$	$\mathcal{O}(nd)$	$\mathcal{O}\left(nr_L^4 \log n \log \frac{1}{\epsilon}\right)$	i.i.d. Bernoulli (m/nd)
NORST-miss (this work)	$\boldsymbol{\Omega((1-\frac{c}{r})nd)}$	$\mathcal{O}\left(nr \log n \log \frac{1}{\epsilon}\right)$	$\mathcal{O}\left(ndr \log \frac{1}{\epsilon}\right)$	$\leq c \cdot d$ **per row** $\leq (1-\frac{c}{r}) \cdot n$ **per column**
Sample-Efficient NORST-miss (this work)	$\Omega(nr_L \log^2 n \log r)$	$\mathcal{O}\left(nr \log n \log \frac{1}{\epsilon}\right)$	$\mathcal{O}\left(ndr \log \frac{1}{\epsilon}\right)$	i.i.d. Bernoulli(ρ_t) where, $\rho_t = 1 - c/r$ for $t \in [t_j, t_j + (K+2)\alpha)$ $\rho_t = r \log^2 n \log r/nd$ other times

Note: Here, $f(n) = \Omega(g(n))$ implies that there exists a $G > 0$ and an $n_0 > 0$ s.t for all $n > n_0$, $|f(n)| \geq G \cdot |g(n)|$

Table 4.4: Comparing robust MC guarantees. We treat the condition number and incoherence parameters as constants for this table.

Algorithm	**Sample complexity**	**Memory**	**Time**	**Observed entries**	**Outliers**
NNM [15]	$\Omega(nd)$	$\mathcal{O}(nd)$	$\mathcal{O}(n^3/\sqrt{\epsilon})$	i.i.d. Bernoulli (c)	i.i.d. Bernoulli (c)
Projected GD [11]	$\Omega(nr^2 \log^2 n)$	$\mathcal{O}(nd)$	$\Omega(nr^3 \log^2 n \log^2(1/\epsilon))$	i.i.d. Bernoulli (m/nd)	bounded fraction ($\mathcal{O}(1/r)$ per row and col)
NORST-miss-rob (this work)	$\Omega(nd(1-1/r))$	$\mathcal{O}(nr \log n \log(1/\epsilon))$	$\mathcal{O}(ndr \log(1/\epsilon))$	bounded frac $\mathcal{O}(1/r)$ per row, $\mathcal{O}(1)$ per col	bounded frac. $\mathcal{O}(1/r)$ per row, $\mathcal{O}(1)$ per col
				Extra assumptions: Slow subspace change and lower bound on outlier magnitude	

Algorithm 8 NORST-miss.

Input: $\boldsymbol{y}_t$, $\mathcal{T}_t$ **Output**: $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}_{(t)}$ **Parameters:** r, $K = C\log(1/\varepsilon)$, $\alpha = Cf^2 r \log n$, $\omega_{evals} = 2\varepsilon^2\lambda^+$
$\hat{\boldsymbol{P}}_0 \leftarrow \boldsymbol{0}_{n\times r}$, $j \leftarrow 1$, $k \leftarrow 1$
phase $\leftarrow$ update; $\hat{t}_0 \leftarrow 0$; $\hat{t}_{-1,fin} = 0$
for $t > 0$ **do**
  $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$; $\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$;
  $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\tilde{\boldsymbol{y}}_t$.
  if phase = update **then**
    if $t = \hat{t}_j + u\alpha - 1$ for $u = 1,\ 2,\ \cdots$, **then**
      $\hat{\boldsymbol{P}}_{j,k} \leftarrow r$-SVD$[\hat{\boldsymbol{L}}_{t;\alpha}]$, $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{j,k}$, $k \leftarrow k+1$.
    else
      $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$
    end if
    if $t = \hat{t}_j + K\alpha - 1$ **then**
      $\hat{t}_{j,fin} \leftarrow t$, $\hat{\boldsymbol{P}}_j \leftarrow \hat{\boldsymbol{P}}_{(t)}$
      $k \leftarrow 1$, $j \leftarrow j+1$, phase $\leftarrow$ detect.
    end if
  end if
  if phase = detect and $t = \hat{t}_{j-1,fin} + u\alpha$ **then**
    $\boldsymbol{\Phi} \leftarrow (\boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}')$, $\boldsymbol{B} \leftarrow \boldsymbol{\Phi}\hat{\boldsymbol{L}}_{t,\alpha}$
    if $\lambda_{\max}(\boldsymbol{B}\boldsymbol{B}') \geq \alpha\omega_{evals}$ **then**
      phase $\leftarrow$ update, $\hat{t}_j \leftarrow t$,
    end if
  end if
end for
Smoothing mode: At $t = \hat{t}_j + K\alpha$ **for** $t \in [\hat{t}_{j-1} + K\alpha, \hat{t}_j + K\alpha - 1]$
$\hat{\boldsymbol{P}}^{\text{smooth}}_{(t)} \leftarrow \text{basis}([\hat{\boldsymbol{P}}_{j-1}, \hat{\boldsymbol{P}}_j])$
$\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}^{\text{smooth}}_{(t)}\hat{\boldsymbol{P}}^{\text{smooth}\prime}_{(t)}$
$\hat{\boldsymbol{\ell}}^{\text{smooth}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{y}_t$

Algorithm 9 NORST-miss-robust. Obtain $\hat{\boldsymbol{P}}_0$ by $C \log r$ iterations of AltProj applied to $\boldsymbol{Y}_{[1;t_{\text{train}}]}$ with $t_{\text{train}} = Cr$ and with setting $(\boldsymbol{y}_t)_{\mathcal{T}_t} = 10$ (or any large nonzero value) for all $t = 1, 2, \ldots, t_{\text{train}}$.

Input: $\boldsymbol{y}_t$, $\mathcal{T}_t$ **Output**: $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}_{(t)}$
Extra Parameters: $\omega_{supp} \leftarrow x_{\min}/2$, $\xi \leftarrow x_{\min}/15$
$\hat{\boldsymbol{P}}_0 \leftarrow$ obtain as given in the caption;
$j \leftarrow 1$, $k \leftarrow 1$, phase $\leftarrow$ update; $\hat{t}_0 \leftarrow t_{\text{train}}$;
for $t > t_{\text{train}}$ **do**
  $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$; $\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$;
  $\hat{\boldsymbol{x}}_{t,cs} \leftarrow \arg\min_{\boldsymbol{x}} \|(\boldsymbol{x})_{\mathcal{T}_t{}^c}\|_1$ s.t $\|\tilde{\boldsymbol{y}}_t - \boldsymbol{\Psi}\boldsymbol{x}\| \le \xi$.
  $\hat{\mathcal{T}}_t \leftarrow \mathcal{T}_t \cup \leftarrow \{i : |\hat{\boldsymbol{x}}_{t,cs}| > \omega_{supp}\}$
  $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\tilde{\boldsymbol{y}}_t$
  Lines $9 - 27$ of Algorithm 8
end for
Offline (RMC solution): line 25 of Algorithm 8.

Table 4.5: (top) Number of samples (frames) required by NORST and its heuristic extensions, and PETRELS to attain $\approx 10^{-16}$ accuracy. The observed entries are drawn from a i.i.d. Bernoulli model with $\rho = 0.7$ fraction of observed entries. Notice that NORST-buffer(4) and NORST-sliding-window ($\beta = 10, R = 1$) converges at the same rate as PETRELS and the time is also comparable. The other variants require more samples to obtain the same error but are faster compared to PETRELS. (bottom) Evaluation of Sample Efficient NORST with $\rho_1 = 0.9$ and $\rho_2 = 0.15$.

Algorithm	NORST	NORST-buffer				NORST-sliding-window and buffer		PETRELS
Parameter R, β		$R = 1$	$R = 2$	$R = 3$	$R = 4$	$\beta = 1$, $R = 0$	$\beta = 10$, $R = 1$	
Time taken (ms)	1.9	10.8	18.6	27.5	34.5	16	35	33
Number of samples	3540	2580	2100	2050	1950	2400	1740	1740

Algorithm	NORST-miss (6)	NORST-samp-eff (1)	PETRELS (15)	GROUSE (2)
Average Error	0.04	0.04	0.02	0.13

Table 4.6: Comparison of $\|\boldsymbol{L} - \hat{\boldsymbol{L}}\|_F / \|\boldsymbol{L}\|_F$ for MC. We report the time taken per sample in milliseconds in parenthesis. Thus the table format is Error (computational time per sample). The first three rows are for the fixed subspace model. The fourth row contains results for time-varying subspace and with noise of standard deviation $0.003\sqrt{\lambda^-}$ added. The last row reports Background Video Recovery results (for the curtain video shown in Fig. 4.4 when missing entries are Bernoulli with $\rho = 0.9$.

Subspace model	NORST-smoothing	nuclear norm min (NNM) solvers		projected-GD
		IALM	SVT	
Fixed (Bern, $\rho = 0.9$)	1.26×10^{-15} (10)	1.43×10^{-12} (150)	7.32×10^{-7} (164)	0.98 (1)
Fixed (Bern, $\rho = 0.3$)	3.5×10^{-6} (11)	5.89×10^{-13} (72)	–	0.98 (9)
Noisy, Changing (Bern, $\rho = 0.9$)	3.1×10^{-4} (3.5)	3.47×10^{-4} (717)	2.7×10^{-3} (256)	0.97 (2)
Video Data	0.0074 (83.7)	0.0891 (57.5)	0.0034 (6177)	–

Table 4.7: Comparing recovery error for Robust MC methods. Missing entries were Bernoulli with $\rho = 0.9$, and the outliers were sparse Moving Objects with $\rho_{\text{sparse}} = 0.95$. The time taken per sample is shown in parentheses.

NORST-miss-rob	GRASTA-RMC	projected-GD
0.0832 (3)	0.1431 (2.9)	0.5699 (2)

CHAPTER 5. FEDERATED OVER-AIR SUBSPACE TRACKING FROM INCOMPLETE AND CORRUPTED DATA

Praneeth Narayanamurthy, Namrata Vaswani, and Aditya Ramamoorthy

Dept. of Electrical and Computer Engineering, Iowa State University, Ames, IA, 50010

Modified from a manuscript under review in *IEEE Transactions on Signal Processing*

Abstract

Subspace tracking (ST) with missing data (ST-miss) or outliers (Robust ST) or both (Robust ST-miss) has been extensively studied in the last many years. This work provides a new simple algorithm and guarantee for both ST with missing data (ST-miss) and RST-miss. Unlike past work on this topic, the algorithm is much simpler (uses fewer parameters) and the guarantee does not make the artificial assumption of piecewise constant subspace change, although it still handles that setting. Secondly, we extend our approach and its analysis to provably solving these problems when the raw data is federated and when the over-air data communication modality is used for information exchange between the K peer nodes and the center.

5.1 Introduction

Subspace tracking (ST) with missing data or outliers or both has been extensively studied in the last few decades [44, 10, 48, 33, 40]. ST with outlier data is commonly referred to as Robust ST (RST); it is the dynamic or "tracking" version of Robust PCA [7, 32]. This work provides a new simple algorithm and guarantee for both ST with missing data (ST-miss) and RST-miss. Secondly, we extend our approach and its analysis to provably solving these problems when the data is federated and when the over-air data communication modality [3] is used for information exchange between the K peer nodes and the central server. (R)ST-miss has important applications

in video analytics [8], social network activity learning [47] (anomaly detection) and recommendation system design [42] (learning time-varying low-dimensional user preferences from incomplete user ratings). The federated setting is most relevant for the latter two. At each time, each local node would have access to user ratings or messaging data from a subset of nearby users, but the subspace learning and matrix completion algorithm needs to use data from all the users.

Federated learning [18] refers to a distributed learning scenario in which individual nodes keep their data private but only share intermediate locally computed summary statistics with the central server at each algorithm iteration. The central server in turn, shares a global aggregate of these iterates with all the nodes. There has been extensive recent work on solving machine learning problems in a federated setting [19, 46, 41, 5, 22] but all these assume a perfect channel between the peer nodes and the central server. This is a valid assumption in the traditional digital transmission mode in which different peer nodes transmit in different time or frequency bands, and appropriate channel coding is done at lower network layers to enable error-free recovery with very high probability.

Advances in wireless communication technology now allow for (nearly) synchronous transmission by the various peer nodes and thus enable an alternate computation/communication paradigm for learning algorithms for which the aggregation step is a summation operation. In this alternate paradigm, the summation can be performed "over-air" using the superposition property of the wireless channel and the summed aggregate or its processed version can be broadcasted to all the nodes [2, 3, 45]. Assuming K peer nodes, this over-air addition is up to K-times more time- or bandwidth-efficient than the traditional mode. In the absence of error control coding at the lower network layers, additive channel noise and channel fading effects corrupt the transmitted data. In general, there exist well-established physical layer communication techniques to estimate and compensate for channel fading [38]. Also, while perfect synchrony in transmission is impossible, small timing mismatches can be handled using standard techniques. We expand upon both these points in Sec 5.4.1. From a signal processing perspective, therefore, the main issue to be tackled is the additive channel noise which now corrupts each algorithm iterate.

Related Work. Provable ST with missing or corrupted data (ST-miss and RST-miss) in the centralized setting has been extensively studied in past work [10, 48, 33, 29, 28, 12]. All existing results are either partial guarantees (need assumptions on intermediate algorithm estimates; do not provide a set of assumptions on algorithm inputs that guarantee that the algorithm output is close to the quantity of interest) [10, 48, 33, 12] or assume piecewise constant subspace change [29, 28]. This assumption is often not valid in practice, e.g, there is no reason for a "subspace change time" in case of slow changing video backgrounds. Existing works assume it in order to obtain simple guarantees for ϵ-accurate subspace recovery for any $\epsilon > 0$ (in the noise-free case) or for any ϵ larger than the noise-level (in the noisy case).

The only other existing works that also study unsupervised learning algorithms with noisy algorithm iterations are [15, 4]; both these works study the noisy iteration version of the power method (PM) for computing the top r singular vectors of a given data matrix. In these works, noise is deliberately added to each algorithm iterate in order to ensure privacy of the data matrix.

It should be noted that other solutions to batch low-rank matrix completion (LRMC) cannot be implemented to respect the federated constraints (the aggregation step needs to be a summation operation). We briefly discuss these in Sec. 5.4. Another somewhat related line of work involves distributed algorithms for PCA; these are reviewed in [42], and there is also one for distributed ST-miss [21], Most of these come without provable guarantees, and most also do not account for either missing data or iteration-noise or both. For example, the recent work [23] aims to optimize communication efficiency but the channel is assumed to be perfect, and so iteration noise is not considered. Moreover, the algorithm is computationally expensive (involves computing a full SVD of a large matrix); and the guarantee provided is a multiplicative one on the PCA reconstruction error. Finally, LRMC in a decentralized setting is studied in [25] with the goal of speeding up computation via parallel processing using multiple computing nodes. In this paper as well, the full data is communicated to the central server and hence this is not a federated setting. Also, no channel noise is considered. It is not clear if this algorithm or guarantee can be modified to deal

with federated data or over-air communication. Finally, there also exist heuristics for various types of distributed LRMC such as [37, 1, 43].

Other works that also develop algorithms for the federated over-air aggregation setting include [3, 14]. However, all these develop stochastic gradient descent (SGD) based algorithms and the focus is on optimizing resource allocation to satisfy transmit power constraints. These do not provide performance guarantees for the resulting perturbed SGD algorithm. A different related line of work is in developing federated algorithms, albeit not in the over-air aggregation mode. Recent works such as [19, 20] attempt to empirically optimize the communication efficiency. Similarly, [13] studies federated PCA but it does not consider over-air communication paradigm, and does not deal with outliers or missing data.

Contributions. This work has two contributions. First, we obtain a new set of results that provide a complete guarantee for ST-miss and RST-miss without assuming piecewise constant subspace change. The tradeoff is our error bounds are a little more complicated. Another advantage of our new result is that it analyzes a much simpler tracking algorithm (only one algorithm parameter needs to be set instead of three). Our guarantee is useful (improves upon the naive approach of standard PCA repeated every α frames) when the subspace changes are indeed slow enough. At the same time, we can still obtain a guarantee for our simpler algorithm that holds under piecewise constant subspace change but does not require an upper bound on the amount of change, i.e, we can still recover the result of [28].

The second contribution of this work is a provable solution to the above problem in the federated data setting when the data communication is done in the over-air mode. As explained above, the main new challenge here is to develop approaches that are provably robust to additive noise in the algorithm iterates. This setting of noisy iterations has received little attention in literature as noted above. To the best of our knowledge, this is the first provable algorithm that studies (R)ST-miss in a federated, over-air paradigm. The main challenges here are (i) a design of an algorithm for this setting (this requires use of a federated over-air power method (FedOA-PM) for solving the PCA step) and (ii) dealing with noise iterates due to the channel noise. For the latter, the main work is

in obtaining a modified result for PCA in sparse data-dependent noise solved via the FedOA-PM; see Lemma 5.84.

Chapter organization. We give the centralized problem formulation next. After this, in Sec 5.3, we develop our solution for just ST-miss in the centralized setting and explain how it successfully relaxes the piecewise constant subspace change assumption made by existing guarantees. Next, we consider RST-miss in the federated over-air setting in Sec 5.4. Simulations are shown in Sec 5.5.

5.2 Notation and Problem Formulation

5.2.1 Notation

We use the interval notation $[a, b]$ to refer to all integers between a and b, inclusive, and we use $[a, b) := [a, b-1]$. We use $[K] := [1, K]$. $\|.\|$ denotes the l_2 norm for vectors and induced l_2 norm for matrices unless specified otherwise. We use $\boldsymbol{I}$ to denote the identity matrix of appropriate dimensions. We use $\boldsymbol{M}_{\mathcal{T}}$ to denote a sub-matrix of $\boldsymbol{M}$ formed by its columns indexed by entries in the set $\mathcal{T}$. A matrix $\boldsymbol{P}$ with mutually orthonormal columns is referred to as a *basis matrix*; it represents the subspace spanned by its columns. For basis matrices $\boldsymbol{P}_1, \boldsymbol{P}_2$, $\mathrm{SE}(\boldsymbol{P}_1, \boldsymbol{P}_2) := \|(\boldsymbol{I} - \boldsymbol{P}_1\boldsymbol{P}_1^\top)\boldsymbol{P}_2\|$ quantifies the Subspace Error (distance) between their respective subspaces. This is equal to the sine of the largest principal angle between the subspaces. If $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ are of the same dimension, $\mathrm{SE}(\boldsymbol{P}_1, \boldsymbol{P}_2) = \mathrm{SE}(\boldsymbol{P}_2, \boldsymbol{P}_1)$. We reuse the letters C, c to denote different numerical constants in each use with the convention that $C \geq 1$ and $c < 1$.

We use *r-SVD* to refer to the matrix of top-r left singular vectors (vectors corresponding to the r largest singular values) of the given matrix. Finally, $\boldsymbol{M}^\dagger := (\boldsymbol{M}^\top\boldsymbol{M})^{-1}\boldsymbol{M}^\top$ is used to denote the pseudo inverse of $\boldsymbol{M}$.

5.2.2 ST with missing data (ST-miss)

Assume that at each time t, we observe an n-dimensional data stream of the form

$$\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\tilde{\boldsymbol{\ell}}_t), \quad t = 1, 2, \cdots, d \tag{5.1}$$

where $\mathcal{P}_{\Omega_t}(\cdot)$ is a binary mask that selects entries in the index set Ω_t (this is known), and $\tilde{\boldsymbol{\ell}}_t$ approximately lies an low (at most r) dimensional subspace that is either constant or changes slowly over time. The goal is to track the subspace(s). This statement can be made precise in several ways. The first is as done in past work [28] (and older work). One assumes a "generative model": $\tilde{\boldsymbol{\ell}}_t = \boldsymbol{P}_t \boldsymbol{a}_t$ with $\boldsymbol{P}_t$ being a $n \times r$ basis matrix. The goal is to track the column span of $\boldsymbol{P}_t$, span($\boldsymbol{P}_t$). To make this problem well-posed (number of unknowns smaller than number of observed scalars), the piecewise constant subspace change model assumption becomes essential as explained in [28]. However, this is a restrictive assumption that is typically not valid for real data, e.g., there is no reason for the subspaces to change at certain select time instants in case of slow changing videos.

A second approach to make our problem statement precise, and the one that we use in this work, is as follows. For an α large enough[1], consider α-length sub-matrices formed by consecutive $\tilde{\boldsymbol{\ell}}_t$'s. Let $\tilde{\boldsymbol{L}}_1 := [\tilde{\boldsymbol{\ell}}_1, \tilde{\boldsymbol{\ell}}_2, \ldots, \tilde{\boldsymbol{\ell}}_\alpha]$; $\tilde{\boldsymbol{L}}_2 := [\tilde{\boldsymbol{\ell}}_{\alpha+1}, \tilde{\boldsymbol{\ell}}_{\alpha+2}, \ldots, \tilde{\boldsymbol{\ell}}_{2\alpha}]$ and so on. Let $\boldsymbol{P}_j$ be the r-SVD (matrix of top r singular vectors) of $\tilde{\boldsymbol{L}}_j$. Slow subspace change means that, for all j,

$$\Delta_j := \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j) \leq \Delta_{tv}$$

for a $\Delta_{tv} \ll 1$. Our guarantee assumes $\Delta_{tv} = 0.1$. The goal is to track (sequentially estimate) the subspace spanned by the columns of $\boldsymbol{P}_j$ as well as the rank-r approximation, $\boldsymbol{L}_j := \boldsymbol{P}_j \boldsymbol{P}_j^\top \tilde{\boldsymbol{L}}_j$. As is well known (Eckart-Young theorem), this minimizes $\|\tilde{\boldsymbol{L}} - \check{\boldsymbol{L}}\|_2$ over all rank r matrices $\check{\boldsymbol{L}}$. We will occasionally refer to $\boldsymbol{L}_j$ and its columns $\boldsymbol{\ell}_t$ as the *true data*.

Let $\boldsymbol{A}_j := \boldsymbol{P}_j^\top \tilde{\boldsymbol{L}}_j$ be the matrix of subspace coefficients along $\boldsymbol{P}_j$. Let $\boldsymbol{V}_j := \tilde{\boldsymbol{L}}_j - \boldsymbol{L}_j$ be the residual noise/error. Clearly, since

$$\tilde{\boldsymbol{L}}_j \overset{\text{SVD}}{=} [\boldsymbol{P}_j \underbrace{\boldsymbol{S}\boldsymbol{B}^\top}_{\boldsymbol{A}_j} + \underbrace{\boldsymbol{P}_{j,\perp}\boldsymbol{S}_\perp \boldsymbol{B}_\perp^\top}_{\boldsymbol{V}_j}] = \underbrace{\boldsymbol{P}_j \boldsymbol{A}_j}_{\boldsymbol{L}_j} + \boldsymbol{V}_j,$$

it is immediate that $\boldsymbol{L}_j \boldsymbol{V}_j^\top = 0$.

Let $\boldsymbol{a}_t$, $\boldsymbol{\ell}_t$ and $\boldsymbol{v}_t$ be the columns of $\boldsymbol{A}_j$, $\boldsymbol{L}_j$, and $\boldsymbol{V}_j$ respectively. Thus, for $t \in \mathcal{J}_j := [(j-1)\alpha + 1, (j-1)\alpha + 2, \ldots j\alpha]$, $\boldsymbol{a}_t = \boldsymbol{P}_j^\top \tilde{\boldsymbol{\ell}}_t$, $\boldsymbol{\ell}_t = \boldsymbol{P}_j \boldsymbol{a}_t$, and $\boldsymbol{v}_t = \tilde{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t$.

[1] as we show later $\alpha \geq Cr \log n$ suffices

Also, let $\mathcal{T}_t = (\Omega_t)^c$ be the index set of missing entries at time t. With this, we can rewrite (5.1) as

$$\begin{aligned}\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\tilde{\boldsymbol{\ell}}_t) &= \tilde{\boldsymbol{\ell}}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}^{\top}\tilde{\boldsymbol{\ell}}_t \\ &= \boldsymbol{\ell}_t + \boldsymbol{v}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}{}^{\top}(\boldsymbol{\ell}_t + \boldsymbol{v}_t)\end{aligned}$$

5.2.3 Robust ST-miss (RST-miss)

Robust ST-miss assumes that there can also be additive sparse outliers in the observed data $\boldsymbol{y}_t$. Thus, for all $t = 1, 2, \cdots, d$,

$$\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\tilde{\boldsymbol{\ell}}_t) + \boldsymbol{s}_t \tag{5.2}$$

where $\boldsymbol{s}_t$ is the sparse outlier with support $\mathcal{T}_{\text{sparse},t}$. The assumptions on Ω_t, $\tilde{\boldsymbol{\ell}}_t$ remain exactly the same as in the previous section. Due to space constraints, we provide the complete algorithm and guarantee for this problem in the supplementary material.

5.2.4 Federated Over-Air Data Sharing Constraints and Iteration Noise

The goal is to also solve RST-miss in a federated over-air fashion. Concretely, this means the following for an iterative algorithm. At iteration l, the central server broadcasts the $(l-1)$-th estimate of the quantity of interest[2] denoted $\hat{\boldsymbol{U}}_{l-1}$ to each of the K nodes. Each node then uses this estimate and its (locally) available data to compute the new local estimate denoted $\tilde{\boldsymbol{U}}_{k,l}$. The nodes then synchronously transmit these to the central server but the transmission is corrupted by channel noise and thus the central server receives

$$\tilde{\boldsymbol{U}}_l := \sum_k \tilde{\boldsymbol{U}}_{k,l} + \boldsymbol{W}_l$$

where $\boldsymbol{W}_l$ is the channel noise. We assume that each entry of $\boldsymbol{W}_l$ is i.i.d. zero-mean Gaussian with variance σ_c^2. The central server then processes $\tilde{\boldsymbol{U}}_l$ to get the new estimate of the quantity of interest, $\hat{\boldsymbol{U}}_l$ which is then broadcast to all K nodes for the next iteration. The presence of $\boldsymbol{W}_l$

[2]The quantity of interest could be a vector or a matrix depending on the application. For the problem we study (subspace learning/tracking), the quantity of interest is a $n \times r$ basis matrix.

in each iteration introduces a new and different set of challenges in algorithm design and analysis compared to what has been largely explored in existing literature.

5.3 ST from Missing Data (ST-miss)

5.3.1 Proposed Algorithm

Recall that we split our data into mini-batches of size α; thus $\boldsymbol{Y}_1 := [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_\alpha]$, $\boldsymbol{Y}_2 := [\boldsymbol{y}_{\alpha+1}, \boldsymbol{y}_{\alpha+2}, \dots \boldsymbol{y}_{2\alpha}]$ and so on. Thus $\boldsymbol{Y}_j := [\boldsymbol{y}_{(j-1)\alpha+1}, \boldsymbol{y}_{(j-1)\alpha+2}, \dots, \boldsymbol{y}_{j\alpha}]$. Without the slow subspace change assumption, the obvious way to solve ST-miss would be to use what can be called *simple PCA*: for each mini-batch j, compute $\hat{\boldsymbol{P}}_j$ as the r-SVD of $\boldsymbol{Y}_j$. However, when slow subspace change is assumed, a better approach is a simplification of our algorithm from [28]. We still initialize via r-SVD: compute $\hat{\boldsymbol{P}}_1$ as the r-SVD of $\boldsymbol{Y}_1$. For the j-th mini-batch, we first obtain *an* estimate of the missing entries for each column using the previous subspace estimate and projected Least Squares (LS) as follows. For every $t \in ((j-1)\alpha, j\alpha]$, we compute

$$\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t} \boldsymbol{\Psi}^\dagger_{\mathcal{T}_t} (\boldsymbol{\Psi} \boldsymbol{y}_t) \tag{5.3}$$

where $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1} \hat{\boldsymbol{P}}_{j-1}^\top$. This step works under the assumption that the span of $\hat{\boldsymbol{P}}_{j-1}$ is a good estimate of that of $\boldsymbol{P}_{j-1}$. By slow subspace change, this also means it is a good estimate of the span of $\boldsymbol{P}_j$. This equation is a compact way to write the following: $(\hat{\boldsymbol{\ell}}_t)_{\mathcal{T}_t^c} = (\boldsymbol{y}_t)_{\mathcal{T}_t^c} = (\tilde{\boldsymbol{\ell}}_t)_{\mathcal{T}_t^c}$ (use the observed entries as is) and $(\hat{\boldsymbol{\ell}}_t)_{\mathcal{T}_t} = \boldsymbol{\Psi}^\dagger_{\mathcal{T}_t} (\boldsymbol{\Psi} \boldsymbol{y}_t)$. To understand this, notice that $\boldsymbol{\Psi} \boldsymbol{y}_t = -\boldsymbol{\Psi}_{\mathcal{T}_t} \boldsymbol{z}_t + (\boldsymbol{\Psi} \boldsymbol{\ell}_t + \boldsymbol{\Psi} \boldsymbol{v}_t)$ where $\boldsymbol{z}_t := (\boldsymbol{I}_{\mathcal{T}_t}{}^\top \tilde{\boldsymbol{\ell}}_t)$ is the vector of missing entries. The second two terms can be treated as small "noise"/disturbance[3] and so we can compute an estimate of $\boldsymbol{z}_t$ from $\boldsymbol{\Psi} \boldsymbol{y}_t$ by LS.

The second step is to compute $\hat{\boldsymbol{P}}_j$ as the r-SVD of $\hat{\boldsymbol{L}}_j := [\hat{\boldsymbol{\ell}}_{(j-1)\alpha+1}, \cdots, \hat{\boldsymbol{\ell}}_{j\alpha}]$.

Finally, we can use $\hat{\boldsymbol{P}}_j$ to obtain an optional improved estimate, $\hat{\hat{\boldsymbol{\ell}}}_t = \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t} \tilde{\boldsymbol{\Psi}}^\dagger_{\mathcal{T}_t} (\tilde{\boldsymbol{\Psi}} \boldsymbol{y}_t)$ where $\tilde{\boldsymbol{\Psi}} = \boldsymbol{I} - \hat{\boldsymbol{P}}_j \hat{\boldsymbol{P}}_j^\top$. We summarize this approach in Algorithm 10. We show next that, under slow

[3] The first is small because of slow subspace change and $\hat{\boldsymbol{P}}_{j-1}$ being a good estimate (if $\text{span}(\hat{\boldsymbol{P}}_{j-1}) = \text{span}(\boldsymbol{P}_j)$ this term would be zero); the second is small because $\|\boldsymbol{v}_t\|$ is small due to the approximate low-rank assumption.

Algorithm 10 STMiss-NoDet

Require: $\boldsymbol{Y}, \mathcal{T}$

Parameters: α

Initialize: $\hat{\boldsymbol{P}}_1 \leftarrow r\text{-}SVD[\boldsymbol{y}_1, \cdots, \boldsymbol{y}_\alpha]$, $j \leftarrow 2$

for $j \geq 2$ **do**

  Projected LS:

  $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}^\top$

  for all $t \in ((j-1)\alpha, j\alpha]$ **do**

    $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t})^\dagger(\boldsymbol{\Psi}\boldsymbol{y}_t)$

  end for

  PCA on $\hat{\boldsymbol{L}}_j$*:*

  $\hat{\boldsymbol{P}}_j \leftarrow r\text{-}SVD(\hat{\boldsymbol{L}}_j)$ where $\hat{\boldsymbol{L}}_j := [\hat{\boldsymbol{\ell}}_{(j-1)\alpha+1}, \cdots, \hat{\boldsymbol{\ell}}_{j\alpha}]$

  for all $t \in ((j-1)\alpha, j\alpha]$ **do** ▷ optional

    $\tilde{\boldsymbol{\Psi}} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_j\hat{\boldsymbol{P}}_j^\top$

    $\hat{\hat{\boldsymbol{\ell}}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}(\tilde{\boldsymbol{\Psi}}_{\mathcal{T}_t})^\dagger(\tilde{\boldsymbol{\Psi}}\boldsymbol{y}_t)$

  end for

end for

Ensure: $\hat{\boldsymbol{P}}_j$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\hat{\boldsymbol{\ell}}}_t$.

subspace change, Algorithm 10 yields a significantly better subspace estimates than simple PCA (PCA on each $\boldsymbol{Y}_j$).

5.3.2 Assumptions and Main Result

It is well known from the LRMC literature [8, 34, 31] that for guaranteeing correct matrix recovery, we need to assume incoherence (w.r.t. the standard basis) of the left and right singular vectors of the matrix. We need a similar assumption on $\boldsymbol{P}_j$'s.

Definition 5.69 (μ-Incoherence of $\boldsymbol{P}_j$s)**.** *Assume that*

$$\max_{j\in[d/\alpha]} \max_{m\in[r]} \|\boldsymbol{P}_j^{(m)}\|_2^2 \leq \frac{\mu r}{n}$$

where $\boldsymbol{P}_j^{(m)}$ *denotes the* m*-th row of* $\boldsymbol{P}_j$ *and* $\mu \geq 1$ *is a constant (incoherence parameter).*

Since we study a tracking algorithm (we want to track subspace changes quickly), we replace the standard right singular vectors' incoherence assumption with the following simple statistical assumption on the subspace coefficients $\boldsymbol{a}_t$. This helps us obtain guarantees on our mini-batch algorithm that operates on α-size mini-batches of the data.

Definition 5.70 (Statistical μ-Incoherence of $\boldsymbol{a}_i$s)**.** *Recall that $\boldsymbol{a}_t = \boldsymbol{P}_j^\top \tilde{\boldsymbol{\ell}}_t$ for all $t \in \mathcal{J}_j$. Assume that the $\boldsymbol{a}_t$'s are zero mean; mutually independent; have identical diagonal covariance matrix $\boldsymbol{\Lambda}$, i.e., that $\mathbb{E}[\boldsymbol{a}_t \boldsymbol{a}_t^\top] = \boldsymbol{\Lambda}$ with $\boldsymbol{\Lambda}$ diagonal; and are bounded, i.e., $\max_t \|\boldsymbol{a}_t\|^2 \le \mu r \lambda^+$, where $\lambda^+ := \lambda_{\max}(\boldsymbol{\Lambda})$ and $\mu \ge 1$ is a small constant. Also, let $\lambda^- := \lambda_{\min}(\boldsymbol{\Lambda})$ and $f := \lambda^+/\lambda^-$.*

If a few complete rows (columns) of the entries are missing, it is impossible to recover the underlying matrix. This can be avoided by either assuming bounds on the number of missing entries in any row and in any column, or by assuming that each entry is observed uniformly at random with probability ρ independent of all others. In this work we assume the former which is a weaker assumption. We need the following definition.

Definition 5.71 (Bounded Missing Entry Fractions)**.** *Consider the $n \times \alpha$ observed matrix $\boldsymbol{Y}_j$ for the j-th mini-batch of data. We use max-miss-frac-col (max-miss-frac-row) to denote the maximum of the fraction of missing entries in any column (row) of this matrix.*

Owing to the assumption that $\tilde{\boldsymbol{L}}_j$ is approximately low-rank, it follows that $\tilde{\boldsymbol{L}}_j - \boldsymbol{L}_j := \boldsymbol{V}_j$ is "small".

Definition 5.72 (Small, bounded, independent modeling error)**.** *Let $\lambda_v^+ := \max_t \|\mathbb{E}[\boldsymbol{v}_t \boldsymbol{v}_t^\top]\|$. We assume that $\lambda_v^+ < \lambda^-$, $\max_t \|\boldsymbol{v}_t\|^2 \le C r \lambda_v^+$ and $\boldsymbol{v}_t$'s are mutually independent over time.*

Main result. We have the following result for the naive algorithm of PCA on every mini-batch of α observed samples $\boldsymbol{Y}_j$. We use the following definition of noise level

$$\text{no-lev} := \sqrt{\lambda_v^+/\lambda^-}$$

Theorem 5.73 (STmiss Algorithm 12)**.** .

Set algorithm parameter $\alpha = C f^2 r \log n$.

Assume that no-lev < 0.02 and the following hold:

1. **Incoherence:** *$\boldsymbol{P}_j$'s satisfy μ-incoherence, and $\boldsymbol{a}_t$'s satisfy statistical right μ-incoherence;*

2. **Missing Entries:** *max-miss-frac-col $\le 0.01/(\mu r)$, max-miss-frac-row $\le 0.0001/f^2$;*

3. **Modeling Error:** *assume Definition 5.72*

4. **Subspace Change:** $\Delta_j := \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j) \leq \Delta_{tv} = 0.1$,

then, with probability at least $1 - 10dn^{-10}$*, we have*

$$\begin{aligned}
&\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \\
&\leq \max(0.1 \cdot 0.3^{j-1} + \Delta_{tv}(0.3 + 0.3^2 ... + 0.3^{j-1}), \textit{no-lev}) \\
&< \max(0.1 \cdot 0.3^{j-1} + 0.5\Delta_{tv}, \textit{no-lev})
\end{aligned}$$

Also, at all j*, and for* $t \in [(j-1)\alpha, j\alpha)$*,* $\|\hat{\tilde{\boldsymbol{\ell}}}_t - \tilde{\boldsymbol{\ell}}_t\| \leq 1.2 \cdot \mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j)\|\tilde{\boldsymbol{\ell}}_t\| + \|\boldsymbol{v}_t\|$ *while* $\|\hat{\boldsymbol{\ell}}_t - \tilde{\boldsymbol{\ell}}_t\| \leq 1.2 \cdot \mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j)\|\tilde{\boldsymbol{\ell}}_t\| + \|\boldsymbol{v}_t\| \leq 1.2 \cdot (\Delta_{tv} + \mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j))\|\tilde{\boldsymbol{\ell}}_t\| + \|\boldsymbol{v}_t\|$

Proof: See Sec. 5.3.4.

Theorem 5.74 (Simple PCA)**.** *Let* $\hat{\boldsymbol{P}}_j$ *be the* r*-SVD of* $\boldsymbol{Y}_j$ *with* $\alpha = Cf^2 r \log n$*. Assume* μ*-incoherence of* $\boldsymbol{P}_j$*s, statistical* μ*-incoherence of* $\boldsymbol{a}_i$*s, modeling error assumption given in Definition 5.72, max-miss-frac-col* $\leq 0.01/(\mu r)$*, max-miss-frac-row* $\leq 0.01/f^2$*. Then, with probability at least* $1 - dn^{-10}$*,*

$$\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \max(0.1 \cdot 0.25, \textit{no-lev})$$

Proof: The proof is the same as that for the initialization step of Algorithm 10; see Sec. 5.3.4.

First consider the practically relevant setting of approximately rank r $\tilde{\boldsymbol{L}}_j$'s so that the noise level $\sqrt{\lambda_v^+/\lambda^-}$ is small. In particular, assume it is smaller than $0.1 \cdot 0.25$. Then, if Δ_{tv} is small enough, the bound of Theorem 5.73 is significantly smaller. If the noise level is larger, then in both cases, the noise level term dominates and both results give the same bound. In summary, in all cases, as long as Δ_{tv} is small (slow subspace change holds), Theorem 5.73 gives an as good or better bound. We demonstrate this point in Fig 5.1.

5.3.3 Guarantee for piecewise constant subspace change

Previous work on provable ST-miss [28] assumed piecewise constant subspace change (required the subspace to be constant for long enough), but did not require an upper bound on the amount of change. As we show next STmiss-NoDet is able to track such changes as well and provide similar tracking guarantees even under a (mild) generalization of the previous model.

Theorem 5.75. *Set algorithm parameter* $\alpha = Cf^2r\log n$. *Assume that no-lev* < 0.02 *and the first three assumptions of Theorem 5.73 hold. Under an approximately piecewise constant subspace change model* ($\Delta_j \leq$ *no-lev for all* j *except for* $j = j_\gamma$, *for* $\gamma = 1, 2, \ldots,$) *with the subspace change times satisfying* $j_\gamma - j_{\gamma-1} > K := C\log(1/\textit{no-lev})$, *then, w.p. at least* $1 - dn^{-10}$,

$$\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq$$

$$\begin{cases} (0.2 + 2\textit{no-lev}) \cdot 0.25 + \textit{no-lev}), & \textit{if} \quad j = j_\gamma \\ (0.2 + 2\textit{no-lev}) \cdot 0.3^{(j-j_\gamma)-1} + \textit{no-lev}, & \textit{if} \quad j_\gamma < j < j_{\gamma+1} \end{cases}$$

Notice that for $j_{\gamma+1} > j > j_\gamma + K$, *the bound is at most* 2*no-lev.*

The subspace change model in this result does not require an upper bound on the amount of subspace change as long as the change occurs infrequently. However, it still allows for small rotations to the subspace at each time. The exponential decay in the subspace recovery error bound is the same as that guaranteed by the results is [28]. STmiss-NoDet does not detect subspace changes. However, a detection step similar to that used in previous work can be included if needed and then a similar detection guarantee can also be proved. We provide these in the Supplementary Material (https://arxiv.org/abs/2002.12873).

5.3.4 Proof of Theorem 5.73 and 5.74

The proof follows by a careful application of a result from [30] that analyzes PCA in sparse data-dependent noise (SDDN) along with simple linear algebra tricks, some of which are also borrowed from there. The novel contribution here is the application of the same ideas for providing a result

that holds under a much simpler and practically valid assumption of slow changing subspaces (without any artificial piecewise constant assumption). Also, the proof provided here is much shorter.

Subspace error bounds. Consider the projected LS step. Recall that $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}^\top$. Since $\boldsymbol{y}_t$ can be expressed as $\boldsymbol{y}_t = \tilde{\boldsymbol{\ell}}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{I}_{\mathcal{T}_t}^\top \tilde{\boldsymbol{\ell}}_t)$, using the idea explained while developing the algorithm,

$$\begin{aligned}
\hat{\boldsymbol{\ell}}_t &= \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{\Psi}_{\mathcal{T}_t}^\dagger \boldsymbol{\Psi}(-\boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}^\top \tilde{\boldsymbol{\ell}}_t + \tilde{\boldsymbol{\ell}}_t) \\
&= \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}(-\boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}^\top \tilde{\boldsymbol{\ell}}_t + \tilde{\boldsymbol{\ell}}_t) \\
&= \boldsymbol{y}_t + \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}^\top \tilde{\boldsymbol{\ell}}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \tilde{\boldsymbol{\ell}}_t \\
&= \tilde{\boldsymbol{\ell}}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \tilde{\boldsymbol{\ell}}_t \\
&= \boldsymbol{\ell}_t + \boldsymbol{v}_t - \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^\dagger \boldsymbol{\Psi}_{\mathcal{T}_t}^\top(\boldsymbol{\ell}_t + \boldsymbol{v}_t)
\end{aligned}$$

This final expression can be reorganized as follows.

$$\begin{aligned}
\hat{\boldsymbol{\ell}}_t &= \boldsymbol{\ell}_t + \underbrace{\boldsymbol{v}_t - \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^\dagger \boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{v}_t}_{\text{small, unstructured noise}} - \underbrace{\boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^\dagger \boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\ell}_t}_{\text{sparse, data dependent noise}} \\
&:= \boldsymbol{\ell}_t + \boldsymbol{e}_t
\end{aligned} \tag{5.4}$$

Thus, recovering $\boldsymbol{P}_j$ from estimates $\hat{\boldsymbol{L}}_j$ is a problem of PCA in sparse data-dependent noise (SDDN): the "noise" $\boldsymbol{e}_t$ consists of two terms, the first is just small unstructured noise (depends on $\boldsymbol{v}_t$) while the second is sparse with support $\mathcal{T}_t$ and depends linearly on the true data $\boldsymbol{\ell}_t$. We studied PCA-SDDN in detail in [30] where we showed the following.

Lemma 5.76 (PCA-SDDN)**.** *For $i = 1, \cdots, \alpha$, assume that $\boldsymbol{z}_i = \boldsymbol{\ell}_i + \boldsymbol{w}_i + \boldsymbol{v}_i$ with $\boldsymbol{w}_i = \boldsymbol{I}_{\mathcal{T}_i}\boldsymbol{B}_i\boldsymbol{\ell}_i$ being sparse, data-dependent noise with support $\mathcal{T}_i$; $\boldsymbol{\ell}_i = \boldsymbol{P}\boldsymbol{a}_i$ with $\boldsymbol{P}$ being an $n \times r$ basis matrix that satisfies μ-incoherence, and $\boldsymbol{a}_i$'s satisfy statistical μ-incoherence; and $\boldsymbol{v}_i$ is small bounded noise with $\lambda_v^+ := \|\mathbb{E}[\boldsymbol{v}_i\boldsymbol{v}_i^\top]\| < \lambda^-$ and $\max_i \|\boldsymbol{v}_i\|^2 \le Cr_v\lambda_v^+$. Let $q := max_i\|\boldsymbol{B}_i\boldsymbol{P}\|$ and let b be the maximum fraction of non-zeros in any row of the matrix $[\boldsymbol{w}_1, \cdots, \boldsymbol{w}_\alpha]$. Let $\hat{\boldsymbol{P}}$ be the matrix of top*

r eigenvectors of $\frac{1}{\alpha}\sum_i \boldsymbol{z}_i\boldsymbol{z}_i^\top$. Assume that $q \leq 3$. Pick an $\epsilon > 0$. If

$$7\sqrt{b}qf + \frac{\lambda_v^+}{\lambda^-} < 0.4\epsilon, \text{ and} \tag{5.5}$$

$$\alpha \geq \alpha^* := C \max\left(\frac{q^2 f^2}{\epsilon^2} r \log n, \frac{\frac{\lambda_v^+}{\lambda^-} f}{\epsilon^2} r \log n\right), \tag{5.6}$$

then, w.p. at least $1 - 10n^{-10}$, $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon$.

This result says that, under the incoherence assumptions, and assuming that the unstructured noise satisfies the stated assumptions, if the support of the SDDN, $\boldsymbol{w}_i$, changes enough over time so that b, which is the maximum fraction of nonzeros in any row of the matrix $[\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_\alpha]$, is sufficiently small, if the unstructured noise power is small enough compared to the r-th eigenvalue of the true data covariance matrix and it is bounded with small effective dimension, $\|\boldsymbol{v}_i\|^2/\lambda_v^+ \leq Cr$, and if α is large enough, then $\mathrm{span}(\hat{\boldsymbol{P}})$, is a good approximation of $\mathrm{span}(\boldsymbol{P})$. Notice here that for SDDN, the true data and noise correlation, $\mathbb{E}[\boldsymbol{\ell}_i\boldsymbol{w}_i{}^\top]$, is not zero, and the noise power, $\mathbb{E}[\boldsymbol{w}_i\boldsymbol{w}_i{}^\top]$, itself is also not small. However, the key idea used to obtain this result is the following: enough support changes over time (small b) helps ensure that the upper bounds on sample averaged values of both these quantities, $\|(1/\alpha)\sum_i \mathbb{E}[\boldsymbol{\ell}_i\boldsymbol{w}_i{}^\top]\|$ and $\|(1/\alpha)\sum_i \mathbb{E}[\boldsymbol{w}_i\boldsymbol{w}_i{}^\top]\|$ are $\sqrt{b}$ times smaller than those on their maximum instantaneous values, $\|\mathbb{E}[\boldsymbol{\ell}_i\boldsymbol{w}_i{}^\top]\|$ and $\|\mathbb{E}[\boldsymbol{w}_i\boldsymbol{w}_i{}^\top]\|$.

Our proof uses Lemma 5.76 applied on the j-th mini-batch of estimates, $\hat{\boldsymbol{L}}_j$ along with the following simple facts.

Fact 5.77. *1. From [33, Remark 3.6] we have: let $\boldsymbol{P}$ be an μ-incoherent, $n \times r$ basis matrix. Then, for any set $\mathcal{T} \subseteq [n]$, we have*

$$\|\boldsymbol{I}_{\mathcal{T}}^\top \boldsymbol{P}\|^2 \leq |\mathcal{T}| \cdot \frac{\mu r}{n}$$

2. For $n \times r$ basis matrices $\boldsymbol{P}$, $\hat{\boldsymbol{P}}$ (useful when the column span of $\hat{\boldsymbol{P}}$ is a good approximation of that of $\boldsymbol{P}$), and any set $\mathcal{T} \subseteq [n]$, we have

$$\|\boldsymbol{I}_{\mathcal{T}}^\top \hat{\boldsymbol{P}}\| \leq \mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) + \|\boldsymbol{I}_{\mathcal{T}}^\top \boldsymbol{P}\|$$

3. For a μ-incoherent $n \times r$ basis matrix, $\boldsymbol{P}$, and any set $\mathcal{T} \subseteq [n]$,

$$\lambda_{\min}(\boldsymbol{I}_{\mathcal{T}}{}^{\top}(\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}^{\top})\boldsymbol{I}_{\mathcal{T}}) = 1 - \|\boldsymbol{I}_{T}{}^{\top}\boldsymbol{P}\|^2$$

Thus, combining the above three facts,

$$\|(\boldsymbol{I}_{\mathcal{T}}{}^{\top}(\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}^{\top})\boldsymbol{I}_{\mathcal{T}})^{-1}\| \leq \frac{1}{1 - (\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) + \sqrt{|\mathcal{T}|\mu r/n})^2}$$

The proof for $j = 1$ is a little different from $j > 1$. For $j = 1$, $\boldsymbol{\Psi} = \boldsymbol{I}$ and $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t$. Also, $i = t$. For $j > 1$, $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}^{\top}$ and $i = t - (j-1)\alpha$. Consider $j = 1$ (initialization). In this case, $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t$ satisfies (5.4) with $\boldsymbol{\Psi} = \boldsymbol{I}$. We apply Lemma 5.76 with $i = t$, $\boldsymbol{z}_i \equiv \hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t$, $\boldsymbol{\ell}_i \equiv \boldsymbol{\ell}_t$, $\boldsymbol{P} \equiv \boldsymbol{P}_1$, $\boldsymbol{w}_i \equiv -\boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}^{\top}\boldsymbol{\ell}_t$, $\boldsymbol{v}_i \equiv \boldsymbol{v}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}^{\top}\boldsymbol{v}_t$, $\boldsymbol{B}_i \equiv \boldsymbol{I}_{\mathcal{T}_t}^{\top}$. Notice that the fraction of non-zeros in the matrix $[\boldsymbol{w}_1, \cdots \boldsymbol{w}_\alpha]$ is bounded by max-miss-frac-row and thus $b \equiv$ max-miss-frac-row. To obtain q, we need to bound $\max_{t \in \mathcal{J}_1} \|\boldsymbol{B}_t\boldsymbol{P}_1\| = \max_{t \in \mathcal{J}_1} \|\boldsymbol{I}_{\mathcal{T}_t}{}^{\top}\boldsymbol{P}_1\|$. By item 1 of Fact 5.77, $\|\boldsymbol{I}_{\mathcal{T}_t}^{\top}\boldsymbol{P}_1\|^2 \leq |\mathcal{T}_t|\mu r/n \leq$ max-miss-frac-col $\cdot\, n\mu r/n$. Under the assumptions of Theorem 5.73, |max-miss-frac-col| $\leq 0.01/\mu r$ and thus $\max_t \|\boldsymbol{B}_t\boldsymbol{P}\| \leq 0.1 = q_1 \equiv q$. We pick $\epsilon = \max(\text{no-lev}, 0.25q_1)$. From the Theorem assumptions, $b =$ max-miss-frac-row $\leq 0.0001/f^2$ and no-lev ≤ 0.2 and so

$$\begin{aligned} 7\sqrt{b}qf + \frac{\lambda_v^+}{\lambda^-} &\leq 7q \cdot 0.01 + \text{no-lev}^2 \\ &\leq 0.07q + 0.2\text{no-lev} \leq 0.4\epsilon_1 \end{aligned}$$

Also, since $\epsilon = \max(\text{no-lev}, 0.25q_1)$, the value of α used in the Theorem satisfies the requirements of Lemma 5.76. Thus, we can apply this lemma to conclude that $\mathrm{SE}(\hat{\boldsymbol{P}}_1, \boldsymbol{P}_1) \leq \epsilon = \max(\text{no-lev}, 0.25q_1)$ with $q_1 = 0.1$. This completes the proof of Theorem 5.74 since simple-PCA just repeats this step at each j.

Now consider any $j > 1$. We claim that for $j > 1$,

$$\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \epsilon_j$$

with ϵ_j satisfying the following recursion: $\epsilon_1 = \max(\text{no-lev}, 0.25q_1)$ with $q_1 = 0.1$, and

$$\epsilon_j = \max(\text{ no-lev},\ 0.25 \cdot 1.2 \cdot (\epsilon_{j-1} + \Delta_{tv})) \tag{5.7}$$

This can be simplified to show that $\epsilon_j \leq \max(\text{no-lev}, (\text{no-lev} + \Delta_{tv}) \sum_{j'=1}^{j-1} (0.3)^{j'}, 0.3^j (0.25 q_1) + \Delta_{tv} \sum_{j'=1}^{j-1} (0.3)^{j'})$. This can be simplified to

$$\epsilon_j \leq 2 \max(\text{no-lev}, 0.3^j (0.25 q_1) + \Delta_{tv} \sum_{j'=1}^{j-1} (0.3)^{j'}) \tag{5.8}$$

To prove the above claim, we use induction. Base case: $j = 1$ done above. Induction assumption: assume $\text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_{j-1}) \leq \epsilon_{j-1}$. The application of the PCA-SDDN lemma is similar to that for $j = 1$ with the difference being that $i = t - (j-1)\alpha$ and $\boldsymbol{B}_i$ is different now. We now have $\boldsymbol{B}_i \equiv (\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}^\top$ and so $\max_{t \in \mathcal{J}_j} \|\boldsymbol{B}_t \boldsymbol{P}\| = \max_t \|(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{P}_j\|$. This can be bounded using Fact 5.77 as follows

$$\begin{aligned}
&\max_t \|(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1} \boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{P}_j\| \\
&\leq \max_t \|(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\| \|\boldsymbol{I}_{\mathcal{T}_t}^\top\| \|\boldsymbol{\Psi} \boldsymbol{P}_j\| \\
&\leq \frac{1}{1 - (\epsilon_{j-1} + \sqrt{0.01})^2} \cdot 1 \cdot \text{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j) \\
&\leq \frac{1}{1 - (\epsilon_{j-1} + \sqrt{0.01})^2} (\epsilon_{j-1} + \Delta_{tv}) := q_j
\end{aligned}$$

Using (5.8), $\epsilon_{j-1} \leq \max(\text{no-lev}, 0.25 q_1 + \Delta_{tv}(3/7) \leq \max(0.02, 0.025 + 0.1(3/7)) < 0.08$. This follows by using $j - 2 < \infty$, and no-lev < 0.02, and $\Delta_{tv} < 0.1$ (from Theorem assumptions). Using this upper bound on ϵ_{j-1} in the denominator expression of above,

$$q_j \leq 1.2(\epsilon_{j-1} + \Delta_{tv}) \tag{5.9}$$

Apply the PCA-SDDN lemma with $q \equiv q_j$ and $\epsilon = \max(\text{no-lev}, 0.25 q_j)$. With this choice of ϵ, it is easy to see that $7\sqrt{b} q_j f + \frac{\lambda_v^+}{\lambda^-} \leq 0.4\epsilon$. Also, α given in the Theorem again satisfies the requirements of the lemma. Applying the PCA-SDDN lemma, and using (5.9) to bound $q \equiv q_j$,

$$\begin{aligned}
\text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) &\leq \max(\text{no-lev}, 0.25 q_j) \\
&\leq \max(\text{no-lev}, 0.25 \cdot 1.2(\epsilon_{j-1} + \Delta_{tv})) = \epsilon_j
\end{aligned}$$

This proves our claim.

Bounds on error in estimating $\tilde{\boldsymbol{\ell}}_t$. From (5.4), $\hat{\boldsymbol{\ell}}_t - \tilde{\boldsymbol{\ell}}_t = -\boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{I}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}\tilde{\boldsymbol{\ell}}_t$ with $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}{}^\top$ for $t \in \mathcal{J}_j$. Using this, $\tilde{\boldsymbol{\ell}}_t = \boldsymbol{\ell}_t + \boldsymbol{v}_t = \boldsymbol{P}_j\boldsymbol{a}_t + \boldsymbol{v}_t$, and Fact 5.77, we can get

$$\|\hat{\boldsymbol{\ell}}_t - \tilde{\boldsymbol{\ell}}_t\| \le \mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j)\|\boldsymbol{\ell}_t\| + \|\boldsymbol{v}_t\| \le (\epsilon_{j-1} + \Delta_{tv})\|\boldsymbol{\ell}_t\| + \|\boldsymbol{v}_t\|$$

Using the same approach that we used to derive (5.4), we get that $\hat{\hat{\boldsymbol{\ell}}}_t - \tilde{\boldsymbol{\ell}}_t$ has the same expression as $\hat{\boldsymbol{\ell}}_t - \tilde{\boldsymbol{\ell}}_t$ but with $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_j\hat{\boldsymbol{P}}_j{}^\top$ for $t \in \mathcal{J}_j$. Thus,

$$\|\hat{\hat{\boldsymbol{\ell}}}_t - \tilde{\boldsymbol{\ell}}_t\| \le \mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j)\|\boldsymbol{\ell}_t\|_2 + \|\boldsymbol{v}_t\| \le \epsilon_{j-1}\|\boldsymbol{\ell}_t\| + \|\boldsymbol{v}_t\|$$

5.3.5 Proof of Theorem 5.75

The proof again follows by using the PCA-SDDN lemma given above along with use of Fact 5.77. The main difference is the use of the following idea.

Consider the interval just before the subspace change, i.e., the j-th interval with $j = j_\gamma - 1$. At this time, by our delay assumption, $\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \le$ 2no-lev and thus, using Fact 5.77, $\|\boldsymbol{I}_{\mathcal{T}_t}{}^\top \hat{\boldsymbol{P}}_j\| \le$ 2no-lev + 0.1. Also, using Fact 5.77,

$$\begin{aligned}
&\max_t \|(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{P}_j\| \\
&\le \max_t \|(\boldsymbol{\Psi}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}_{\mathcal{T}_t})^{-1}\|\|\boldsymbol{I}_{\mathcal{T}_t}^\top \boldsymbol{\Psi}\boldsymbol{P}_j\| \\
&\le \frac{1}{1-(\text{2no-lev}+0.1)^2} \cdot (\|\boldsymbol{I}_{\mathcal{T}_t}^\top \hat{\boldsymbol{P}}_{j-1}\| + \|\boldsymbol{I}_{\mathcal{T}_t}^\top \boldsymbol{P}_j\|) \\
&\le \frac{1}{1-(\text{2no-lev}+0.1)^2} \cdot ((0.1 + \text{2no-lev}) + 0.1)
\end{aligned}$$

Combining with the bound from the previous section, the final bound for this term is

$$\frac{\min(\mathrm{SE}(\hat{\boldsymbol{P}}_{j-1}, \boldsymbol{P}_j), ((0.1 + \text{2no-lev}) + 0.1))}{1-(\text{2no-lev}+0.1)^2}$$

5.4 Federated Over-Air Robust ST-Miss

In this section, we study robust ST-miss in the federated, over-air learning paradigm. There are two important distinctions with respect to the centralized ST-miss problem from Sec. 5.3 namely (a) data is now available across different nodes and the proposed algorithm must obey the federated

data sharing constraints and (b) the proposed algorithm must be able to deal with gross and sparse outliers. A key observation that allows us to build upon Sec. 5.3 is that only Line 10 of Algorithm 10 needs to be federated (all other operations are performed locally on each vector). To this end, we first explain why tackling iteration noise is sufficient to satisfy the Fed-OA constraints in Sec. 5.4.1, we then present our result for PCA in the Fed-OA setting in Sec. 5.4.2 (federated version of Line 10 of Algorithm 10), and finally show how this is used to develop an algorithm that solves Robust ST-Miss in the Fed-OA setting in Sec. 5.4.3.

5.4.1 Dealing with mild asynchrony and channel fading

As discussed previously, the three key challenges while working with over-air aggregation are (a) small timing mismatches, (b) channel fading, and (c) iteration noise. There exist a plethora of techniques within physical layer communications for dealing with channel fading and mild asynchrony. The main idea is to use carefully designed pilot sequences. Pilot sequences are symbols that the transmitter-receiver pairs agree on in advance and are transmitted in the beginning of a data frame. For instance, suppose that there are only $K = 2$ transmitters and the relative offsets between the transmitters is at most j symbols. In this case, both transmitters can use pilot sequences of length $2j + 1$, $[a_1, a_1, \ldots, a_1]$ and $[a_2, a_2, \ldots, a_2]$ respectively. Since the offset is at most j, the central node receives at least one symbol with values $a_1 + a_2$. It can determine the relative offset by determining the start location of the value $a_1 + a_2$. Once the estimated offset is communicated back to the nodes, the center can then receive the correct sum by having the nodes appropriately zero pad their transmissions. Extensions of these ideas can be utilized to handle the case of $K > 2$ nodes. Similarly, channel fading is compensated for by estimating the fading coefficients which can be done since the values of the pilot symbols are assumed to be known. These techniques are by now quite well-known in the single and multiple antenna scenarios [38]. Thus, the main problem to be addressed is iteration noise which is the focus of this paper.

5.4.2 Federated Over-Air PCA via the Power Method (PM)

Here we provide a result for subspace learning while obeying the federated data sharing constraints.

Problem setting. The goal of PCA (subspace learning) is to compute an r-dimensional subspace approximation in which a given data matrix $\boldsymbol{Z} \in \boldsymbol{R}^{n\times d}$ approximately lies. The k-th node observes a columns' sub-matrix $\boldsymbol{Z}_k \in \boldsymbol{R}^{n\times d_k}$. We have $\boldsymbol{Z} := [\boldsymbol{Z}_1, \cdots, \boldsymbol{Z}_k, \cdots, \boldsymbol{Z}_K] \in \boldsymbol{R}^{n\times d}$ with $d = \sum_{k=1}^{K} d_k$ and the goal of PCA is to find an $n \times r$ basis matrix $\boldsymbol{U}$ that minimizes $\|\boldsymbol{Z} - \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{Z}\|_F^2$. As is well known, the solution, $\boldsymbol{U}$, is given by the top r eigenvectors of $\boldsymbol{Z}\boldsymbol{Z}^\top$. Thus the goal is to estimate the span of $\boldsymbol{U}$ in a federated over-air (FedOA) fashion.

Federated Over-Air Power Method (FedOA-PM). The simplest algorithm for computing the top eigenvectors is the Power Method (PM) [11]. The distributed PM is well known, but most previous works assume the iteration-noise-free setting, e.g., see the review in [42]. On the other hand, there is recent work that studies the iteration-noise-corrupted PM [15, 4] but in the centralized setting. In this line of work, the authors consider two models for iteration-noise. The noise could either be deterministic, or statistical noise could be added to ensure differential privacy. Our setting is easier than the deterministic noise model, since we assume a statistical channel noise model, but is harder than the privacy setting since we do not have control over the amount of noise observed at the central server (here use the term channel noise and iteration-noise interchangeably).

The vanilla PM estimates $\boldsymbol{U}$ by iteratively updating $\tilde{\boldsymbol{U}}_l = \boldsymbol{Z}\boldsymbol{Z}^\top \hat{\boldsymbol{U}}_{l-1}$ followed by QR decomposition to get $\hat{\boldsymbol{U}}_l$. FedOA-PM approximates this computation as follows. At iteration l, each node k computes $\tilde{\boldsymbol{U}}_{k,l} := \boldsymbol{Z}_k \boldsymbol{Z}_k^\top \hat{\boldsymbol{U}}_{l-1}$ and synchronously transmits it to the central server which receives the sum corrupted by channel noise, i.e., it receives

$$\tilde{\boldsymbol{U}}_l := \sum_{k=1}^{K} \tilde{\boldsymbol{U}}_{k,l} + \boldsymbol{W}_l = \boldsymbol{Z}\boldsymbol{Z}^\top \hat{\boldsymbol{U}}_{l-1} + \boldsymbol{W}_l.$$

since $\sum_k \boldsymbol{Z}_k \boldsymbol{Z}_k^\top = \boldsymbol{Z}\boldsymbol{Z}^\top$. Here $\boldsymbol{W}_l$ is the channel noise. It then computes a QR decomposition of $\tilde{\boldsymbol{U}}_l$ to get a basis matrix $\hat{\boldsymbol{U}}_l$ which is broadcast to all the K nodes for use in the next iteration. We summarize this complete FedOA-PM algorithm in Algorithm 11. If no initialization is available,

Algorithm 11 FedOA-PM: Federated Over-Air PM

Require: $\boldsymbol{Z}$ (data matrix), r (rank), L (# iterations), $\hat{\boldsymbol{U}}_0$ (optional initial subspace estimate)

1: K nodes, $\boldsymbol{Z}_k \in \boldsymbol{R}^{n \times d_k}$ local data at k-th node.
2: If no initial estimate provided, at central node, do $\tilde{\boldsymbol{U}}_0 \overset{i.i.d.}{\sim} \mathcal{N}(0, I)_{n \times r}$; $\hat{\boldsymbol{U}}_0 \leftarrow \tilde{\boldsymbol{U}}_0$, transmit to all K nodes.
3: **for** $l = 1, \ldots, L$ **do**
4: At k-th node, for all $k \in [K]$, compute $\tilde{\boldsymbol{U}}_{k,l} = \boldsymbol{Z}_k \boldsymbol{Z}_k^\top \hat{\boldsymbol{U}}_{l-1}$
5: All K nodes transmit $\tilde{\boldsymbol{U}}_{k,l}$ synchronously to central node.
6: Central node receives $\tilde{\boldsymbol{U}}_l := \sum_k \tilde{\boldsymbol{U}}_{k,l} + \boldsymbol{W}_l$.
7: Central node computes $\hat{\boldsymbol{U}}_l \boldsymbol{R}_l \overset{QR}{\leftarrow} \boldsymbol{U}_l$
8: Central node broadcasts $\hat{\boldsymbol{U}}_l$ to all nodes
9: **end for**
10: At k-th node, compute $\tilde{\boldsymbol{U}}_{k,L+1} = \boldsymbol{Z}_k \boldsymbol{Z}_k^\top \hat{\boldsymbol{U}}_L$
11: All K nodes transmit $\tilde{\boldsymbol{U}}_{k,L+1}$ synchronously to the central node.
12: Central node receives $\tilde{\boldsymbol{U}}_{L+1} := \sum_k \tilde{\boldsymbol{U}}_{k,L+1} + \boldsymbol{W}_{L+1}$
13: Central node computes $\hat{\boldsymbol{\Lambda}} = \hat{\boldsymbol{U}}_L^\top \tilde{\boldsymbol{U}}_{L+1}$ and its top eigenvalue, $\hat{\sigma}_1 = \lambda_{\max}(\hat{\boldsymbol{\Lambda}})$.

Ensure: $\hat{\boldsymbol{U}}_L$, $\hat{\sigma}_1$.

it starts with a random initialization. When we use FedOA-PM for subspace tracking in the next section, the input will be the subspace estimate from the previous time instant.

We use σ_i to denote the i-th largest eigenvalue of $\boldsymbol{Z}\boldsymbol{Z}^\top$, i.e., $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_n \geq 0$. We have the following guarantee for Algorithm 11.

Lemma 5.78 (FedOA-PM)**.** *Consider Algorithm 11. Pick the desired final accuracy $\epsilon \in (0, 1/3)$. Assume that, at each iteration, the channel noise $\boldsymbol{W}_l \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_c^2)$ with (i) $\sigma_c < \epsilon\sigma_r/(5\sqrt{n})$ and (ii) $R := \sigma_{r+1}/\sigma_r < 0.99$.*

When using random initialization, if the number of iterations, $L = \Omega\left(\frac{1}{\log(1/R)} \log\left(\frac{nr}{\epsilon}\right)\right)$. then, with probability at least $0.9 - L\exp(-cr)$, $\mathrm{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_L) \leq \epsilon$.

When using an available initialization with $\mathrm{SE}(\hat{\boldsymbol{U}}_0, \boldsymbol{U}) < \epsilon_0$, if $L = \Omega\left(\frac{1}{\log(1/R)} \log\left(\frac{1}{\epsilon\sqrt{1-\epsilon_0^2}}\right)\right)$, then, with probability at least $1 - L\exp(-cr)$, $\mathrm{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_L) \leq \epsilon$.

Lemma 5.78 is similar to the one proved in [15, 4] for private PM but with a few key differences which we discuss in the Supplementary Material (Appendix 5.10) due to space constraints. We also provide a guarantee for the convergence of the maximum eigenvalue (Lines $10 - 13$ of Algorithm 11) below.

Lemma 5.79 (FedOA-PM: Maximum eigenvalue)**.** *Let σ_i be the i-th largest eigenvalue of $\boldsymbol{Z}\boldsymbol{Z}^\top$. Under the assumptions of Lemma 5.78, $\hat{\sigma}_1$ computed in line 13 of Algorithm 11 satisfies*

$$(1-4\epsilon^2)\sigma_1 - \epsilon^2\sigma_{r+1} - \epsilon\sigma_r \leq \hat{\sigma}_1 \leq (1+\epsilon)\sigma_1$$

To our best knowledge, the Lemma 5.79 has not been proved in earlier work. This result is useful because threshlding the top eigenvalue of an appropriately defined matrix is typically used for subspace change detection, see for example [28]. The proof of Lemma 5.79 given in Supplementary Material requires use of Weyl's inequality and the careful bounding of two error terms.

Note: The reason we obtain a constant probability 0.9 in the Lemma 5.78 is as follows: for any given r-dimensional subspace, $\boldsymbol{U}$ and a random Gaussian matrix $\hat{\boldsymbol{U}}$, the matrix $\hat{\boldsymbol{U}}^\top\boldsymbol{U}$ is an $r \times r$ random Gaussian matrix with independent entries. The singular values of $\hat{\boldsymbol{U}}^\top\boldsymbol{U}$ equal the cosine of the r principal angles between $\hat{\boldsymbol{U}}_0$ and $\boldsymbol{U}$. For successfully estimation (through *any* iterative method) it is necessary that none of the principal angles are $\pi/2$. To ensure this, we need to lower bound the smallest singular value of $\hat{\boldsymbol{U}}^\top\boldsymbol{U}$. This is difficult because the smallest singular value of square or "almost" square random matrices can be arbitrarily close to zero [35, 36]. The same issue is also seen in [15, 4] [4]. In fact, this is an issue for any randomized algorithm for estimating only the top r singular vectors (without a full SVD), e.g., see [27, 26, 17].

We next define the federated over-air robust subspace tracking with missing entries (Fed-OA-RSTMiss) problem, and show how Algorithm 11 and Lemma 5.78 is used to solve Fed-OA-RSTMiss.

5.4.3 Fed-OA-RSTMiss: Problem setting

In this section, we use α_k to denote the number of data points at node k at time t and $\alpha := \sum_k \alpha_k$ to denote the total number at time t. We do this to differentiate from d (in Sec. 5.4.2) which is used to indicate the total number of data vectors. Thus, at time t, $d = t\alpha$ and $d_k = t\alpha_k$. At time t and node k, we observe a possibly incomplete and noisy data matrix $\boldsymbol{Y}_{k,t}$ of dimension $n \times \alpha_k$ with the missing entries being replaced by a zero. This means the following: let $\tilde{\boldsymbol{L}}_{k,t}$ denote the

[4] These papers also provide a more general result that allows one to compute an r'-dimensional subspace approximation for an $r' > r$. If r' is picked sufficiently large, e.g., if $r' = 2r$, then the guarantee holds with probability at least $1 - 0.1^r$.

unknown, complete, approximately low-rank matrix at node k at time t. Then

$$\boldsymbol{Y}_{k,t} = \mathcal{P}_{\Omega_{k,t}}(\tilde{\boldsymbol{L}}_{k,t} + \boldsymbol{G}_{k,t}) = \mathcal{P}_{\Omega_{k,t}}(\tilde{\boldsymbol{L}}_{k,t}) + \boldsymbol{S}_{k,t}$$

where $\boldsymbol{G}_{k,t}$'s are sparse outliers and $\boldsymbol{S}_{k,t} := \mathcal{P}_{\Omega_{k,t}}(\boldsymbol{G}_{k,t})$, and $\mathcal{P}_{\Omega_{k,t}}$ sets entries outside the set $\Omega_{k,t}$ to zero. The full matrix available from all nodes at time t is denoted $\boldsymbol{Y}_t := [\boldsymbol{Y}_{1,t}, \boldsymbol{Y}_{2,t}, \ldots, \boldsymbol{Y}_{K,t}]$. This is of size $n \times \alpha$. The true (approximately) rank-r matrix $\tilde{\boldsymbol{L}}_t$ is similarly defined. Define the index sets $\mathcal{I}_{1,t} := [1, 2, \ldots, \alpha_1]$, $\mathcal{I}_{2,t} := [\alpha_1 + 1, \alpha_1 + 2, \ldots, \alpha_1 + \alpha_2]$ and so on. Denote the i-th column of $\boldsymbol{Y}_t$ by $\boldsymbol{y}_i$, $i = 1, 2, \ldots, \alpha$. And with slight abuse of notation, we define (the matrix binary masks) $\Omega_{1,t} := [(\mathcal{T}_{1,t})^c, (\mathcal{T}_{2,t})^c, \cdots, (\mathcal{T}_{\alpha_1,t})^c]$, $\Omega_{2,t} := [(\mathcal{T}_{\alpha_1+1,t})^c, (\mathcal{T}_{\alpha_1+2,t})^c, \cdots, (\mathcal{T}_{\alpha_1+\alpha_2,t})^c]$ and so on where $\mathcal{T}_{i,t}$ is the set of missing entries in column i of the data matrix at time t, $(\mathcal{T}_{i,t})^c$ is its complement w.r.t $[n]$. Thus, the observations satisfy

$$\boldsymbol{y}_i = \mathcal{P}_{\mathcal{T}_{i,t}^c}(\tilde{\boldsymbol{\ell}}_i) + \boldsymbol{s}_i, \quad i \in \mathcal{I}_{k,t}, \quad k \in [K] \tag{5.10}$$

where $\boldsymbol{s}_i$ are sparse vectors with support $\mathcal{T}_{\text{sparse},i}$. Notice that it is impossible to reover $\boldsymbol{g}_i$ on the set $\mathcal{T}_{i,t}$ and so by definition, $\mathcal{T}_{\text{sparse},i}, \mathcal{T}_{i,t}$ are disjoint. Let $\boldsymbol{P}_t$ denote the ($n \times r$ dimensional) matrix of top r left singular vectors of $\tilde{\boldsymbol{L}}_t$. In general, our assumptions imply that $\tilde{\boldsymbol{L}}_t$ is only approximately rank r. As done in our result for ST-miss (in a centralized setting), we define the matrix of the principal subspace coefficients at time t as $\boldsymbol{A}_t := \boldsymbol{P}_t^\top \tilde{\boldsymbol{L}}_t$, the rank-$r$ approximation, $\boldsymbol{L}_t := \boldsymbol{P}_t \boldsymbol{P}_t^\top \tilde{\boldsymbol{L}}_t$ and the "noise" orthogonal to the span($\boldsymbol{P}_t$) as $\boldsymbol{V}_t := \tilde{\boldsymbol{L}}_t - \boldsymbol{L}_t$. With these definitions, for all $i \in \mathcal{I}_{k,t}$ and $k \in [K]$, we can equivalently express the measurements as follows

$$\begin{aligned}
\boldsymbol{y}_i &= \mathcal{P}_{\mathcal{T}_{i,t}^c}(\tilde{\boldsymbol{\ell}}_i) + \boldsymbol{s}_i \\
&= \tilde{\boldsymbol{\ell}}_i - \boldsymbol{I}_{\mathcal{T}_{i,t}} \boldsymbol{I}_{\mathcal{T}_{i,t}}^\top \tilde{\boldsymbol{\ell}}_i + \boldsymbol{s}_i \\
&:= \tilde{\boldsymbol{\ell}}_i + \boldsymbol{z}_i + \boldsymbol{s}_i \\
&= \boldsymbol{\ell}_i + \boldsymbol{z}_i + \boldsymbol{s}_i + \boldsymbol{v}_i
\end{aligned}$$

The goal is to track the subspaces $\boldsymbol{P}_t$ quickly and reliably, and hence also reliably estimate the columns of the rank r matrix $\boldsymbol{L}_t$, under the FedOA constraints given earlier. Our problem can also be understood as a dynamic (changing subspace) version of robust matrix completion [9].

5.4.4 Algorithm

The overall idea of the solution is similar to that for ST-miss. The algorithm still consists of two parts: (a) obtain *an* estimate of the columns $\tilde{\boldsymbol{L}}_t$ using the previous subspace estimate $\hat{\boldsymbol{P}}_{t-1}$; and (b) use this estimated matrix $\hat{\boldsymbol{L}}_t$ to update the subspace estimate, i.e., obtain $\hat{\boldsymbol{P}}_t$ by r-SVD. The algorithm can be initialized via r-SVD (as done in ST-miss) if we assume that $\boldsymbol{Y}_1$ (the set of data available at $t = 1$) contains no outliers and if not, one would need to use a batch RPCA approach such as AltProj [32] to obtain the initial subspace estimate $\hat{\boldsymbol{P}}_1$.

In the federated setting (a) is done locally at each node, while (b) requires a Fed-OA algorithm for SVD which is done using Algorithm 11. If one were to consider a federated but noise-free setting, there would be no need for new analysis (standard guarantees for PM would apply).

For step (a) (obtaining an estimate of $\tilde{\boldsymbol{L}}_t$ column-wise), we use the projected Compressive Sensing (CS) idea [33]. This relies on the slow-subspace change assumption. Let $\hat{\boldsymbol{P}}_{t-1}$ denote the subspace basis estimate from the previous time and let $\boldsymbol{\Phi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{P}}_{t-1}^\top$. Projecting $\boldsymbol{y}_i$ orthogonal to $\hat{\boldsymbol{P}}_{t-1}$ helps mostly nullify $\boldsymbol{\ell}_i$ but gives projected measurements of the missing entries, $\boldsymbol{I}_{\mathcal{T}_i}\boldsymbol{I}_{\mathcal{T}_i}^\top\boldsymbol{\ell}_i$ and the sparse outliers, $\boldsymbol{s}_i$ as follows

$$\boldsymbol{\Phi}\boldsymbol{y}_i = \underbrace{\boldsymbol{\Phi}(\boldsymbol{s}_i - \boldsymbol{I}_{\mathcal{T}_i}\boldsymbol{I}_{\mathcal{T}_i}^\top\boldsymbol{\ell}_i)}_{\text{projected sparse vector}} + \underbrace{\boldsymbol{\Phi}(\boldsymbol{\ell}_i + \boldsymbol{v}_i)}_{\text{error}}$$

If the previous subspace estimate is good enough, and the noise is small, the error term above will be small. Now recovering the vector $\boldsymbol{s}_i - \boldsymbol{I}_{\mathcal{T}_i}\boldsymbol{I}_{\mathcal{T}_i}^\top\boldsymbol{\ell}_i$ is from $\boldsymbol{\Phi}\boldsymbol{y}_i$ is a problem of noisy compressive sensing with partial support knowledge (since we know $\mathcal{T}_i$). We first recover the support of $\boldsymbol{s}_i$ using the approach of [24], and then perform a least-squares based debiasing to estimate the magnitude of the entries. Following this, *an* estimate of the true data, $\hat{\boldsymbol{\ell}}_i$ is computed by subtraction from the observed data $\boldsymbol{y}_i$. We show in Lemma 5.83 that $\hat{\boldsymbol{\ell}}_i$ satisfies

$$\hat{\boldsymbol{\ell}}_i = \boldsymbol{\ell}_i - \boldsymbol{I}_{\hat{\mathcal{T}}_i}\left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}\right)^{-1}\boldsymbol{I}_{\hat{\mathcal{T}}_i}^\top\boldsymbol{\Psi}(\boldsymbol{\ell}_i + \boldsymbol{v}_i) + \boldsymbol{v}_i \tag{5.11}$$

Now we have $\hat{\boldsymbol{L}}_t := [\hat{\boldsymbol{L}}_{1,t}, \hat{\boldsymbol{L}}_{2,t}, \cdots, \hat{\boldsymbol{L}}_{K,t}]$ with $\hat{\boldsymbol{L}}_{k,t}$ available only at node k. To goal is to compute an estimate ($\hat{\boldsymbol{P}}_t$) of its top r left singular vectors while obeying the federated data sharing

Algorithm 12 Fed-OA-RSTMiss-NoDet

Require: $\boldsymbol{Y}$, $\mathcal{T}$

Parameters: $L \leftarrow C \log(1/\text{no-lev})$, ω_{supp}, ξ, α

Init: $\tau \leftarrow 1$, $j \leftarrow 1$, $\hat{\boldsymbol{P}}_1$

for $t > 1$ **do**

  $\hat{\boldsymbol{L}}_t \leftarrow$ Fed-ModCS($\boldsymbol{y}_i$, $\mathcal{I}_{k,t}$, $\mathcal{T}_i$, $\hat{\boldsymbol{P}}_{t-1}$)

  $\hat{\boldsymbol{P}}_t \leftarrow$ FedOA-PM($\hat{\boldsymbol{L}}_t$, r, L, $\hat{\boldsymbol{P}}_{t-1}$)

  $\hat{\hat{\boldsymbol{L}}}_t \leftarrow$ Fed-ModCS($\boldsymbol{y}_i$, $\mathcal{I}_{k,t}$, $\mathcal{T}_i$, $\hat{\boldsymbol{P}}_t$) ▷ optional

end for

Ensure: $\hat{\boldsymbol{P}}$

Algorithm 13 Federated Modified Compressed Sensing

procedure Fed-ModCS($\boldsymbol{y}_i$, $\mathcal{I}_{k,t}$, $\mathcal{T}_i$, $\hat{\boldsymbol{P}}_{t-1}$)

  for all node k, $i \in \mathcal{I}_{k,t}$ **do**

    $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{P}}_{t-1}^{\top}$

    $\tilde{\boldsymbol{y}}_i \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_i$

    $\hat{\boldsymbol{s}}_{i,cs} \leftarrow \arg\min_{\boldsymbol{s}} \|(\boldsymbol{s})_{(\mathcal{T}_i)^c}\|_1$ s.t. $\|\tilde{\boldsymbol{y}}_i - \boldsymbol{\Psi}\boldsymbol{s}\| \leq \xi$.

    $\hat{\mathcal{T}}_i \leftarrow \mathcal{T}_i \cup \{j : |(\hat{\boldsymbol{s}}_{i,cs})_j| > \omega_{supp}\}$

    $\hat{\boldsymbol{\ell}}_i \leftarrow \boldsymbol{y}_i - \boldsymbol{I}_{\hat{\mathcal{T}}_i}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i})^{\dagger}\tilde{\boldsymbol{y}}_i$.

  end for

  Output: $\hat{\boldsymbol{L}}_t$

end procedure

constraints. We implement this through FedOA-PM (Algorithm 11) with $\boldsymbol{Z}_k \equiv \hat{\boldsymbol{L}}_{k,t}$ being the data matrix at node k. We invoke FedOA-PM with an initial estimate $\hat{\boldsymbol{P}}_{t-1}$. This simple change allows the probability of success of the overall algorithm to be close to 1 rather than 0.9 which is what the result of Lemma 5.78 predicts. This result is obtained by carefully combining the result for PCA-SDDN in a centralized setting (Lemma 5.76) and the result for FedOA-PM (Lemma 5.78). The result is summarized in Lemma 5.84. Applying these results in exactly the same manner as we did in Sec. 5.3.4 (with a few minor differences we point out in the next section), we get the main result.

5.4.5 Guarantee for Fed-OA RST-miss

Before we state the main result, we need a few definitions.

Definition 5.80 (Sparse outlier fractions)**.** *Consider the $n \times \alpha$ sparse outlier matrix $\boldsymbol{S}_t := [\boldsymbol{S}_{1,t}, \ldots, \boldsymbol{S}_{K,t}]$ at time t. We use max-outlier-frac-col (max-outlier-frac-row) to denote the maximum of the fraction of non-zero elements in any column (row) of this matrix. Also define $x_{\min} = \min_{i \in \mathcal{I}_{k,t}} \min_{j \in \mathcal{T}_{\text{sparse},i}} |(\boldsymbol{s}_i)_j|$.*

Let $\lambda_v^+ := \max_{i \in \mathcal{I}_{k,t}} \|\mathbb{E}[\boldsymbol{v}_i \boldsymbol{v}_i^\top]\|$ and $\max_{i \in \mathcal{I}_{k,t}} \|\boldsymbol{v}_i\|^2 \leq Cr\lambda_v^+$ for all $k \in [K]$.

Theorem 5.81 (Federated Robust Subspace Tracking NoDet)**.** *Consider Algorithm 12. Assume that $\sqrt{\lambda_v^+/\lambda^-} :=$ no-lev ≤ 0.2. Set $L = C \log(1/\text{no-lev})$ and $\omega_{supp} = x_{\min}/2$, $\xi = x_{\min}/15$. Assume that the following hold:*

1. *At $t = 1$ we are given a $\hat{\boldsymbol{P}}_1$ s.t. $\text{SE}(\boldsymbol{P}_1, \hat{\boldsymbol{P}}_1) \leq \epsilon_{\text{init}}$.*
2. **Incoherence:** *$\boldsymbol{P}_t$'s satisfy μ-incoherence, and $\boldsymbol{a}_i$'s satisfy statistical right μ-incoherence;*
3. **Missing Entries:** *max-miss-frac-col $\in O(1/\mu r)$, max-miss-frac-row $\in O(1)$;*
4. **Sparse Outliers:** *max-outlier-frac-col $\in O(1/\mu r)$, max-outlier-frac-row $\in O(1)$;*
5. **Channel Noise:** *the channel noise seen by each FedOA-PM iteration is mutually independent at all times, isotropic, and zero mean Gaussian with standard deviation $\sigma_c \leq$ no-lev$\lambda^-/10\sqrt{n}$.*
6. **Subspace Model:** *The total data available at each time t, $\alpha \in \Omega(r \log n)$ and $\Delta_{tv} := \max_t \text{SE}(\boldsymbol{P}_{t-1}, \boldsymbol{P}_t)$ s.t.*

$$0.3\epsilon_{\text{init}} + 0.5\Delta_{tv} \leq 0.28 \quad \textit{and}$$
$$C\sqrt{r\lambda^+}(0.3^{t-1}\epsilon_{\text{init}} + 0.5\Delta_{tv}) + \sqrt{r_v\lambda_v^+} \leq x_{\min}$$

then, with probability at least $1 - 10dn^{-10}$, for $t > 1$, we have

$$\begin{aligned}
&\text{SE}(\hat{\boldsymbol{P}}_t, \boldsymbol{P}_t) \\
&\leq \max(0.3^{t-1}\epsilon_{\text{init}} + \Delta_{tv}(0.3 + 0.3^2 ... + 0.3^{t-1}), \text{no-lev}) \\
&< \max(0.3^{t-1}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \text{no-lev})
\end{aligned}$$

Also, at all times t, $\|\hat{\hat{\boldsymbol{\ell}}}_i - \boldsymbol{\ell}_i\| \leq 1.2 \cdot \text{SE}(\hat{\boldsymbol{P}}_t, \boldsymbol{P}_t)\|\boldsymbol{\ell}_i\| + \|\boldsymbol{v}_i\|$ for all $i \in \mathcal{I}_{k,t}, k \in [K]$.

Discussion. Items 2-4 of Theorem 5.81 are necessary to ensure that the RST-miss and robust matrix completion problems are well posed [9, 28]. The initialization assumption of Theorem 5.81 is different from the requirement of Theorem 5.73 due to the presence of outliers. Just performing a r-SVD on $\boldsymbol{Y}_1$ as done in Algorithm 10 does not work since even a few outliers can make the output arbitrarily far from the "true subspace". Additionally, without a "good initialization" Algorithm 12 cannot obtain good estimates of the sparse outliers since the noise in the sparse recovery step would be too large. One possibility to extend our result is to assume that there are no outliers at $t = 1$, i.e., $\boldsymbol{S}_1 = \boldsymbol{0}$ in which case, we use the initialization idea of Algorithm 10 (see Remark 5.82). Item 5 is standard in the federated learning/differential privacy literature [15, 4] as without bounds on iteration noise, it is not possible to obtain a final estimate that is close to the ground truth. Finally, consider item 6: the first part is required to ensure that the projection matrices, $\boldsymbol{\Psi}$'s satisfy the restricted isometry property [6, 24] which is necessary for provable sparse recovery (with partial support knowledge). This is a more stringent assumption than $\Delta_{tv} \leq 0.1$ assumed in Theorem 5.73 due to the presence of outliers. The second part of item 6 is an artifact of our analysis and arises due to the fact that it is hard to obtiain element-wise error bounds for Compressive Sensing.

In Theorem 5.81 we assumed that we are given a good enough initialization. If however, $\boldsymbol{S}_1$ were 0, we have the following result.

Remark 5.82. *Under the conditions of Theorem 5.81, if $\boldsymbol{S}_1 = 0$, then all conclusions of Theorem 5.81 hold with the following changes*

1. *The number of iterations is set as $L = C\log(n/\textit{no-lev})$*
2. *The subspace model (item 6 satisfies all conditions with ϵ_{init} replaced by $0.01 \cdot 0.3$*
3. *The probability of success is now $0.9 - 10dn^{-10}$.*

5.4.6 Proof Outline

Here we prove our main result for robust ST-Miss under the federated data sharing constraints. The proof relies on two main results given below – (i) the result of (centralized) RST-Miss proved

in the Supplementary Material (Appendix. 5.9) and (ii) our result for federated over-air power method from Sec. 5.4.2.

Lemma 5.83 (Projected-CS with partial support knowledge)**.** *Consider Lines* $5-7$ *of Algorithm 13. Under the conditions of Theorem 5.81, we have for all* t *and all* $i \in \mathcal{I}_{k,t}$*, the error seen by the compressed sensing step satisfies*

$$\|\boldsymbol{\Psi}(\boldsymbol{\ell}_i + \boldsymbol{v}_i)\| \leq (0.3^{t-1}\epsilon_{\text{init}} + 2.5\Delta_{tv})\sqrt{\mu r \lambda^+} + \sqrt{r_v \lambda_v^+}$$

$\|\hat{\boldsymbol{s}}_{i,cs} - \boldsymbol{s}_i\| \leq 7x_{\min}/15 < x_{\min}/2$*,* $\hat{\mathcal{T}}_{\text{sparse},i} = \mathcal{T}_{\text{sparse},i}$*, the error* $\boldsymbol{e} := \hat{\boldsymbol{\ell}}_i - \boldsymbol{\ell}_i$ *satisfies*

$$\begin{aligned}\boldsymbol{e} &= -\boldsymbol{I}_{\hat{\mathcal{T}}_i}\left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}\right)^{-1} \boldsymbol{I}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}(\boldsymbol{\ell}_i + \boldsymbol{v}_i) + \boldsymbol{v}_i, \\ &= (\boldsymbol{e}_i)_{\boldsymbol{\ell}} + (\boldsymbol{e}_i)_{\boldsymbol{v}} + \boldsymbol{v}_i \end{aligned} \tag{5.12}$$

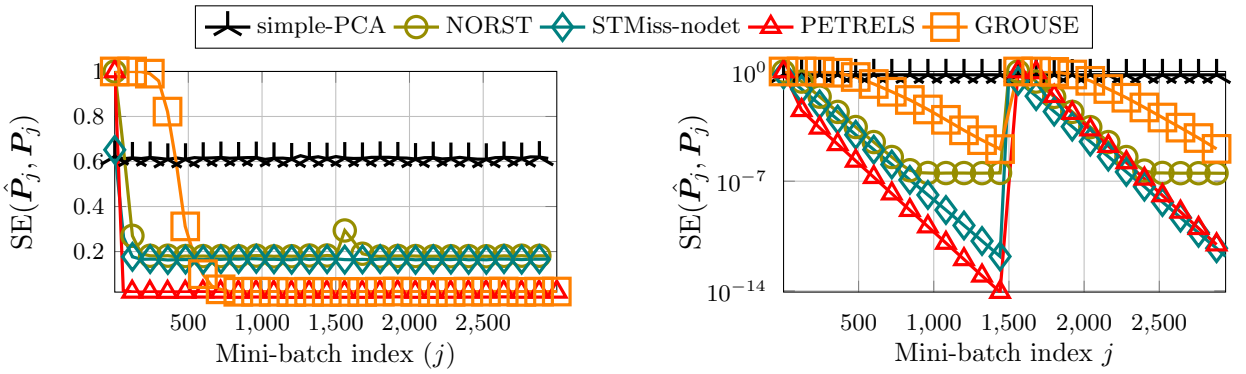
and $\|\boldsymbol{e}_i\| \leq 1.2(0.3^{t-1}\epsilon_{\text{init}} + 2.5\Delta_{tv})\sqrt{\mu r \lambda^+} + 2.2\sqrt{r_v \lambda_v^+}$*. Here,* $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{P}}_{t-1}^\top$

Lemma 5.84 (FedOA PCA-SDDN (available init))**.** *Consider the output* $\hat{\boldsymbol{P}}$ *of FedOA-PM (Algorithm 11) applied on data vectors* $\boldsymbol{z}_i$ *distributed across* K *nodes, when* $\boldsymbol{z}_i = \boldsymbol{\ell}_i + \boldsymbol{e}_i + \boldsymbol{v}_i$*,* $i = 1, 2, \ldots, \alpha$ *with* $\boldsymbol{\ell}_i = \boldsymbol{P}\boldsymbol{a}_i$*,* $\boldsymbol{e}_i = \boldsymbol{I}_{\mathcal{T}_i}\boldsymbol{B}_i\boldsymbol{\ell}_i$ *being sparse, data-dependent noise with support* $\mathcal{T}_i$*; the modeling error* $\boldsymbol{v}_i$ *is bounded with* $\max_i \|\boldsymbol{v}_i\|^2 \leq C r_v \lambda_v^+$ *where* $\lambda_v^+ := \|\mathbb{E}[\boldsymbol{v}_i \boldsymbol{v}_i^\top]\|$*. The matrix of top-*$r$ *left singular vectors,* $\boldsymbol{P}$ *satisfies* μ*-incoherence, and* $\boldsymbol{a}_i$*'s satisfy* μ*-statistical right-incoherence. The channel noise is zero mean i.i.d. Gaussian with standard deviation* $\sigma_c \leq \epsilon_{PM}\lambda^-/10\sqrt{n}$ *and is independent of the* $\boldsymbol{\ell}_i$*'s. Let* $q := \max_i \|\boldsymbol{B}_i\boldsymbol{P}\|$ *and let* b *denote the fraction of non-zeros in any row of the SDDN matrix* $\boldsymbol{E} = [\boldsymbol{e}_1, \cdots, \boldsymbol{e}_\alpha]$*. Pick an* $\epsilon_{PM} > 0$*. If*

$$7\sqrt{b}qf + \lambda_v^+/\lambda^- < 0.4\epsilon_{PM},$$

$\alpha \geq Cr\log n \max(\frac{q^2}{\epsilon_{PM}^2}f^2, \frac{\frac{\lambda_v^+}{\lambda^-}}{\epsilon_{PM}^2}f)$*, and if FedOA-PM is initialized with a matrix* $\boldsymbol{P}_{\text{init}}$ *such that* $\text{SE}(\boldsymbol{P}_{\text{init}}, \boldsymbol{P}) \leq \epsilon_{\text{init},PM}$*, then after* $L = C\log(1/(\epsilon_{PM}\sqrt{1-\epsilon_{\text{init},PM}^2}))$ *iterations, with probability at least* $1 - L\exp(-cr) - n^{-10}$*,* $\hat{\boldsymbol{P}}$ *satisfies* $\text{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon_{PM}$*.*

With these two Lemmas, the proof of Theorem 5.81 is similar to the proof of Theorem 5.73. Firstly, consider the projected CS with partial support knowledge step. Lemma 5.83 applied to



(a) Small rotations at each time

(b) Piecewise Constant Subspace – Large

Figure 5.1: Comparison of ST-Miss Algorithms in the centralized setting.

each vector locally gives us $\hat{\boldsymbol{\ell}}_i = \boldsymbol{\ell}_i - \boldsymbol{e}_i$ with $\boldsymbol{e}_i$ satisfying (5.12). Next, at each time t, we update the subspace as the top r left singular vectors of $\hat{\boldsymbol{L}}_t$, where the k-th node only has access to the sub-matrix $\hat{\boldsymbol{L}}_{k,t}$. For a $t > 1$, we assume that the previous subspace estimate, $\hat{\boldsymbol{P}}_{t-1}$ satisfies $\mathrm{SE}(\hat{\boldsymbol{P}}_{t-1}, \boldsymbol{P}_{t-1}) \leq \max(0.3^{t-2}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \text{no-lev})$. We invoke Lemma 5.84 with $\hat{\boldsymbol{P}}_{init} \equiv \hat{\boldsymbol{P}}_{t-1}$ and thus, $\epsilon_{\text{init},PM} \equiv \max(0.3^{t-2}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \text{no-lev})$; $\boldsymbol{z}_i \equiv \hat{\boldsymbol{\ell}}_i, i \in \mathcal{I}_{k,t}$; $\boldsymbol{P} \equiv \boldsymbol{P}_t$, $\boldsymbol{e}_i \equiv (\boldsymbol{e}_i)_{\boldsymbol{\ell}}$, $\boldsymbol{v}_i \equiv (\boldsymbol{e}_i)_{\boldsymbol{v}} + \boldsymbol{v}_i$; and $\epsilon_{PM} \equiv \max(0.3^{t-2}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \text{no-lev})$. Under the conditions of theorem 5.81, we conclude that w.h.p., $\mathrm{SE}(\hat{\boldsymbol{P}}_t, \boldsymbol{P}_t) \leq \max(0.3^{t-1}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \text{no-lev})$. Thus, applying this argument inductively proves the result. For the second optional FedOA-PM step, the same ideas from the proof of Theorem 5.73 apply.

5.5 Numerical Experiments

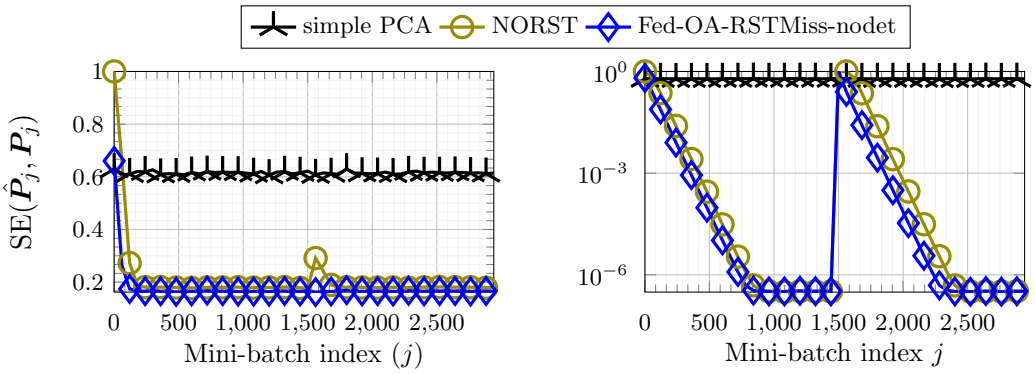
Experiments are performed on a Desktop Computer with Intel® Xeon 8-core CPU with 32GB RAM and the results are averaged over 100 independent trials. The codes are available at https://github.com/praneethmurthy/distributed-pca.

5.5.1 Centralized STMiss

Small Rotations at each time. We first consider the centralized setting for Subspace Tracking with missing data (Sec. 5.3). We demonstrate results under two sets of subspace change

models. First we consider the "rotation model" that has been commonly used in the literature [10, 48]. At each time t, we generate a $n \times r$ dimensional subspace $\boldsymbol{P}_{(t)} = e^{-\delta_t B_t} \boldsymbol{P}_{(t-1)}$ with $\boldsymbol{P}_{(0)}$ generated by orthonormalizing the columns of a i.i.d. standard Gaussian matrix and B_t is some skew symmetric matrix to simulate rotations and δ_t controls the amount of rotation (for this experiment we set $\delta_t = 10^{-4}$ which ensures that $\Delta_{tv} \approx 10^{-2}$). We generate matrix $\tilde{A}$ as a i.i.d. uniform random matrix of size $r \times d$ and set the t-th column of the true data matrix $\tilde{\boldsymbol{\ell}}_t = \boldsymbol{P}_{(t)} \tilde{\boldsymbol{a}}_t$. Thus, in the notation of our result, $\boldsymbol{P}_j$ is the matrix of the top r left singular vectors of $\tilde{\boldsymbol{L}}_j = [\tilde{\boldsymbol{\ell}}_{(j-1)\alpha+1}, \cdots \tilde{\boldsymbol{\ell}}_{j\alpha}]$ and $\boldsymbol{A}_j = \boldsymbol{P}_j^\top \tilde{\boldsymbol{L}}_j$. In all experiments, we choose $n = 1000$ and $d = 3000$. We simulate the set of observed entries using a bernoulli model where each element of the matrix is observed with probability 0.9. For all experiments, we set $r = 30$ and the fraction of missing entries to be 0.1. We implement STMiss-nodet (Algorithm 10) and set $r = 30$. We compare with NORST [28] (the state-of-the-art theoretically), GROUSE [48], and PETRELS [10] (the state-of-the-art experimentally). For all algorithms, we used default parameters mentioned in the codes. We also implement the simple PCA method wherein we estimate $\hat{\boldsymbol{P}}_j$ as the top-r left singular vectors of $\boldsymbol{Y}_j$ for each mini-batch. For all algorithms, the mini-batch size was chosen as $\alpha = 60$. The results are shown in Fig. 5.1(a). We see that as specified by Theorem 5.74, the simple PCA algorithm does not improve the recovery errors since it is not exploiting slow subspace change. However, all other algorithms exploit slow-subspace change and thus are able to provide better estimates with time. We also notice that PETRELS is the fastest to converge, followed by NORST and STMiss-nodet, and finally GROUSE. This is consistent with the previous set of results in [28].

Piecewise Constant. Next, we consider a piecewise constant subspace change model that has been considered in the provable subspace tracking literature [28]. In this, we simulate a large subspace change at $t_1 = 1500$. The subspace is fixed until then, i.e., $\boldsymbol{P}_j = \boldsymbol{P}_1$ for all $j \in [1, \lceil t_1/\alpha \rceil)$ and $\boldsymbol{P}_j = \boldsymbol{P}_2$ for all $j \in [\lceil t_1/\alpha \rceil, \lceil d/\alpha \rceil]$. The results are shown in Fig. 5.1(b). Notice that NORST and STMiss-nodet significantly outperform simple PCA as both exploit slow subspace change. Additionally, even though the change is large (in the notation of Definition 5.85 given in the supplementary material, $\Delta_{\text{large}} \approx 1$ and $\Delta_{tv} = 0$), STMiss-nodet is also able to adapt without



(a) Small rotations at each time

(b) Piecewise Constant Subspace – Large

Figure 5.2: Corroborating the claims of Theorem 5.81.

requiring a detection step. Finally, since the updates are always improving, after a certain time, NORST stops improving the subspace estimates, but STMiss-nodet improves it and gets a better result.

5.5.2 Fedrated ST-Miss

We also implement Algorithm 12 to corroborate our theoretical claims. We use the exact data generation method as we did in the centralized setting. To simulate over-air communication, we replace the inbuilt `SVD` routine of matlab by a power method code snippet, and by adding iteration noise. In each iteration, we add i.i.d. Gaussian noise with variance 10^{-6}. The results are presented in Fig. 5.2. Notice that in both cases, Algorithm 12 works as well as NORST even though NORST cannot deal with iteration noise. Additionally, as opposed to the centralized setting (Fig. 5.1(b)), the error of Fed-OA-RSTMiss-nodet in Fig. 5.2 does not improve beyong the iteration noise level of 10^{-6}.

5.6 References

[1] Alistarh, D., Allen-Zhu, Z., and Li, J. Byzantine stochastic gradient descent. In *Advances in Neural Information Processing Systems* (2018), pp. 4613–4623.

[2] Amiri, M. M., and Gündüz, D. Federated learning over wireless fading channels. *arXiv preprint arXiv:1907.09769* (2019).

[3] Amiri, M. M., and Gündüz, D. Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air. In *2019 IEEE International Symposium on Information Theory (ISIT)* (2019), IEEE, pp. 1432–1436.

[4] Balcan, M.-F., Du, S. S., Wang, Y., and Yu, A. W. An improved gap-dependency analysis of the noisy power method. In *Conference on Learning Theory* (2016), pp. 284–309.

[5] Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konecny, J., Mazzocchi, S., and McMahan, H. B. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046* (2019).

[6] Candes, E. The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris Serie I* (2008).

[7] Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *J. ACM 58*, 3 (2011).

[8] Candes, E. J., and Recht, B. Exact matrix completion via convex optimization. *Found. of Comput. Math*, 9 (2008), 717–772.

[9] Cherapanamjeri, Y., Gupta, K., and Jain, P. Nearly-optimal robust matrix completion. *ICML* (2016).

[10] Chi, Y., Eldar, Y. C., and Calderbank, R. Petrels: Parallel subspace estimation and tracking by recursive least squares from partial observations. *IEEE Transactions on Signal Processing* (December 2013).

[11] Golub, G. H., and Van Loan, C. F. Matrix computations. *The Johns Hopkins University Press, Baltimore, USA* (1989).

[12] Gonen, A., Rosenbaum, D., Eldar, Y. C., and Shalev-Shwartz, S. Subspace learning with partial information. *Journal of Machine Learning Research 17*, 52 (2016), 1–21.

[13] Grammenos, A., Mendoza-Smith, R., Mascolo, C., and Crowcroft, J. Federated pca with adaptive rank estimation. *arXiv preprint arXiv:1907.08059* (2019).

[14] Gunduz, D., de Kerret, P., Sidiropoulos, N. D., Gesbert, D., Murthy, C. R., and van der Schaar, M. Machine learning in the air. *IEEE Journal on Selected Areas in Communications 37*, 10 (2019), 2184–2199.

[15] Hardt, M., and Price, E. The noisy power method: A meta algorithm with applications. In *Advances in Neural Information Processing Systems* (2014), pp. 2861–2869.

[16] Horn, R. A., and Johnson, C. R. *Matrix analysis*. Cambridge university press, 2012.

[17] Jain, P., Jin, C., Kakade, S. M., Netrapalli, P., and Sidford, A. Streaming pca: Matching matrix bernstein and near-optimal finite sample guarantees for oja?s algorithm. In *Conference on learning theory* (2016), pp. 1147–1164.

[18] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977* (2019).

[19] Konecny, J., McMahan, H. B., Ramage, D., and Richtárik, P. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527* (2016).

[20] Konecny, J., McMahan, H. N., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).

[21] Kopsinis, Y., Chouvardas, S., and Theodoridis, S. Distributed robust subspace tracking. In *2015 23rd European Signal Processing Conference (EUSIPCO)* (2015), IEEE, pp. 2531–2535.

[22] Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine 37*, 3 (2020), 50–60.

[23] Liang, Y., Balcan, M.-F. F., Kanchanapally, V., and Woodruff, D. Improved distributed principal component analysis. In *NIPS* (2014), pp. 3113–3121.

[24] Lu, W., and Vaswani, N. Modified basis pursuit denoising (modified-bpdn) for noisy compressive sensing with partially known support. In *IEEE Intl. Conf. Acous. Speech. Sig.Proc.(ICASSP)* (2010).

[25] Mackey, L., Talwalkar, A., and Jordan, M. I. Distributed matrix completion and robust factorization. *The Journal of Machine Learning Research 16*, 1 (2015), 913–960.

[26] Mitliagkas, I., Caramanis, C., and Jain, P. Memory limited, streaming pca. In *Advances in neural information processing systems* (2013), pp. 2886–2894.

[27] Musco, C., and Musco, C. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *Advances in Neural Information Processing Systems* (2015), pp. 1396–1404.

[28] Narayanamurthy, P., Daneshpajooh, V., and Vaswani, N. Provable subspace tracking from missing data and matrix completion. *IEEE Transactions on Signal Processing* (2019), 4245–4260.

[29] Narayanamurthy, P., and Vaswani, N. Provable dynamic robust pca or robust subspace tracking. *IEEE Transactions on Information Theory 65*, 3 (2019), 1547–1577.

[30] Narayanamurthy, P., and Vaswani, N. Fast robust subspace tracking via pca in sparse data-dependent noise. *Journal of Selected Areas in Information Theory* (2021).

[31] Netrapalli, P., Jain, P., and Sanghavi, S. Low-rank matrix completion using alternating minimization. In *STOC* (2013).

[32] Netrapalli, P., Niranjan, U. N., Sanghavi, S., Anandkumar, A., and Jain, P. Non-convex robust pca. In *NIPS* (2014).

[33] Qiu, C., Vaswani, N., Lois, B., and Hogben, L. Recursive robust pca or recursive sparse recovery in large but structured noise. *IEEE Trans. Info. Th.* (August 2014), 5007–5039.

[34] Recht, B. A simpler approach to matrix completion. *Journal of Machine Learning Research 12*, Dec (2011), 3413–3430.

[35] Rudelson, M., and Vershynin, R. The littlewood–offord problem and invertibility of random matrices. *Advances in Mathematics 218*, 2 (2008), 600–633.

[36] Rudelson, M., and Vershynin, R. Smallest singular value of a random rectangular matrix. *Communications on Pure and Applied Mathematics 62*, 12 (2009), 1707–1739.

[37] Teflioudi, C., Makari, F., and Gemulla, R. Distributed matrix completion. In *2012 ieee 12th international conference on data mining* (2012), IEEE, pp. 655–664.

[38] Tse, D., and Viswanath, P. *Fundamentals of wireless communication*. Cambridge university press, 2005.

[39] Vershynin, R. *High-dimensional probability: An introduction with applications in data science*, vol. 47. Cambridge university press, 2018.

[40] Wang, C., Eldar, Y. C., and Lu, Y. M. Subspace estimation from incomplete observations: A high-dimensional analysis. *JSTSP* (2018).

[41] Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications 37*, 6 (2019), 1205–1221.

[42] Wu, S. X., Wai, H.-T., Li, L., and Scaglione, A. A review of distributed algorithms for principal component analysis. *Proceedings of the IEEE 106*, 8 (2018), 1321–1340.

[43] Xie, C., Koyejo, S., and Gupta, I. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. *arXiv preprint arXiv:1903.03936* (2019).

[44] Yang, B. Projection approximation subspace tracking. *IEEE Trans. Sig. Proc.* (1995), 95–107.

[45] Yang, K., Jiang, T., Shi, Y., and Ding, Z. Federated learning via over-the-air computation. *IEEE Transactions on Wireless Communications* (2020).

[46] Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST) 10*, 2 (2019), 1–19.

[47] Zare, A., Ozdemir, A., Iwen, M. A., and Aviyente, S. Extension of pca to higher order data structures: An introduction to tensors, tensor decompositions, and tensor pca. *Proceedings of the IEEE 106*, 8 (2018), 1341–1358.

[48] Zhang, D., and Balzano, L. Global convergence of a grassmannian gradient descent algorithm for subspace estimation. In *AISTATS* (2016).

5.7 Appendix A: Proof of Key Lemmas for Theorem 5.81

Proof of Lemma 5.83. Recall from Algorithm 13 that we need solve

$$\hat{\boldsymbol{s}}_{i,cs} = \arg\min_{\boldsymbol{s}} \|(\boldsymbol{s})_{\mathcal{T}_i^c}\|_1 \text{ s.t.} \|\boldsymbol{\Phi}\boldsymbol{y}_t - \boldsymbol{\Phi}\boldsymbol{s}\| \leq \xi$$

This is a problem of sparse recovery from partial subspace knowledge. To prove the correctness of the result, we first need to bound the s-level RIC of $\boldsymbol{\Psi} = \boldsymbol{I} - \hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{P}}_{t-1}^\top$ where $s := (2\text{max-outlier-frac-col} + \text{max-miss-frac-col}) \cdot n$. Under the assumptions of Theorem 5.81 (we only assumed that max-outlier-frac-col $\in P(1/\mu r)$ and max-miss-frac-col $\in O(1/\mu r)$ but the actual requirement is $(2\text{max-miss-frac-col} + \text{max-outlier-frac-col}) \cdot n \leq 0.01/\mu r$), and Fact 5.77, we have that

$$\begin{aligned}\delta_s(\boldsymbol{I} - \hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{P}}_{t-1}^\top) &= \max_{|\mathcal{T}|\leq s} \|\boldsymbol{I}_{\mathcal{T}}^\top \hat{\boldsymbol{P}}_{t-1}\|^2 \\ &\leq \max_{|\mathcal{T}|\leq s} (\text{SE}(\hat{\boldsymbol{P}}_{t-1}, \boldsymbol{P}_{t-1}) + \|\boldsymbol{I}_{\mathcal{T}}^\top \boldsymbol{P}_{t-1}\|)^2\end{aligned}$$

Recall that for $t > 1$, $\text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \max(0.1 \cdot 0.3^{j-1}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \text{no-lev}) \leq 0.2$ and from the incoherence assumption on $\boldsymbol{P}_t$'s, the second term above is upper bounded by 0.01. Thus, $\delta_s(\boldsymbol{\Phi}) \leq$

$0.3^2 < 0.15$. Next, consider the error seen by the modified-CS step,

$$\begin{aligned}\|\boldsymbol{b}_i\| = \|\boldsymbol{\Psi}(\boldsymbol{\ell}_i + \boldsymbol{v}_i)\| \leq & \left\|(\boldsymbol{I} - \hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{P}}_{t-1})\boldsymbol{P}_t\boldsymbol{a}_i\right\| + \|\boldsymbol{v}_i\| \\ \leq & \operatorname{SE}(\hat{\boldsymbol{P}}_{t-1}, \boldsymbol{P}_t)\,\|\boldsymbol{a}_i\| + \|\boldsymbol{v}_i\| \\ \leq & (\operatorname{SE}(\hat{\boldsymbol{P}}_{t-1}, \boldsymbol{P}_{t-1}) + \operatorname{SE}(\boldsymbol{P}_{t-1}, \boldsymbol{P}_t))\sqrt{\mu r \lambda^+} + C\sqrt{r\lambda_v^+} \\ \leq & (0.3^{t-1}\epsilon_{\text{init}} + 1.5\Delta_{tv})\sqrt{\mu r \lambda^+} + C\sqrt{r_v \lambda_v^+}\end{aligned}$$

under the assumptions of Theorem 5.81, the RHS of the above is bounded by $x_{\min}/15$. This is why we have set $\xi = x_{\min}/15$ in Algorithm 12. Using these facts, and $\delta_s(\boldsymbol{\Psi}) < 0.15$, we have that

$$\|\hat{\boldsymbol{s}}_{i,cs} - \boldsymbol{s}_i\| \leq 7\xi = 7x_{\min}/15 < x_{\min}/2$$

Consider support recovery. From above,

$$|(\hat{\boldsymbol{s}}_{i,cs} - \boldsymbol{s}_i)_m| \leq \|\hat{\boldsymbol{s}}_{i,cs} - \boldsymbol{s}_i\| \leq 7x_{\min}/15 < x_{\min}/2$$

The Algorithm sets $\omega_{supp} = x_{\min}/2$. Consider an index $m \in \mathcal{T}_{\text{sparse},i}$. Since $|(\boldsymbol{s}_i)_m| \geq x_{\min}$,

$$\begin{aligned}x_{\min} - |(\hat{\boldsymbol{s}}_{i,cs})_m| &\leq |(\boldsymbol{s}_i)_m| - |(\hat{\boldsymbol{s}}_{i,cs})_m| \\ &\leq |(\boldsymbol{s}_i - \hat{\boldsymbol{s}}_{i,cs})_m| < \frac{x_{\min}}{2}\end{aligned}$$

Thus, $|(\hat{\boldsymbol{s}}_{i,cs})_m| > \frac{x_{\min}}{2} = \omega_{supp}$ which means $m \in \hat{\mathcal{T}}_{\text{sparse},i}$. Hence $\mathcal{T}_{\text{sparse},i} \subseteq \hat{\mathcal{T}}_{\text{sparse},i}$. Next, consider any $m \notin \mathcal{T}_{\text{sparse},i}$. Then, $(\boldsymbol{s}_i)_m = 0$ and so

$$|(\hat{\boldsymbol{s}}_{i,cs})_m| = |(\hat{\boldsymbol{s}}_{i,cs})_m)| - |(\boldsymbol{s}_i)_m| \leq |(\hat{\boldsymbol{s}}_{i,cs})_m - (\boldsymbol{s}_i)_m| < \frac{x_{\min}}{2}$$

which implies $m \notin \hat{\mathcal{T}}_{\text{sparse},i}$ and $\hat{\mathcal{T}}_{\text{sparse},i} \subseteq \mathcal{T}_{\text{sparse},i}$ implying that $\hat{\mathcal{T}}_{\text{sparse},i} = \mathcal{T}_{\text{sparse},i}$ and consequently that $\hat{\mathcal{T}}_i := \mathcal{T}_i \cup \hat{\mathcal{T}}_{\text{sparse},i} = \mathcal{T}_i \cup \mathcal{T}_{\text{sparse},i}$.

With $\hat{\mathcal{T}}_{\text{sparse},i} = \mathcal{T}_{\text{sparse},i}$ and since $\mathcal{T}_{\text{sparse},i}$ is the support of $\boldsymbol{s}_i$, $\boldsymbol{s}_i = \boldsymbol{I}_{\mathcal{T}_{\text{sparse},i}}\boldsymbol{I}_{\mathcal{T}_{\text{sparse},i}}^\top \boldsymbol{s}_i$, and so

$$\begin{aligned}\hat{\boldsymbol{s}}_i &= \boldsymbol{I}_{\hat{\mathcal{T}}_i}\left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}\right)^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top(\boldsymbol{\Psi}\boldsymbol{\ell}_i + \boldsymbol{\Psi}\boldsymbol{z}_i + \boldsymbol{\Psi}\boldsymbol{s}_i + \boldsymbol{\Psi}\boldsymbol{v}_i) \\ &= \boldsymbol{I}_{\hat{\mathcal{T}}_i}\left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}\right)^{-1}\boldsymbol{I}_{\hat{\mathcal{T}}_i}^\top\boldsymbol{\Psi}(\boldsymbol{\ell}_i + \boldsymbol{v}_i) + \boldsymbol{s}_i + \boldsymbol{z}_i\end{aligned}$$

Thus, the estimate of the true-data $\hat{\boldsymbol{\ell}}_i = \boldsymbol{y}_i - \hat{\boldsymbol{s}}_i$ satisfies

$$\hat{\boldsymbol{\ell}}_i = \boldsymbol{\ell}_i + \boldsymbol{v}_i - \boldsymbol{I}_{\hat{\mathcal{T}}_i} \left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}_{\hat{\mathcal{T}}_i} \right)^{-1} \boldsymbol{I}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi} (\boldsymbol{\ell}_i + \boldsymbol{v}_i)$$

and thus $\boldsymbol{e}_i = \hat{\boldsymbol{\ell}}_i - \boldsymbol{\ell}_i$ satisfies

$$\begin{aligned}
\boldsymbol{e}_i &= -\boldsymbol{I}_{\hat{\mathcal{T}}_i} \left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}_{\hat{\mathcal{T}}_i} \right)^{-1} \boldsymbol{I}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi} (\boldsymbol{\ell}_i + \boldsymbol{v}_i) + \boldsymbol{v}_i \\
\|\boldsymbol{e}_i\| &\leq \left\| \left(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi}_{\hat{\mathcal{T}}_i} \right)^{-1} \right\| \|\boldsymbol{I}_{\hat{\mathcal{T}}_i}^\top \boldsymbol{\Psi} (\boldsymbol{\ell}_i + \boldsymbol{v}_i)\| + \|\boldsymbol{v}_i\| \\
&\leq 1.2 \|\boldsymbol{b}_i\| + \|\boldsymbol{v}_i\|
\end{aligned}$$

□

We next prove Lemma 5.84. But before we prove this, under the conditions of Lemma 5.76, the result from [30] also shows the following:

$$\begin{aligned}
\|\text{perturb}\| &:= \left\| \frac{1}{\alpha} \sum_i (\boldsymbol{z}_i \boldsymbol{z}_i^\top - \boldsymbol{\ell}_i \boldsymbol{\ell}_i^\top) \right\| \\
&\leq \left\| \frac{1}{\alpha} \sum_i \boldsymbol{e}_i \boldsymbol{e}_i^\top \right\| + 2 \left\| \frac{1}{\alpha} \sum_i \boldsymbol{\ell}_i \boldsymbol{e}_i^\top \right\| + 2 \left\| \frac{1}{\alpha} \sum_i \boldsymbol{\ell}_i \boldsymbol{v}_i^\top \right\| \\
&+ 2 \left\| \frac{1}{\alpha} \sum_i \boldsymbol{v}_i \boldsymbol{e}_i^\top \right\| + \left\| \frac{1}{\alpha} \sum_i \boldsymbol{v}_i \boldsymbol{v}_i^\top \right\|, \\
&\leq \left(6.6 \sqrt{b} q f + 4.4 \frac{\lambda_v^+}{\lambda^-} \right) \lambda^- \qquad (5.13)
\end{aligned}$$

and

$$\lambda_r \left(\frac{1}{\alpha} \sum_i \boldsymbol{\ell}_i \boldsymbol{\ell}_i^\top \right) \geq 0.99 \lambda^-.$$

Proof of Lemma 5.84. Before we prove There are the following two parts in the proof:

1. First, we show that $\hat{\boldsymbol{P}}$ is *close* to $\tilde{\boldsymbol{P}}$ where $\tilde{\boldsymbol{P}}$ is the top r left singular vectors of $\boldsymbol{Z}$. In particular, we show that $\text{SE}(\hat{\boldsymbol{P}}, \tilde{\boldsymbol{P}}) \leq \epsilon_{PM}/2$. This relies on application of Lemma 5.78 to the matrix $\boldsymbol{Z}\boldsymbol{Z}^\top/\alpha$ with the appropriate parameters.

2. Next, we use centralized Principal Components Analysis in Sparse, Data-Dependent Noise (PCA SDDN) with $\boldsymbol{z}_i \equiv \boldsymbol{y}_i$ to show that the $\tilde{\boldsymbol{P}}$ is *close* to the true subspace, $\boldsymbol{P}$. Here too we show that $\mathrm{SE}(\tilde{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon_{PM}/2$. Combining the above two results, and the triangle inequality gives $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \mathrm{SE}(\hat{\boldsymbol{P}}, \tilde{\boldsymbol{P}}) + \mathrm{SE}(\tilde{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon_{PM}$.

Notice from (5.13), with high probability, the matrix $\boldsymbol{Z}\boldsymbol{Z}^\top$ has a good eigen-gap, i.e.,

$$\begin{aligned}
\lambda_r(\boldsymbol{Z}\boldsymbol{Z}^\top) = \lambda_r(\boldsymbol{L}\boldsymbol{L}^\top + \text{perturb}) &\geq \lambda_r(\boldsymbol{L}\boldsymbol{L}^\top) - \|\text{perturb}\| \\
&\geq 0.99\lambda^- - \left(7.7\sqrt{b}qf + 4.4\frac{\lambda_v^+}{\lambda^-}\right)\lambda^- \\
\lambda_{r+1}(\boldsymbol{Z}\boldsymbol{Z}^\top) \leq \lambda_{r+1}(\boldsymbol{L}\boldsymbol{L}^\top) + \|\text{pertub}\| & \\
&\leq \left(7.7\sqrt{b}qf + 4.4\frac{\lambda_v^+}{\lambda^-}\right)\lambda^-
\end{aligned}$$

Under the assumptions of Lemma 5.84, $7.7\sqrt{b}qf + 4.4\lambda_v^+/\lambda^- \leq 2.5\epsilon_{SE}$. Thus, for this matrix, $R < 0.99$ with high probability. The standard deviation of the channel noise in each iteration satisfies, $\sigma_c \leq \epsilon_{PM}\lambda^-/10\sqrt{n}$. Furthermore, since we initialize Fed-PM with $\boldsymbol{P}_{\text{init}}$ that satisfies $\mathrm{SE}(\boldsymbol{P}_{\text{init}}, \boldsymbol{P}) \leq \epsilon_{\text{init},PM}$ it follows from second part of Lemma 5.78 that after $L = C\log(1/(\epsilon_{PM}\sqrt{1-\epsilon_{\text{init},PM}^2}))$ iterations, with probability at least $1 - L\exp(-cr)$, the output $\hat{\boldsymbol{P}}$ satisfies $\mathrm{SE}(\hat{\boldsymbol{P}}, \tilde{\boldsymbol{P}}) \leq \epsilon_{PM}/2$.

Next, observe that the conditions required to apply Lemma 5.76 is satisfied under the assumptions of Lemma 5.84. Thus, we apply Lemma 5.76 with $\epsilon_{\text{SE}} \equiv \epsilon_{PM}/2$. This ensures that with probability at least $1 - 10n^{-10}$, the eigenvectors of the empirical covariance are close to that of the the population covariance, i.e., $\mathrm{SE}(\tilde{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon_{PM}/2$.

Combining the above two results we have with probability at least $1 - L\exp(-cr) - 10n^{-10}$, $\mathrm{SE}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \mathrm{SE}(\hat{\boldsymbol{P}}, \tilde{\boldsymbol{P}}) + \mathrm{SE}(\tilde{\boldsymbol{P}}, \boldsymbol{P}) \leq \epsilon_{PM}$. □

The proof of the subspace detection step (Lemma 5.88) is similar to that of [30] applied with Lemma 5.79.

Proof of Lemma 5.78. The proof of Lemma 5.78 is a special case of Lemma 5.91 that is proved in the Supplementary Material. The proof of Lemma 5.79 is also provided in the Supplementary Material. □

The Supplementary Material is organized as follows. In Appendix 5.8, we provide the setting, algorithm and the guarantee for (a) a generalization of Theorem 5.73 wherein we provide our result to provably detect and track large, but infrequent subspace changes; and (c) a generalization of Theorem 5.81 wherein we again deal with large infrequent subspace changes but in under the federated over-air constraints. In Appendix 5.9, we provide the guarantee for robust ST-miss in a centralized setting. And finally, in Appendix 5.10, we prove the convergence of Algorithm 11, i.e., Lemma 5.78 and Lemma 5.79 (in fact, we prove a stronger result there, but only provide a special case of it in the main paper).

5.8 Appendix B: Extensions of Theorem 5.73 and Theorem 5.81

5.8.1 Generalization to detect and track larger subspace changes for centralized ST-miss

When Δ_{tv} is small enough, the bound given by Theorem 5.73 holds and is better than that for simple PCA given in Theorem 5.74. When Δ_{tv} is very small but there are occasional large changes, then the guarantee of Theorem 5.75 applies. However, the result does not guarantee change detection (only tracking), this is because the algorithm itself does not contain a detection step. In this section, we provide a modification of our algorithm that contains a detection step and a corollary that also guarantees quick enough detection. The proof is essentially a direct combination of the ideas given in the main paper and those used in [28] for quick and reliable subspace change detection.

A simple modification to Algorithm 10 given in Algorithm 14 allows us to deal with such a model. Our next result shows that under such a subspace change model, we are able to recover the result of [28]. Concretely, consider the following subspace change model

Definition 5.85 (Small frequent and abrupt infrequent subspace change model)**.** *Assume that the* γ*-th* large subspace changes *occurs at* $t = t_\gamma \alpha$ *for* $\tau = 1, \cdots \Gamma$ *such that* $t_{\gamma+1} - t_\gamma > (J^* + 2)$ *with* $J^* := C \log(1/\epsilon)$ *where* ϵ *chosen by the user denotes the desired final accuracy. In addition, assume*

that

$$\min_{\gamma\in[\Gamma]} \Delta_{t_\gamma} \geq \Delta_{\text{large}} \geq \Delta_{tv} \geq \max_{\{j:j\neq t_\gamma,\gamma\in[\Gamma]\}} \Delta_j$$

Notice that this is a generalization of the "model" considered in previous literature on provable subspace tracking [28] and the model considered above.

To deal with the large subspace changes, we need a few minor changes to Algorithm 10, which we briefly summarize below. Firstly, the algorithm now has two phases, the subspace update phase and the subspace detect phase, akin to the algorithms of [28, 30]. Second, as opposed to the algorithms studied in [28, 30], the current algorithm updates the subspace even in the detect phase (this is necessary since we only assume an approximate piecewise-constant subspace change model). The pseudo-code is provided as Algorithm 14 in the Appendix. With these changes, we have the following result

Corollary 5.86 (Subspace tracking in the presence of infrequent, abrupt changes)**.** *Assume that data satisfies the subspace change model in Definition 5.85 such that* $\Delta_{\text{large}} > 9\sqrt{f}\max(0.1\cdot 0.3^{J^*-1} + 1.5\Delta_{tv}, \textit{no-lev})$ *Then, under the conditions of Theorem 5.73 and using Algorithm 14, with probability at least* $1 - dn^{-10}$*, the* γ*-th large subspace change is detected within* 1 *mini-batch of* α *frames, i.e.,* $t_\gamma \leq \hat{t}_\gamma \leq t_\gamma + 1$ *and*

$$\text{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \leq \begin{cases} \max(0.1\cdot 0.25, \textit{no-lev}), & \textit{if} \quad j^* = 1 \\ \max(0.1\cdot 0.3^{j^*-1} + 0.5\Delta_{tv}, \textit{no-lev}), & \textit{if} \quad j^* \in [2, J^*] \\ \epsilon, & \textit{if} \quad j^* > J^* \end{cases}$$

where $j^* = \min_\gamma(\hat{t}_\gamma - j)$

Proof of Corollary 5.86. The proof follows from the idea of the result of [30]. The analysis of the subspace update step is exactly as mentioned in the proof of Theorem 5.73. The proof of the subspace update step requires the following changes to [30, Lemma 6.20]: consider the case when the subspace has not changed, but $K = C\log(1/\epsilon)$ subspace updates have been completed. In this

case, q_K (the subspace error between the previous estimate and the current actual subspace) from [30, Lemma 6.20] gets replaced with $\epsilon + \Delta_{tv}$ using the triangle inequality for subspace errors. Next consider the case when the subspace has changed by a quantity of Δ_{large}. In this case, q_1 (the subspace error between the current algorithm estimate and the subspace *after* the large subspace change) gets replaced with $\epsilon + \Delta_{tv} + \Delta_{\text{large}}$. Once we make these changes, the rest of the proof follows exactly in the same fashion, and we get that (i) if the subspace has changed, $\lambda_{\max}(\boldsymbol{\Phi}\hat{\boldsymbol{L}}_j\hat{\boldsymbol{L}}_j^\top\boldsymbol{\Phi}) \geq 5(\epsilon + \Delta_{tv} + \Delta_{\text{large}})^2\lambda^+$, and (ii) if the subspace has not changed, $\lambda_{\max}(\boldsymbol{\Phi}\hat{\boldsymbol{L}}_j\hat{\boldsymbol{L}}_j^\top\boldsymbol{\Phi}) \leq 1.5\epsilon^2\lambda^+$. Thus, under the conditions of Corollary 5.86, as long as $\omega_{evals} = 2\epsilon^2\lambda^+$, w.h.p. the large subspace change is detected. □

Generalization to detect and track large subspace changes in FedOA-RST-miss. Recall that $\boldsymbol{P}_t$ is the matrix of top-r left singular vectors of data, $\boldsymbol{Y}_t = [\boldsymbol{Y}_{1,t}, \cdots, \boldsymbol{Y}_{K,t}]$. Assume that at $t = t_\gamma$ for $\gamma = 1, \cdots, \Gamma$, such that $t_{\gamma+1} - t_\gamma > (J^* + 2)$ with $J^* = C\log(1/\epsilon)$ where ϵ is chosen by the user to denote the desired final accuracy. In addition, assume that

$$\min_{\gamma\in[\Gamma]} \text{SE}(\boldsymbol{P}_{t_\gamma+1}, \boldsymbol{P}_{t_\gamma}) \geq \Delta_{\text{large}}$$

$$\max_{\{t:t\neq t_\gamma,\gamma\in[\Gamma]\}} \text{SE}(\boldsymbol{P}_{t+1}, \boldsymbol{P}_t) \leq \Delta_{tv}$$

Corollary 5.87. *Assume that the data satisfies the subspace change model specified above such that* $\Delta_{\text{large}} > 9\sqrt{f}\max(0.1\cdot 0.3^{J^*-1} + 1.5\Delta_{tv}, \textit{no-lev})$ *. Then, under the conditions of Theorem 5.81 and with minor modifications to Algorithm 12, with probability at least* $1 - dn^{-10}$*, the* γ*-th large subspace change is detected within* 1 *time instant, i.e.,* $t_\gamma \leq \hat{t}_\gamma \leq t_\gamma + 1$ *and*

$$\text{SE}(\hat{\boldsymbol{P}}_t, \boldsymbol{P}_t) \leq$$
$$\begin{cases} \max(0.1\cdot 0.25, \textit{no-lev}), & \textit{if} \quad j^* = 1 \\ \max(0.1\cdot 0.3^{j^*-1} + 0.5\Delta_{tv}, \textit{no-lev}), & \textit{if} \quad j^* \in [2, J^*] \\ \epsilon, & \textit{if} \quad j^* > J^* \end{cases}$$

where $j^* = \min_\gamma(\hat{t}_\gamma - t)$

Algorithm 14 STMiss – Infrequent Abrupt Changes

Require: $\boldsymbol{Y}$, $\mathcal{T}$

1: **Parameters:** α, ϵ,
2: **Init:** $\hat{\boldsymbol{P}}_1 \leftarrow r\text{-}SVD[\boldsymbol{y}_1, \cdots, \boldsymbol{y}_\alpha]$, $j \leftarrow 2$, $k \leftarrow 2$, phase $\leftarrow$ update, $K = C\log(1/\epsilon)$
3: **for** $j \geq 2$ **do**
4: **if** $k = 1$ **then**
5: $\hat{\boldsymbol{P}}_j \leftarrow r\text{-}SVD[\boldsymbol{y}_{(j-2)\alpha+1}, \cdots, \boldsymbol{y}_{(j-1)\alpha}]$
6: **else**
7: **if** phase = update **then**
8: $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{j-1}\hat{\boldsymbol{P}}_{j-1}^\top$
9: **for all** $t \in ((j-1)\alpha, j\alpha]$ **do**
10: $\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$; $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\mathcal{T}_t}(\boldsymbol{\Psi}_{\mathcal{T}_t})^\dagger \tilde{\boldsymbol{y}}_t$.
11: **end for**
12: $\hat{\boldsymbol{P}}_j \leftarrow r\text{-}SVD[\hat{\boldsymbol{\ell}}_{(j-1)\alpha+1}, \cdots, \hat{\boldsymbol{\ell}}_{j\alpha}]$
13: $k \leftarrow k + 1$
14: **if** $k = K$ **then**
15: phase $\leftarrow$ detect
16: **end if**
17: **end if**
18: **end if**
19: **if** phase = detect **then**
20: **if** $\lambda_{\max}(\boldsymbol{\Phi}\hat{\hat{\boldsymbol{L}}}_j\hat{\hat{\boldsymbol{L}}}_j^\top\boldsymbol{\Phi}) \geq 2\alpha\epsilon^2\lambda^+$ **then**
21: phase $\leftarrow$ update, $k \leftarrow 1$
22: **else**
23: Repeat lines 5 - 12
24: **end if**
25: **end if**
26: **end for**

Ensure: $\hat{\boldsymbol{P}}_j$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\hat{\boldsymbol{\ell}}}_t$.

For the proof of Corollary 5.87, the approach is the same as Corollary 5.86. One key difference is how we perform the subspace detection step since this needs to be done while obeying the federated, over-air data sharing constraints. To do this, we leverage Lemma 5.79 and derive the following result:

Lemma 5.88 (Subspace Change Detection)**.** *Consider α data vectors at time $t > t_{\gamma-1}$. Assume that the $(t-1)$-th subspace has been estimated to ϵ-accuracy, i.e.,* $\mathrm{SE}(\hat{\boldsymbol{P}}_{t-1}, \boldsymbol{P}_{t-1}) \leq \epsilon$. *Let the*

number of iterations of Fed-PM be $L_{\text{det}} = C \log nr$. *Let the detection threshold* $\omega_{evals} = 2\epsilon^2 \alpha \lambda^+$. *Then, under the assumptions of Theorem 5.81, the following holds.*

1. *If the subspace changes, i.e.,* $t > t_\gamma$. *At this time, with probability at least* $1 - 10n^{-10}$,

$$\hat{\lambda}_{det} \geq 0.9\lambda_{\max}\left(\boldsymbol{\Phi}\hat{\boldsymbol{L}}_t\hat{\boldsymbol{L}}_t^\top\boldsymbol{\Phi}\right) \geq 4.5(\epsilon + \Delta_{tv} + \Delta_{\text{large}})^2\alpha\lambda^+$$

2. *If the subspace has not changed, then with probability at least* $1 - 10n^{-10}$,

$$\hat{\lambda}_{det} \leq 1.1\lambda_{\max}\left(\boldsymbol{\Phi}\hat{\boldsymbol{L}}_t\hat{\boldsymbol{L}}_t^\top\boldsymbol{\Phi}\right) \leq 1.6\epsilon^2\alpha\lambda^+$$

5.9 Appendix C: Robust Subspace Tracking with Missing Data

In this section, we provide the concrete problem setting, algorithm and result for RST-miss in the centralized setting. Assume that at each time t, we observe an n-dimensional data stream of the form

$$\boldsymbol{y}_t = \mathcal{P}_{\Omega_t}(\tilde{\boldsymbol{\ell}}_t + \boldsymbol{g}_t), \quad t = 1, 2, \cdots, d \tag{5.14}$$

where $\boldsymbol{g}_t$'s are the sparse outliers and $\tilde{\boldsymbol{\ell}}_t$, $\mathcal{P}_{\Omega_t}(\cdot)$ etc are defined exactly as done before. We let $\boldsymbol{s}_t := \mathcal{P}_{\Omega_t}(\boldsymbol{g}_t)$ and let $\mathcal{T}_{\text{sparse},t}$ denote the support of $\boldsymbol{s}_t$. Notice that it is impossible to recover $\boldsymbol{g}_t$ on the set $\mathcal{T}_t$ and thus we only work with $\boldsymbol{s}_t$ in the sequel. Furthermore, by definition, $\boldsymbol{s}_t$ is supported outside $\mathcal{T}_t$ and thus $\mathcal{T}_t$ and $\mathcal{T}_{\text{sparse},t}$ are disjoint. With $\boldsymbol{s}_t$ defined as above, the measurements can also be expressed as

$$\begin{aligned}
\boldsymbol{y}_t &= \mathcal{P}_{\Omega_t}(\tilde{\boldsymbol{\ell}}_t) + \boldsymbol{s}_t \\
&= \tilde{\boldsymbol{\ell}}_t - \boldsymbol{I}_{\mathcal{T}_t}\boldsymbol{I}_{\mathcal{T}_t}^\top\tilde{\boldsymbol{\ell}}_t + \boldsymbol{s}_t \\
&:= \tilde{\boldsymbol{\ell}}_t + \boldsymbol{z}_t + \boldsymbol{s}_t = \boldsymbol{\ell}_t + \boldsymbol{z}_t + \boldsymbol{s}_t + \boldsymbol{v}_t.
\end{aligned}$$

One main difference required in the algorithm is how we estimate the sparse vector, $\tilde{\boldsymbol{s}}_t = \boldsymbol{z}_t + \boldsymbol{s}_t$. Recovering $\tilde{\boldsymbol{s}}_t$ is a problem of sparse recovery with partial support knowledge, $\mathcal{T}_t$. In this paper, we use noisy modified CS [24] which was introduced to solve exactly this problem. Another main

Algorithm 15 Robust Subspace Tracking with missing entries (RST-miss)

Require: $\boldsymbol{Y}$, $\mathcal{T}$

1: **Parameters:** α, ω_{evals}, ξ
2: **Init:** $\hat{\boldsymbol{P}}_1 \leftarrow \text{AltProj}[\boldsymbol{y}_1, \cdots, \boldsymbol{y}_\alpha]$, $j \leftarrow 2$
3: Lines 3-13 of Algorithm 10 with line 6 replaced by
$\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$
$\hat{\boldsymbol{s}}_{t,cs} \leftarrow \arg\min_{\boldsymbol{s}} \|(\boldsymbol{s})_{(\mathcal{T}_t)^c}\|_1$ s.t. $\|\tilde{\boldsymbol{y}}_t - \boldsymbol{\Psi}\boldsymbol{s}\| \leq \xi$.
$\hat{\mathcal{T}}_t \leftarrow \mathcal{T}_t \cup \{m : |(\hat{\boldsymbol{s}}_{t,cs})_m| > \omega_{supp}\}$
$\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^\dagger \tilde{\boldsymbol{y}}_t$.
4: Line 11 replaced by
$\tilde{\boldsymbol{y}}_t \leftarrow \tilde{\boldsymbol{\Psi}}\boldsymbol{y}_t$
$\hat{\boldsymbol{s}}_{t,cs} \leftarrow \arg\min_{\boldsymbol{s}} \|(\boldsymbol{s})_{(\mathcal{T}_t)^c}\|_1$ s.t. $\|\tilde{\boldsymbol{y}}_t - \tilde{\boldsymbol{\Psi}}\boldsymbol{s}\| \leq \xi$.
$\hat{\mathcal{T}}_t \leftarrow \mathcal{T}_t \cup \{m : |(\hat{\boldsymbol{s}}_{t,cs})_m| > \omega_{supp}\}$
$\hat{\hat{\boldsymbol{\ell}}}_t \leftarrow \boldsymbol{y}_t - \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\tilde{\boldsymbol{\Psi}}_{\hat{\mathcal{T}}_t})^\dagger \tilde{\boldsymbol{y}}_t$.

Ensure: $\hat{\boldsymbol{P}}_j$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\hat{\boldsymbol{\ell}}}_t$, $\hat{\mathcal{T}}_t$.

difference is in the initialization step. Observe that due to the presence of sparse outliers, a simple PCA step does not ensure a "good enough" initialization in this case.

Assumptions. We need all the assumptions from the previous section. In addition, it is well known from the RPCA literature that the fraction of outliers in each row and column of the matrix $\boldsymbol{S}_j$ needs to be bounded.

Definition 5.89 (Sparse outlier fractions)**.** *Consider the sparse outlier matrix* $\boldsymbol{S}_j := [\boldsymbol{s}_{(j-1)\alpha+1}, \ldots, \boldsymbol{s}_{j\alpha}]$ *. We use max-outlier-frac-col (max-outlier-frac-row) to denote the maximum of the fraction of non-zero elements in any column (row) of this matrix. Also define* $x_{\min} = \min_{t\in((j-1\alpha, j\alpha]} \min_{i\in\mathcal{T}_{\text{sparse},t}} |(\boldsymbol{s}_t)_i|$.

Algorithm and Main Result.

We have the following result for robust subspace tracking with missing entries

Theorem 5.90 (Robust Subspace Tracking with missing entries)**.** *Consider Algorithm 12. Assume that no-lev* ≤ 0.2*. Set* $\omega_{supp} = x_{\min}/2$ *and* $\xi = x_{\min}/15$*. Assume that the following hold:*

1. *At* $t = 1$ *we are given a* $\hat{\boldsymbol{P}}_1$ *s.t.* $\text{SE}(\hat{\boldsymbol{P}}_1, \boldsymbol{P}_1) \leq \epsilon_{\text{init}}$.

2. **Incoherence:** $\boldsymbol{P}_j$*'s satisfy* μ*-incoherence, and* $\boldsymbol{a}_t$*'s satisfy statistical right* μ*-incoherence;*

3. **Missing Entries:** *max-miss-frac-col* $\in O(1/\mu r)$, *max-miss-frac-row* $\in O(1)$;

4. **Sparse Outliers:** *max-outlier-frac-col* $\in O(1/\mu r)$, *max-outlier-frac-row* $\in O(1)$;

5. **Subspace Model:** *let* $\Delta_{tv} := \max_j \mathrm{SE}(\boldsymbol{P}_{j-1}, \boldsymbol{P}_j)$ *s.t.*

$$0.3\epsilon_{\text{init}} + 0.5\Delta_{tv} \leq 0.28 \quad \textit{and}$$
$$C\sqrt{r\lambda^+}(0.3^{j-1}\epsilon_{\text{init}} + 0.5\Delta_{tv}) + \sqrt{r_v\lambda_v^+} \leq x_{\min}$$

then, with probability at least $1 - 10dn^{-10}$, *for all* $j > 1$, *we have*

$$\begin{aligned}
&\mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j) \\
&\leq \max(0.3^{j-1}\epsilon_{\text{init}} + \Delta_{tv}(0.3 + 0.3 + 0.3^2 ... + 0.3^{j-1}), \textit{no-lev}) \\
&< \max(0.3^{j-1}\epsilon_{\text{init}} + 0.5\Delta_{tv}, \textit{no-lev})
\end{aligned}$$

Also, at all j *and* $t \in ((j-1)\alpha, j\alpha]$, $\|\hat{\tilde{\boldsymbol{\ell}}}_t - \boldsymbol{\ell}_t\| \leq 1.2 \cdot \mathrm{SE}(\hat{\boldsymbol{P}}_j, \boldsymbol{P}_j)\|\tilde{\boldsymbol{\ell}}_t\| + \|\boldsymbol{v}_t\|$.

5.10 Appendix D: Convergence Analysis for FedPM

5.10.1 Eigenvalue convergence

First we present the proof of the eigenvalue convergence result (Lemma 5.79). To our best knowledge, this has not been studied in the federated ML literature.

Proof of Lemma 5.79. We now wish to compute the error bounds of in convergence of eigenvalues. To this end, at the end of L iterations, we compute $\hat{\boldsymbol{\Sigma}} = \hat{\boldsymbol{U}}_L^\top \boldsymbol{A}\hat{\boldsymbol{U}}_L + \hat{\boldsymbol{U}}_L^\top \boldsymbol{W}_L$. The intuition is that if the eigenvectors are estimated well, then this matrix will be approximately diagonal (off diagonal entries $\approx \epsilon$), and the diagonal entries will be close to the true eigenvalues. Furthermore, in the application of this result for the Subspace Change detection problem, we will only consider the

largest eigenvalue of $\hat{\boldsymbol{\Sigma}}$ and thus we have

$$\begin{aligned}\lambda_{\max}(\hat{\boldsymbol{\Sigma}}) &= \lambda_{\max}(\hat{\boldsymbol{U}}_L^\top \boldsymbol{A}\hat{\boldsymbol{U}}_L + \hat{\boldsymbol{U}}_L^\top \boldsymbol{W}_L) \\ &= \lambda_{\max}(\boldsymbol{\Sigma} + (\hat{\boldsymbol{U}}_L^\top \boldsymbol{A}\hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}) + \hat{\boldsymbol{U}}_L^\top \boldsymbol{W}_L) \\ &\geq \lambda_{\max}(\boldsymbol{\Sigma}) - \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{A}\hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| - \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{W}_L\| \\ &\geq \sigma_1 - \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{A}\hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| - \|\boldsymbol{W}_L\|\end{aligned}$$

The second term can be upper bounded as follows

$$\begin{aligned}&\|\hat{\boldsymbol{U}}_L^\top \boldsymbol{A}\hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| \\ &= \|(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}) + \hat{\boldsymbol{U}}_L^\top \boldsymbol{U}_\perp \boldsymbol{\Sigma}_\perp \boldsymbol{U}_\perp^\top \hat{\boldsymbol{U}}_L\| \\ &\leq \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| + \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}_\perp \boldsymbol{\Sigma}_\perp \boldsymbol{U}_\perp^\top \hat{\boldsymbol{U}}_L\| \\ &\leq \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| + \|\boldsymbol{\Sigma}_\perp\|\|\boldsymbol{U}_\perp^\top \hat{\boldsymbol{U}}_L\|^2 \\ &= \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| + \|\boldsymbol{\Sigma}_\perp\|\|\boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top \hat{\boldsymbol{U}}_L\|^2 \\ &\leq \|\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| + \sigma_{r+1}\mathrm{SE}^2(\hat{\boldsymbol{U}}_L, \boldsymbol{U})\end{aligned}$$

The first term above can be bounded as

$$\begin{aligned}&\|\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{\Sigma}\| \\ &= \|(\boldsymbol{I} - \boldsymbol{I} + \hat{\boldsymbol{U}}_L^\top \boldsymbol{U})\boldsymbol{\Sigma}(\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L + \boldsymbol{I} - \boldsymbol{I}) - \boldsymbol{\Sigma}\| \\ &\leq \|(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U} - \boldsymbol{I})\boldsymbol{\Sigma}\| + \|\boldsymbol{\Sigma}(\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{I})\| \\ &+ \|(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U} - \boldsymbol{I})\boldsymbol{\Sigma}(\boldsymbol{U}^\top \hat{\boldsymbol{U}}_L - \boldsymbol{I})\| \\ &\leq \sigma_1(2\|\boldsymbol{I} - \hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\| + \|\boldsymbol{I} - \hat{\boldsymbol{U}}_L^\top \boldsymbol{U}\|^2) \\ &\leq \sigma_1(2(1 - \sigma_r(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U})) + (1 - \sigma_r(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}))^2)\end{aligned}$$

and since $\mathrm{SE}^2(\hat{\boldsymbol{U}}_L, \boldsymbol{U}) = 1 - \sigma_r^2(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}) \leq \epsilon^2$ and thus we get that $\sigma_r(\hat{\boldsymbol{U}}_L^\top \boldsymbol{U}) \geq \sqrt{1-\epsilon^2} \geq 1 - \epsilon^2$. Finally, the assumption on the channel noise implies that with high probability, $\|\boldsymbol{W}_L\| \leq C\sqrt{n}\sigma_c \leq 1.5\sigma_r\epsilon$. Thus,

$$\lambda_{\max}(\hat{\boldsymbol{\Sigma}}) \geq \sigma_1(1 - 4\epsilon^2) - \sigma_{r+1}\epsilon^2 - \sigma_r\epsilon$$

Algorithm 16 FedPM: Federated Noise-Tolerant Power Method

Require: $\boldsymbol{Z}$, r, L, η, K nodes, for each $i \in \mathcal{I}_k$, data $\boldsymbol{y}_i$

1: At central server, $\tilde{\boldsymbol{U}}_0 \overset{i.i.d.}{\sim} \mathcal{N}(0, I)_{n\times r}$; $\hat{\boldsymbol{U}}_0 \leftarrow \tilde{\boldsymbol{U}}_0$, transmit to all K workers.
2: **for** $l = 1, \ldots, L\eta$ **do**
3: At k-th node, do $\tilde{\boldsymbol{U}}_{k,l} = \boldsymbol{Z}_k \boldsymbol{Z}_k^\top \hat{\boldsymbol{U}}_{l-1}$
4: All k nodes transmit $\tilde{\boldsymbol{U}}_{k,l}$ synchronously to the central server.
5: Central server receives $\tilde{\boldsymbol{U}}_\eta := \sum_k \tilde{\boldsymbol{U}}_{k,l} + \boldsymbol{W}_{k,l}$, with $\sum_k \boldsymbol{W}_{k,l} = \boldsymbol{W}_l$.
6: $\hat{\boldsymbol{U}}_l \leftarrow \hat{\boldsymbol{U}}_{l-1}$
7: **if** $(l \mod \eta) = 0$ **then** $\hat{\boldsymbol{U}}_l \boldsymbol{R}_l \overset{QR}{\leftarrow} \tilde{\boldsymbol{U}}_l$ **end if**
8: Central server broadcasts $\hat{\boldsymbol{U}}_l$ to all nodes
9: **end for**
10: All k nodes compute $\boldsymbol{Z}_k \boldsymbol{Z}_k^\top \hat{\boldsymbol{U}}_L$, transmit synchronously to central server
11: Central server receives $\boldsymbol{B} = \sum_k \boldsymbol{Z}_k \boldsymbol{Z}_k^\top \hat{\boldsymbol{U}}_L + \boldsymbol{W}_L$, computes the top eigenvalue, $\hat{\sigma}_1 = \lambda_{\max}(\hat{\boldsymbol{U}}_L^\top \boldsymbol{B})$.

Ensure: $\hat{\boldsymbol{U}}_L$, $\hat{\sigma}_1$.

We also get

$$\begin{aligned}\lambda_{\max}(\hat{\boldsymbol{\Sigma}}) &\leq \lambda_{\max}(\hat{\boldsymbol{U}}_L^\top \boldsymbol{B}\boldsymbol{B}^\top \hat{\boldsymbol{U}}_L) + \|\boldsymbol{W}_L\| \\ &\leq \|\hat{\boldsymbol{U}}_L\|^2 \|\boldsymbol{B}\boldsymbol{B}^\top\| + \|\boldsymbol{W}_L\| = \lambda_{\max}(\boldsymbol{B}\boldsymbol{B}^\top) + 1.5\sigma_r\epsilon\end{aligned}$$

This completes the proof. □

5.10.2 The Noise Tolerant FedOA-PM, Algorithm, and Guarantee

Next, we present a "robust" version of the FedOA-PM algorithm. As mentioned earlier, by normalizing the subspace estimates once every $\eta \geq 1$ iterations allows for a larger noise tolerance than the vanilla FedOA-PM algorithm. This is summarized in Algorithm 16 and the main result is provided below.

Before we state the main result, we need to define the following quantities. For this section we use $\boldsymbol{A} = \boldsymbol{Z}\boldsymbol{Z}^\top$ and let $\boldsymbol{A} \overset{EVD}{=} \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top + \boldsymbol{U}_\perp \boldsymbol{\Sigma}_\perp \boldsymbol{U}_\perp^\top$ denote its eigenvalue decomposition. Recall that $\boldsymbol{U} \in \boldsymbol{R}^{n\times r}$ denote the principal subspace that we are interesting in estimating. We also use σ_i to denote the i-th eigenvalue of $\boldsymbol{A}$ with $\sigma_1 \geq \cdots \geq \sigma_r > \sigma_{r+1} \geq \cdots \geq \sigma_n \geq 0$. We also let the ratio of $(r+1)$-th to r-th eigenvalue, $R := \sigma_{r+1}/\sigma_r$, the noise to signal ratio, NSR $:= \sigma_c/\sigma_r$, and $\tilde{R} := \max(R, 1/\sigma_r)$. We use $\mathrm{SE}_l := \mathrm{SE}(\hat{\boldsymbol{U}}_l, \boldsymbol{U})$.

We have the following main result:

Theorem 5.91. *Consider Algorithm 16 with initial subspace estimation error* SE_0.

1. *Let* $\eta = 1$. *Assume that* $R < 0.99$. *If, at each iteration, the channel noise* $\boldsymbol{W}_l$ *satisfies* $\mathrm{NSR} < c \min\left(\frac{\epsilon}{\sqrt{n}}, 0.2\sqrt{\frac{1-\mathrm{SE}_{l-1}^2}{r}}\right)$ *then, after* $L = \Omega\left(\frac{1}{\log(1/R)}\left(\log\frac{1}{\epsilon} + \log\frac{1}{\sqrt{1-\mathrm{SE}_0^2}}\right)\right)$ *iterations, with probability at least* $1 - L\exp(-cr)$, $\mathrm{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_L) \leq \epsilon$.

2. *Consider Algorithm 11 with* $\eta > 1$. *If,* $\sigma_r > 1$, *and if* $\mathrm{NSR} < c \min\left(\frac{\epsilon}{\sqrt{n}} \cdot \frac{1}{\sqrt{\eta}R^{\eta-1}}, 0.2\sqrt{\frac{\sigma_r^2-1}{\sigma_r^2}} \cdot \sqrt{\frac{1-\mathrm{SE}_{(l-1)\eta}^2}{r}}\right)$, *then the above conclusion holds.*

3. *If* $\tilde{\boldsymbol{U}}_0 \overset{i.i.d}{\sim} \mathcal{N}(0, \boldsymbol{I})_{n\times r}$, *then* $\mathrm{SE}_0 = \mathcal{O}(\sqrt{1 - 1/\gamma nr})$ *with probability* $1 - 1/\gamma$.

To understand the above theorem, first consider $\eta = 1$. In this case, we require $\mathrm{NSR}\sqrt{n} < \epsilon$ to achieve ϵ-accurate recovery of the subspace. In this setting, with a random initialization, our result essentially recovers the main result of [15, 4]. But we can choose to pick $\eta > 1$. To understand its advantage, suppose that $\lambda_r > 1.5$ (this is easy to satisfy by assuming that all the data transmitted is scaled by a large enough factor). Then, clearly, $\lambda_r^2/(\lambda_r^2 - 1) < 3$ and so the first term in the upper bound of NSR dominates. Thus, as η is increased, we only require $\mathrm{NSR}\sqrt{n} \cdot \sqrt{\eta}R^{\eta-1} \leq \epsilon$ which is a significantly weaker requirement. Thus, a larger η means we can allow the noise variance to be larger. However, we cannot pick η too large because it will lead to numerical problems (bit overflow problems) and may also result in violation of the transmit power constraint. As an example, if we set $\eta = C\log n$, for a constant C that is large enough (depends on $\tilde{R}$), then the we only require $(\mathrm{NSR}\sqrt{n}/\log n) \leq \epsilon$ which provides a $\log n$ factor of noise robustness. Observe that the number of iterations needed, L, depends on the initialization. If $\mathrm{SE}_0 < c_0$ with c_0 being a constant, then we only need $L = \Omega\left(\frac{1}{\log(1/R)}\log(1/\epsilon)\right)$ iterations (which we leverage in the ST-miss result). Finally, if we use random initialization we need $L = \Omega\left(\frac{1}{\log(1/R)}\log(nr/\epsilon)\right)$, i.e., $O(\log nr)$ more iterations. We provide a comparison with [15, 4] in Table 5.1.

Table 5.1: Comparing bounds on channel noise variance σ_c^2 and on number of iterations L. Let $\text{gap}_1 := \lambda_r - \lambda_{r+1}$, $\text{gap}_q := \lambda_r - \lambda_{q+1}$ for some $r \le q \le r'$. Also, we assume $\epsilon \le c/r$.

	Noisy Power Method [15, 4]	This Work
$\eta = 1$ $r' = r$	$\sigma_c = \mathcal{O}\left(\frac{\text{gap}_1 \epsilon}{\sqrt{n}}\right)$	$\sigma_c = \mathcal{O}\left(\frac{\lambda_r \epsilon}{\sqrt{n}}\right)$, $R < 0.99$
Random init	$L = \Omega\left(\frac{\lambda_r}{\text{gap}_q} \log\left(\frac{n}{\epsilon}\right)\right)$	$L = \Omega\left(\frac{1}{\log(1/R)} \log\left(\frac{n}{\epsilon}\right)\right)$
Good init ($\text{SE}_0 \le c_0$)	-	$L = \Omega\left(\frac{1}{\log(1/R)} \log\left(\frac{1}{\epsilon}\right)\right)$
$\eta = 1$ $r' > r$	$\sigma_c = \mathcal{O}\left(\frac{\text{gap}_q \epsilon}{\sqrt{n}}\right)$	–
$\eta > 1$ $r' = r$	–	$\sigma_c = \mathcal{O}\left(\frac{\lambda_r \epsilon}{R^\eta \eta}\right)$, $R < 0.99$, $\lambda_r > 1$
$\eta = \mathcal{O}(\log n)$ $r' = r$	–	$\sigma_c = \mathcal{O}\left(\frac{\lambda_r \epsilon n}{\log n}\right)$, $R < 0.99$, $\lambda_r > 1$

5.10.3 Proof of Theorem 5.91

Before we state the proof, we define two auxiliary quantities

$$\Gamma_{num}^2(\eta) := \frac{1 + \sigma_{r+1}^2 + \sigma_{r+1}^4 + \ldots \sigma_{r+1}^{2\eta-2}}{\sigma_r^{2\eta-2}},$$
$$\Gamma_{denom}^2(\eta) := \frac{1 + \sigma_r^2 + \sigma_r^4 + \cdots + \sigma_r^{2\eta-2}}{\sigma_r^{2\eta-2}}$$

Intuitively, $\Gamma_{num}(\eta)$ captures the effect of the ratio of the "effective channel noise orthogonal to the signal space", to the signal energy, while $\Gamma_{denom}(\eta)$ captures the "effective channel noise along the signal space" and the signal energy. The following lemma bounds the reduction in error from iteration $(l-1)\eta$ to $l\eta$.

Lemma 5.92 (Descent Lemma, general η)**.** *Consider Algorithm 16. Assume that $R < 0.99$. With probability at least $1 - \exp(-cr)$, the following holds:*

$$\text{SE}_{l\eta} \le \frac{R^\eta \ \text{SE}_{(l-1)\eta} + \sqrt{n} \ NSR \ \Gamma_{num}(\eta)}{0.9\sqrt{1 - \text{SE}_{(l-1)\eta}^2} - \sqrt{r} \ NSR \ \Gamma_{denom}(\eta)}$$

By recursively applying the above lemma at each iteration, we have the following. It assumes that the initial subspace estimate has error $\mathrm{SE}_0 := \mathrm{SE}(\tilde{\boldsymbol{U}}_0, \boldsymbol{U})$. The proof is provided in Appendix 5.10.

Proof of Lemma 5.92. Consider the setting where we normalize our subpsace estimates every η iterations. In other words, we start with a basis matrix estimate at $l = l_0$, and then analyze the subspace error after η iterations. In this case, the subspace update equations can be written as

$$\begin{aligned}
\tilde{\boldsymbol{U}}_{l_0+1} &= \boldsymbol{A}\hat{\boldsymbol{U}}_{l_0} + \boldsymbol{W}_{l_0+1} \\
\tilde{\boldsymbol{U}}_{l_0+2} = \boldsymbol{A}\tilde{\boldsymbol{U}}_{l_0+1} + \boldsymbol{W}_{l_0+2} &= \boldsymbol{A}^2\hat{\boldsymbol{U}}_{l_0} + \boldsymbol{A}\boldsymbol{W}_{l_0+1} + \boldsymbol{W}_{l_0+2} \\
&\vdots \\
\tilde{\boldsymbol{U}}_{l_0+\eta} = \tilde{\boldsymbol{U}}_l &= \boldsymbol{A}^\eta\hat{\boldsymbol{U}}_{l_0} + \sum_{i=1}^{\eta} \boldsymbol{A}^{\eta-i}\boldsymbol{W}_{l_0+i}
\end{aligned}$$

Recall that $\hat{\boldsymbol{U}}_{l_0} \overset{QR}{=} \tilde{\boldsymbol{U}}_{l_0}\boldsymbol{R}_{l_0}$. Thus, we have

$$\begin{aligned}
\tilde{\boldsymbol{U}}_l &= \boldsymbol{A}^\eta\tilde{\boldsymbol{U}}_{l_0}\boldsymbol{R}_{l_0}^{-1} + \sum_{i=1}^{\eta} \boldsymbol{A}^{\eta-i}\boldsymbol{W}_{l_0+i} \\
&= \boldsymbol{A}^\eta(\boldsymbol{U}\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_{l_0} + \boldsymbol{U}_\perp\boldsymbol{U}_\perp^\top\tilde{\boldsymbol{U}}_{l_0})\boldsymbol{R}_{l_0}^{-1} \\
&+ \sum_{i=1}^{\eta} \boldsymbol{A}^{\eta-i}(\boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{W}_{l_0+i} + \boldsymbol{U}_\perp\boldsymbol{U}_\perp^\top\boldsymbol{W}_{l_0+i}) \\
&= \boldsymbol{U}\boldsymbol{\Sigma}^\eta(\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_{l_0})\boldsymbol{R}_{l_0}^{-1} + \boldsymbol{U}_\perp\boldsymbol{\Sigma}_\perp^\eta(\boldsymbol{U}_\perp^\top\tilde{\boldsymbol{U}}_{l_0})\boldsymbol{R}_{l_0}^{-1} \\
&+ \sum_{i=1}^{\eta} \left[\boldsymbol{U}\boldsymbol{\Sigma}^{\eta-i}(\boldsymbol{U}^\top\boldsymbol{W}_{l_0+i}) + \boldsymbol{U}_\perp\boldsymbol{\Sigma}_\perp^{\eta-i}(\boldsymbol{U}_\perp^\top\boldsymbol{W}_{l_0+i})\right]
\end{aligned}$$

and thus, $\text{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_l) = \text{SE}(\boldsymbol{U}, \tilde{\boldsymbol{U}}_l) = \|\boldsymbol{U}_\perp^\top \tilde{\boldsymbol{U}}_l \boldsymbol{R}_l^{-1}\|$ simplifies to

$$
\begin{aligned}
&\text{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_l) \\
&= \left\| \left[\boldsymbol{\Sigma}_\perp^\eta (\boldsymbol{U}_\perp^\top \tilde{\boldsymbol{U}}_{l_0}) \boldsymbol{R}_{l_0}^{-1} + \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}) \right] \boldsymbol{R}_t^{-1} \right\| \\
&\leq \left(\|\boldsymbol{\Sigma}_\perp^\eta\| \|\boldsymbol{U}_\perp^\top \tilde{\boldsymbol{U}}_{l_0} \boldsymbol{R}_{l_0}^{-1}\| + \left\| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}) \right\| \right) \|\boldsymbol{R}_t^{-1}\| \\
&= \left(\|\boldsymbol{\Sigma}_\perp^\eta\| \text{SE}(\boldsymbol{U}, \tilde{\boldsymbol{U}}_{l_0}) + \left\| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}) \right\| \right) \|\boldsymbol{R}_t^{-1}\| \\
&\leq \frac{\|\boldsymbol{\Sigma}_\perp^\eta\| \text{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_{l_0}) + \left\| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}) \right\|}{\sigma_r(\boldsymbol{R}_t)}
\end{aligned}
$$

We also have that

$$
\begin{aligned}
\sigma_r^2(\boldsymbol{R}_t) &= \sigma_r^2(\tilde{\boldsymbol{U}}_t) \\
&= \lambda_{\min}((\boldsymbol{U}\boldsymbol{U}^\top \tilde{\boldsymbol{U}}_t + \boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top \hat{\boldsymbol{U}}_t)^\top (\boldsymbol{U}\boldsymbol{U}^\top \tilde{\boldsymbol{U}}_t + \boldsymbol{U}_\perp \boldsymbol{U}_\perp^\top \hat{\boldsymbol{U}}_t)) \\
&\geq \lambda_{\min}(\tilde{\boldsymbol{U}}_t^\top \boldsymbol{U}\boldsymbol{U}^\top \hat{\boldsymbol{U}}_t) = \sigma_r^2(\boldsymbol{U}^\top \hat{\boldsymbol{U}}_t) \\
\implies \sigma_r(\boldsymbol{U}^\top \tilde{\boldsymbol{U}}_t) &= \sigma_r \left(\boldsymbol{\Sigma}^\eta \left(\boldsymbol{U}^\top \hat{\boldsymbol{U}}_{l_0} + \sum_{i=1}^{\eta} \boldsymbol{\Sigma}^{-i} \boldsymbol{U}^\top \boldsymbol{W}_{l_0+i} \right) \right) \\
&\geq \sigma_r^\eta \left[\sigma_r(\boldsymbol{U}^\top \hat{\boldsymbol{U}}_{l_0}) - \left\| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}^{-i} \boldsymbol{U}^\top \boldsymbol{W}_{l_0+i} \right\| \right]
\end{aligned}
$$

We define $\text{SE}(\boldsymbol{U}, \tilde{\boldsymbol{U}}_{l_0}) = \text{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_{l_0}) = \text{SE}_{l_0}$ and $R = \sigma_{r+1}/\sigma_r$, $\tilde{R} = \max(1, \sigma_{r+1})/\sigma_r$ and thus we have

$$
\begin{aligned}
&\text{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_l) \\
&\leq \frac{\|\boldsymbol{\Sigma}_\perp^\eta\| \text{SE}(\boldsymbol{U}, \hat{\boldsymbol{U}}_{l_0}) + \left\| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}) \right\|}{\sigma_r^\eta \left[\sqrt{1 - \text{SE}^2(\boldsymbol{U}, \tilde{\boldsymbol{U}}_{l_0})} - \| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}^{-i} \boldsymbol{U}^\top \boldsymbol{W}_{l_0+i} \| \right]} \\
&\leq \frac{R^\eta \text{SE}_{l_0} + \sigma_r^{-\eta} \| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} \boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i} \|}{\sqrt{1 - \text{SE}_{l_0}^2} - \| \sum_{i=1}^{\eta} \boldsymbol{\Sigma}^{-i} \boldsymbol{U}^\top \boldsymbol{W}_{l_0+i} \|}
\end{aligned}
$$

notice that the entries of $\boldsymbol{U}^\top \boldsymbol{W}_{l_0+i}$ and $\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}$ are i.i.d. Gaussian r.v's with variance σ_c^2. Next we define the matrix $M = \sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i})$ and we apply Theorem 5.93 to M. We can

apply this theorem because we know that each entry of M is a weighted sum of η indepdendent Gaussian r.v.'s. In other words

$$\begin{aligned} M_{jk} &= \sum_{i=1}^{\eta} (\sigma_\perp)_j^{\eta-i} (\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i})_{jk} \\ \implies M_{jk} &\sim \mathcal{N}\left(0, \sigma_c^2 \sum_{i=1}^{\eta} (\lambda_\perp)_j^{2(\eta-i)}\right) \\ \implies \max_{jk} \|(M)_{jk}\|_{\psi_2} &= \sigma_c \sqrt{\sum_{i=1}^{\eta} \sigma_{r+1}^{2(\eta-i)}} \end{aligned}$$

Recall that there is a factor of $\sigma_r^{-\eta}$ multiplying M so effectively, the sub-Gaussian norm is $K = \sigma_r^{-\eta}\sigma_c\sqrt{\sum_{i=1}^{\eta}\sigma_{r+1}^{2(\eta-i)}} = \text{NSR}\cdot\Gamma_{num}(\eta)$. Now, using Theorem 5.93, we get that with probability at least $1-e^{-\epsilon^2}$

$$\|\sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i}\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}\| \le C\text{NSR}\cdot\Gamma_{num}(\eta)\cdot(\sqrt{n-r}+\sqrt{r}+\epsilon)$$

and now picking $\epsilon = 0.01\sqrt{n}$ followed by simple algebra yields

$$\begin{aligned} \Pr\left(\|\sum_{i=1}^{\eta} \boldsymbol{\Sigma}_\perp^{\eta-i}\boldsymbol{U}_\perp^\top \boldsymbol{W}_{l_0+i}\| \le \sqrt{n}\text{NSR}\cdot\Gamma_{num}(\eta)\right) \\ \ge 1-\exp(-cn) \end{aligned}$$

Next consider the denominator term. Again, we notice that the matrix $M = \sum_{i=1}^{\eta}\boldsymbol{\Sigma}^{-i}\boldsymbol{U}^\top\boldsymbol{W}_{l_0+i}$ has entries that are gaussian r.v.'s and are independent. Moreover, the sub Gaussian norm bound is

$$\begin{aligned} M_{jk} &= \sum_{i=1}^{\eta} \sigma_j^{-i}(\boldsymbol{U}^\top \boldsymbol{W}_{l_0+i})_{jk} \\ \implies M_{jk} &\sim \mathcal{N}\left(0, \sigma_c^2 \sum_{i=1}^{\eta} \sigma_j^{-2i}\right) \\ \implies \max_{jk} \|(M)_{jk}\|_{\psi_2} &= \sigma_c \sqrt{\sum_{i=1}^{\eta} \sigma_r^{-2i}} := \text{NSR}\cdot\Gamma_{denom}(\eta) \end{aligned}$$

Now we apply Theorem 5.93 to get that with probability $1-\exp(-\epsilon^2)$

$$\left\|\sum_{i=1}^{\eta}\boldsymbol{\Sigma}^{-i}\boldsymbol{U}^\top\boldsymbol{W}_{l_0+i}\right\| \le \text{NSR}\cdot\Gamma_{denom}(\eta)\cdot(2\sqrt{r}+\epsilon)$$

picking $\epsilon = 0.01\sqrt{r}$ yields that

$$\Pr\left(\left\|\sum_{i=1}^{\eta} \boldsymbol{\Sigma}^{-i}\boldsymbol{U}^\top \boldsymbol{W}_{l_0+i}\right\| \leq \sqrt{r}\mathrm{NSR}\cdot\Gamma_{denom}(\eta)\right) \geq 1-\exp(-cr)$$

This completes the proof of Lemma 5.92. □

Proof of Theorem 5.91. The idea for proving Theorem 5.91 is a straightforward extension from Lemma 5.92. Consider $\eta = 1$, and assume that the initial subspace estimtate, $\tilde{\boldsymbol{U}}_0$ satisfies $\mathrm{SE}(\tilde{\boldsymbol{U}}_0, \boldsymbol{U}) = \mathrm{SE}_0 < 1$ we know that with probability $1-\exp(-cr)-\exp(-cn)$,

$$\begin{aligned}\mathrm{SE}(\tilde{\boldsymbol{U}}_\eta, \boldsymbol{U}) &\leq \frac{R^\eta \mathrm{SE}_0 + \sqrt{n}\mathrm{NSR}\Gamma_{num}(\eta)}{0.9\sqrt{1-\mathrm{SE}_0^2} - \sqrt{r}\mathrm{NSR}\Gamma_{denom}(\eta)} \\ &= \frac{R\mathrm{SE}_0 + \sqrt{n}\mathrm{NSR}}{0.9\sqrt{1-\mathrm{SE}_0^2} - \sqrt{r}\mathrm{NSR}}\end{aligned}$$

thus, as long as $\mathrm{NSR} \leq 0.2\sqrt{\frac{1-\mathrm{SE}_0^2}{r}}$ the denominator is positive. Next, to achieve an ϵ-accurate estimate, we note that the second term in the numerator is the larger term (since $R<1$ and this goes to 0 with every iteration) and thus as long as $\mathrm{NSR} \leq \frac{\epsilon}{\sqrt{n}}$ we can ensure that the numerator is small enough. Combining the two bounds, followed by a union bound over L iterations gives the final conclusion.

Finally, consider the case of $\eta > 1$ and the l-th iteration. Assume that $\sigma_r > 1$. This is used to simplify the $\Gamma_{denom}(\eta)$ expression as follows: $\Gamma^2_{denom}(\eta) = (1+\sigma_r^2+\cdots+\sigma_r^{2\eta-2})/\sigma_r^{2\eta-2} = \sum_{i=0}^{\eta-1} 1/\sigma_r^{2i} \leq \sum_{i=0}^{\infty} 1/\sigma_r^{2i} = \frac{\sigma_r^2}{\sigma_r^2-1}$. Using the same reasoning as in the $\eta=1$ case, as long as

$$\mathrm{NSR} \leq 0.2\sqrt{\frac{\sigma_r^2-1}{\sigma_r^2}}\cdot\sqrt{\frac{1-\mathrm{SE}^2_{(l-1)\eta}}{r}}$$

the denominator is positive. We also have that $\Gamma^2_{num}(\eta) = \sum_{i=1}^{\eta}\sigma_{r+1}^{2(\eta-i)}/\sigma_r^{2\eta} \leq \eta R^{2\eta-2}$. Thus, as long as $\mathrm{NSR} \leq \frac{\epsilon}{\sqrt{n}}\cdot\frac{1}{\sqrt{\eta}R^{\eta-1}}$ the first term of the numerator is small enough and this gives us the final result. □

Random Initialization Lemma. Finally, we provide the proof for random initialization. This is a well known result as shown in [36, 15] but we prove it here for completeness.

Proof of Item 3 of Theorem 5.91. The proof follows by application of Theorem 5.93, 5.94 to a standard normal random matrix, and definition of principal angles. Recall that $(\tilde{\boldsymbol{U}}_0)_{ij} \overset{iid}{\sim} \mathcal{N}(0,1)$ and consider its reduced QR decomposition, $\tilde{\boldsymbol{U}}_0 = \hat{\boldsymbol{U}}_0 \boldsymbol{R}_0$. We know that

$$\begin{aligned}
\mathrm{SE}^2(\tilde{\boldsymbol{U}}_0, \boldsymbol{U}) &= \|(\boldsymbol{I} - \hat{\boldsymbol{U}}_0\hat{\boldsymbol{U}}_0^\top)\boldsymbol{U}\|^2 = \lambda_{\max}(\boldsymbol{I} - \boldsymbol{U}^\top\hat{\boldsymbol{U}}_0\hat{\boldsymbol{U}}_0^\top\boldsymbol{U}) \\
&= 1 - \lambda_{\min}(\boldsymbol{U}^\top\hat{\boldsymbol{U}}_0\hat{\boldsymbol{U}}_0^\top\boldsymbol{U}) \\
&= 1 - \lambda_{\min}(\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_0\boldsymbol{R}_0^{-1}(\boldsymbol{R}_0^{-1})^\top\tilde{\boldsymbol{U}}_0^\top\boldsymbol{U}) \\
&\overset{(a)}{\leq} 1 - \lambda_{\min}(\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_0\tilde{\boldsymbol{U}}_0^\top\boldsymbol{U})\lambda_{\min}(\boldsymbol{R}_0^{-1}(\boldsymbol{R}_0^{-1})^\top)) \\
&= 1 - \frac{\sigma_{\min}^2(\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_0)}{\|\tilde{\boldsymbol{U}}_0\|_2^2}
\end{aligned}$$

where (a) follows from Ostrowski's Theorem (Theorem 4.5.9, [16]) and the last relation follows since reduced qr decomposition preserves the singular values. It is easy to see that $(\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_0)_{ij} \sim \mathcal{N}(0,1)$. We can apply Theorem 5.94 to get that with probability at least $1 - \exp(-cr) - (c/\gamma)$,

$$\sigma_{\min}(\boldsymbol{U}^\top\tilde{\boldsymbol{U}}_0) \geq c(\sqrt{r} - \sqrt{r-1})/\gamma$$

and we also know that $\sqrt{r} - \sqrt{r-1} = O(1/\sqrt{r})$. Additionally, the denominator term is bounded using Theorem 5.93 as done before and thus, with probability $1 - \exp(-\epsilon^2)$,

$$\|\tilde{\boldsymbol{U}}_0\| \leq C(\sqrt{n} + \sqrt{r} + \epsilon)$$

and now picking $\epsilon = 0.01\sqrt{n}$ we get that with probability at least $1 - \exp(-cn) - \exp(-cr) - (1/c\gamma)$,

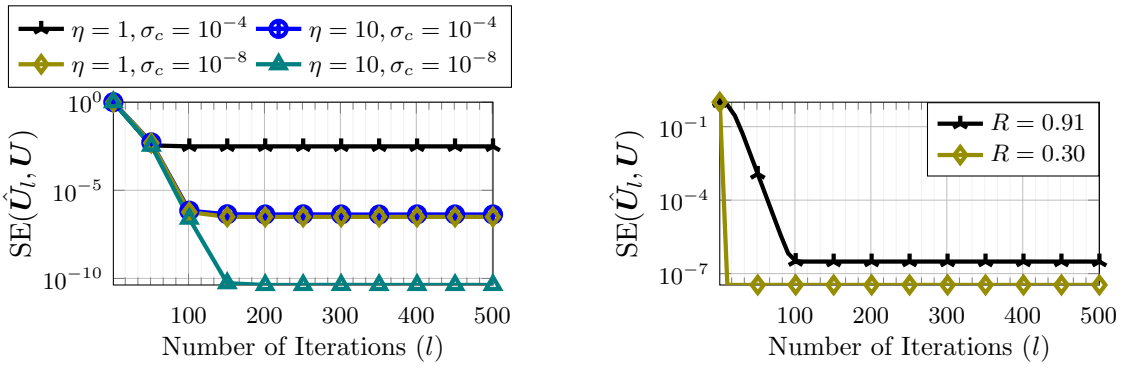
$$\mathrm{SE}^2(\tilde{\boldsymbol{U}}_0, \boldsymbol{U}) \leq 1 - \frac{1}{\gamma n r}$$

which completes the proof. □

While invoking the above result, to simplify notation, we set $\gamma = 10$.

5.10.4 Numerical Verificaion of Theorem 5.91

We generate $\boldsymbol{S} = \boldsymbol{U}\Lambda\boldsymbol{V}^T + \boldsymbol{U}_\perp\Lambda_\perp\boldsymbol{V}_\perp^T$ with $\boldsymbol{U}^* = [\boldsymbol{U}, \boldsymbol{U}_\perp]$, $\boldsymbol{V}^* = [\boldsymbol{V}, \boldsymbol{V}_\perp]$ being orthonormal matrices of appropriate dimensions. We then set $\boldsymbol{Y} = \boldsymbol{S}\boldsymbol{S}^T$ and the goal is to estimate the span of



(a) Varying σ_c and η

(b) Varying the EigenGap, R

Figure 5.3: Numerical verification of Theorem 5.91: **Left:** increasing η increases robustness to noise; **Right:** Increasing the "gap" helps achieve faster, better convergence.

the $n \times r$ dimensional matrix, $\boldsymbol{U}$. We choose $n = 1000$ and $r = 30$. We consider two settings where $\Lambda = 1.1\boldsymbol{I}$, $\Lambda_\perp = \boldsymbol{I}$ so that $R = 0.91$; and $\Lambda = 3.3\boldsymbol{I}$, $\Lambda_\perp = \boldsymbol{I}$ so that $R = 0.33$. At each iteration we generate channel noise as i.i.d. $\mathcal{N}(0, \sigma_c^2)$. We verify the claims of Theorem 5.91 and (i) show that choosing a larger value of η considerably increases robustness to noise. We set $R = 0.91$, and consider $\eta = 1, 10$ and $\sigma_c = 10^{-4}, 10^{-4}$. See from Fig. 5.3(a) that increasing η has a similar effect as that of reducing σ_c (the $\eta = 10, \sigma_c = 10^{-8}$ plot overlaps with $\eta = 1, \sigma_c = 10^{-8}$); and (ii) in Fig. 5.3(b) we show that choosing a smaller value of R speeds up convergence, and also increases noise robustness. Here we use $\sigma_c = 10^{-8}$ and consider two eigengaps, $R = \{0.91, 0.30\}$.

5.11 Appendix E: Preliminaries

The following result is Theorem 4.4.5, [39]

Theorem 5.93 (Upper Bounding Spectral Norm)**.** *Let A be a $m \times n$ random matrix whose entries are independent zero-mean sub-Gaussian r.v.'s and let $K = \max_{i,j} \|A_{i,j}\|_{\psi_2}$. Then for any $\epsilon > 0$ with probability at least $1 - 2\exp(-\epsilon^2)$,*

$$\|A\| \leq CK(\sqrt{m} + \sqrt{n} + \epsilon)$$

The following result (Theorem 1.1, [36]) bounds the smallest singular value of a random rectangular matrix.

Theorem 5.94 (Lower Bounding Smallest Singular Value for Rectangular matrices)**.** *. Let A be a $m \times n$ random matrix whose entries are independent zero-mean sub-Gaussian r.v.'s. Then for any $\epsilon > 0$ we have*

$$\sigma_{\min}(A) \geq \epsilon C_K(\sqrt{m} - \sqrt{n-1})$$

with probability at least $1 - \exp(-c_K n) - (c_K \epsilon)^{m-n+1}$. Here, $K = max_{i,j}\|A_{i,j}\|_{\psi_2}$.

CHAPTER 6. CONCLUSIONS AND FUTURE WORK

In this work we designed and analyzed provable algorithms for learning and tracking low-dimensional linear subspaces. We showed that under mild conditions, we can efficiently learn, detect, and track subspaces. In particular we considered the problem of tracking the underlying subspace in presence of gross outliers in Chapters 2 and 3. In Chapter 4, we also allowed for missing data due to failures in data acquisition, and transmission pipeline. In Chapter 5, we studied the subspace learning problem in a federated setting which also takes into account the data being available to a central server in a distributed fashion. In all our results, we show that our algorithm enjoys several desirable properties such as fast run time, improved outlier tolerance, and in some cases, near-optimal memory and sample complexity.

There are several avenues for possible future work (a) an immediate extension of Chapter 5 is to provide a guarantee for differentially private (robust) subspace tracking wherein, one deliberately adds noise to each algorithm iterate so that given the output, it is not possible for a malicious agent to ascertain whether a particular data point exists in the database or not; (b) whereas we considered learning linear subspaces in all work, it is certainly possible to extend this to learning non-linear low-dimensional models (such as manifolds); and finally, (c) extending our results to the case where measurements are a (3-rd order) tensor rather than a matrix would provide more room to exploit the underlying low-dimensional structure.