Efficient Federated Low Rank Matrix Completion

Ahmed Ali Abbasi and Namrata Vaswani

Abstract

In this work, we develop and analyze a novel Gradient Descent (GD) based solution, called Alternating GD and Minimization (AltGDmin), for efficiently solving the low rank matrix completion (LRMC) in a federated setting. Here "efficient" refers to communication-, computation- and sample- efficiency. LRMC involves recovering an $n \times q$ rank-r matrix $\boldsymbol{X}^\star$ from a subset of its entries when $r \ll \min(n, q)$. Our theoretical bounds on the sample complexity and iteration complexity of AltGDmin imply that it is the most communication-efficient solution while also been one of the most computation- and sample- efficient ones. We also extend our guarantee to the noisy LRMC setting. In addition, we show how our lemmas can be used to provide an improved sample complexity guarantee for the Alternating Minimization (AltMin) algorithm for LRMC. AltMin is one of the fastest centralized solutions for LRMC; with AltGDmin having comparable time cost even for the centralized setting.

Index Terms

Low rank matrix completion, Federated learning, Alternating Gradient Minimization (AltGDMin), Alternating Minimization (AltMin).

I. Introduction

In this work, we introduce a fast and communication-efficient solution for solving the low rank matrix completion (LRMC) problem [2, 3, 4, 5, 6, 6, 7, 8, 9, 10] in a federated setting. Our proposed algorithm, and the guarantees that we prove for it, are applicable in centralized settings as well. LRMC finds important applications in recommender systems' design, survey data analysis, and video inpainting. Federation means that (i) different subsets of the data are acquired at different distributed nodes; and (ii) all nodes can only communicate with the central node or "center".

Communication efficiency is a key concern with all distributed algorithms, including federated ones. Privacy of the data is another concern in a federated setting. In this work, "privacy" means the following: the nodes' raw data (observed matrix entries) cannot be shared with the center and the center should not be able reconstruct the unknown LR matrix.

A. Notation and Problem Setup

1) Notation: For any matrix $\boldsymbol{M}$, $\boldsymbol{m}_k$ denotes its k-th column while $\boldsymbol{m}^j$ denotes its j-th row transposed (so it is a column vector). $(\cdot)^\intercal$ denotes the matrix/vector transpose. We use $\boldsymbol{I}$ to denote the identity matrix and $\boldsymbol{e}_k$ to denote its k-th column (this is a 1-0 vector with 1 at the k-th location and zero everywhere else). We use $\|\cdot\|$ to denote either the ℓ_2 norm of a vector or the

Symbol	Description
$\boldsymbol{X}^\star \in \mathbb{R}^{n \times q}$	The unknown rank-r matrix $\boldsymbol{X}^\star \overset{\text{SVD}}{=} \boldsymbol{U}^* \Sigma^* \boldsymbol{V}^* = \boldsymbol{U}^* \boldsymbol{B}^*$
$\boldsymbol{u}^j \in \mathbb{R}^n$	j-th row of $\boldsymbol{U}^*$
$\boldsymbol{v}_k^* \in \mathbb{R}^q$	k-th column of $\boldsymbol{V}^*$
$\mu \in \mathbb{R}$	Coherence parameter μ such that $\max_{j \in [n]} \lVert \boldsymbol{u}^{*j} \rVert \leq \mu \sqrt{r/n}, \max_{k \in [q]} \lVert \boldsymbol{v}_k^* \rVert \leq \mu \sqrt{r/q}$
$\mu_{\boldsymbol{u}} \in \mathbb{R}$	$\mu_{\boldsymbol{u}} = 20\kappa^2\mu.$
$\mathcal{U}$	The set of $n \times r$ row-incoherent matrices. $\mathcal{U} := \{\check{\boldsymbol{U}} : \max_{j \in [n]} \lVert \check{\boldsymbol{u}}^j \rVert \leq \mu \sqrt{r/n}\}$
$\kappa \in \mathbb{R}$	Condition number $\kappa = \sigma^*_{\max}/\sigma^*_{\min}$, where $\sigma^*_{\max} = \sigma_1(\boldsymbol{X}^\star)$, $\sigma^*_{\min} = \sigma_r(\boldsymbol{X}^\star)$
ξ_{jk}	i.i.d Bernoulli random variable with $\Pr[\xi_{jk} = 1] = p$
Ω	$\Omega :- \{(j,k) \mid \xi_{jk} = 1\}$, the set of observed entries of $\boldsymbol{X}^\star$
Ω_k	$\Omega_k := \{j \in [n] \mid \xi_{jk} = 1\}$ Set of indices of observed entries of k-th column of $\boldsymbol{X}$
$\boldsymbol{S}_k \in \mathbb{R}^{n \times n}$	Row sampling matrix, $\boldsymbol{S}_k := \boldsymbol{I}_{\Omega_k}$, $k \in [q]$
$\boldsymbol{y}_k \in \mathbb{R}^n$	$\boldsymbol{y}_k = \boldsymbol{S}_k \boldsymbol{x}_k^\star$, the k-th column of $\boldsymbol{Y}$
γ	Total number of nodes; we assume $\gamma = O(1)$ in this work
$\mathcal{R}_\ell$	The indices of the subset of columns of $\boldsymbol{Y}$ available at node ℓ
$\text{GradU} \in \mathbb{R}^{n \times r}$	The gradient of $f(\boldsymbol{U}, \boldsymbol{B}) = \lVert (\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{B})_\Omega \rVert_F^2$ with respect to $\boldsymbol{U}$

TABLE I: Table summarizing the notation used in this paper.

The authors are at the Department of Electrical and Computer Engineering, Iowa State University. Email: namrata@iastate.edu. A part of this work (only simulation experiments) appeared in Allerton 2023 [1].

induced ℓ_2 norm of a matrix ($\|\boldsymbol{M}\| := \max_{\boldsymbol{z}:\|\boldsymbol{z}\|=1} \|\boldsymbol{M}\boldsymbol{z}\| = \sigma_{\max}(\boldsymbol{M})$), while $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. The notation $[q] := \{1, 2, \ldots, q\}$. For a tall matrix $\boldsymbol{M} \in \mathbb{R}^{n\times r}$, $\boldsymbol{M}^\dagger \triangleq (\boldsymbol{M}^\intercal\boldsymbol{M})^{-1}\boldsymbol{M}^\intercal$ and $\mathrm{QR}(\boldsymbol{M})$ maps $\boldsymbol{M}$ to $\boldsymbol{Q} \in \mathbb{R}^{n\times r}$ such that $\boldsymbol{M} = \boldsymbol{Q}\boldsymbol{R}$ is the QR decomposition of $\boldsymbol{M}$. In this work, we only use QR decomposition for tall $n \times r$ matrices with $r < n$. For tall $n \times r$ matrices $\boldsymbol{U}_1, \boldsymbol{U}_2$ with orthonormal columns, the two commonly used measures of subspace distance (SD) between their column spans are [11, Sec 2.2] $\mathrm{SD}_2(\boldsymbol{U}_1, \boldsymbol{U}_2) := \|(\boldsymbol{I}-\boldsymbol{U}_1\boldsymbol{U}_1^\intercal)\boldsymbol{U}_2\|$ and $\mathrm{SD}_F(\boldsymbol{U}_1, \boldsymbol{U}_2) := \|(\boldsymbol{I}-\boldsymbol{U}_1\boldsymbol{U}_1^\intercal)\boldsymbol{U}_2\|_F$ with $\mathrm{SD}_F \le \sqrt{r}\mathrm{SD}_2$. These have been used in past works on LR matrix recovery, e.g., [4, 12, 13]. We use $\boldsymbol{M} = \mathrm{diag}(m_k, k \in [q])$ to denote a diagonal $q \times q$ matrix with scalar entries m_k and $\boldsymbol{M} = \mathrm{blockdiag}(\boldsymbol{M}_k, k \in [q])$ to denote a $qr \times qr$ block-diagonal matrix with diagonal $r\times r$ blocks $\boldsymbol{M}_k$. Also, we use $vec(\boldsymbol{M})$ to vectorize the matrix $\boldsymbol{M}$ column-wise; thus $\|\boldsymbol{M}\|_F = \|vec(\boldsymbol{M})\|$. For a set Ω, we use $|\Omega|$ to denote its cardinality. For a set of matrix entry indices Ω, we use $\boldsymbol{M}_\Omega$ to refer to the matrix $\boldsymbol{M}$ with all entries whose indices are not in Ω zeroed out. For a set of matrix row indices Ω_{row}, we use $\boldsymbol{I}_{\Omega_{row}}$ to denote the identity matrix with diagonal entries whose indices are not in Ω_{row} zeroed out. *We reuse the letters c, C to denote different numerical constants in each use with the convention $c < 1$ and $C \ge 1$.*

2) LRMC problem: LRMC involves recovering a rank-r matrix $\boldsymbol{X}^\star \in \mathbb{R}^{n\times q}$, where $r \ll \min(n, q)$, from a subset of its entries. Entry j of column k, denoted $\boldsymbol{X}^\star_{jk}$, is observed, independently of all other observations, with probability p. Let $\xi_{jk} \overset{\text{iid}}{\sim} \text{Bernoulli}(p)$ for $j \in [n], k \in [q]$. Then, the set of observed entries, denoted by Ω, is

$$\Omega := \{(j, k) : \xi_{jk} = 1\}$$

By setting the unobserved entries to zero, the observed data matrix $\boldsymbol{Y} \in \mathbb{R}^{n\times q}$ can be defined as

$$\boldsymbol{Y}_{jk} := \begin{cases} \boldsymbol{X}^\star_{jk} & \text{if } (j,k) \in \Omega, \\ 0 & \text{otherwise.} \end{cases} \quad \text{or, equivalently,} \quad \boldsymbol{Y} := \boldsymbol{X}^\star_\Omega \tag{1}$$

We use $\Omega_k := \{j \in [n] \mid \xi_{jk} = 1\}$ to denote the set of indices of the observed entries in column k and we define a diagonal row-sampling matrix $\boldsymbol{S}_k \in \mathbb{R}^{n\times n}$ as

$$\boldsymbol{S}_k := \boldsymbol{I}_{\Omega_k}$$

Thus $(\boldsymbol{S}_k)_{j,j} = \xi_{jk}$, for all $j \in [n]$. With this, we can express

$$\boldsymbol{y}_k := \boldsymbol{S}_k\boldsymbol{x}^\star_k \text{ for all } k \in [q].$$

Here $\boldsymbol{y}_k$, $\boldsymbol{x}^\star_k$ denote the k-th columns of $\boldsymbol{Y}$, $\boldsymbol{X}^\star$ respectively. Let $\boldsymbol{X}^\star \overset{\text{SVD}}{=} \boldsymbol{U}^\star(\boldsymbol{\Sigma}^*\boldsymbol{V}^*) := \boldsymbol{U}^\star\boldsymbol{B}^\star$ denote its reduced singular value decomposition (SVD) with $\boldsymbol{U}^\star \in \mathbb{R}^{n\times r}$ with orthonormal columns, $\boldsymbol{V}^\star \in \mathbb{R}^{r\times q}$ with orthonormal rows, and $\boldsymbol{\Sigma}^*$ being a diagonal $r \times r$ matrix. We use $\kappa = \sigma^*_{\max}/\sigma^*_{\min}$ to denote the condition number of the diagonal $r \times r$ matrix $\boldsymbol{\Sigma}^*$. Here $\sigma^*_{\max}, \sigma^*_{\min}$ are its largest, smallest singular values. Also, we let $\boldsymbol{B}^\star := \boldsymbol{\Sigma}^*\boldsymbol{V}^*$ so that $\boldsymbol{X}^\star = \boldsymbol{U}^\star\boldsymbol{B}^\star$.

3) Assumption: As in all past works on LRMC [2, 3, 4, 5, 6, 6, 7, 8, 9, 10], we need the following assumption on the singular vectors of $\boldsymbol{X}^\star$. This is a way to guarantee that the rows and columns of $\boldsymbol{X}^\star$ are dense (non-sparse). This and the LR assumption, along with the i.i.d. Bernoulli observed entries selection, help ensure that one can correctly interpolate (fill in) the missing entries even with observing only a few entries of each row or column.

Assumption 1.1 (μ-incoherence of singular vectors of $\boldsymbol{X}^\star$)**.** *Assume row norm bounds on $\boldsymbol{U}^\star$: $\max_{j\in[n]} \|\boldsymbol{u}^{*j}\| \le \mu\sqrt{r/n}$, and column norm bounds on $\boldsymbol{V}^*$: $\max_{k\in[q]} \|\boldsymbol{v}^*_k\| \le \mu\sqrt{r/q}$ for a μ that is not too large. In most of the discussion in this work, we assume that μ is a numerical constant. Since $\boldsymbol{B}^\star = \boldsymbol{\Sigma}^*\boldsymbol{V}^*$, this implies that $\|\boldsymbol{b}^\star_k\| \le \mu\sqrt{r/q}\sigma^*_{\max}$.*

Remark 1.2. *The above assumption can also be interpreted as follows. Let $\mu :- \max\left(\max_{j\in[n]} \|\boldsymbol{u}^{*j}\| \cdot \sqrt{n/r}, \max_{k\in[q]} \|\boldsymbol{v}^*_k\| \cdot \sqrt{q/r}\right)$. We are assuming that μ is not too large; our discussion of complexities treats it as a numerical constant.*

The above definition of μ was used in [4, 6, 10]. Other works such as [14] defined μ as the square of the above quantity. A different and weaker notion of coherence was used in [15, 16].

To understand the need for the above assumption, we repeat the example from [2, Sec 1.1.1]. Let $\boldsymbol{X}^\star = C\boldsymbol{e}_1\boldsymbol{e}_1^\intercal \in \mathbb{R}^{n\times n}$ be a rank $r = 1$ matrix. For this matrix, $\boldsymbol{U}^\star = \boldsymbol{e}_1$ and $\boldsymbol{V}^* = \boldsymbol{e}_1^\intercal$. Thus , $\mu = \sqrt{n} = \sqrt{n/r}$ (since $r = 1$). This is a very large value of μ (in fact this is the largest value that μ can take). In this case, it is impossible to estimate $\boldsymbol{X}^\star$ even with almost all observed entries. If the set of observed entries Ω does not contain the (1,1) entry, the observed matrix $\boldsymbol{Y}$ would be the all zeros matrix.

4) Federation: We assume that there are a total of γ nodes, with $\gamma \le q$. Each node has access to a different subset of the columns of the observed data matrix $\boldsymbol{Y}$. We use $\mathcal{R}_\ell$ to denote the subset of columns of $\boldsymbol{Y}$ available at node ℓ. The sets $\mathcal{R}_\ell$ form a partition of $[q]$, i.e., they are mutually disjoint and $\cup_{\ell=1}^{\gamma}\mathcal{R}_\ell = [q]$. Let $\boldsymbol{Y}_\ell = [\boldsymbol{y}_k, k \in \mathcal{R}_\ell]$, $\boldsymbol{X}^\star_\ell = [\boldsymbol{x}^\star_k, k \in \mathcal{R}_\ell]$, and $\boldsymbol{B^\star}_\ell = [\boldsymbol{b}^\star_k, k \in \mathcal{R}_\ell]$. We have

$$\boldsymbol{Y} = [\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots \boldsymbol{Y}_\gamma] \text{ where } \boldsymbol{Y}_\ell = (\boldsymbol{X}^\star_\ell)_{\Omega_{(\ell)}} = (\boldsymbol{U}^\star\boldsymbol{B^\star}_\ell)_{\Omega_{(\ell)}}$$

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2025.3563450

Algorithm	Computation Comp.	Communic. Comp.	Sample Comp.
AltGDMin **(this work)** (Private)	$\kappa^2 \frac{\lvert\Omega\rvert}{\gamma} r^2 \log(\frac{1}{\epsilon})$	$\kappa^2 nr \log(\frac{1}{\epsilon})$	$\kappa^6 \mu^2 qr^2 \log q \log(\frac{1}{\epsilon})$ **(this work)**
FactGD [9, 10] (Private)	$\kappa\mu \frac{\lvert\Omega\rvert}{\gamma} r^2 \log(\frac{1}{\epsilon})$	$\kappa\mu nr^2 \log(\frac{1}{\epsilon})$	$\kappa^4 \mu^2 qr^2 \log q$
AltMin [4] (Private)	$\frac{\lvert\Omega\rvert}{\gamma} r \log^2(\frac{1}{\epsilon})$	$nr \log^2(\frac{1}{\epsilon})$	$\kappa^4 \mu^2 qr^{4.5} \log q \log(\frac{1}{\epsilon})$ [4] $\kappa^4 \mu^2 qr^2 \log n \log(1/\epsilon)$ **(this work)**
AltMin [4] (Not-Private)	$\frac{\lvert\Omega\rvert}{\gamma} r^2 \log(\frac{1}{\epsilon})$	$\frac{\lvert\Omega\rvert}{\gamma} \log(\frac{1}{\epsilon})$	$\kappa^4 \mu^2 qr^{4.5} \log q \log(\frac{1}{\epsilon})$ [4] $\kappa^4 \mu^2 qr^2 \log n \log(1/\epsilon)$ **(this work)**
Smooth-AltMin [14] (Not-Private)	$\frac{\lvert\Omega\rvert}{\gamma} r^2 \log q \log(\frac{1}{\epsilon})$	$\frac{\lvert\Omega\rvert}{\gamma} \log q \log(\frac{1}{\epsilon})$	$\kappa^2 \mu^2 qr^3 \log q \log(\frac{1}{\epsilon})$ [14] $C\kappa^2 \mu^2 qr^2 (\log(n/\epsilon) \log^2 n)$ **(this work)**
ProjGD [6, 18] (Not-Private)	$\mu^4 \frac{\lvert\Omega\rvert}{\gamma} r \log^2(\frac{1}{\epsilon})$	$\mu^4 \frac{\lvert\Omega\rvert}{\gamma} \log^2(\frac{n}{\epsilon})$	$\mu^4 qr^2 \log^2 n \log^2(\frac{1}{\epsilon})$

TABLE II: **Comparing computation, communication and sample complexities for federated LRMC.** $\gamma = 1$ **gives the centralized LRMC computation cost.** *Here Communic Comp* $= T \cdot \max($*Communic.(node), Communic.(center)*$)$*. Similarly for the computation cost. For the sample complexities, we assume* $\max(n, q) = q$*. Treating* κ, μ, γ *as numerical constants, and assuming* $r < \log(1/\epsilon)$ *and* $|\Omega| \geq qr^2$ *(sample complexity needed for accurate recovery), clearly, AltGDmin is the most communication-efficient, while all of AltGDmin, AltMin and FactGD are equally computationally efficient.*

To keep notation simple, we assume that q is a multiple of γ and $|\mathcal{R}_\ell| = q/\gamma$. This federated LRMC setting is also considered in [17]. Since the LRMC problem is symmetric, if each node had access to a different subset of rows of $\boldsymbol{Y}$, we would transpose both $\boldsymbol{Y}$ and $\boldsymbol{X}^\star$ and convert the problem to this one.

In a federated setting, the two desirable properties are communication-efficiency and "privacy". *In this work, "privacy" means the following. The nodes' raw data cannot be shared with the center and the center should not be able reconstruct* $\boldsymbol{X}^\star$*.*

Our discussion treats γ as a numerical constant. Thus order $|\Omega|/\gamma$ is equal to order $|\Omega|$. Also, $|\Omega| \geq (n+q)r$ is a necessary condition since the number of observed matrix entries needs to be larger than the number of unknowns for specifying an $n \times q$ rank r matrix. Such a matrix can always be expressed as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$ where $\boldsymbol{U}, \boldsymbol{B}$ have r columns and rows respectively. .

5) Applications: An important application where the LRMC problem occurs is movie, or any product, recommendation system design. The goal is to fill in the missing entries of the $n \times q$ product-ratings' matrix; this is a matrix with q total users and n products, and with column k denoting the preferences of user k for the n products. This matrix can be modeled as being low-rank (LR) as first argued in [2, 4]. The idea is that user preferences are governed by much fewer factors (r factors) than either n or q [2, 4]. Users rate only a small subset of movies/products and hence we only have a few observations of this LR matrix. An important setting where the above federation is relevant is a movie recommendation system designed for a set of γ dorms within a university, or for a set of apartment buildings in a small town. Node ℓ is the router for dorm or apartment building ℓ. Each node (building) has samples of a different subset of q/γ users (residents). Another application noted in [17] involves γ different hospitals having records for different groups of patients (users) about some of a large set of diseases (items); hospitals want to cooperatively train a patient-disease prediction model. In all these examples, the number of nodes γ is much smaller than the number of users at a node, q/γ.

B. Related Work

Starting with the seminal work of [2, 19] which introduced a nuclear norm based convex relaxation, the LRMC problem has been extensively studied in the last decade and a half [2, 3, 4, 5, 6, 6, 7, 8, 9, 10, 19]. Two classes of algorithms feature prominently in this literature - solutions to convex relaxations and direct iterative algorithms. The former [2, 19] are slow: the required number of iterations for ϵ accuracy (iteration complexity) grows as $1/\sqrt{\epsilon}$ [4]. The first provably accurate iterative solution was the Alternating Minimization (AltMin) algorithm with a spectral initialization [3, 4]. AltMin was shown to converge geometrically, i.e., its error was shown to decay as c^t with iteration count t, for a numerical constant $c < 1$, with a sample complexity of order $\kappa^4 \mu^2 nr^{4.5} \log(1/\epsilon)$ in [4]. AltMin factorizes the unknown LR matrix $\boldsymbol{X}$ as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$. After initializing $\boldsymbol{U}$, it alternately updates $\boldsymbol{B}$ and $\boldsymbol{U}$ using minimization for one, keeping the other fixed. Subsequent work [14] considered a modified version of AltMin and improved its sample complexity to order $\kappa^2 \mu^2 nr^3 \log(1/\epsilon)$. The work of [8] introduced a complicated modification of AltMin with the goal of proving a result that has almost no dependence on κ; however their guarantee instead has a much worse dependence of r^9 on the matrix rank r.

Later works proposed two gradient descent (GD) based algorithms - Projected GD (ProjGD) [6, 18] and Factorized GD (FactGD) [9, 10] - that reduced the sample complexity dependence on the rank to r^2. ProjGD involves GD, followed by projection onto the space of rank r matrices after each GD iteration. FactGD factorizes the unknown LR matrix $\boldsymbol{X}$ as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$, where $\boldsymbol{U}, \boldsymbol{B}$ have r columns and rows respectively, and updates both by GD as follows. At each iteration, it updates $\boldsymbol{U}$ and $\boldsymbol{B}$ by one GD step for the cost function $g(\boldsymbol{U}, \boldsymbol{B}) := f(\boldsymbol{U}, \boldsymbol{B}) + c_1 \|\boldsymbol{U}^\intercal \boldsymbol{U} - \boldsymbol{B}\boldsymbol{B}^\intercal\|_F^2$; followed by projecting each of them onto the set of matrices with incoherent rows and columns respectively. Initialization involves computing the top r left singular vectors of $\boldsymbol{Y}$ followed by projecting their matrix on the set of row incoherent matrices. The second term of $g(\boldsymbol{U}, \boldsymbol{B})$ is a norm balancing term that ensures the norm of $\boldsymbol{U}$ does not keep increasing with iterations while that of $\boldsymbol{B}$ decreases (or vice versa). The best ProjGD guarantee needs a sample complexity of $\mu^4 nr^2 \log^2 n \log^2(1/\epsilon)$ while FactGD reduces this even further to $\kappa^4\mu^2 nr^2 \log^2 n$. For the error to decay exponentially with iterations, a constant GD step size suffices for ProjGD. On the other hand, FactGD needs the GD step size to be of order $1/r$. Because of this, its iteration complexity is worse than that of ProjGD by a factor of r. However, in terms of per-iteration time cost, ProjGD is significantly slower. Consequently, overall in terms of total cost, it is slower. Numerically, it is much slower than all other methods, because each ProjGD iteration involves a rank r SVD step (constants per iteration are much larger); see of [1, Fig. 1b and 1d]. This discussion is summarized in Table II.

To our best knowledge, *there is no existing work on provably accurate federated LRMC*. Federation requires a communication-efficient and private solution, with "private" as defined above.

The Alternating GD and Minimization (AltGDmin) algorithm was introduced in our past work [13, 20] as a fast solution to the LR column-wise compressive sensing (LRCS) problem. It was also shown to be the most communication-efficient in a federated setting. LRCS involves recovering $\boldsymbol{X}^\star$ from $\boldsymbol{y}_k := \boldsymbol{A}_k \boldsymbol{x}^\star_k$ when $\boldsymbol{A}_k$'s are $m \times n$ i.i.d. random Gaussian matrices (each entry of each $\boldsymbol{A}_k$ is i.i.d. standard Gaussian), and the right singular vectors of $\boldsymbol{X}^\star$ satisfy the incoherence assumption.

In tangentially related work [21, 22], distributed-computing solutions to LRMC are studied. These do not consider the federated setting, instead these assume that all data is observed centrally and then is distributed to different nodes to parallelize the computation; and these develop an approximate solution to the convex relaxation which is known to be very slow. Other somewhat related works include [17, 23, 24] which study differential privacy or attack resilience for LRMC; and [15, 25, 26, 27, 28] which focus on the fully sampled matrix factorization problem.

C. Contributions and Novelty

In this work, we develop and analyze the AltGDmin algorithm for solving the LRMC problem. The design of AltGDmin is motivated by a federated setting. To our best knowledge, our work provides the first theoretical guarantees for solving LRMC in a federated setting; see Theorem 2.1. The sample and iteration complexity bounds that we prove are applicable in centralized settings as well. In addition, we also show how our lemmas can be used to provide an improved sample complexity guarantee for AltMin which is the fastest centralized solution for LRMC.

Using our results we can argue that, in a federated setting, AltGDmin is the most communication-efficient solution. It is also one of the two fastest private solutions and has the second smallest sample complexity. See Table II. Our new corollary for AltMin, and smooth AltMin, proves that both have the same sample complexity as that of AltGDmin. This discussion treats κ, μ and γ (number of nodes) as numerical constants.

1) Novelty of Proof Techniques: The most important difference between LRCS and LRMC is that the LRMC proof requires incoherence of each algorithm iterate $\boldsymbol{U}$ and $\boldsymbol{B}$. When analyzing AltGDmin for LRCS [13, 20], this was needed only for $\boldsymbol{B}$ because LRCS measurements are column-wise global. To show the incoherence of $\boldsymbol{U}$ at each iteration, we cannot borrow ideas from existing work because the update of $\boldsymbol{U}$ in AltGDmin is different from that in all existing LRMC solutions. FactGD [10] projects $\boldsymbol{U}$ onto the space of row incoherent matrices after each GD step; this automatically ensures its incoherence after each update. AltMin [4] updates $\boldsymbol{U}$ by solving a least squares problem (and not by GD) and so its incoherence proof is different. ProjGD [6] implements projected GD for $\boldsymbol{X}$, it does not factorize $\boldsymbol{X}$. There are other important proof differences too between our work and the two works from which we borrow some proof ideas – [20] and [4]. We explain these in Sec. III.

2) Paper Organization: We develop the AltGDmin algorithm and give our main theoretical guarantee, Theorem 2.1, for it, along with a detailed discussion comparing it to existing work, in Sec. II. Sec. III describes the novel proof ideas. Theorem 2.1 is proved in Sec.IV. The lemmas used in this proof are proved in Sec. V. The corollary for noisy LRMC is provided in Sec. VI and proved in the Appendix. We show how we can use our lemmas to also prove an improved guarantee for AltMin and one for Smooth AltMin in Sec. VII. Numerical experiments are described and discussed in Sec. VIII. We conclude in Sec. IX.

We summarize the symbols used in this work in Table I.

II. Alternating GD and Minimization (AltGDMin) Algorithm and Guarantees

A. AltGDmin overall idea

The goal is to minimize the following squared loss cost function

$$\min_{\boldsymbol{B}, \boldsymbol{U}:\, \boldsymbol{U}^\intercal \boldsymbol{U} = \boldsymbol{I}} f(\boldsymbol{U}, \boldsymbol{B}), \ f(\boldsymbol{U}, \boldsymbol{B}) := \|(\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{B})_\Omega\|_F^2 \tag{2}$$

We impose the orthonormal columns constraint on $\boldsymbol{U}$ to ensure that $\|\boldsymbol{U}\| = 1$ always. Since $\boldsymbol{U}\boldsymbol{B} = \boldsymbol{U}\boldsymbol{M}\boldsymbol{M}^{-1}\boldsymbol{B}$ for any invertible $r \times r$ matrix $\boldsymbol{M}$, if we do not impose this constraint, the norm of $\boldsymbol{U}$ can keep increasing over iteration, while that of $\boldsymbol{B}$ decreases (or vice versa).

The AltGDmin algorithmic framework was introduced in [13, 20] for solving the LRCS problem. More generally, it is useful for solving any partly decoupled optimization problem. To understand what this means, consider solving $\min_{\boldsymbol{Z}} f(\boldsymbol{Z})$. This is

Algorithm 1 AltGDMin

```
Require: partial observations $\boldsymbol{Y}$, rank $r$, step size $\eta$, and number of iterations $T$
Partition $\boldsymbol{Y}$ into $2T+1$ subsets $\boldsymbol{Y}_{\Omega^{(0)}}, \cdots, \boldsymbol{Y}_{\Omega^{(2T)}}$. Define $\boldsymbol{S}_k^{(t)} = \boldsymbol{I}_{\Omega_k}$, for all $k \in [q]$, for each subset.
$\boldsymbol{U}^{(00)} \leftarrow$ top $r$ left-singular vectors of $\boldsymbol{Y}_{\Omega^{(0)}}$
$\boldsymbol{M}^{(0)} \leftarrow \Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)})$
$\boldsymbol{U}^{(0)} \leftarrow \mathrm{QR}(\boldsymbol{M}^{(0)})$
for $t \in 1 \cdots T$ do
    $\boldsymbol{U}_k \leftarrow \boldsymbol{S}_k^{(t-1)} \boldsymbol{U}^{(t-1)}$ for all $k \in [q]$
    $\boldsymbol{b}_k^{(t)} \leftarrow (\boldsymbol{U}_k)^\dagger \boldsymbol{y}_k^{(t)}$ for all $k \in [q]$ and $\boldsymbol{B}^{(t)} = [\boldsymbol{b}_1^{(t)}, \boldsymbol{b}_2^{(t)}, \dots \boldsymbol{b}_q^{(t)}]$
    $\tilde{\boldsymbol{U}}^{(t)} \leftarrow \boldsymbol{U}^{(t-1)} - \eta(\boldsymbol{U}^{(t-1)}\boldsymbol{B}^{(t)} - \boldsymbol{Y})_{\Omega^{(T+t)}} \boldsymbol{B}^{(t)\intercal}$
    $\boldsymbol{U}^{(t)} \leftarrow \mathrm{QR}(\tilde{\boldsymbol{U}}^{(t)})$ , i.e. $\tilde{\boldsymbol{U}}^{(t)} \overset{\mathrm{QR}}{=} \boldsymbol{U}^{(t)}\boldsymbol{R}(t)$
Return $\boldsymbol{U}^{(T)}, \boldsymbol{B}^{(T)}$ and $\boldsymbol{X}^{(T)} := \boldsymbol{U}^{(T)}\boldsymbol{B}^{(T)}$
```

Algorithm 2 AltGDmin-Federated

```
Partition $\boldsymbol{Y}$ into $2T+1$ subsets $\boldsymbol{Y}_{\Omega^{(0)}}, \cdots, \boldsymbol{Y}_{\Omega^{(2T)}}$. Define $\boldsymbol{S}_k = \boldsymbol{I}_{\Omega_k}$, for all $k \in [q]$, for each subset.
//Initialize $\boldsymbol{U}^{(0)}$ by Power Method
Center: Initialize random $\boldsymbol{U} \in \mathbb{R}^{n \times r}$; push to nodes.
for $t \in 1 \cdots T_{\text{init}}$ do
    Node $\ell$, $\ell \in [\gamma]$
     $\boldsymbol{M}_\ell \leftarrow \sum_{k \in \mathcal{R}_\ell} \boldsymbol{y}_{\Omega_k^{(0)}} \boldsymbol{y}^\intercal_{\Omega_k^{(0)}} \boldsymbol{U}$; push to center.
    Center: $\boldsymbol{U} \leftarrow \mathrm{QR}(\sum_\ell \boldsymbol{M}_\ell)$; push to nodes.
Center: Clip and orthonormalize $\boldsymbol{U}$ as given in Algorithm 1
//AltGDMin Iterations
for $t \in 1 \cdots T$ do
    Node $\ell$, $\ell \in [\gamma]$:
    $\boldsymbol{U}_k \leftarrow \boldsymbol{S}_k^{(t-1)} \boldsymbol{U}^{(t-1)}$ for all $k \in \mathcal{R}_\ell$
    $\boldsymbol{b}_k^{(t)} \leftarrow (\boldsymbol{U}_k)^\dagger \boldsymbol{y}_k^{(t)}$ for all $k \in \mathcal{R}_\ell$
    $\mathrm{GradU}_\ell \leftarrow \sum_{k \in \mathcal{R}_\ell} (\boldsymbol{U}_k \boldsymbol{b}_k^{(t)} - \boldsymbol{y}_k^{(t)})(\boldsymbol{b}_k^{(t)})^\intercal$; push to center.
    Center:
    $\boldsymbol{U}^{(t)} \leftarrow \mathrm{QR}(\boldsymbol{U}^{(t-1)} - \eta \sum_\ell \mathrm{GradU}_\ell)$; push to nodes.
Return $\boldsymbol{U}^{(T)}, \boldsymbol{B}^{(T)}$ and $\boldsymbol{X}^{(T)} := \boldsymbol{U}^{(T)}\boldsymbol{B}^{(T)}$
```

partly-decoupled if we can split the set of optimization variables $\boldsymbol{Z}$ into two blocks, $\boldsymbol{Z} = \{\boldsymbol{Z}_a, \boldsymbol{Z}_b\}$, so that the minimization over $\boldsymbol{Z}_b$, keeping $\boldsymbol{Z}_a$ fixed, is decoupled. This means that it can be solved by solving many smaller-dimensional, and hence much faster, minimization problems over disjoint subsets of $\boldsymbol{Z}_b$. That over $\boldsymbol{Z}_a$, keeping $\boldsymbol{Z}_b$ fixed, may or may not satisfy such a property. If such a decoupling holds for $\boldsymbol{Z}_b$, and if the data is federated across the nodes in such a way that the smaller problems can be solved locally at the nodes, then AltGDmin provides a faster and more communication-efficient solution than AltMin. After initializing $\boldsymbol{Z}_a$, it alternatively updates $\boldsymbol{Z}_b, \boldsymbol{Z}_a$ using minimization for $\boldsymbol{Z}_b$ and GD for $\boldsymbol{Z}_a$. In the LRCS problem studied in our past work [13, 20], factoring $\boldsymbol{X}$ as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$, the optimization to be solved was $\min_{\boldsymbol{U},\boldsymbol{B}} \sum_k \|\boldsymbol{y}_k - \boldsymbol{A}_k \boldsymbol{U} \boldsymbol{b}_k\|_2^2$. Clearly in this case, $\boldsymbol{Z}_a = \boldsymbol{U}$, $\boldsymbol{Z}_b = \boldsymbol{B}$ since the problem is decoupled over $\boldsymbol{B}$ but is coupled over $\boldsymbol{U}$. For the LRMC problem being studied here, the decoupling holds for both $\boldsymbol{U}$ (with $\boldsymbol{B}$ fixed) and for $\boldsymbol{B}$ (with $\boldsymbol{U}$ fixed). Thus, we can pick either of them to be $\boldsymbol{Z}_b$. The choice of which one to pick as $\boldsymbol{Z}_b$ depends on how the data is federated. In our case, since the data is vertically federated, we use $\boldsymbol{Z}_b = \boldsymbol{B}$ again and $\boldsymbol{Z}_a = \boldsymbol{U}$.

B. AltGDmin algorithm for LRMC

AltGDmin for LRMC proceeds as follows. We initialize $\boldsymbol{U}$ as explained below; this approach is adapted from that in FactGD [10]. After the initialization, different from AltMin [4], which used alternating exact minimization for both $\boldsymbol{U}$ and $\boldsymbol{B}$, and different from FactGD [10], which used GD for updating both $\boldsymbol{U}$ and $\boldsymbol{B}$, AltGDmin alternates between exact minimization over $\boldsymbol{B}$ and a single GD step for $\boldsymbol{U}$. The GD step is followed by an orthonormalization (QR) step. The use of exact minimization for one of the variables helps ensure that AltGDmin provably converges with a nearly constant step size. Because of this, the AltGDmin iteration complexity is better than that of FactGD by a factor of r.

1) AltGDmin for LRMC: Initialization: As in most previous work [4, 10], the first step of our initialization computes the top r left singular vectors of $\boldsymbol{Y}$. Denote the $n \times r$ matrix formed by these left singular vectors by $\boldsymbol{U}^{(00)}$. This has computation cost [29] $|\Omega| r \cdot \log(1/\delta_0)$ where δ_0 is the accuracy level needed for the initialization step. Our guarantee given below proves that we need $\delta_0 = c/\kappa^2$; see Remark 2.2. Thus, this step has time cost $|\Omega| r \log \kappa$. This is followed by a step to make $\boldsymbol{U}^{(00)}$ incoherent. We borrow this step from [10]. It involves projecting $\boldsymbol{U}^{(00)}$ onto the space of row incoherent matrices,

$$\mathcal{U} := \{\boldsymbol{U} : \max_{j\in[n]} \|\check{\boldsymbol{u}}^j\| \leq \mu\sqrt{r/n}\} \tag{3}$$

i.e., computing the $n \times r$ matrix $\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) := \min_{\boldsymbol{U}\in\mathcal{U}} \|\boldsymbol{U} - \boldsymbol{U}^{(00)}\|_F$. It is easy to see that this projection can be obtained in closed form by the following row norm clipping operation:

$$[\Pi_{\mathcal{U}}(\boldsymbol{M})]^j = \boldsymbol{m}^j \cdot \min\left(1, \frac{\mu\sqrt{r/n}}{\|\boldsymbol{m}^j\|}\right), \quad \text{for all } j \in [n]$$

In words, if a row of $\boldsymbol{M}$ has ℓ_2 norm that is more than the threshold $\mu\sqrt{r/n}$, then one renormalizes the row so that its norm equals the threshold. If the norm is less than this threshold, then we do not change it. Clearly this is an order nr time operation. Finally, we obtain $\boldsymbol{U}^{(0)}$ by orthonormalizing $\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)})$ using QR decomposition. This needs time of order nr^2.

2) AltGDmin iterations: We alternately update $\boldsymbol{B}$ using minimization and $\boldsymbol{U}$ using GD. The minimization over $\boldsymbol{B}$ is a decoupled least squares (LS) problem since $f(\boldsymbol{U}, \boldsymbol{B})$ decouples as $f(\boldsymbol{U}, \boldsymbol{B}) = \sum_{k\in[q]} \|\boldsymbol{y}_k - \boldsymbol{S}_k\boldsymbol{U}\boldsymbol{b}_k\|^2$. We update $\boldsymbol{b}_k$ as

$$\boldsymbol{b}_k = \operatorname*{argmin}_{\check{\boldsymbol{b}}} \|\boldsymbol{y} - \boldsymbol{S}_k\boldsymbol{U}\check{\boldsymbol{b}}\|^2 = (\boldsymbol{S}_k\boldsymbol{U})^\dagger \boldsymbol{y}_k, \quad \text{for all } k \in [q]$$

Let $\boldsymbol{U}_k := \boldsymbol{S}_k\boldsymbol{U}$, then $\boldsymbol{b}_k := \boldsymbol{U}_k^\dagger \boldsymbol{y}_k$. Recall that $\boldsymbol{M}^\dagger := (\boldsymbol{M}^\intercal\boldsymbol{M})^{-1}\boldsymbol{M}^\intercal$. We emphasise here that we write things as above for notational ease. The computational complexity for computing $\boldsymbol{b}_k$ depends only on the sub-matrix of $\boldsymbol{U}_k$ with nonzero rows. This is of size $|\Omega_k| \times r$. Thus the cost of computing $\boldsymbol{b}_k$ is order $|\Omega_k| r^2$ for a given k. Hence, the total cost for all nodes is $\sum_k |\Omega_k| r^2 = |\Omega| r^2$.

We update $\boldsymbol{U}$ by one GD step followed by orthnormalization using QR, i.e.,

$$\tilde{\boldsymbol{U}}^+ = \boldsymbol{U} - \eta\nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B})), \text{ and } \boldsymbol{U}^+ = \mathrm{QR}(\tilde{\boldsymbol{U}}^+)$$

The gradient is

$$\nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B}) = 2((\boldsymbol{U}\boldsymbol{B})_\Omega - \boldsymbol{Y})\boldsymbol{B}^\intercal = 2\sum_{k=1}^{q}(\boldsymbol{S}_k\boldsymbol{U}\boldsymbol{b}_k - \boldsymbol{y}_k)\boldsymbol{b}_k^\intercal$$

For the gradient computation, the computational cost is $\sum_k |\Omega_k| r = |\Omega| r$. The QR step needs time of order nr^2.

We summarize the complete algorithm in Algorithm 1. Sample splitting (line 1) is assumed, as is common in most structured data recovery literature, e.g., [4, 6, 14, 18]. In fact, as we discuss in Sec. II-E, sample-splitting is assumed for obtaining provable guarantees for all iterative solutions for LRMC (and those for LRCS) in which one or both of the alternating steps is a minimization step.

C. Federated AltGDmin

The federated AltGDmin algorithm is given Algorithm 2. At each algorithm iteration, t, each node $\ell = 1, 2, \ldots, \gamma$ performs two operations i) updating $\boldsymbol{b}_k$ by the LS solution, for all $k \in \mathcal{R}_\ell$; and ii) computation of the partial gradient $\sum_{k\in\mathcal{R}_\ell} [\nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B})]_k = \sum_{k\in\mathcal{R}_\ell} (\boldsymbol{U}_k\boldsymbol{b}_k - \boldsymbol{y})_{\Omega_k} \boldsymbol{b}_k^\intercal$. Only the $n \times r$ partial gradient needs to be sent to the center. The center sums the received partial gradients, implements the GD step, and computes the QR decomposition, and broadcasts the updated $\boldsymbol{U}^+$ to all the nodes. This is used by the nodes in the next iteration. The communication cost from nodes to center is equal to the number of nonzero entries in $\nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B})$; this is $r \cdot \min(n, \sum_{k\in\mathcal{R}_\ell} |\Omega_k|) = r \cdot \min(n, (|\Omega|/\gamma)) = nr$ since we assume γ is a numerical constant and $|\Omega| \geq (n+q)r$ is a necessary lower bound for any approach to work. The center to nodes communication cost is also nr. The computation cost is as explained above with $|\Omega|$ replaced by $|\Omega|/\gamma$.

The initialization can be federated using the power method applied to $\boldsymbol{Y}\boldsymbol{Y}^\intercal$. This has time cost $\sum_{k\in\mathcal{R}_\ell} |\Omega_k| r = |\Omega| r/\gamma$ per power method iteration. The upstream (node to center) per-node communication cost is $\min(n, \sum_{k\in\mathcal{R}_\ell} |\Omega_k|) r = nr$ and the downstream cost is also nr. The power method converges linearly and thus, for δ_0 accuracy, $\log(1/\delta_0)$ iterations are required. Our guarantee given below proves that we need $\delta_0 = c/\kappa^2$; see Remark 2.2.

D. AltGDmin Guarantee

Recall from above that the per-iteration per-node computation and communication complexities of AltGDmin are $\max(nr^2, (|\Omega|/\gamma) r^2)$ and $\max(nr, \min(n, (|\Omega|/\gamma)) r) = nr$. We use this and the iteration complexity (expression for T) derived in the result below to provide an expression for its total per-node computation and communication complexities.

For LRMC, the sample complexity $|\Omega|$ is a random variable. Thus, the term "sample complexity" always refers to the expected sample complexity $\mathbb{E}[|\Omega|]$. This is equal to $nq \cdot p$ for each iteration. Theorem 2.1 given next provides a lower bound on the required value of p in each algorithm iteration.

Theorem 2.1. *Consider Algorithm 1 or 2. Let* $n_{mx} := \max(n, q)$ *and* $n_{mn} := \min(n, q)$*. Pick an* $\epsilon < 1$*. Assume that Assumption 1.1 holds and that, at each iteration t, entries of* $\boldsymbol{X}^\star$ *are observed independently with probability* p *satisfying* $nqp > C\kappa^4\mu^2 n_{mx} r^2 \log n_{mx}$*. Set* $\eta = 0.5/(p\sigma_{\max}^{*2})$ *and* $T = C\kappa^2 \log(1/\epsilon)$*. Then, with probability (w.p.) at least* $1 - 4T/n_{mn}^3$*,*

$$\mathrm{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^\star) \leq \epsilon \text{ and } \|\boldsymbol{X}^{(T)} - \boldsymbol{X}^\star\|_F \leq \epsilon \|\boldsymbol{X}^\star\|. \tag{4}$$

The total sample complexity is $T \cdot nqp \geq C\kappa^6\mu^2 n_{mx} r^2 \log n_{mx} \log(1/\epsilon)$*. The total per-node computation complexity is* $T \cdot \max(n, |\Omega|/\gamma) r^2 = C\kappa^2 \log(1/\epsilon) \cdot \max(n, |\Omega|/\gamma) r^2$ *and total per-node communication complexity is* $T \cdot nr = C\kappa^2 \log(1/\epsilon) \cdot nr$*.*

Remark 2.2. *We state Theorem 2.1 for one value of the step size* η *because this makes some of our proof arguments cleaner notation-wise. However, our proof will go through for any step size* $\eta = c_\eta/(p{\sigma_{\max}^*}^2)$ *for a constant* $c_\eta < c_1 < 1$*. We can actually show the following. If at each iteration,* $nq \cdot p > C\kappa^4\mu^2 n_{mx} r^2 \log n_{mx}$*., then with probability (w.p.) at least* $1 - 4t/n_{mn}^3$*, at iteration* $t \geq 0$*,*

$$\mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^\star) \leq \left(1 - \frac{cc_\eta}{\kappa^2}\right)^t \frac{c_0}{\kappa^2}$$

Proof. In our proofs, instead of using n_{mx}, n_{mn}, we just assume $n \leq q$ for simplicity. The proof is given in Sec. IV and V. Before this, we discuss this result. We describe the novel ideas used in our proof in Sec. III. □

Treating κ, μ as numerical constants, the above result says the following. As long as we observe order $qr^2 \log q \log(1/\epsilon)$ matrix entries, and we set the GD step size, η, and the total number of iterations, T, as stated, then, with high probability (w.h.p.), we can fill in the rest of the entries accurately: the normalized Frobenius norm of the error in this estimation is at most ϵ. Also, we can estimate the column span of $\boldsymbol{X}^\star$ with ϵ accuracy. The number of iterations needed is order $\kappa^2 \log(1/\epsilon)$. Recall from (2) that our cost function $f(\boldsymbol{U}, \boldsymbol{B}) := \|(\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{B})_\Omega\|_F^2$. Thus, $\mathbb{E}[f(\boldsymbol{U}, \boldsymbol{B})] = p\|\boldsymbol{X}^\star - \boldsymbol{U}\boldsymbol{B}\|_F^2$. To keep notation cleaner, we do not use a $1/p$ factor in the cost function. Consequently, it does not appear in the gradient expression either. This is why the step-size η contains a factor of $1/p$.

1) Parameter Setting: AltGDmin needs to set three parameters: the rank r, the GD step size η and the total number of iterations T. The rank can be set by computing the approximate rank of the initialization matrix $\boldsymbol{Y}$, while also ensuring that r is sufficiently smaller than the the number of samples per column, so that the LS step estimate of $\boldsymbol{b}_k$ can be accurately computed. As an example, we could use the heuristic introduced in [30]. This sets r as the smallest integer $\hat{r}$ for which the sum of squares of the first $\hat{r}$ singular values is more than 85% of the sum of squares of the first $\min_k m_k/10$ singular values with m_k being the number of observed samples for column k. We have $\mathbb{E}[m_k] = np$. The step size can be set as $\eta = c/(p{\sigma_{\max}^*}^2)$ with a $c < 1$. Practically, $\sigma_{\max}^*$ will not be known. It can be replaced by $\|\boldsymbol{Y}\|/p$ since $\mathbb{E}[\boldsymbol{Y}] = p\boldsymbol{X}^\star$. The total number of iterations T should be replaced by the commonly used stopping criterion for GD: use a very large value of T but exit the iterations loop if the norm of the gradient becomes small enough.

E. Discussion

In this discussion, we treat κ, μ and γ as numerical constants (as is commonly done in many past works, e.g., [4]), and we assume that = $\epsilon < \exp(-r)$ or equivalently $r < \log(1/\epsilon)$. The average (expected value of) the required sample complexity is $nqp \cdot T$ where T is the iteration complexity. Our guarantee given in Theorem 2.1 shows that the AltGDmin sample complexity is $\kappa^6\mu^2 n_{mx} r^2 \log n_{mx} \log(1/\epsilon)$. This is worse than that of FactGD by a factor of $\kappa^2 \log(1/\epsilon)$. The reason that AltGDmin needs a factor of $\log(1/\epsilon)$ in its sample complexity is because its proof uses sample-splitting. To our best knowledge, all guarantees for all solutions for LRMC or LRCS that factor $\boldsymbol{X}$ as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$, and in which one (or both) of the two alternating steps is a minimization step, need to use sample-splitting [4, 12, 13, 14, 20, 31] [1]. Both problems involve recovery from non-global measurements (no scalar measurement depends on the entire $\boldsymbol{X}^\star$).

The iteration complexity of AltGDmin is $T = \kappa^2 \log(1/\epsilon)$ while that of FactGD is $T = \kappa\mu r \log(1/\epsilon)$, which is r times larger. Per iteration, both have the same communication cost. Consequently, the total communication cost of AltGDmin is lower than that of FactGD by a factor of r. Consider the computation cost. Per iteration, the FactGD cost is lower than that of AltGDmin by a factor of r. Since the FactGD iteration complexity is larger by a factor of r, thus, the total computation cost of both algorithms is comparable. AltGDmin is private and FactGD can be made private with two data exchanges per iteration (as explained in [1]).

Consider AltMin. AltMin can be made private by implementing the minimization step for $\boldsymbol{U}$ using multiple GD iterations to solve the LS problem (details in [1]). The total computation cost of AltMin (private) is higher than that of AltGDmin by a factor of $(\log(1/\epsilon))/r$. Its total communication cost is also higher by a factor of $\log(1/\epsilon)$. Consider AltMin (Not-Private), this requires sharing all the observed data with the center first and then implementing LS using the closed form expression. Its communication cost is much higher, it is higher than that of AltGDmin by a factor of $|\Omega|/(\gamma nr)$. Its computation cost is comparable to that of AltGDmin though. The original sample complexity of AltMin proved in [4] is also much higher than

[1] Some of this cited work provides guarantees for LR phase retrieval (LRPR) which is a phaseless measurements' generalization of LRCS; and hence any LRPR solution automatically solves LRCS.

that of AltGDmin or FactGD (see Table II). However, the corollary that we prove for AltMin later in Sec VII shows that the AltMin sample complexity is $\kappa^4\mu^2 nr^2 \log n \log(1/\epsilon)$ and its iteration complexity is $\log(1/\epsilon)$. This is comparable to that of AltGDmin. The same is true for the comparison with Smooth AltMin as well [14]. Its original sample complexity is higher by a factor of r but our guarantee for it given in Sec VII reduces it to $C\kappa^2\mu^2 qr^2(\log(n/\epsilon)\log^2 n)$.

The ProjGD sample complexity is worse than that of AltGDmin by a factor of $\log n \log(1/\epsilon)$. Moreover, the ProjGD communication cost per iteration is much higher than that of AltGDmin and FactGD. The same is true for its computation cost making it one of the slowest and most communication inefficient.

Finally, notice that we have specified the computation cost per node and the communication cost per node. Communication cost per node is relevant because in practical distributed settings, the node data can be transmitted to the center in parallel using well known schemes such as frequency division multiplexing or code division multiple access.

The above discussion is summarized in Table II.

Comparisons for Centralized LRMC. In the centralized setting, there is no communication cost and no notion of privacy. The computation cost is as given in Table II with $\gamma = 1$. The sample cost is as given there too. In a centralized setting, clearly, all of AltMin, AltGDmin and FactGD have similar computation cost. If we also consider κ, μ dependence, then AltMin (AltMin-Not-Private from the table) is the fastest

A not-typical heterogenous federated setting. Our federated setting is a heterogeneous one because the data sub-matrix at node ℓ depends on $\boldsymbol{Y}_\ell = (\boldsymbol{X}_\ell^\star)_{\Omega_{(\ell)}} = (\boldsymbol{U}^\star \boldsymbol{B}^\star{}_\ell)_{\Omega_{(\ell)}}$ and $\boldsymbol{B}^\star{}_\ell$ is a different column sub-matrix of $\boldsymbol{B}^\star$. The unknowns in our case are $\boldsymbol{U}^\star$ and $\{\boldsymbol{B}^\star{}_\ell, \ell \in [\gamma]\}$. The unknown $\boldsymbol{U}^\star$ is common across all nodes. However, the unknown $\boldsymbol{B}^\star{}_\ell$ is different for each node. If we increase the total number of nodes γ, the number of unknowns in $\boldsymbol{B}^\star = [\boldsymbol{B}^\star{}_1, \boldsymbol{B}^\star{}_2, \dots, \boldsymbol{B}^\star{}_\gamma]$ increases. Our federated setting is different from the homogenous one, studied in [32, 33] and other works, which involves learning from data that is identically distributed at different nodes. In this case, increasing γ implies that the total amount of available data increases, while the size of the unknown quantity to be estimated remains fixed. It is also different from a typical heterogenous setting which involves estimating a common set of unknowns from data that is not identically distributed across nodes, e.g., the data variance (or other distributional parameters) are different across different nodes. In both these cases, one would expect improvement in algorithm convergence speed and noise robustness with increasing γ. In our case, we cannot make a clear claim since the number of unknowns also increases.

III. Proof Novelty

The overall structure of our proof is similar to that developed in [20] for providing a simple correctness proof for AltGDmin for the LR columnwise sensing problem. However there are important differences that we describe next. The discussion below treats κ, μ as numerical constants and assumes $n \le q$ so that $n_{mx} = q$.

1) The most important difference between LRCS and LRMC is that the LRMC proofs require incoherence of each updated estimate $\boldsymbol{U}$ and $\boldsymbol{B}$. When analyzing AltGDmin for LRCS in [13, 20], this was needed only for $\boldsymbol{B}$ because LRCS measurements are column-wise global (matrix $\boldsymbol{A}_k$ is dense and so each entry of $\boldsymbol{y}_k$ depends on the entire column $\boldsymbol{x}_k^\star$). Our proof of incoherence of each updated $\boldsymbol{U}$ needs a different approach than that used in all past work on LRMC. The details of this approach are in the proof of Lemma 4.6 and in its use in proving Theorem 2.1.
   The reason that a new approach is needed is because AltGDmin is a different algorithm. (i) ProjGD [6] does not use a factorized representation for $\boldsymbol{X}$ and hence the gradient w.r.t. $\boldsymbol{U}$ does not exist for it. (ii) We do not update $\boldsymbol{U}$ by LS and hence we cannot use the approach used for AltMin or Smooth AltMin [4, 14]. (iii) FactGD [10] uses row norm clipping at each iteration to ensure incoherence by construction. AltGDmin uses row norm clipping only for the initialization step and for the iterations. If we used row norm colipping for the iterations, it would not be possible to borrow the overall proof structure for bounding $\mathrm{SD}_F(\boldsymbol{U}^+, \boldsymbol{U}^\star)$ in terms of $\mathrm{SD}_F(\boldsymbol{U}, \boldsymbol{U}^\star)$ from [20].
2) We use SD_F as the subspace distance measure (instead of SD_2 that was used in the LRCS work [20] (whose proof approach we partly borrow). With our proof approach, use of SD_F allows us to show convergence under a sample complexity lower bound of order qr^2 per iteration. By using SD_2, we would need order qr^3.
   The reason for this is as follows. (i) The analysis of row norm clipping for the initialization step relies on the fact that this operation is a projection (in Frobenius norm) onto a convex set. To efficiently use this fact, we need to use SD_F to bound initialization error. With it, we show that $\mathrm{SD}_F(\boldsymbol{U}^{(0)}, \boldsymbol{U}^\star) \le \delta_0$ w.h.p. if $nqp \gtrsim nr^2/\delta_0^2$. If SD_2 was used, it would need an extra factor of r. (ii) At iteration t, we show that $\|\boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{X}^\star\|_F \le \epsilon_1 \mathrm{SD}_F(\boldsymbol{U}^\star, \boldsymbol{U})\sigma_{\max}^* \le \epsilon_1\sqrt{r}\mathrm{SD}_2(\boldsymbol{U}^\star, \boldsymbol{U})\sigma_{\max}^*$ w.h.p. if $nqp \gtrsim qr^2/\epsilon_1^2$. This bound is used to lower bound $\sigma_{\min}(\boldsymbol{B})$. To get the lower bound to be at least a constant, say 0.9, we need $\|\boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{X}^\star\|_F \le c\sigma_{\min}^*$ for a constant $c \le 0.8$. This can be guaranteed by setting $\epsilon_1 = c$ and requiring $\mathrm{SD}_F(\boldsymbol{U}^\star, \boldsymbol{U}) \le 1/\kappa$ for all iterations t including $t = 0$ (initialization). This requires setting $\delta_0 = 1/\kappa$. With $\epsilon_1 = c$ and $\delta_0 = 1/\kappa$, the sample complexity per iteration remains order qr^2 (treating κ as a numerical constant). If we use SD_2, we would need $\epsilon_1\sqrt{r}\mathrm{SD}_2(\boldsymbol{U}^\star, \boldsymbol{U}) \le c/\kappa$. If we set $\epsilon_1 = c$, this would require $\mathrm{SD}_2(\boldsymbol{U}^\star, \boldsymbol{U}) \le 1/\kappa\sqrt{r}$ for all iterations t including $t = 0$ (initialization). Since we can only bound $\mathrm{SD}_2 \le \mathrm{SD}_F$ (there is no factor of $\sqrt{r}$ in this upper bound), this would mean that we would now need $\delta_0 = 1/\kappa\sqrt{r}$. This then implies that the initialization step would require qr^3 samples. Alternatively, we could move the $\sqrt{r}$ factor into ϵ_1, but then each iteration would need qr^3 sample complexity.

3) We need to use the matrix Bernstein inequality [34] for bounding all of terms, instead of the concentration bounds used in [20] (sub-Gausian Hoeffding or sub-exponential Bernstein inequality followed by an epsilon-net argument).
4) Unlike LRCS, LRMC measurements are both row-wise and column-wise local. Consequently, it is not possible to get a tight column-wise bound (bound on $\|\boldsymbol{b}_k - \boldsymbol{U}^\intercal \boldsymbol{x}^\star_k\|$, and hence on $\|\boldsymbol{x}_k - \boldsymbol{x}^\star_k\|$, for each k) under the desired sample complexity. We can only bound $\|\boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{X}^\star\|_F$ and $\|\boldsymbol{X} - \boldsymbol{X}^\star\|_F$. The bound on $\|\boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{X}^\star\|_F$ that we prove needs only roughly qr^2 samples and is better than the one proved in [4] which needs more samples.
5) Also, our initialization guarantee is better than that of AltMin and Smooth AltMin [4, 14] by a factor of r^3 and r respectively, and comparable to that of FactGD. Using it and using our $\|\boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{X}^\star\|_F$ bound, we are able to provide significantly improved, roughly qr^2 sample complexity bound for AltMin. Using our initialization guarantee we can also provide a similar sample complexity guarantee for Smoothed AltMin [4, 14].

IV. Proving Theorem 2.1

Some readers find it easier to directly read the proof than the outline. We thus provide an outline in Appendix A.

In the proof below, we assume $n \le q$ for simplicity. This allows us to bound $1/q$ by $1/n$ and $1/\sqrt{nq} < 1/n$, etc, at various places.

A. Definitions and Expressions

Let $\boldsymbol{U} \equiv \boldsymbol{U}^{(t-1)}$, $\boldsymbol{B} \equiv \boldsymbol{B}^{(t)}$, and $\boldsymbol{X} \equiv \boldsymbol{X}^{(t)} = \boldsymbol{U}^{(t-1)}\boldsymbol{B}^{(t)}$. Also let $\boldsymbol{U}^+ \equiv \boldsymbol{U}^{(t)}$, $\boldsymbol{B}^+ \equiv \boldsymbol{B}^{(t+1)}$. Define

$$\boldsymbol{G} := \boldsymbol{U}^\intercal \boldsymbol{X}^\star = [\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_q] \text{ with } \boldsymbol{g}_k := \boldsymbol{U}^\intercal \boldsymbol{U}^\star \boldsymbol{b}^\star_k,$$

$$\boldsymbol{P}_{*,\perp} := \boldsymbol{I} - \boldsymbol{U}^\star \boldsymbol{U}^{\star\intercal},$$

and

$$\delta^{(t)} := \mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^*) = \mathrm{SD}_F(\boldsymbol{U}^+, \boldsymbol{U}^*) = \|\boldsymbol{P}_{*,\perp}\boldsymbol{U}^+\|_F$$

Thus, $\delta^{(t-1)} = \mathrm{SD}_F(\boldsymbol{U}, \boldsymbol{U}^\star)$, $\delta^{(t)} = \mathrm{SD}_F(\boldsymbol{U}^+, \boldsymbol{U}^\star) = \|\boldsymbol{P}_{*,\perp}\boldsymbol{U}^+\|_F$, and $\boldsymbol{P}_{*,\perp}\boldsymbol{X}^\star = \boldsymbol{0}$.

Let $\mu_u := 20\kappa^2\mu$. All our proofs use c, C to denote different numerical constants in each use. The numerical values in our intermediate steps are often loose bounds to make the analysis simpler.

Recall that $\boldsymbol{U}_k := \boldsymbol{S}_k\boldsymbol{U}$. Let $\boldsymbol{U}^\star{}_k := \boldsymbol{S}_k\boldsymbol{U}^\star$. Since

$$\boldsymbol{S}_k^\intercal \boldsymbol{S}_k = \boldsymbol{S}_k\boldsymbol{S}_k = \boldsymbol{S}_k = \mathrm{diag}(\xi_{jk}, j \in [n])$$

$$\boldsymbol{U}_k = \boldsymbol{S}_k\boldsymbol{U} = \sum_{j=1}^n \xi_{jk}\boldsymbol{e}_j\boldsymbol{u}^{j\intercal},$$

$$\boldsymbol{U}^\star{}_k := \boldsymbol{S}_k\boldsymbol{U}^\star = \sum_{j=1}^n \xi_{jk}\boldsymbol{e}_j\boldsymbol{u}^{*j\intercal},$$

and

$$\boldsymbol{U}_k^\intercal \boldsymbol{U}^\star{}_k = \boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U}^\star = \sum_{j=1}^n \xi_{jk}\boldsymbol{u}^j\boldsymbol{u}^{*j\intercal}$$

Using this,

$$\boldsymbol{b}_k = \boldsymbol{U}_k^\dagger \boldsymbol{y}_k = (\boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U})^{-1}\boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U}^\star \boldsymbol{b}^\star_k$$

Thus, $\boldsymbol{b}_k - \boldsymbol{g}_k = \boldsymbol{b}_k - \boldsymbol{U}^\intercal \boldsymbol{U}^\star \boldsymbol{b}^\star_k$ simplifies to

$$\boldsymbol{b}_k - \boldsymbol{g}_k = (\boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U})^{-1}[\boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U}^\star \boldsymbol{b}^\star_k - (\boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U})\boldsymbol{U}^\intercal \boldsymbol{U}^\star \boldsymbol{b}^\star_k] = (\underbrace{\boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U}}_{\boldsymbol{F}_k})^{-1}\underbrace{\boldsymbol{U}^\intercal \boldsymbol{S}_k[\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\intercal]\boldsymbol{U}^\star \boldsymbol{b}^\star_k}_{\boldsymbol{d}_k} \tag{5}$$

Defining

$$\boldsymbol{F} := \mathrm{blockdiag}(\boldsymbol{F}_k, k \in [q]) \quad \text{with} \quad \boldsymbol{F}_k := \boldsymbol{U}^\intercal \boldsymbol{S}_k\boldsymbol{U},$$

$$\boldsymbol{D} := [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots \boldsymbol{d}_q] = \sum_k \boldsymbol{d}_k\boldsymbol{e}_k^\intercal, \quad \text{with } \boldsymbol{d}_k := \boldsymbol{U}^\intercal \boldsymbol{S}_k[\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\intercal]\boldsymbol{U}^\star \boldsymbol{b}^\star_k,$$

we have

$$vec(\boldsymbol{B} - \boldsymbol{G}) = \boldsymbol{F}^{-1}vec(\boldsymbol{D}),$$

In the above $\boldsymbol{F} \in \mathbb{R}^{rq \times rq}$ and $\boldsymbol{D} \in \mathbb{R}^{r \times q}$. Using this,

$$\|\boldsymbol{B} - \boldsymbol{G}\|_F = \|vec(\boldsymbol{B} - \boldsymbol{G})\| = \|\boldsymbol{F}^{-1} vec(\boldsymbol{D})\| \leq \|\boldsymbol{F}^{-1}\| \; \|\boldsymbol{D}\|_F \\ \leq \frac{\sqrt{r}\|\boldsymbol{D}\|}{\min_k \sigma_{\min}(\boldsymbol{F}_k)}.$$

Since $\boldsymbol{S}_k$ is a diagonal matrix with each diagonal entry being a $Bernoulli(p)$ r.v., thus,

$$\mathbb{E}[\boldsymbol{S}_k] = p\boldsymbol{I}$$

and so,

$$\mathbb{E}[\boldsymbol{d}_k] = p\boldsymbol{U}^\intercal[\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\intercal]\boldsymbol{U}^\star \boldsymbol{b}_k^\star = \boldsymbol{0}, \text{ and } \mathbb{E}[\boldsymbol{D}] = \boldsymbol{0}$$

Finally, define

$$\text{GradU} := \nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B}) = 2(\boldsymbol{X}_\Omega - \boldsymbol{Y})\boldsymbol{B}^\intercal = 2\sum_{k=1}^{q} \boldsymbol{S}_k(\boldsymbol{x}_k - \boldsymbol{x}_k^\star)\boldsymbol{b}_k^\intercal = 2\sum_{k=1}^{q}\sum_{j=1}^{n} \xi_{jk}\boldsymbol{e}_j(\boldsymbol{x}_{jk} - \boldsymbol{x}_{jk}^\star)\boldsymbol{b}_k^\intercal \tag{6}$$

where $\boldsymbol{x}_{jk} = \boldsymbol{u}^{j\intercal}\boldsymbol{b}_k$ and $\boldsymbol{x}_{jk}^\star = \boldsymbol{u}^{*j\intercal}\boldsymbol{b}_k^\star$. Also,

$$\text{GradU}^j := \boldsymbol{e}_j^\intercal \nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B}) = 2\sum_{k=1}^{q} \xi_{jk}(\boldsymbol{x}_{jk} - \boldsymbol{x}_{jk}^\star)\boldsymbol{b}_k^\intercal,$$

Observe that

$$\mathbb{E}[\text{GradU}] = p(\boldsymbol{X} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal \text{ and } \mathbb{E}[\text{GradU}^j] = p\boldsymbol{e}_j^\intercal(\boldsymbol{X} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal$$

B. Lemmas for proving Theorem 2.1

All lemmas below assume Assumption 1.1 (singular vectors' incoherence) holds. Also, everywhere $\epsilon < 1$.

Lemma 4.1 (Initialization)**.** *Pick a $\delta \leq 0.2$. Assume $p \geq C\kappa^2 r^2 \mu \log q/(n\delta^2)$. Then, w.p. at least $1 - n^{-10}$.*

1) $\text{SD}_F(\boldsymbol{U}^{(0)}, \boldsymbol{U}^*) \leq \delta$.
2) *$\boldsymbol{U}^{(0)}$ is 1.5μ row-incoherent, i.e., $\|\boldsymbol{u}^{j(0)}\| \leq 1.5\mu\sqrt{r/n}$ for all $j \in [n]$.*

Lemma 4.2 (LS step analysis: error bound for $\boldsymbol{B}$)**.** *Let $\mu_u := 20\kappa^2\mu$. Recall that $\delta^{(t-1)} = \text{SD}_F(\boldsymbol{U}, \boldsymbol{U}^\star)$. Assume that $\|\boldsymbol{u}^j\| \leq \mu_u\sqrt{r/n}$. Then, w.p. at least $1 - \exp(\log q - c\frac{\epsilon^2 pn}{\mu_u^2 r^2})$,*

$$\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \epsilon\delta^{(t-1)}\sigma_{\max}^*$$

Lemma 4.3 (Implications of error bound for $\boldsymbol{B}$)**.** *Assume $\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \delta^{(t-1)}\sigma_{\max}^*$. Then,*

1) $\|\boldsymbol{X} - \boldsymbol{X}^\star\|_F \leq 2\delta^{(t-1)}\sigma_{\max}^*$.
2) $\sigma_{\max}(\boldsymbol{B}) \leq (1 + \delta^{(t-1)})\sigma_{\max}^*$ *and* $\sigma_{\min}(\boldsymbol{B}) \geq \sqrt{1 - \delta^{(t-1)2}}\sigma_{\min}^* - \delta^{(t-1)}\sigma_{\max}^*$.
3) *Thus, if $\delta^{(t-1)} \leq c/\kappa$, then $\sigma_{\min}(\boldsymbol{B}) \geq 0.9\sigma_{\min}^*$ and $\sigma_{\max}(\boldsymbol{B}) \leq 1.1\sigma_{\max}^*$.*

Lemma 4.4 (Incoherence of $\boldsymbol{B}$)**.** *Let $\mu_u := 20\kappa^2\mu$. Assume that $\|\boldsymbol{u}^j\| \leq \mu_u\sqrt{r/n}$. Then, w.p. greater than $1 - \exp(\log q - c\frac{np}{\mu_u^2 r})$,*

$$\|\boldsymbol{b}_k\| \leq 1.1\sigma_{\max}^*\mu\sqrt{r/q} \;\; \textit{for all } k \in [q].$$

Lemma 4.5 (Gradient expression and bounds)**.** *Let $\mu_u := 20\kappa^2\mu$. Recall that $\delta^{(t-1)} = \text{SD}_F(\boldsymbol{U}, \boldsymbol{U}^\star)$. Assume that $\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \delta^{(t-1)}\sigma_{\max}^*$ with $\delta^{(t-1)} \leq c/\kappa$, $\|\boldsymbol{u}^j\| \leq \mu_u\sqrt{r/n}$, $\|\boldsymbol{b}_k\| \leq \sigma_{\max}^*\mu\sqrt{r/q}$. Then,*

1) $\|\mathbb{E}[\text{GradU}]\| \leq \|\mathbb{E}[\text{GradU}]\|_F \leq 2.5p\delta^{(t-1)}\sigma_{\max}^{*2}$.
2) $\|\text{GradU} - \mathbb{E}[\text{GradU}]\| \leq \epsilon p\delta^{(t-1)}\sigma_{\min}^{*2}$, *w.p. at least* $1 - \exp(\log q - c\frac{\epsilon^2 pn}{\max(\kappa^4\mu^2, \kappa^2\mu_u\mu)r})$
3) $\|\text{GradU} - \mathbb{E}[\text{GradU}]\|_F \leq \epsilon p\delta^{(t-1)}\sigma_{\min}^{*2}$, *w.p. at least* $1 - \exp(\log q - c\frac{\epsilon^2 pn}{\max(\kappa^4\mu^2, \kappa^2\mu_u\mu)r^2})$

Lemma 4.6 (Incoherence of $\boldsymbol{U}$)**.** *Recall that $\delta^{(t-1)} = \text{SD}_F(\boldsymbol{U}, \boldsymbol{U}^\star)$. Assume that $\|\boldsymbol{b}_k\| \leq 1.1\sigma_{\max}^*\mu\sqrt{r/q}$. Then, w.p. greater than $1 - \exp(\log q - c\epsilon^2 pn/\kappa^4\mu^2 r^2)$*

1) $\|\text{Grad}\boldsymbol{u}^j - \mathbb{E}[\text{Grad}\boldsymbol{u}^j]\| \leq \epsilon p \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)\sigma_{\min}^{*2}$
2) *Further, if $\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \delta^{(t-1)}\sigma_{\max}^*$ with $\delta^{(t-1)} < c/\kappa$, and GD step size $\eta \leq 0.5/(p{\sigma_{\max}^*}^2)$, then*

$$\|{\boldsymbol{u}^j}^+\| \leq (1 - 0.15/\kappa^2)\|\boldsymbol{u}^j\| + 2\|\boldsymbol{u}^{*j}\|.$$

Proof. All these lemmas are proved in Sec V. □

C. Proof of Theorem 2.1

Proof of Theorem 2.1. Theorem 2.1 follows from claim (i) of Claim 4.7 stated below along with using the following:

- To ensure $\delta^{(T)} \le \epsilon$, we need $(1-c/\kappa^2)^T \cdot (c/\kappa^2) \le \epsilon$ or that $T = C\kappa^2 \log(1/\epsilon)$.
- Since we use sample splitting, the total number of samples needed for all T iterations is $2T \cdot npq \ge C\kappa^4\mu^2 qr^2 \log q \cdot \kappa^2 \log(1/\epsilon)$.

□

Claim 4.7. *Assume everything stated in Theorem 2.1 and $np \ge C\kappa^4\mu^2 r^2 \log q$. For all times $\tau \ge 0$, the following hold w.p. at least $1-\tau/n^3$: (i) $\delta^{(\tau)} \le (1-c/\kappa^2)^\tau \cdot (c/\kappa^2)$; and (ii) $\|\boldsymbol{u}^{j^{(\tau)}}\| \le (1-\frac{0.15}{\kappa^2})^\tau 1.5\mu\sqrt{r/n} + \left(\sum_{\tau'=0}^{\tau-1}(1-\frac{0.15}{\kappa^2})^{\tau'}\right) 2\|\boldsymbol{u}^{*j}\|$*

We prove this claim next using an induction argument and the lemmas from Sec. IV-B.

Base case: Lemma 4.1 shows that $\delta^{(0)} \le \delta = 0.1/\kappa^2$ and $\|\boldsymbol{u}^{j^{(0)}}\| \le 1.5\mu\sqrt{r/n}$. This proves (i) and (ii) for $\tau = 0$.

Induction assumption: Assume that the claim holds for $\tau = t-1$.

Induction step: Consider $\tau = t$. Recall from Algorithm that $\tilde{\boldsymbol{U}}^+ = \boldsymbol{U} - \eta\text{GradU}$, and $\boldsymbol{U}^+ = \tilde{\boldsymbol{U}}^+ \boldsymbol{R}^{+-1}$ where $\tilde{\boldsymbol{U}}^+ \overset{\text{QR}}{=} \boldsymbol{U}^+\boldsymbol{R}^+$. Recall from Sec. IV-A that $\boldsymbol{P}_{*,\perp}\boldsymbol{X}^\star = \boldsymbol{0}$, $\mathbb{E}[\text{GradU}] = p(\boldsymbol{U}\boldsymbol{B} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal$, $\|\boldsymbol{P}_{*,\perp}\boldsymbol{U}\|_F = \delta^{(t-1)}$. Also, using Weyl's inequality, $\sigma_{\min}(\boldsymbol{R}^+) = \sigma_{\min}(\tilde{\boldsymbol{U}}^+) \ge \sigma_{\min}(\boldsymbol{U}) - \eta\|\text{GradU}\| = 1 - \eta\|\text{GradU}\|$. Using these,

$$\begin{aligned}
\delta^{(t)} &= \text{SD}_F(\boldsymbol{U}^+, \boldsymbol{U}^\star) = \|\boldsymbol{P}_{*,\perp}\boldsymbol{U}^+\|_F \\
&\le \|\boldsymbol{P}_{*,\perp}\tilde{\boldsymbol{U}}^+\|_F \cdot \|(\boldsymbol{R}^+)^{-1}\| = \|\boldsymbol{P}_{*,\perp}\tilde{\boldsymbol{U}}^+\|_F / \sigma_{\min}(\tilde{\boldsymbol{U}}^+) \\
&\le \frac{\|\boldsymbol{P}_{*,\perp}(\boldsymbol{U} - \eta\mathbb{E}[\text{GradU}] + \eta\mathbb{E}[\text{GradU}] - \eta\text{GradU})\|_F}{(1-\eta\|\text{GradU}\|)} \\
&\le \frac{\|\boldsymbol{P}_{*,\perp}(\boldsymbol{U} - \eta p(\boldsymbol{U}\boldsymbol{B} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal)\|_F + \eta\|\mathbb{E}[\text{GradU}] - \text{GradU})\|_F}{(1-\eta\|\mathbb{E}[\text{GradU}]\| - \eta\|\mathbb{E}[\text{GradU}] - \text{GradU}\|)} \\
&\le \frac{\delta^{(t-1)} \cdot \|\boldsymbol{I} - \eta p\boldsymbol{B}\boldsymbol{B}^\intercal\| + \eta\|\mathbb{E}[\text{GradU}] - \text{GradU})\|_F}{(1-\eta\|\mathbb{E}[\text{GradU}]\| - \eta\|\mathbb{E}[\text{GradU}] - \text{GradU}\|)}.
\end{aligned} \tag{7}$$

By the induction assumption, $\delta^{(t-1)} \le (1-c/\kappa^2)^{t-1} \cdot (c/\kappa^2) \le (c/\kappa^2)$ and

$$\begin{aligned}
&\|\boldsymbol{u}^j\| = \|\boldsymbol{u}^{j^{(t-1)}}\| \\
&\le (1-\frac{0.15}{\kappa^2})^{t-1}\|\boldsymbol{u}^{j^{(0)}}\| + [1 + (1-\frac{0.15}{\kappa^2}) + \cdots + (1-\frac{0.15}{\kappa^2})^{t-2}]2\|\boldsymbol{u}^{*j}\| \\
&\le \|\boldsymbol{u}^{j^{(0)}}\| + \frac{\kappa^2}{0.15}2\|\boldsymbol{u}^{*j}\| \le (1.5\mu + 14\kappa^2\mu)\sqrt{r/n} \le \mu_u\sqrt{r/n}
\end{aligned} \tag{8}$$

The last inequality above used the infinite geometric series bound. This shows that $\boldsymbol{U}$ is μ_u-row-incoherent.

Using (8) and $\delta^{(t-1)} \le (c/\kappa^2)$, Lemmas 4.2 and 4.4 hold, i.e. the bound on $\|\boldsymbol{B} - \boldsymbol{G}\|_F$ and $\|\boldsymbol{b}_k\|$ holds. This then implies that all claims of Lemma 4.3 hold too. This then implies that Lemmas 4.5 and 4.6 hold. All the lemmas hold with probability at least $1 - 1/n^3$ if p is such that all the probabilities stated in all the lemmas are lower bounded by $1 - 0.1/n^3$. Using $\mu_u = 8\kappa^2\mu$, clearly, this is true if

$$np \ge C\kappa^4\mu^2 r^2 \log q$$

By Lemmas 4.2 and 4.3, $\sigma_{\min}(\boldsymbol{B}) \ge 0.9\sigma^*_{\min}$ and $\|\boldsymbol{B}\| \le 1.1\sigma^*_{\max}$. Using these, if $\eta \le 0.5/(p\sigma^{*2}_{\max})$, then $\boldsymbol{I} - \eta p\boldsymbol{B}\boldsymbol{B}^\intercal$ is positive semi-definite (psd) and hence $\|\boldsymbol{I} - \eta p\boldsymbol{B}\boldsymbol{B}^\intercal\| = \lambda_{\max}(\boldsymbol{I} - \eta p\boldsymbol{B}\boldsymbol{B}^\intercal) \le 1 - 0.8\eta p\sigma^{*2}_{\min}$. Using Lemma 4.4, $\boldsymbol{B}$ is 1.1μ-column-incoherent. Using Lemma 4.5, $\|\mathbb{E}[\text{GradU}]\| \le 2.5p\delta^{(t-1)}\sigma^{*2}_{\max}$ and $\|\text{GradU} - \mathbb{E}[\text{GradU}]\|_F \le \epsilon p\delta^{(t-1)}\sigma^{*2}_{\min}$. Substituting these into (7) with $\epsilon = 0.01$,

$$\begin{aligned}
\delta^{(t)} &\le \frac{\delta^{(t-1)}\left(1 - 0.8\eta p\sigma^{*2}_{\min} + 0.01\eta p\sigma^{*2}_{\min}\right)}{1 - \delta^{(t-1)}\left(2.55\eta p\sigma^{*2}_{\max}\right)}. \\
&\le \delta^{(t-1)}(1 - 0.79\eta p\sigma^{*2}_{\min})(1 + 2\delta^{(t-1)}(2.55\eta p\sigma^{*2}_{\max})) \\
&\le \delta^{(t-1)}(1 - (0.79 - \delta^{(t-1)}5.1\kappa^2)\eta p\sigma^{*2}_{\min}) \le \delta^{(t-1)}(1 - \eta p\sigma^{*2}_{\min}(0.79 - 0.051)) \le \delta^{(t-1)}(1 - 0.3\eta p\sigma^{*2}_{\min})
\end{aligned}$$

In the above, for the denominator term we used $1/(1-x) \le 1 + 2x$ for $x < 0.5$ and $\delta^{(t-1)} \le 0.01/\kappa^2$. Setting $\eta = 0.5/(p\sigma^{*2}_{\max})$ in the final expression given above, we can conclude that

$$\delta^{(t)} \le (1 - 0.15/\kappa^2)\delta^{(t-1)} \le (1 - 0.15/\kappa^2)^t \cdot (0.1/\kappa^2).$$

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2025.3563450

Thus claim (i) holds for $\tau = t$. Next we prove claim (ii) for $\tau = t$. By Lemma 4.6 and the induction assumption,

$$\begin{aligned}\|\boldsymbol{u}^{j(t)}\| &= \|\boldsymbol{u}^{j+}\| \leq (1-0.15/\kappa^2)\|\boldsymbol{u}^j\| + 2\|\boldsymbol{u}^{*j}\| \\ &\leq (1-0.15/\kappa^2)^t 1.5\mu\sqrt{r/n} + (1-0.15/\kappa^2)[1+(1-\frac{0.15}{\kappa^2}) + \cdots + (1-\frac{0.15}{\kappa^2})^{t-2}]2\|\boldsymbol{u}^{*j}\| + 2\|\boldsymbol{u}^{*j}\| \\ &= (1-0.15/\kappa^2)^t 1.5\mu\sqrt{r/n} + [1+(1-\frac{0.15}{\kappa^2}) + \cdots + (1-\frac{0.15}{\kappa^2})^{t-1}]2\|\boldsymbol{u}^{*j}\|\end{aligned}$$

V. Proofs of the Lemmas

All the proofs below use the matrix Bernstein inequality [34, 35].

Proposition 5.1 (Matrix Bernstein). *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots \boldsymbol{X}_m$ be independent, zero-mean, $d_1 \times d_2$ matrices with $\|\boldsymbol{X}_i\| \leq L$ for all $i = 1, 2, \ldots m$. Define the "variance parameter" of the sum*

$$\sigma^2 := \max\left(\|\sum_i \mathbb{E}[\boldsymbol{X}_i\boldsymbol{X}_i^\top]\|, \|\sum_i \mathbb{E}[\boldsymbol{X}_i^\top\boldsymbol{X}_i]\|\right).$$

$$\begin{aligned}\Pr\left(\|\sum_{i=1}^m \boldsymbol{X}_i\| \geq t\right) &\leq (d_1+d_2)\exp\left(-c\frac{t^2}{\sigma^2 + Lt/3}\right) \\ &\leq 2\exp\left(\log\max(d_1,d_2) - c\min\left(\frac{t^2}{\sigma^2}, \frac{t}{L}\right)\right).\end{aligned}$$

Consider matrices $\boldsymbol{Z}_i$ that are not zero mean but otherwise satisfy everything above. We would then apply the above result with $\boldsymbol{X}_i = \boldsymbol{Z}_i - \mathbb{E}[\boldsymbol{Z}_i]$.

A. Proof of Lemma 4.1

By Lemmas 2.5 and 2.6 of [36], for two $n \times r$ matrices with orthonormal columns, $\boldsymbol{U}_1, \boldsymbol{U}_2$,

$$\mathrm{SD}_F(\boldsymbol{U}_1, \boldsymbol{U}_2) \leq \min_{\boldsymbol{Q}\in\mathbb{R}^{r\times r}, \boldsymbol{Q}^\intercal\boldsymbol{Q}=\boldsymbol{Q}\boldsymbol{Q}^\intercal=\boldsymbol{I}} \|\boldsymbol{U}_1 - \boldsymbol{U}_2\boldsymbol{Q}\|_F \leq \sqrt{2}\mathrm{SD}_F(\boldsymbol{U}_1, \boldsymbol{U}_2) \tag{9}$$

and a similar bound holds for SD_2 as well. We use this fact frequently below.

Recall from the Algorithm that $\boldsymbol{Y} \overset{\mathrm{SVD}}{=} \boldsymbol{U}^{(00)}\Sigma^{(00)}\boldsymbol{V}^{(00)}$, and $\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) \overset{\mathrm{QR}}{=} \boldsymbol{U}^{(0)}\boldsymbol{R}^{(0)}$. By Theorem 3.22 of [36] along with using (9) and $\mathrm{SD}_2(\boldsymbol{U}_1, \boldsymbol{U}_2) \leq \sqrt{r}\mathrm{SD}_F(\boldsymbol{U}_1, \boldsymbol{U}_2)$, we have: with probability at least $1 - Cq^{-10}$,

$$\mathrm{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^\star) \leq C\sqrt{\frac{\kappa^2\mu r^2\log q}{np}}$$

Let $\boldsymbol{Q}_{*,00} = \arg\min_{\boldsymbol{Q}\in\mathbb{R}^{r\times r}, \boldsymbol{Q}^\intercal\boldsymbol{Q}=\boldsymbol{Q}\boldsymbol{Q}^\intercal=\boldsymbol{I}} \|\boldsymbol{U}^{(00)} - \boldsymbol{U}^\star\boldsymbol{Q}\|_F$. By (9),

$$\|\boldsymbol{U}^{(00)} - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F \leq \sqrt{2}\mathrm{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^\star)$$

Recall that the set $\mathcal{U}$ is defined in (3). Next we use the above two bounds and the fact that $\mathcal{U}$ is a convex set[2] to bound $\|\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F$ as follows.

$$\|\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F = \|\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \Pi_{\mathcal{U}}(\boldsymbol{U}^\star\boldsymbol{Q}_{*,00})\|_F \leq \|\boldsymbol{U}^{(00)} - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F \leq \sqrt{2}\mathrm{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^\star) \leq C\sqrt{\frac{\kappa^2\mu r^2\log q}{np}}$$

with probability at least $1 - Cq^{-10}$. The first equality above uses the fact that, for any unitary $\boldsymbol{Q}$, $\boldsymbol{U}^\star\boldsymbol{Q}$ belongs to $\mathcal{U}$. This holds since $\|(\boldsymbol{U}^\star\boldsymbol{Q})^j\| = \|\boldsymbol{e}_j^\intercal\boldsymbol{U}^\star\boldsymbol{Q}\| \leq \|\boldsymbol{u}^{*j}\| \leq \mu\sqrt{r/n}$ (using $\|\boldsymbol{Q}\| = 1$ and the incoherence Assumption 1.1). The second one uses the facts that projection onto $\mathcal{U}$, which is a convex set, is non-expansive [37, eq. (9),(10)], [38, eq. (1.5)]). The projection $\Pi_{\mathcal{U}}(\boldsymbol{M}) := \min_{\boldsymbol{U}\in\mathcal{U}} \|\boldsymbol{U} - \boldsymbol{M}\|_F$ is non-expansive means that $\|\Pi_{\mathcal{U}}(\boldsymbol{M}) - \Pi_{\mathcal{U}}(\boldsymbol{M}_2)\|_F \leq \|\boldsymbol{M} - \boldsymbol{M}_2\|_F$.

[2] To see that $\mathcal{U}$ is a convex set, let matrices $\boldsymbol{M}_1, \boldsymbol{M}_2 \in \mathcal{U}$. Let $\boldsymbol{m}_1^j$ and $\boldsymbol{m}_2^j$ denote the rows of $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$, respectively. For any $0 < \theta < 1$, $\|\theta\boldsymbol{m}_1^j + (1-\theta)\boldsymbol{m}_2^j\| \leq \theta\|\boldsymbol{m}_1^j\| + (1-\theta)\|\boldsymbol{m}_2^j\| \leq \theta\mu\sqrt{r/n} + (1-\theta)\mu\sqrt{r/n} = \mu\sqrt{r/n}$.

Finally, we use the above to bound $\mathrm{SD}_F(\boldsymbol{U}^{(0)}, \boldsymbol{U}^\star) = \|\boldsymbol{P}_{*,\perp}\boldsymbol{U}^{(0)}\|_F = \|\boldsymbol{P}_{*,\perp}\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)})\boldsymbol{R}^{(0)-1}\|_F$. Using $\boldsymbol{P}_{*,\perp}\boldsymbol{U}^\star\boldsymbol{Q}_{*,00} = \boldsymbol{0}$, and $x/(1-x) < 1.5x$ for any $0 \le x \le 0.3$, we have: if $C\sqrt{\frac{\kappa^2\mu r^2 \log q}{np}} < 0.3$, then

$$\begin{aligned}
\mathrm{SD}_F(\boldsymbol{U}^{(0)}, \boldsymbol{U}^\star) &= \|\boldsymbol{P}_{*,\perp}\boldsymbol{U}^{(0)}\|_F \le \|\boldsymbol{P}_{*,\perp}\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)})\|_F \|\boldsymbol{R}^{(0)-1}\| \\
&\le \frac{\|\boldsymbol{P}_{*,\perp}\boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F + \|\boldsymbol{P}_{*,\perp}(\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00})\|_F}{\sigma_{\min}(\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)})} \\
&\le \frac{\|\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F}{1 - \|\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F} \\
&\le \frac{C\sqrt{\frac{\kappa^2\mu r^2 \log q}{np}}}{1 - C\sqrt{\frac{\kappa^2\mu r^2 \log q}{np}}} \\
&\le 1.5C\sqrt{\frac{\kappa^2\mu r^2 \log q}{np}}
\end{aligned}$$

The denominator bound follows by Weyl's inequality, $\sigma_{\min}(\boldsymbol{U}^\star\boldsymbol{Q}_{*,00}) = 1$, and $x/(1-x) < 1.5x$ for $0 < x < 0.3$. Thus, for a $\delta < 0.3$, w.p. at least $1 - q^{-10}$, $\mathrm{SD}_F(\boldsymbol{U}^{(0)}, \boldsymbol{U}^\star) \le \delta$ if $p \ge C\kappa^2\mu r^2 \log q / n\delta^2$.

1) Proof of μ-incoherence: We have $\boldsymbol{u}^{j\,(0)} = (\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}))^j\boldsymbol{R}^{(0)-1}$. Thus, using the denominator bound from above,

$$\|\boldsymbol{u}^{j\,(0)}\| \le \|(\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}))^j\|\|\boldsymbol{R}^{(0)-1}\| = \frac{\|(\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}))^j\|}{\sigma_{\min}(\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)})} \le \frac{\mu\sqrt{r/n}}{1 - C\sqrt{\frac{\kappa^2\mu r^2 \log q}{np}}}$$

Our assumed bound on p implies that the denominator is at least $1 - \delta$ for a $\delta < 0.3$. This then implies the above is upper bounded by $1.5\mu\sqrt{r/n}$.

B. Proof of Lemma 4.2

Let $\delta = \delta^{(t-1)}$. Recall from Sec. IV-A that $vec(\boldsymbol{B} - G) = \boldsymbol{F}^{-1}vec(\boldsymbol{D})$ with $\boldsymbol{F}, \boldsymbol{D}$ defined there. Here $\boldsymbol{F} \in \mathbb{R}^{rq \times rq}$ and $\boldsymbol{D} \in \mathbb{R}^{r \times q}$. Thus,

$$\|\boldsymbol{B} - \boldsymbol{G}\|_F = \|vec(\boldsymbol{B} - \boldsymbol{G})\| = \|\boldsymbol{F}^{-1}vec(\boldsymbol{D})\| \le \|\boldsymbol{F}^{-1}\| \cdot \|vec(\boldsymbol{D})\| = \|\boldsymbol{F}^{-1}\| \cdot \|\boldsymbol{D}\|_F \le \|\boldsymbol{F}^{-1}\| \cdot \sqrt{r}\|\boldsymbol{D}\|, \tag{10}$$

We first bound $\|\boldsymbol{D}\| = \|\boldsymbol{D} - \mathbb{E}[\boldsymbol{D}]\|$ using the matrix Bernstein inequality [34]. We summarize it above in Proposition 5.1. Let

$$\boldsymbol{M} := [(\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\intercal)\boldsymbol{U}^\star]$$

Recall from Sec. IV-A that

$$\boldsymbol{D} = \sum_k \boldsymbol{d}_k\boldsymbol{e}_k^\intercal = \boldsymbol{U}^\intercal\boldsymbol{S}_k\boldsymbol{M}\boldsymbol{b}_k^\star\boldsymbol{e}_k^\intercal = \sum_k\sum_j \underbrace{\xi_{jk}\boldsymbol{u}^j\boldsymbol{M}^{j\,\intercal}\boldsymbol{b}_k^\star\boldsymbol{e}_k^\intercal}_{\boldsymbol{Z}_{jk}} \text{ and } \mathbb{E}[\boldsymbol{D}] = \boldsymbol{0} \tag{11}$$

Using $n \le q$,

$$L = \max_{jk}\|\xi_{jk}\boldsymbol{u}_j\boldsymbol{M}^{j\,\intercal}\boldsymbol{b}_k^\star\boldsymbol{e}_k^\intercal\| \le \max_{jk}\|\boldsymbol{M}^j\|\|\boldsymbol{u}_j\|\|\boldsymbol{b}_k^\star\| \le \delta\sigma^*_{\max}\mu\mu_u r/n$$

where we used $\|\boldsymbol{M}^j\| \le \|\boldsymbol{M}\| \le \delta$. Also, using $\mathbb{E}[\xi_{jk}] = \mathbb{E}[\xi_{jk}^2] = p$,

$$\begin{aligned}
\sigma_1^2 = \|\mathbb{E}[\sum_{jk}\boldsymbol{Z}_{jk}^\intercal\boldsymbol{Z}_{jk}]\| &= p\|\sum_{jk}(\boldsymbol{M}^{j\,\intercal}\boldsymbol{b}_k^\star)^2\|\boldsymbol{u}_j\|^2\boldsymbol{e}_k\boldsymbol{e}_k^\intercal\| \\
&\le p\mu_u^2\frac{r}{n}\|\boldsymbol{M}\boldsymbol{B}^*\|_F^2 \le 2p\mu_u^2\frac{r}{n}\|\boldsymbol{M}\|_F^2\|\boldsymbol{B}^*\|^2 \le 2p\mu_u^2\frac{r}{n}\delta^2\sigma^{*2}_{\max}.
\end{aligned}$$

and, proceeding in a very similar fashion,

$$\begin{aligned}
\sigma_2^2 = \|\mathbb{E}[\sum_{jk}\boldsymbol{Z}_{jk}\boldsymbol{Z}_{jk}^\intercal]\| &= p\|\sum_{jk}(\boldsymbol{M}^{j\,\intercal}\boldsymbol{b}_k^\star)^2\boldsymbol{u}_j\boldsymbol{u}_j^\intercal\| \\
&\le p\mu_u^2\frac{r}{n}\|\boldsymbol{M}\boldsymbol{B}^*\|_F^2 \le 2p\mu_u^2\frac{r}{n}\|\boldsymbol{M}\|_F^2\|\boldsymbol{B}^*\|^2 \le 2p\mu_u^2\frac{r}{n}\delta^2\sigma^{*2}_{\max}.
\end{aligned}$$

Thus, $\sigma^2 = \max(\sigma_1^2, \sigma_2^2) = \sigma_2^2$, for $r < q$. Setting $t = \epsilon p\delta\sigma^*_{\max}$, we have

$$\frac{t}{L} = \frac{\epsilon pn}{r\mu\mu_u}, \quad \frac{t^2}{\sigma^2} = \frac{\epsilon^2 pn}{2r\mu_u^2}. \tag{12}$$

Thus, by the Matrix-Bernstein inequality, and using $\mu \le \mu_u$,

$$\|\boldsymbol{D}\| = \|\boldsymbol{D} - \mathbb{E}[\boldsymbol{D}]\| \le \epsilon p\delta\sigma^*_{\max}, \text{w.p. greater than } 1 - \exp(\log q - c\frac{\epsilon^2 pn}{r\mu_u^2}). \tag{13}$$

1) Final bound: The bounding of $\|\boldsymbol{F}^{-1}\| = 1/\min_k \sigma_{\min}(\boldsymbol{F}_k)$ is similar to that in [4]. By using the matrix-Bernstein inequality, $\|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}_k - p\boldsymbol{I}\| \leq \epsilon p$, w.p. greater than $1 - \exp(\log 2r - c\frac{\epsilon^2 pn}{\mu_u^2 r})$. Applying a union bound over all q diagonal blocks, and using the fact that $\sigma_{\min}(\boldsymbol{F}) = \min_k \sigma_{\min}(\boldsymbol{U}_k^{\intercal}\boldsymbol{U}_k))$ (since $\boldsymbol{F}$ is block diagonal),

$$\|\boldsymbol{F}^{-1}\| \leq \frac{1}{(1-\epsilon)p}, \quad \text{w.p. greater than } 1 - \exp(\log q + \log 2r - c\frac{\epsilon^2 pn}{\mu_u^2 r}). \tag{14}$$

(the above also follows by Lemma C.7 of [4]).

Using the above bound and the bound on $\|\boldsymbol{D}\|$ from above

$$\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \sqrt{r}\|\boldsymbol{F}^{-1}\|\|\boldsymbol{D}\| \leq 2\epsilon\sqrt{r}\delta\sigma^*_{\max}, \text{ w.p. greater than } 1 - \exp(\log q - \frac{\epsilon^2 pn}{r\mu_u^2}). \tag{15}$$

Let $\epsilon = \epsilon_1/2\sqrt{r}$. Then,

$$\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \|\boldsymbol{F}^{-1}\| \cdot \sqrt{r}\|\boldsymbol{D}\| \leq \epsilon_1\delta\sigma^*_{\max}, \text{ w.p. greater than } 1 - \exp(\log q - \frac{\epsilon_1^2 pn}{r^2\mu_u^2}). \tag{16}$$

C. *Proof of Lemma 4.3*

Let $\delta = \delta^{(t-1)}$. Writing $\boldsymbol{X}^\star = \boldsymbol{U}\boldsymbol{G} + (\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^{\intercal})\boldsymbol{X}^\star$, and $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$, we have $\|\boldsymbol{X}^\star - \boldsymbol{X}\|_F \leq \|\boldsymbol{B} - \boldsymbol{G}\|_F + \|(\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^{\intercal})\boldsymbol{U}^*\boldsymbol{B}^*\|_F \leq \|\boldsymbol{B} - \boldsymbol{G}\|_F + \|(\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^{\intercal})\boldsymbol{U}^*\|_F\|\boldsymbol{B}^*\| \leq c\delta\sigma^*_{\max} + \delta\sigma^*_{\max}$. For ii), using the bound on $\|\boldsymbol{B} - \boldsymbol{G}\|_F$, $\|\boldsymbol{B}\| = \|\boldsymbol{B} - \boldsymbol{G} + \boldsymbol{G}\| \leq \|\boldsymbol{B} - \boldsymbol{G}\| + \|\boldsymbol{G}\| \leq \delta\sigma^*_{\max} + \sigma^*_{\max}$. For iii), $\sigma_{\min}(\boldsymbol{B}) \geq \sigma_{\min}(\boldsymbol{G}) - \sigma_{\max}(\boldsymbol{B} - \boldsymbol{G}) \geq \sqrt{1-\delta^2}\sigma^*_{\min} - \delta\sqrt{r}\sigma^*_{\max}$. Here we used $\sigma_{\min}(\boldsymbol{G}) = \sigma_{\min}(\boldsymbol{U}^{\intercal}\boldsymbol{U}^\star\boldsymbol{B}^\star) \geq \sigma_{\min}(\boldsymbol{U}^{\intercal}\boldsymbol{U}^*)\sigma^*_{\min} \geq \sqrt{1-\delta^2}\sigma^*_{\min}$.

D. *Proof of Lemma 4.4*

Since $\boldsymbol{b}_k = (\boldsymbol{U}_k^{\intercal}\boldsymbol{U}_k)^{-1}\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k\boldsymbol{b}_k^\star$,

$$\|\boldsymbol{b}_k\| \leq \|(\boldsymbol{U}_k^{\intercal}\boldsymbol{U}_k)^{-1}\| \cdot \|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k\| \cdot \|\boldsymbol{b}_k^\star\|$$

By Lemma C.6 of [4]:

$$\|(\boldsymbol{U}_k^{\intercal}\boldsymbol{U}_k)^{-1}\| \leq \frac{1}{(1-\epsilon)p} \quad \text{w.p. at least } 1 - \exp(\log r - \epsilon^2 pn/\mu_u^2 r) \tag{17}$$

To bound $\|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k\|$, first note that

$$\mathbb{E}[\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k] = p\boldsymbol{U}^{\intercal}\boldsymbol{U}^\star.$$

We bound $\|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k - p\boldsymbol{U}^{\intercal}\boldsymbol{U}^\star\|$ using matrix Bernstein inequality [34]. We summarize it above in Proposition 5.1. Recall the expression for $\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k$ from Sec. IV-A. Let $\boldsymbol{Z}_j = (\xi_{jk} - p)\boldsymbol{u}^j\boldsymbol{u}^{*j\intercal}$. As done in earlier proofs, we can show that

$$L = \max_j \|\boldsymbol{Z}_j\| \leq \mu\mu_u\sqrt{r/n}, \quad \text{and}$$

$$\sigma^2 = \max(\|\sum_j \mathbb{E}[\boldsymbol{Z}_j\boldsymbol{Z}_j^{\intercal}]\|, \|\sum_j \mathbb{E}[\boldsymbol{Z}_j^{\intercal}\boldsymbol{Z}_j]\|) \leq 2p\mu_u^2\frac{r}{n}.$$

For this, we used the fact that $\mathbb{E}[(\xi_{jk} - p)^2] = 2p(1-p) \leq 2p$, $\|\sum_j \|\boldsymbol{u}^*{}_j\|^2\boldsymbol{u}_j\boldsymbol{u}_j^{\intercal}\| \leq \mu^2(r/n)\|\boldsymbol{U}\| = \mu^2(r/n)$ and $\|\sum_j \|\boldsymbol{u}^j\|^2\boldsymbol{u}^*{}_j\boldsymbol{u}^*{}_j{}^{\intercal}\| \leq \mu_u^2(r/n)\|\boldsymbol{U}^\star\| = \mu_u^2(r/n)$ and $\mu \leq \mu_u$.

Thus, by matrix Bernstein, $\|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k - p\boldsymbol{U}^{\intercal}\boldsymbol{U}^*\| \leq \epsilon p$ w.p. at least $1 - \exp(\log r - \epsilon^2 pn/\mu_u^2 r)$. Hence, with this probability,

$$\|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k\| \leq \|\boldsymbol{U}_k^{\intercal}\boldsymbol{U}^\star{}_k - \boldsymbol{U}^{\intercal}\boldsymbol{U}^\star\| + p\|\boldsymbol{U}^{\intercal}\boldsymbol{U}^\star\| \leq \epsilon p + p = (1+\epsilon)p. \tag{18}$$

Thus, letting $\epsilon = 0.1$, w.p. at least $1 - \exp(\log r - cpn/\mu_u^2 r)$, $\|\boldsymbol{b}_k\| \leq 1.1\|\boldsymbol{b}_k^\star\| \leq 1.1\mu\sqrt{r/q}\sigma^*_{\max}$. Here we obtained a bound on $\|\boldsymbol{b}_k\|$ for a given k. By union bound, the above bound holds for all $k \in [q]$ w.p. at least $1 - q\exp(\log r - cpn/\mu_u^2 r) = 1 - \exp(\log q + \log r - cpn/\mu_u^2 r)$.

E. Proof of Lemma 4.5

Let $\delta = \delta^{(t-1)}$. Recall that $\text{GradU} = \sum_{jk} \xi_{jk} \boldsymbol{e}_j (\boldsymbol{x}_{jk} - \boldsymbol{x}^*_{jk}) \boldsymbol{b}_k^\intercal$. Using the lemma assumption, $\|\boldsymbol{B} - \boldsymbol{G}\|_F \le \delta \sigma^*_{\max}$. Using this and Lemma 4.3, $\|\boldsymbol{X} - \boldsymbol{X}^\star\|_F \le 2\delta\sigma^*_{\max}$, $\|\boldsymbol{B}\| \le 1.1\sigma^*_{\max}$, We use these bounds in the proof below.

Observe that

$$\mathbb{E}[\text{GradU}] = p(\boldsymbol{X} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal,$$

Thus, using the assumed $\boldsymbol{B} - \boldsymbol{G}$ bound,

$$\|\mathbb{E}[\text{GradU}]\|_F \le p\|\boldsymbol{X} - \boldsymbol{X}^\star\|_F \|\boldsymbol{B}\| \le 2.2 p \delta \sigma^{*2}_{\max}$$

Next we bound the deviation using matrix Bernstein inequality. Writing $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{B}$, $\boldsymbol{X}^\star = \boldsymbol{U}\boldsymbol{G} + (\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\intercal)\boldsymbol{X}^\star$, and using $n \le q$

$$\begin{aligned}
\max_{jk} |\boldsymbol{x}_{jk} - \boldsymbol{x}^*_{jk}| &= |\boldsymbol{e}_j^\intercal (\boldsymbol{X} - \boldsymbol{X}^\star) \boldsymbol{e}_k| \\
&\le \|\boldsymbol{e}_j^\intercal \boldsymbol{U}\| \|\boldsymbol{B} - \boldsymbol{G}\| + \|(\boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\intercal)\boldsymbol{U}^\star\| \|\boldsymbol{B}^\star \boldsymbol{e}_k\| \\
&\le \mu_u \sqrt{r/n}\delta\sigma^*_{\max} + \mu\sqrt{r/q}\delta\sigma^*_{\max} \le 2\mu_u\sqrt{r/n}\delta\sigma^*_{\max}
\end{aligned}$$

Using this,

$$L = \max_{jk} \|(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})\boldsymbol{b}_k^\intercal\| \le \max_{jk} \|(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})\| \max_k \|\boldsymbol{b}_k\| \le 2\mu_u\sqrt{r/n}\delta\sigma^*_{\max} \cdot \mu\sqrt{r/q}\sigma^*_{\max} \le 2\mu_u\mu(r/n)\delta\sigma^{*2}_{\max},$$

To bound $\sigma_1^2 = \|\sum_{jk} \mathbb{E}[\boldsymbol{Z}_{jk}\boldsymbol{Z}_{jk}^\intercal]\|$ and $\sigma_2^2 = \|\sum_{jk} \mathbb{E}[\boldsymbol{Z}_{jk}^\intercal \boldsymbol{Z}_{jk}]\|$, where $\boldsymbol{Z}_{jk} = \xi_{jk}\boldsymbol{e}_j(\boldsymbol{x}_{jk} - \boldsymbol{x}^*_{jk})\boldsymbol{b}_k^\intercal$, using $\mathbb{E}[(\xi_{jk} - p)^2] = 2p(1-p) \le 2p$ we have

$$\sigma_1^2 = 2p\|\sum_{jk}(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})^2 \boldsymbol{e}_j \boldsymbol{b}_k^\intercal \boldsymbol{b}_k \boldsymbol{e}_j^\intercal\| \le 2p\|\boldsymbol{b}_k\|^2 \|\boldsymbol{X} - \boldsymbol{X}^\star\|_F^2 \le 2p\mu^2(r/q)\sigma^{*2}_{\max} \cdot (\delta\sigma^*_{\max})^2 = 2p\mu^2(r/q)\delta^2\sigma^{*4}_{\max}.$$

$$\sigma_2^2 = 2p\|\sum_{jk}(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})^2 \boldsymbol{e}_j^\intercal \boldsymbol{e}_j \boldsymbol{b}_k \boldsymbol{b}_k^\intercal\| \le 2p\|\boldsymbol{b}_k\|^2 \|\boldsymbol{X} - \boldsymbol{X}^\star\|_F^2 = \sigma_1^2.$$

Setting $t = \epsilon p \delta \sigma^{*2}_{\min}$, we have

$$\frac{t^2}{\sigma^2} = c\frac{\epsilon^2 p^2 \delta^2 {\sigma^*_{\min}}^4}{p\mu^2(r/q)\delta^2\sigma^{*4}_{\max}} = \frac{\epsilon^2 pq}{\kappa^4\mu^2 r}, \quad \frac{t}{L} = c\frac{\epsilon p \delta \sigma^{*2}_{\min}}{\mu_u\mu(r/n)\delta\sigma^{*2}_{\max}} = \frac{\epsilon pn}{\kappa^2\mu_u\mu r}.$$

Thus,

$$\min\left(\frac{t^2}{\sigma^2}, \frac{t}{L}\right) = c\frac{\epsilon^2 pn}{\max(\kappa^4\mu^2, \kappa^2\mu_u\mu)r}$$

and so, by matrix Bernstein, w.p. at least $1 - \exp(\log q - c\frac{\epsilon^2 pn}{\max(\kappa^4\mu^2, \kappa^2\mu_u\mu)r})$, $\|\text{GradU} - \mathbb{E}[\text{GradU}]\| \le \epsilon p \delta \sigma^{*2}_{\min}$.

By setting $\epsilon = \epsilon_1/\sqrt{r}$, $\|\text{GradU} - \mathbb{E}[\text{GradU}]\|_F \le \sqrt{r}\|\text{GradU} - \mathbb{E}[\text{GradU}]\| \le \epsilon_1 p \delta \sigma^{*2}_{\min}$ w.p. at least $1 - \exp(\log q - \frac{\epsilon_1^2 pn}{\max(\kappa^4\mu^2, \kappa^2\mu_u\mu)r^2})$.

F. Proof of Lemma 4.6

Let $\delta = \delta^{(t-1)}$. This is used only for proving the second part.

Recall from Sec. IV-A that

$$\text{Grad}\boldsymbol{u}^j = \sum_k \xi_{jk}(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})\boldsymbol{b}_k^\intercal,$$

To apply matrix Bernstein, we need to bound $\sigma_1^2 \equiv \|\mathbb{E}[\sum_k \xi_{jk}^2 (\boldsymbol{x}_{jk} - \boldsymbol{x}^*_{jk})^2 \boldsymbol{b}_k^\intercal \boldsymbol{b}_k]\|$, $\sigma_2^2 \equiv \|\mathbb{E}[\sum_k \xi_{jk}^2 (\boldsymbol{x}_{jk} - \boldsymbol{x}^*_{jk})^2 \boldsymbol{b}_k \boldsymbol{b}_k^\intercal]\|$, and $L = \max_k \|(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})\boldsymbol{b}_k\|$. Using

$$|\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk}| \le 2\max(|\boldsymbol{x}_{jk}|, |\boldsymbol{x}^\star_{jk}|) \le 2\max(\|\boldsymbol{u}^j\|\|\boldsymbol{b}_k\|, \|\boldsymbol{u}^{*j}\|\|\boldsymbol{b}^\star_k\|) \le 2\max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)\max(\|\boldsymbol{b}_k\|, \|\boldsymbol{b}^\star_k\|).$$

and $\max(\|\boldsymbol{b}^\star_k\|, \|\boldsymbol{b}_k\|) \le 1.1\mu\sqrt{r/q}\sigma^*_{\max}$,

$$\begin{aligned}
L \le \max_k |\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk}| \max_k \|\boldsymbol{b}_k\| &\le 2\max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) \max_k \max(\|\boldsymbol{b}_k\|, \|\boldsymbol{b}^\star_k\|) \max_k \|\boldsymbol{b}_k\| \\
&\le 2\max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)\mu^2(r/q)\sigma^{*2}_{\max}. \qquad (19)
\end{aligned}$$

We use the above seemingly loose bound because we do not need $\delta^{(t-1)}$ in this bound. Instead we need to the bound to be of the form $\max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^*{}_j\|)$ times a factor of $\sqrt{r/q}\sigma^*_{\max}$

The variance $\sigma_1^2 = \|\mathbb{E}[\sum_k \xi_{jk}(\boldsymbol{x}_{jk} - \boldsymbol{x}^*_{jk})^2 \boldsymbol{b}_k^\intercal \boldsymbol{b}_k]\|$ can be bounded using similar ideas as follows.

$$\begin{aligned}\sigma_1^2 = \| \sum_k p(\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk})^2 \boldsymbol{b}_k^\intercal \boldsymbol{b}_k \| &\leq 2p\mathrm{max}(\|\boldsymbol{u}^j\|_2, \|\boldsymbol{u}^{*j}\|_2)^2 \max_k(\|\boldsymbol{b}_k\|, \|\boldsymbol{b}^*_k\|)^2 \sum_k \|\boldsymbol{b}_k\|_2^2 \\ &\leq 2p \max\left(\|\boldsymbol{u}^j\|_2, \|\boldsymbol{u}^{*j}\|_2\right)^2 \mu^2 (r^2/q) \sigma^{*4}_{\max}. \end{aligned} \tag{20}$$

Similarly, $\sigma_2^2 \leq 2p \max\left(\|\boldsymbol{u}^j\|_2, \|\boldsymbol{u}^{*j}\|_2\right)^2 \mu^2 (r^2/q) \sigma^{*4}_{\max}$. By the matrix Bernstein inequality with $t = \epsilon p \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)\sigma^{*2}_{\min}$, and noting that $\sigma^2 = \max(\sigma_1^2, \sigma_2^2) = \sigma_1^2$, we have w.p. at least $1 - \exp(\log q - \epsilon^2 pn/\mu^2\kappa^4 r^2)$,

$$\|\mathbb{E}[\mathrm{Grad}\boldsymbol{u}^j] - \mathrm{Grad}\boldsymbol{u}^j\| \leq \epsilon p \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)\sigma^{*2}_{\min}. \tag{21}$$

This completes the proof for the first part of the lemma.

1) Proof of second part: In line 7 of Algorithm 1, adding/subtracting $\mathbb{E}[\mathrm{Grad}\boldsymbol{u}^j] = p(\boldsymbol{u}^j \boldsymbol{B}\boldsymbol{B}^\intercal - \boldsymbol{u}^{*j}\boldsymbol{B}^\star\boldsymbol{B}^\intercal)$,

$$\tilde{\boldsymbol{u}}^{j+} = \boldsymbol{u}^j - \eta \mathrm{Grad}\boldsymbol{u}^j = \boldsymbol{u}^j(\boldsymbol{I} - \eta p \boldsymbol{B}\boldsymbol{B}^\intercal) + \eta p \boldsymbol{u}^{*j}\boldsymbol{B}^\star\boldsymbol{B}^\intercal + \eta(\mathbb{E}[\mathrm{Grad}\boldsymbol{u}^j] - \mathrm{Grad}\boldsymbol{u}^j), \tag{22}$$

Since we assumed $\delta \leq c/\kappa$, by Lemma 4.3, $\sigma_{\min}(\boldsymbol{B}) \geq 0.9\sigma^*_{\min}$ and $\sigma_{\max}(\boldsymbol{B}) \leq 1.1\sigma^*_{\max}$. Thus, if $\eta < 0.5/p\sigma^{*2}_{\max}$ then, $\boldsymbol{I} - \eta p \boldsymbol{B}\boldsymbol{B}^\intercal$ is positive semi-definite (psd) and so $\|\boldsymbol{I} - \eta p \boldsymbol{B}\boldsymbol{B}^\intercal\| = \lambda_{\max}(\boldsymbol{I} - \eta p \boldsymbol{B}\boldsymbol{B}^\intercal) = 1 - \eta p \sigma^2_{\min}(\boldsymbol{B}) \leq 1 - 0.9\eta p {\sigma^*_{\min}}^2$.

Thus, using the above bound on $\|\mathbb{E}[\mathrm{Grad}\boldsymbol{u}^j] - \mathrm{Grad}\boldsymbol{u}^j\|$, if $\eta < 0.5/p\sigma^{*2}_{\max}$,

$$\begin{aligned}\|\tilde{\boldsymbol{u}}^{j+}\| &\leq \|\boldsymbol{u}^j\|(1 - 0.9\eta p \sigma^{*2}_{\min}) + \eta p \|\boldsymbol{u}^{*j}\|\sigma^{*2}_{\max} + \epsilon \eta p \sigma^{*2}_{\min} \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) \\ &\leq (1 - (0.9 - \epsilon)\eta p \sigma^{*2}_{\min}) \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) + \eta p \sigma^{*2}_{\max}\|\boldsymbol{u}^{*j}\|, \end{aligned} \tag{23}$$

w.p. at least $1 - \exp(\log q - \epsilon^2 pn/\mu^2\kappa^4 r^2)$. We bound $\|\boldsymbol{u}^{j(t+1)}\| \leq \|(\boldsymbol{R}^+)^{-1}\| \cdot \|\tilde{\boldsymbol{u}}^{j(t+1)}\|$, where $\tilde{\boldsymbol{U}}^+ \overset{\mathrm{QR}}{=} \boldsymbol{U}^+\boldsymbol{R}^+$ next. Using Lemma 4.5,

$$\begin{aligned}\|(\boldsymbol{R}^+)^{-1}\| = \frac{1}{\sigma_{\min}(\boldsymbol{U} - \eta\|\mathrm{GradU}\|)} &\leq \frac{1}{1 - \eta 2.1 p\delta\sigma^{*2}_{\max}} \\ &\leq \frac{1}{1 - 0.25\eta p \sigma^{*2}_{\max}} \leq 1 + 0.5\eta p \sigma^{*2}_{\max}. \end{aligned} \tag{24}$$

w.p. given in Lemma 4.5. Thus, using (23) and (24) in $\|{\boldsymbol{u}^j}^+\| \leq \|(\boldsymbol{R}^+)^{-1}\| \cdot \|\tilde{\boldsymbol{u}}^{j(t+1)}\|$,

$$\begin{aligned}\|\boldsymbol{u}^{j+}\| &\leq (1 + 0.5\eta p\sigma^{*2}_{\max})(1 - (0.9 - \epsilon)\eta p \sigma^{*2}_{\min}) \max(\|\boldsymbol{u}^j\|\|\boldsymbol{u}^{*j}\|) + (1 + 0.5\eta p \sigma^{*2}_{\min})\eta p \sigma^{*2}_{\max}\|\boldsymbol{u}^{*j}\| \\ &\leq (1 - (0.4 - \epsilon)\eta p \sigma^{*2}_{\min}) \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) + \cdots \\ &\cdots (1 + 0.5\eta p\sigma^{*2}_{\max})\eta p \sigma^{*2}_{\max}\|\boldsymbol{u}^{*j}\| \\ &\leq (1 - (0.4 - \epsilon)\eta p \sigma^{*2}_{\min}) \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) + (1 + 0.25/\kappa^2)0.5\|\boldsymbol{u}^{*j}\| \\ &\leq (1 - \frac{0.15}{\kappa^2}) \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) + 0.7\|\boldsymbol{u}^{*j}\| \end{aligned}$$

where the last bound follows by setting $\eta = 0.5/p\sigma^{*2}_{\max}$ and setting $\epsilon = 0.1$ in (21). Thus, we have shown that if $\eta = 0.5/p\sigma^{*2}_{\max}$, and if $\delta \leq c/\kappa^2$, with probability exceeding $1 - 4/n^3$,

$$\begin{aligned}\|{\boldsymbol{u}^j}^+\| &\leq (1 - \frac{0.15}{\kappa^2}) \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|) + 0.7\|\boldsymbol{u}^{*j}\| \\ &\leq (1 - \frac{0.15}{\kappa^2})\|\boldsymbol{u}^j\| + 2\|\boldsymbol{u}^{*j}\|. \end{aligned}$$

VI. Guarantee for Noisy LRMC

Consider the noisy LRMC problem defined as follows. We observe

$$\boldsymbol{Y} := \boldsymbol{X}^\star_\Omega + \boldsymbol{W}_\Omega$$

We do not make any assumption on the noise $\boldsymbol{W}$ (deterministic $\boldsymbol{W}$). Unlike the standard Gaussian assumption, this assumption is weaker: it allows the noise entries to be anything, all entries could be positive too for example.

We can show the following by borrowing the overall proof approach for modifying a noise-free case guarantee for an iterative algorithm from past works [12, 39]; the latter used a similar approach for analyzing the AltMin algorithm for LR phase retrieval.

Corollary 6.1. *Let*

$$\epsilon_{noise} := \max_{jk} \frac{|w_{jk}|}{|x^*_{jk}|}$$

In the setting of Theorem 2.1, if $\epsilon_{noise} \leq \frac{c}{\sqrt{r}\kappa^3}$*, then, by setting* $T = C\kappa^2 \log(1/\epsilon_{noise})$*, we can guarantee that*

$$\mathrm{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^\star) \leq \kappa^2 \sqrt{r}\epsilon_{noise}$$

and $\|\boldsymbol{X}^{(T)} - \boldsymbol{X}^\star\|_F \le \mathrm{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^\star)\sigma^*_{\max}$. *In general, for any* ϵ, *by setting* $T = C\kappa^2 \log(1/\epsilon)$, *we can ensure that*

$$\mathrm{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^\star) \le \max(\epsilon, \kappa^2\sqrt{r}\epsilon_{noise})$$

and $\|\boldsymbol{X} - \boldsymbol{X}^\star\|_F \le \mathrm{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^\star)\sigma^*_{\max}$.

Proof. Corollary 6.1 extends our noise-free case proof to the noisy case using the following simple ideas. First we assume that the noise level is small enough so that accurate initialization is possible; this requires noise level, $\epsilon_{noise}\sqrt{r}$, to be of order $c\delta^{(0)}\sigma^*_{\min}$ for our required value of $\delta^{(0)} = c/\kappa^2$. This helps ensure that the initialization error is bounded by $\delta^{(0)}$.

Next, at each iteration, we attempt to bound terms and prove incoherence for the updated $\boldsymbol{U}, \boldsymbol{B}$, in order to guarantee error decay similar to the noise-free case. For this, we need the noise level to be such that (i) the error in recovering $\boldsymbol{B}^\star$ is of the same order as in the noise-free case; and (ii) the same is true for the bounds on the gradient norm. Both of these are ensured if $\epsilon_{noise}\kappa^2\sqrt{r} \le c\delta^{(t-1)}$ for a $c < 1$, e.g., $c = 0.1$. Details are given in Appendix B. □

An entry-wise noise bound is assumed also in all other works that study iterative algorithms for noisy LRMC. The work on Smooth AltMin [14] considers the noisy case under a similar assumption to ours. It replaces the noise $\boldsymbol{N}$ by $\boldsymbol{W} := (\boldsymbol{I} - \boldsymbol{U}^\star\boldsymbol{U}^{\star\intercal})\boldsymbol{N}$ and then requires the following bounds on it:

$$\max_j \|e_j^\intercal \boldsymbol{W}\|^2 \le \mu{\sigma^*_{\min}}^2/q, \ \max_{jk} |\boldsymbol{W}_{jk}| \le \mu\|\boldsymbol{X}^\star\|_F/q$$

In addition, it needs a sample complexity of order $n_{mx}r^3$. Notice that the second bound above is on the maximum entry magnitude $\max_{jk}|\boldsymbol{W}_{jk}|$ similar to ours. Also note that the right hand side of the bound contains $1/q$ and not $1/\sqrt{q}$, making it a strong assumption. A similar result is proved in [8] as well. We note that the works on AltMin [4], FactGD [10] and ProjGD [6] do not provide noisy case bounds. The latter two do consider sparse outliers which are handled differently; the magnitude of any entry of a sparse outlier is not bounded, but the number of nonzero entries is bounded. On the other hand, noise means a small and bounded disturbance in all observed entries.

The above guarantee shows how the noise-free case proof approach can be directly extended to also handle the noisy case. Notice from Theorem 6.1 that the sample complexity requirement does not depend on the noise level. This is why our result can only tolerate small noise levels. A guarantee that holds for any value of noise would require the sample complexity to grow with NSR/ϵ^2 where NSR is the noise to signal ratio (can be defined as $(q\sigma_w^2)/{\sigma^*_{\min}}^2$). It should be possible to obtain such a result, and one that assumes Gaussian noise instead of bounded noise, by adapting ideas from [40].

VII. Improved Guarantees for AltMin and Smooth-AltMin

Using the same lemmas used to prove the AltGDmin guarantee, we are also able to improve the guarantee for AltMin and Smooth-AltMin [14], as long as both are initialized as given in Algorithm 1: the initial estimate of $\boldsymbol{U}$ is clipped using row-wise clipping. We borrow this clipping idea from [10].

A. Improving the result for AltMin

Corollary 7.1 (Improved AltMin [4] Guarantee)**.** *Consider the AltMin algorithm initialized using our initialization (lines 2-4) of Algorithm 1. Assume that Assumption 1.1 holds and that, at each iteration* t, *entries of* $\boldsymbol{X}^\star$ *are observed independently with probability* p *satisfying*

$$np > C\kappa^4\mu^2 r^2 \log q. \tag{25}$$

Then, w.p. at least $1 - 3/n^3$, *the iterates* $\boldsymbol{U}^{(t)}$ *of AltMin [4] satisfy* $\mathrm{SD}_F(\boldsymbol{U}^{(t+1)}, \boldsymbol{U}^\star) \le 0.25\mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^*)$.

Consequently, if the expected number of samples observed across all iterations satisfies $\mathbb{E}[|\Omega|] = T \cdot nqp > \frac{C\kappa^4\mu^2 r^2 \log q}{n}\log(1/\epsilon)$, *then, after* $T = C\log(1/\epsilon)$ *iterations,* $\mathrm{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^\star) \le \epsilon$.

We prove this in Appendix C. The proof is an easy corollary of Lemmas 4.1, 4.2, 4.3 and 4.4. Comparing (25) to Theorem 2.5 of [4], we observe that the sample complexity for AltMinComplete has reduced in r from $r^{4.5}\log r$ to r^2.

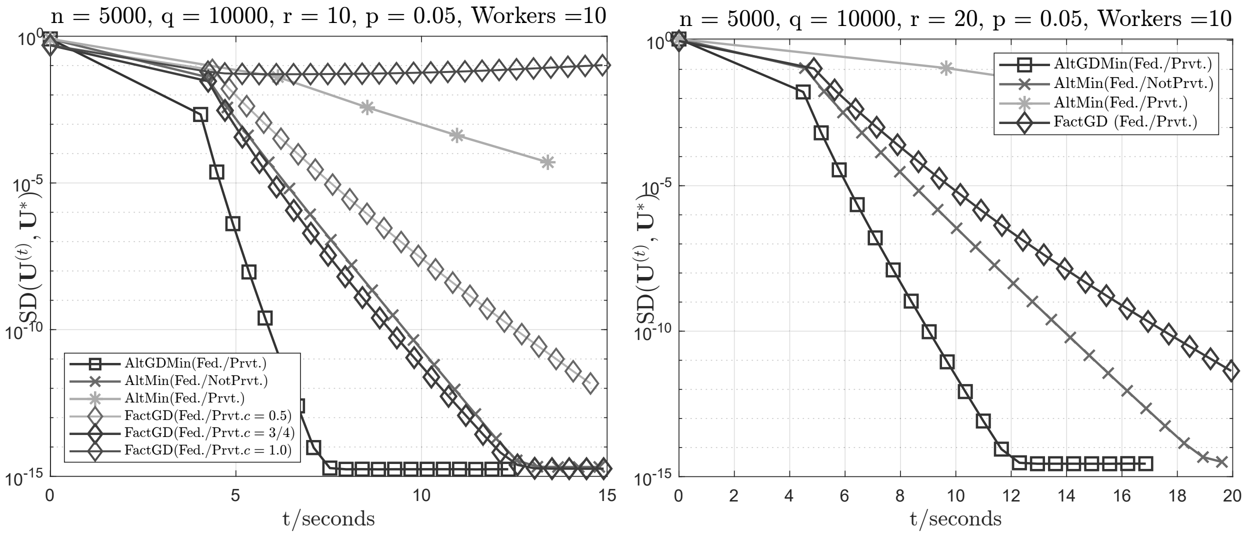
B. Improved Guarantee for Smooth AltMin [14]

Consider Smooth AltMin given in Fig 1 of [14]. By replacing its initialization by ours or that from [10], we can improve its sample complexity guarantee by a factor of r. We summarize this next.

Corollary 7.2. *Let* $\epsilon > 0$. *Consider the output of Smooth AltMin given in Fig 1 of [14] with its Initialization procedure replaced by lines 2-4 of AltGDmin given in our Algorithm 1. Let* $\boldsymbol{X}^\star$ *be a symmetric* $n \times n$ *matrix. Set* $T = O(\log(n/\epsilon))$. *The algorithm output satisfies* $\mathrm{SD}_F(\boldsymbol{U}, \boldsymbol{U}^*) \le \epsilon$ *with probability* $9/10$, *provided that sampling probability* $p \ge p_0 + p_{LS}$, *where*

$$p_0 \ge C\kappa^2 r^2 \mu(\log n)/n \quad p_{\mathrm{LS}} \ge C\kappa^2\mu^2 r^2(\log(n/\epsilon)\log^2 n)/n. \tag{26}$$

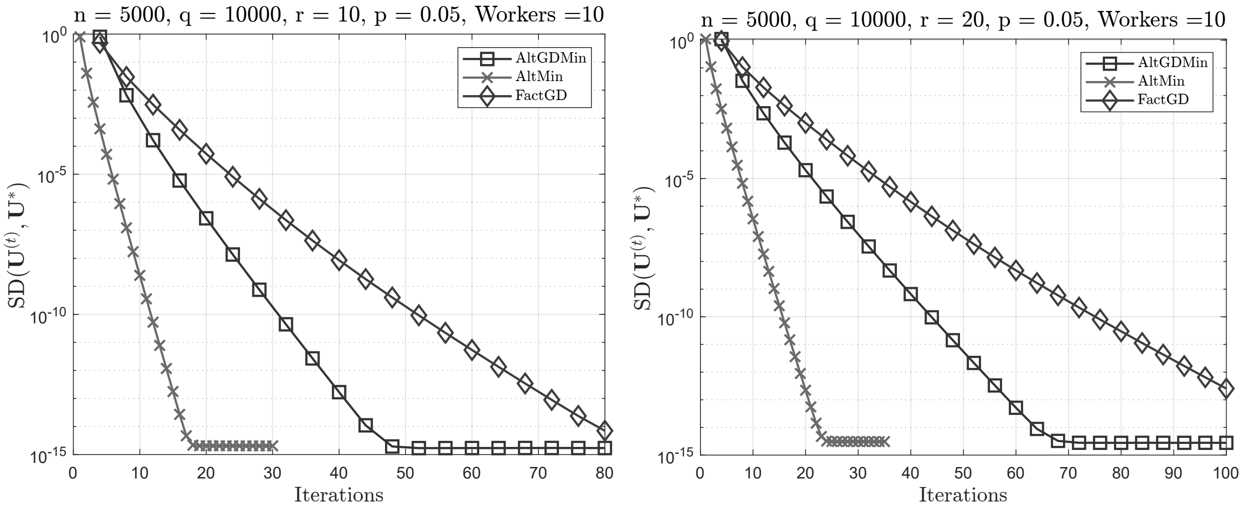
Thus, its sample complexity is $nqp \cdot T \ge C\kappa^2\mu^2 q r^2(\log(n/\epsilon)\log^2 n)$.



(a) Federated setting: Error against time-taken, $r = 10$

(b) Federated setting: Error against time-taken, $r = 10$

Fig. 1: *Comparing federated implementations of AltGDmin (proposed), FactGD and AltMin. For AltMin we compare two versions - the private one (which uses multiple GD iterations to solve the minimization step for updating U) and the not private one. The results match what our theory (sufficient conditions) predicts. AltGDmin is the fastest due to its lowest communication-efficiency and due to all three having comparable computation cost. In (a), we also compare FactGD with three choices of step size. See experiments' description for details.*



(a) Error against iterations, $r = 10$.

(b) Error against iterations, $r = 20$.

Fig. 2: *Iteration complexity comparisons: We plot the numerically computed subspace recovery error against iteration count for the same two cases as those in Fig 2. This remains the same whether the implementation is centralized or federated. The results match theory once again. AltMin iteration complexity is order* $\log(1/\epsilon)$*, AltGDmin is* $\kappa^2 \log(1/\epsilon)$ *and FactGD is* $\kappa\mu r \log(1/\epsilon)$.

This is a direct corollary of Theorem 6.1 of [14] combined with our initialization lemma, Lemma 4.1. The only thing we change for Smooth AltMin is its initialization step. For the proof also we replace use of the initialization guarantee given in Theorem 7.1 of [14] by our Lemma 4.1.

VIII. Simulation Results

1) Experimental Setup: We plot averaged subspace distance at the iteration t against the average time taken until iteration t, with the averages being computed over 100 Monte Carlo runs. The averaging is over the observed entries which are generated uniformly at random. The matrix $\boldsymbol{X}^\star = \boldsymbol{U}^\star \boldsymbol{B}^\star$ was generated once: we let $\boldsymbol{U}^\star$ be an orthonormalized $n \times r$ random Gaussian matrix and $\boldsymbol{B}^\star$ an $r \times q$ random Gaussian matrix. We used the 'parfor' loop in MATLAB to distribute the computation across 10 workers, with each worker being an individual core of a multi-core CPU (13-th Gen. Intel Core i7 with 32GB RAM and 16 cores). To compute the left-singular vectors needed for the initialization of federated algorithms, we used the federated power method and performed 15 power iterations. Also, our experiments we do not sample-split, i.e., we run each iteration of the algorithms on all observed entries.

For federated versions of all algorithms, we plot recovery error at iteration t against total time taken until iteration t in Fig. 1. In Fig. 2 we numerically compare the iteration complexity of all three algorithms. This plot is just error versus iteration t and will look the same regardless of which machine it is run on.

2) Step-size and other parameters: For FactGD (Centralized), we used the code provided by the authors of that work [10]. The step size was set $\eta_{\text{FactGD}} = pc/\|\boldsymbol{Y}\|$, as also done in the authors' own implementation of their algorithm. Setting $c = 0.75$ showed the fastest convergence for our simulations (see Fig. 1a). For AltMin (Fed./NotPrvt.) and AltMin(Fed./Prvt.), we wrote our own MATLAB implementation; the latter uses GD to solve the LS problem for updating $\boldsymbol{U}$. We set the number of GD iterations for solving each LS problem to 10 with step size $\eta = p/\|\boldsymbol{Y}\|^2$, which showed the fastest convergence. Note that the $\boldsymbol{U}$-update least-squares problem is convex, and the chosen step-size is an upper bound on the Lipschitz constant of the expected gradient of the objective function at $t = 0$ [41]. AltGDMin is also implemented with step-size $\eta = p/\|\boldsymbol{Y}\|^2$, which approximates the step size choice suggested by Theorem 2.1, since $||\mathbb{E}[\boldsymbol{Y}]||^2 = p^2 \sigma_{\max}^{*2}$. AltMin and AltGDMin were both initialized by lines 2-4 of Algorithm 1. The incoherence parameter μ was estimated as $\hat{\mu} = \text{argmin}_{\mu} \|\boldsymbol{u}^{(0)j}\| \leq \mu\sqrt{r/n}$ for all $j \in [n]$.

3) Observations: The proposed algorithm AltGDMin (Fed.) is faster than all benchmark methods, especially at the higher value of $r = 20$. Specifically, *AltGDMin (Fed.) converges to $\boldsymbol{U}^\star$ in approximately 12 seconds, compared to nearly 20 seconds for the second fastest AltMin (Fed. NotPrvt.).* We interpret/explain these observations and the benchmark methods below.

4) AltGDMin: AltGDMin (Fed.) has faster convergence because of low communication complexity and low time complexity at the center. The upstream communication complexity, $\min(\sum_{k \in \boldsymbol{S}_\ell} \nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{b}_k), nr) = nr$, is low because the nodes sum the column-wise gradients

5) FactGD: $\nabla_{\boldsymbol{U}} f = \nabla_{\boldsymbol{X}} \boldsymbol{B}^\intercal \in \mathbb{R}^{n \times r}$ and $\boldsymbol{B}\boldsymbol{B}^\intercal \in \mathbb{R}^{r \times r}$ are computed at the nodes and transmitted upstream. The total upstream communication cost is $nr + r^2$, higher than the nr communication cost of AltGDMin. $\boldsymbol{B}$ is updated locally (GD iteration and normalizing) at the nodes but this requires two data exchanges with the center. This is because the gradient of $\boldsymbol{B}$, $(\boldsymbol{B}\boldsymbol{B}^\intercal - \boldsymbol{U}^\intercal\boldsymbol{U})\boldsymbol{B}$, with respect to the norm balancing term, $\|\boldsymbol{U}^\intercal\boldsymbol{U} - \boldsymbol{B}\boldsymbol{B}^\intercal\|_F^2$, cannot be computed at the nodes. The nodes compute and transmit $\sum_{k \in \mathcal{S}_\ell} \boldsymbol{b}_k \boldsymbol{b}_k^\intercal$, which are summed at the center to form $\boldsymbol{B}\boldsymbol{B}^\intercal$, and transmitted back to the nodes. The center also computes and transmits $\boldsymbol{U}^\intercal\boldsymbol{U}$. Then, at the nodes, the partial gradient $(\boldsymbol{B}\boldsymbol{B}^\intercal - \boldsymbol{U}^\intercal\boldsymbol{U}) \sum_{k \in \mathcal{S}_\ell} \boldsymbol{b}_k \boldsymbol{b}_k^\intercal$ is computed, followed by column-normalizing. The federated and centralized implementations of FactGD do not use 'for' loops at all, either at the nodes or at the center. But, FactGD is slower than AltMin and AltGDMin because of its higher communication complexity, $O(nr + r^2)$ and 2 data exchanges, compared to 1 data exchange and $O(nr)$, $O(qr)$ communication complexity for AltGDMin and AltMin (Fed.NotPrvt.), respectively.

6) AltMin: AltMin (Fed./NotPrvt.) is slower than AltGDMin (Fed.) because the n $\boldsymbol{U}$-update LS problems are solved sequentially at the center with complexity $|\Omega| r^2$, compared to the nr^2 complexity of computing the QR decomposition for AltGDMin. Also, AltMin (Fed./NotPrvt.) is not private because the nodes communicate the updated $\boldsymbol{b}_k^{(t+1)}$ to the center. For AltMin(Fed./Prvt.), the $\boldsymbol{U}$-update LS problems are solved by multiple gradient descent iterations at the node. While private, the GD version of AltMin is slow because of the communication overhead of transmitting/receiving the gradients several times ($\log(1/\epsilon)$ times for ϵ-accuracy) in each iteration. AltMin (Cntrl.) is slower than federated methods because both $\boldsymbol{U}$, $\boldsymbol{B}$ LS problems are solved sequentially by the closed form solution at the center, but it is still the fastest centralized algorithm overall, possibly because of $\log(1/\epsilon)$ iteration complexity, which is lower than that of other methods.

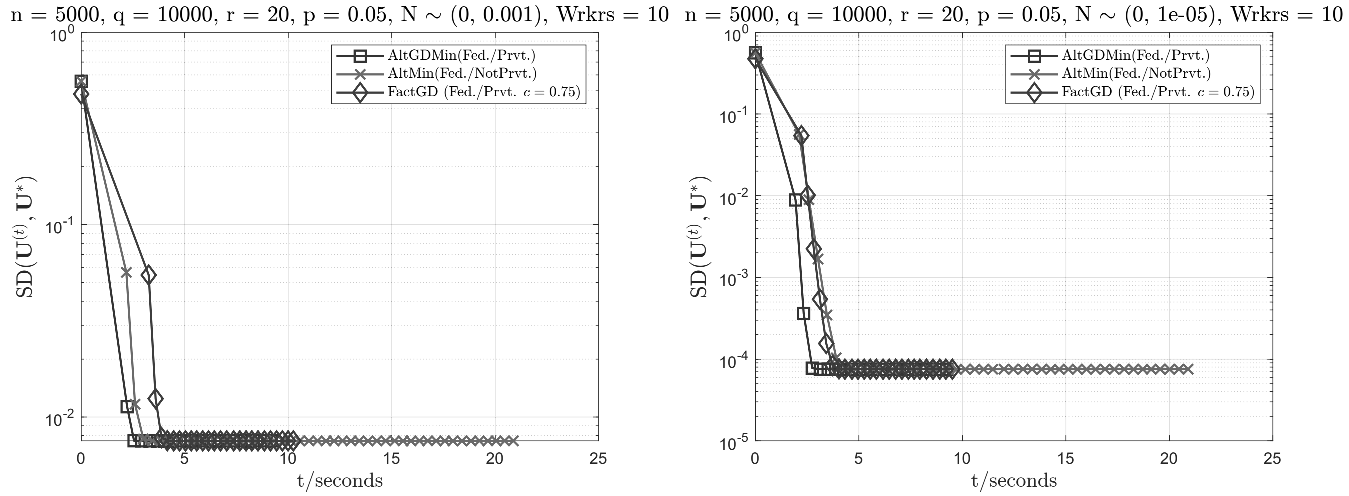
7) Simulations for Noisy LRMC: In Figure 3, we show simulation results for noisy LRMC. $\boldsymbol{X}^\star$ and the set of observed entries Ω were generated as in earlier experiments. We simulated $\boldsymbol{Y} = \boldsymbol{X}_\Omega^\star + \boldsymbol{W}_\Omega$, with $\boldsymbol{W} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$ and three choices of σ^2. Notice that all algorithms converge to a final error value that is proportional to the noise level $\sigma^2/({\sigma_{\min}^*}^2/q)$. For $\sigma = 0.001$, the algorithm converges to error level 0.003. For $\sigma = 0.00001$, it converges to error level 0.00005 and so on.

IX. Conclusions

In this work we developed and analyzed the Alternating GD and Minimization (AltGDmin) algorithm for solving the LRMC problem. The design of AltGDmin is motivated by a federated setting. Using our results (sample and iteration complexity bounds) we argued that, in a federated setting, AltGDmin is the most communication-efficient solution. It is also one of the two fastest private solutions and has the second smallest sample complexity. In addition, we were able to use our lemmas to prove an improved sample complexity guarantee for AltMin, which is the fastest centralized solution for LRMC.

Appendix A
Proof Outline

1) Proof outline of Theorem 2.1: Let $\boldsymbol{U} = \boldsymbol{U}^{(t-1)}$, $\boldsymbol{G} := \boldsymbol{U}^\intercal \boldsymbol{X}^\star$ and $\boldsymbol{U}^+ = \boldsymbol{U}^{(t)}$. To prove Theorem 2.1, (i) we obtain a bound on $\delta^{(t)} := \text{SD}_F(\boldsymbol{U}^+, \boldsymbol{U}^\star)$ in terms of $\delta^{(t-1)} = \text{SD}_F(\boldsymbol{U}, \boldsymbol{U}^\star)$ that can be used to show exponential error decay, w.h.p., under the desired sample complexity. (ii) This bound requires the initialization error $\delta^{(0)}$ to be small enough. Hence we also need to analyze the initialization step to show that this is true w.h.p.. Initialization is analyzed in Lemma 4.1 (proof outline given below). The overall idea for (i) is borrowed from [20] but there are differences because we need each new $\boldsymbol{U}$ to be incoherent. We use induction. The induction step assumes a certain bound on $\delta^{(t-1)}$ and on the row norms of $\boldsymbol{U}$ and proves the same for its updated version, $\boldsymbol{U}^+$. To do this, we first use the induction assumption (which implies $\kappa^2\mu$ row-incoherence of



(a) Noise variance $\sigma^2 = 10^{-6}$

(b) Noise variance $\sigma^2 = 10^{-10}$

Fig. 3: *Noisy LRMC: All results are averages over 25 MC runs. Notice that all algorithms converge to the noise level. The error value to which they converge decreases as the amount of noise decreases.*

$\boldsymbol{U}$) and the fact that $\mathbb{E}[\nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B})] = p(\boldsymbol{X} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal$ to get a deterministic bound on $\delta^{(t)}$. Next, we get high probability bounds on the terms of this bound using matrix Bernstein and appropriate linear algebra. These are obtained in the lemmas given in Sec. IV-B. The last step is to use the induction assumption and Lemma 4.6 to bound the row norms of $\boldsymbol{U}^+$.

Finally, we simplify the bounds in order to show that, w.h.p., $\delta^{(t)}$ decays exponentially with t as long as η is at most $0.5/(p{\sigma^*_{\max}}^2)$ and $\delta^{(0)} \leq c/\kappa^2$. The proof is in Sec. IV-C and it relies on the six lemmas stated in Sec. IV-B.

2) Proof outlines of the lemmas used to prove Theorem 2.1: Initialization Lemma 4.1 uses the following ideas, many of which are borrowed from initialization step analysis of FactGD [10] (with filling in the missing details from there). (i) Using the results of [36] (Lemma 3.21, Theorem 3.22), we can bound $\mathrm{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^\star)$. (ii) The row norm clipping step can be interpreted as projecting its input onto a convex set[3], $\mathcal{U} := \{\boldsymbol{U} : \|\boldsymbol{u}^j\| \leq \mu\sqrt{r/n}\}$, with the projection being in Frobenius norm. Projection onto convex sets is non-expansive, i.e., $\|\Pi_{\mathcal{U}}(\boldsymbol{U}_1) - \Pi_{\mathcal{U}}(\boldsymbol{U}_2)\|_F \leq \|\boldsymbol{U}_1 - \boldsymbol{U}_2\|_F$ [37, eq (9),(10)],[10]. Also, $\Pi_{\mathcal{U}}(\boldsymbol{U}^\star\boldsymbol{Q}) = \boldsymbol{U}^\star\boldsymbol{Q}$ for any $r \times r$ unitary matrix $\boldsymbol{Q}$ (since $\boldsymbol{U}^\star$ as well as $\boldsymbol{U}^\star$ times any unitary matrix belong to $\mathcal{U}$).

(iii) Let $\boldsymbol{Q}_{*,00} := \arg\min_{\boldsymbol{Q}\ unitary} \|\boldsymbol{U}^{(00)} - \boldsymbol{U}^\star Q\|_F$. By Lemmas 2.5 and 2.6 of [36], $\mathrm{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^\star) \approx \|\boldsymbol{U}^{(00)} - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F$ (upper and lower bounded by RHS with a factor of $\sqrt{2}$ for upper bound). (iv) The above ideas help bound $\|\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) - \boldsymbol{U}^\star\boldsymbol{Q}_{*,00}\|_F$. In the last step, we use this bound and $\Pi_{\mathcal{U}}(\boldsymbol{U}^{(00)}) \overset{\mathrm{QR}}{=} \boldsymbol{U}^{(0)}\boldsymbol{R}^{(0)}$ to bound $\mathrm{SD}_F(\boldsymbol{U}^{(0)}, \boldsymbol{U}^\star)$. The proof is in Sec. V-A.

Lemma 4.2 uses incoherence of $\boldsymbol{U}$ and $\delta^{(t-1)}$ to bound $\|\boldsymbol{B} - \boldsymbol{G}\|_F$ where $\boldsymbol{G} = \boldsymbol{U}^\intercal\boldsymbol{X}^\star$. This is proved by writing $vec(\boldsymbol{B} - \boldsymbol{G}) = \boldsymbol{F}^{-1}vec(\boldsymbol{D})$, where $\boldsymbol{F} \in \mathbb{R}^{qr \times qr}$ and $\boldsymbol{D} \in \mathbb{R}^{r \times q}$ are defined below in Sec. IV-A, bounding $\|\boldsymbol{F}^{-1}\|$ and $\|\boldsymbol{D}\|$ by the matrix Bernstein inequality, and using $\|\boldsymbol{B} - \boldsymbol{G}\|_F = \|vec(\boldsymbol{B} - \boldsymbol{G})\| \leq \|\boldsymbol{F}^{-1}\| \cdot \|vec(\boldsymbol{D})\| = \|\boldsymbol{F}^{-1}\| \cdot \|\boldsymbol{D}\|_F \leq \|\boldsymbol{F}^{-1}\| \cdot \sqrt{r}\|\boldsymbol{D}\|$. The bound on $\|\boldsymbol{F}^{-1}\|$ is borrowed from [4], while the rest of our bounding is different and simpler than the approach used in [4]. We use matrix Bernstein which provides a much simpler proof and the resulting bound holds with a smaller sample complexity (better dependence on r). The proof is in Sec. V-B. Lemma 4.3 uses the $\|\boldsymbol{B} - \boldsymbol{G}\|_F$ bound to bound $\|\boldsymbol{X} - \boldsymbol{X}\|_F$, and the minimum and maximums singular values of $\boldsymbol{B}$. The proof idea is similar to that in [13]. We provide the proof in Sec. V-C anyway since it is very short.

Lemma 4.4 uses incoherence of $\boldsymbol{U}$ to show incoherence of $\boldsymbol{B}$. It follows by writing $\boldsymbol{b}_k = (\boldsymbol{U}_k^\intercal\boldsymbol{U}_k)^{-1}(\boldsymbol{U}_k^\intercal\boldsymbol{U}^\star{}_k)\boldsymbol{b}_k^\star$, using the fact that $\mathbb{E}[\boldsymbol{U}_k^\intercal\boldsymbol{U}_k] = p\boldsymbol{U}^\intercal\boldsymbol{U} = p\boldsymbol{I}$, $\mathbb{E}[\boldsymbol{U}_k^\intercal\boldsymbol{U}^\star{}_k] = p\boldsymbol{U}^\intercal\boldsymbol{U}^\star$, $\|\boldsymbol{U}^\intercal\boldsymbol{U}^\star\| \leq 1$, and using matrix Bernstein to bound deviations of both matrices from their expected values. This proof is also much simpler than the one given in [4] and the result needs a smaller sample complexity (better dependence on r). The proof is in Sec. V-D.

Lemma 4.5 uses the $\boldsymbol{B} - \boldsymbol{G}$ bound and its implications, $\delta^{(t-1)} < c/\kappa^2$, and incoherence of $\boldsymbol{U}$ and of $\boldsymbol{B}$ to bound the gradient norm. Let $\mathrm{GradU} := \nabla_{\boldsymbol{U}} f(\boldsymbol{U}, \boldsymbol{B})$. It relies on the following ideas. $\mathbb{E}[\mathrm{GradU}] = p(\boldsymbol{X} - \boldsymbol{X}^\star)\boldsymbol{B}^\intercal$. This holds since the expectation is taken with respect to an independent set of samples at each iteration (possible because of sample splitting). We bound the deviation from the expected value using matrix Bernstein inequality and the assumed bounds in the lemma assumptions. The proof is in Sec. V-E.

Lemma 4.6 uses incoherence of $\boldsymbol{B}$ to show incoherence of $\boldsymbol{U}$. (i) We first bound the deviation of the j-th row of GradU, denoted GradU^j, from its expected value in terms of $\max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)$. To get a high probability bound of this form, we use

[3] To see that $\mathcal{U}$ is a convex set, let matrices $\boldsymbol{M}_1, \boldsymbol{M}_2 \in \mathcal{U}$. Let $\boldsymbol{m}_1^j$ and $\boldsymbol{m}_2^j$ denote the rows of $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$, respectively. For any $0 < \theta < 1$, $\|\theta\boldsymbol{m}_1^j + (1-\theta)\boldsymbol{m}_2^j\| \leq \theta\|\boldsymbol{m}_1^j\| + (1-\theta)\|\boldsymbol{m}_2^j\| \leq \theta\mu\sqrt{r/n} + (1-\theta)\mu\sqrt{r/n} = \mu\sqrt{r/n}$.

matrix Bernstein inequality and the fact that, for showing incoherence, we do not need the bound to contain $\delta^{(t-1)}$. Hence, we can use a seemingly loose bound of the form $|\boldsymbol{x}_{jk} - \boldsymbol{x}^\star_{jk}| \le 2\max(|\boldsymbol{x}_{jk}|, |\boldsymbol{x}^\star_{jk}|) \le 2.2\mu\sqrt{r/q}\sigma^*_{\max} \cdot \max(\|\boldsymbol{u}^j\|, \|\boldsymbol{u}^{*j}\|)$. This bound is, in fact, tighter than a bound that contains $\delta^{(t-1)}$ for the initial iterations when $\delta^{(t-1)}$ is allowed to be larger than order $\sqrt{r/q}$. (ii) Next we use $\tilde{\boldsymbol{u}}^{j^{(t+1)}} = \boldsymbol{u}^{j^{(t)}} - \eta\text{Grad}\boldsymbol{u}^j \pm \mathbb{E}[\text{Grad}\boldsymbol{u}^j]$, $\mathbb{E}[\text{GradU}^j] = p(\boldsymbol{u}^{j\intercal}\boldsymbol{B}\boldsymbol{B}^\intercal - \boldsymbol{u}^{*j\intercal}\boldsymbol{B}^\star\boldsymbol{B}^\intercal)$, and an upper bound on the GD step size η to show the second part of this lemma: $\|\boldsymbol{u}^{j^+}\| \le (1 - c/\kappa^2)\|\boldsymbol{u}^j\| + 2\|\boldsymbol{u}^{*j}\|$. The proof is provided in Sec. V-F.

Appendix B
Proof of the Corollary 6.1 (noisy LRMC)

We prove the following by directly modifying the noise-free case proof.

Claim B.1. *Let the observed matrix $\boldsymbol{Y} = \boldsymbol{X}^*_\Omega + \boldsymbol{W}_\Omega$ be corrupted by additive noise $\boldsymbol{W}$ such that $|\boldsymbol{W}_{jk}| \le \epsilon_{noise}|\boldsymbol{X}^*_{jk}|$. Assuming also that $p \ge C\kappa^6 r^2\mu\log q\log(1/\epsilon)/n$ and $\epsilon_{noise} \le 1/(\sqrt{r}\kappa^3)$, then, after $T = \kappa^2\log(1/\epsilon_{noise})$ iterations, $\text{SD}_F(\boldsymbol{U}^{(T)}, \boldsymbol{U}^*) \le \kappa^2\epsilon_{noise}\sqrt{r}$.*

A. Proof

Restricting $\epsilon_{noise} \le 1/(\sqrt{r}\kappa^3)$ (as done in the statement of Theorem B.1) ensures that the initialization bound in Lemma 4.1 differs only by a constant factor (see Lemma B.2).

With noise $\boldsymbol{W}_\Omega$, the Least-Squares update (5) changes to

$$\boldsymbol{b}_k - \boldsymbol{g}_k = \boldsymbol{F}_k^{-1}(\boldsymbol{d}_k + \boldsymbol{U}_k^\intercal\boldsymbol{w}_k), \text{ for all } k \in [q],$$

where $\boldsymbol{w}_k$ is the k-th column of $\boldsymbol{W}_\Omega$. Consequently, the bound in (10) changes to $\|\boldsymbol{B} - \boldsymbol{G}\|_F \le \|\boldsymbol{F}^{-1}\|(\|\boldsymbol{D}\|_F + \|\boldsymbol{W}_{\boldsymbol{B}}\|_F)$, where $\boldsymbol{W}_B = \sum_k \boldsymbol{U}_k^\top\boldsymbol{w}_k\boldsymbol{e}_k^\top$. In Lemma B.3, we bound $\|\boldsymbol{W}_{\boldsymbol{B}}\|_F \le 2\sqrt{r}\epsilon_{noise}p\sigma^*_{\max}$, ensuring that $\|\boldsymbol{B} - \boldsymbol{G}\|_F \le \max(\delta^{(t-1)}, 2\sqrt{r}\epsilon_{noise})\sigma^*_{\max}$. Assuming $2\sqrt{r}\epsilon_{noise} \le \delta^{(t-1)}$, Lemmas 4.2 and 4.3 continue to hold. Because $\|\boldsymbol{U}_k^\intercal\boldsymbol{w}_k\| \le 2\epsilon_{noise}p\|\boldsymbol{b}^*_k\|_2$ (Lemma B.3), $\|(\boldsymbol{U}_k^\intercal\boldsymbol{U}_k)^{-1}\boldsymbol{U}_k^\intercal\boldsymbol{w}_k\|_2 \le 2\epsilon_{noise}\|\boldsymbol{b}^*_k\|_2$, which, together with $\|\boldsymbol{b}_k\|_2 \le 1.1\sigma^*_{\max}\mu\sqrt{r/q}$ (Lemma 4.4), bounds the noisy least-squares update $\|\tilde{\boldsymbol{b}}_k\|_2 \le (1.1 + 2\epsilon_{noise})\mu\sigma^*_{\max}\sqrt{r/q}$, thereby proving μ-incoherence in the noisy setting.

With noise, the gradient is $\widetilde{\text{GradU}}$= GradU $- \boldsymbol{W}_{\boldsymbol{U}}$, where $\boldsymbol{W}_{\boldsymbol{U}} = \boldsymbol{W}_\Omega\boldsymbol{B}^\intercal$. We bound $\|\boldsymbol{W}_{\boldsymbol{U}}\|_F \le 2\epsilon_{noise}\sqrt{r}p\sigma^{*2}_{\max}$ in Lemma B.3, and assume $\epsilon_{noise} \le \delta^{(t-1)}/\sqrt{r}$ so that the bound in Lemma 4.5 continues to hold. To prove row-incoherence in the noisy setting, the proof of Lemma 4.6 needs to be only slightly modified. (21) still applies and (22) has the additional row-vector term $\eta p\boldsymbol{w}^j\boldsymbol{B}^{*\intercal}$, which can be bounded $\|\boldsymbol{w}^j\boldsymbol{B}^{*\intercal}\|_2 \le \|\boldsymbol{w}^j\|_2\|\boldsymbol{B}^*\| \le \epsilon_{noise}\|\boldsymbol{x}^{*j}\|_2\sigma^*_{\max} \le \epsilon_{noise}\|\boldsymbol{u}^{*j}\|\sigma^{*2}_{\max}$. This term contributes $\epsilon_{noise}\|\boldsymbol{u}^{*j}\|$in the final bound, that is, $\|\tilde{\boldsymbol{u}}^{+j}\|_2 \le (1 - 0.15/\kappa^2)\|\boldsymbol{u}^j\|_2 + 2\|\boldsymbol{u}^{*j}\|_2 + \epsilon_{noise}\|\boldsymbol{u}^{*j}\|$, which ensures $\kappa^2\mu$ incoherence for the noisy gradient update $\tilde{\boldsymbol{u}}^{+j}$, since $\epsilon_{noise} \le 1$.

The proof details are the same as those for proving Theorem 2.1 given earlier in Sec. IV-C.

B. Noisy case lemmas

Our assumption $|\boldsymbol{W}_{jk}| \le \epsilon_{noise}|\boldsymbol{X}^*_{jk}|$ also implies that

$$\max\Big(\frac{\|\boldsymbol{w}_k\|}{\|\boldsymbol{x}^*_k\|}, \frac{\|\boldsymbol{w}^j\|}{\|\boldsymbol{x}^{*j}\|}\Big) \le \epsilon_{noise}, \ |\boldsymbol{W}_{jk}| \le \epsilon_{noise}\mu^2\sqrt{\frac{r}{q}}\sqrt{\frac{r}{n}}\sigma^*_{\max} \tag{27}$$

This fact is used in proving both lemmas below.

Lemma B.2. *Let the observed matrix $\boldsymbol{Y} = \boldsymbol{X}^*_\Omega + \boldsymbol{W}_\Omega$ be corrupted by additive noise $\boldsymbol{W}$ such that $|\boldsymbol{W}_{jk}| \le \epsilon_{noise}|\boldsymbol{X}^*_{jk}|$. Assuming also that $p \ge C\kappa^6 r^2\mu\log q/n$ and $\epsilon_{noise} \le 1/\sqrt{r}\kappa^3$, then, $\text{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^*) \le c/\kappa^2$, where $\boldsymbol{U}^{(00)} \in \mathbb{R}^{n\times r}$ are the left-singular vectors of $\boldsymbol{Y}$.*

Proof. By Wedin's sin theta theorem (Frob norm version),

$$\begin{aligned}\text{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^\star) &\le C\frac{\|\boldsymbol{Y} - p\boldsymbol{X}^\star\|_F}{\sigma^*_{\min}} \\ &\le C\frac{\|\boldsymbol{X}^\star_\Omega - p\boldsymbol{X}^\star\|_F + \|\boldsymbol{W}_\Omega\|_F}{p\sigma^*_{\min}} \\ &\le C\sqrt{\frac{\kappa^2\mu r^2\log n}{p}} + \frac{\|\boldsymbol{W}_\Omega\|_F}{p\sigma^*_{\min}},\end{aligned} \tag{28}$$

where we have used Lemma 3.21 of [11], with $p \ge C\kappa^6 r^2\mu\log q/n$, to bound $(\|\boldsymbol{X}^*_\Omega - p\boldsymbol{X}^*\|_F/p\sigma^*_{\min}) \le c/\kappa^2$. By using matrix-Bernstein inequality, we can show that $\|\boldsymbol{W}_\Omega\|_F \le 2\epsilon_{noise}\sqrt{r}p\sigma^*_{\max}$ w.h.p.. By the assumed upper bound on ϵ_{noise}, we can then argue that $\mathrm{SD}_F(\boldsymbol{U}^{(00)}, \boldsymbol{U}^*) \leq c/\kappa^2$. The rest of the proof is exactly the same as in the noiseless case (Lemma 4.1). □

Lemma B.3. *Assume that the observed matrix $\boldsymbol{Y} = \boldsymbol{X}^*_\Omega + \boldsymbol{W}_\Omega$ is corrupted by additive noise $\boldsymbol{W}$ such that $|\boldsymbol{W}_{jk}| \leq \epsilon_{noise}|\boldsymbol{X}^*_{jk}|$. Assuming also $\|\boldsymbol{u}^j\| \leq \mu_{\boldsymbol{u}}\sqrt{r/n}$, where $\mu_u = \kappa^2\mu$, $\|\boldsymbol{b}_k\| \leq \sigma^*_{\max}\mu\sqrt{r/q}$ and $\sigma_{\max}(\boldsymbol{B}) \leq C\sigma^*_{\max}$. For $\epsilon_{noise} \leq 1/(\sqrt{r}\kappa^3)$,*

1) $\|\boldsymbol{W}_B - \mathbb{E}[\boldsymbol{W}_{\boldsymbol{B}}]\|_F \leq \epsilon_{noise}\sqrt{r}p\sigma^*_{\max}$, *w.p. greater than* $1 - \exp(\log 2q - np/(\mu^2_{\boldsymbol{u}}r^2))$
2) $\|\boldsymbol{W}_{\boldsymbol{B}}\|_F \leq 2\sqrt{r}\epsilon_{noise}p\sigma^*_{\max}$, *w.p. same as 1).*
3) $\|\boldsymbol{U}_k^\intercal \boldsymbol{w}_k\|_2 \leq 2\epsilon_{noise}p\|\boldsymbol{b}^*_k\|_2$, *w.p. same as 1)*
4) $\|\boldsymbol{W}_{\boldsymbol{U}} - \mathbb{E}[\boldsymbol{W}_{\boldsymbol{U}}]\|_F \leq \epsilon_{noise}\sqrt{r}p\sigma^{*2}_{\max}$, *w.p. greater than* $1 - \exp(\log 2q - np/(\mu^2 r^2))$
5) $\|\boldsymbol{W}_{\boldsymbol{U}}\|_F \leq 2\epsilon_{noise}\sqrt{r}p\sigma^{*2}_{\max}$, *w.p. same as 4).*

Proof. All terms are bounded using (27) and the matrix Bernstein inequality. □

Appendix C
Proof of Corollary 7.1 (Improved guarantee for AltMin for LRMC)

In section 4.2 of [4], it is shown that the least-squares updates for $\boldsymbol{U} \in \mathbb{R}^{n\times r}$ and $\boldsymbol{B} \in \mathbb{R}^{r\times q}$ are equivalent to the following QR-based updates:

$$\boldsymbol{B}^{(t)} = \underset{\boldsymbol{B}}{\operatorname{argmin}}\|\boldsymbol{Y} - \boldsymbol{U}^{(t)}\boldsymbol{B}\|_F^2, \ \ \boldsymbol{B}^{(t)\intercal} \overset{\mathrm{QR}}{=} \boldsymbol{V}^{(t)}\boldsymbol{R}_{\boldsymbol{B}} \tag{29}$$

$$\widetilde{\boldsymbol{U}}^{(t+1)} = \underset{\widetilde{\boldsymbol{U}}}{\operatorname{argmin}}\|\boldsymbol{Y}^\intercal - \boldsymbol{B}^{(t)}\widetilde{\boldsymbol{U}}\|_F^2, \ \ \widetilde{\boldsymbol{U}}^{(t+1)^\intercal} \overset{\mathrm{QR}}{=} \boldsymbol{U}^{(t+1)}\boldsymbol{R}_{\boldsymbol{U}}. \tag{30}$$

1) $\mathbf{B}$*-update* (29)/$\mathrm{SD}_F(\mathbf{V}^{(t)}, \mathbf{V}^*)$ *Bound:* Note that the $\boldsymbol{B}$-update in (29) is exactly the same as the $\boldsymbol{B}$-update for AltGDMin. Therefore, we use Lemma 4.2 to bound $\mathrm{SD}_F(\boldsymbol{V}^{(t)}, \boldsymbol{V}^*)$. Let $\boldsymbol{P}_{*,\perp} :- \mathbf{I} - \boldsymbol{V}^*\boldsymbol{V}^{*\intercal}$. Then,

$$\begin{aligned}\mathrm{SD}_F(\boldsymbol{V}^{(t)}, \boldsymbol{V}^*) = \|\boldsymbol{P}_{*,\perp}\boldsymbol{V}^{(t)}\|_F &= \|\boldsymbol{P}_{*,\perp}\boldsymbol{B}^\intercal\boldsymbol{R}_B^{-1}\|_F \\ &\leq \|\boldsymbol{P}_{*,\perp}\boldsymbol{B}^\intercal\|_F\|\boldsymbol{R}_B^{-1}\| \\ &\leq \|(\boldsymbol{B} - \boldsymbol{G})^\intercal\|_F/\sigma_{\min}(\boldsymbol{B}),\end{aligned} \tag{31}$$

where $\boldsymbol{P}_{*,\perp}\boldsymbol{B}^\intercal = \boldsymbol{P}_{*,\perp}(\boldsymbol{G}^\intercal + (\boldsymbol{B} - \boldsymbol{G})^\intercal) = \boldsymbol{P}_{*,\perp}(\boldsymbol{B} - \boldsymbol{G})^\intercal$, because $\boldsymbol{G}^\intercal = \boldsymbol{X}^{*\intercal}\boldsymbol{U}^\intercal$ and the column-span of $\boldsymbol{X}^{*\intercal}$ is orthogonal to the span of $\boldsymbol{P}_{*,\perp}$. By Lemma 4.2, $\|\boldsymbol{B} - \boldsymbol{G}\|_F \leq \sigma^*_{\min}\delta^{(t-1)}/4$. By Lemma 4.3, assuming $\delta^{(t-1)} \leq 1/(4\kappa)$, $\sigma_{\min}(\boldsymbol{B}) \geq \sigma^*_{\min}/2$. Substituting these bounds in (31) and recalling that $\delta^{(t-1)} :- \mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^*)$, w.p. greater than $1 - \exp(\log q - pn/(r^2\mu^2\kappa^4))$,

$$\mathrm{SD}_F(\boldsymbol{V}^{(t)}, \boldsymbol{V}^*) \leq \frac{1}{2}\mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^*). \tag{32}$$

2) Incoherence of $\mathbf{V}^{(t)}$*:* Recall that $\widetilde{\boldsymbol{B}}^\intercal \overset{\mathrm{QR}}{=} \boldsymbol{V}\boldsymbol{R}$. Then, the k-th row, $\|\boldsymbol{v}^k\|_2 \leq \|\boldsymbol{b}_k\|_2/\sigma^*_{\min}(\boldsymbol{B}) \leq C\kappa\mu\sqrt{r/q}$, where the last inequality follows from Lemmas 4.3 and 4.4. Thus, $\boldsymbol{V}$ is $\kappa\mu$-incoherent.

3) $\mathbf{U}$*-Update* (30)/$\mathrm{SD}_F(\mathbf{U}^{(\mathbf{t+1})}, \mathbf{U}^*)$ *Bound and Incoherence of* $\mathbf{U}^{(t+1)}$*:* The two AltMin steps are symmetric, so arguments analogous to the above help show that

$$\mathrm{SD}_F(\boldsymbol{U}^{(t+1)}, \boldsymbol{U}^*) \leq \frac{1}{2}\mathrm{SD}_F(\boldsymbol{V}^{(t)}, \boldsymbol{U}^*). \tag{33}$$

Combining (32) and (33), we have $\mathrm{SD}_F(\boldsymbol{U}^{(t+1)}, \boldsymbol{U}^*) \leq \mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^*)/4$. For initialization we use the few lines of our Algorithm 1 and Lemma 4.1. Consequently, AltMinComplete [4] initialized as given in the first few lines of our Algorithm 1 needs $T = \log(1/\epsilon)$ iterations for $\mathrm{SD}_F(\boldsymbol{U}^{(t)}, \boldsymbol{U}^*) \leq \epsilon$.

References

[1] A. A. Abbasi, S. Moothedath, and N. Vaswani, "Fast federated low rank matrix completion," in *2023 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2023, pp. 1–6.

[2] E. J. Candes and B. Recht, "Exact matrix completion via convex optimization," *Found. of Comput. Math*, no. 9, pp. 717–772, 2008.

[3] R. Keshavan, A. Montanari, and S. Oh, "Matrix completion from a few entries," *IEEE Trans. Info. Th.*, vol. 56, no. 6, pp. 2980–2998, 2010.

[4] P. Netrapalli, P. Jain, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Annual ACM Symp. on Th. of Comp. (STOC)*, 2013.

[5] R. Sun and Z.-Q. Luo, "Guaranteed matrix completion via non-convex factorization," *IEEE Trans. Info. Th.*, vol. 62, no. 11, pp. 6535–6579, 2016.

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

[7] C. Ma, K. Wang, Y. Chi, and Y. Chen, "Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval, matrix completion and blind deconvolution," in *Intl. Conf. Machine Learning (ICML)*, 2018.

[8] M. Hardt and M. Wootters, "Fast matrix completion without the condition number," in *Conf. on Learning Theory*, 2014.

[9] Q. Zheng and J. Lafferty, "Convergence analysis for rectangular matrix completion using burer-monteiro factorization and gradient descent," *arXiv preprint arXiv:1605.07051*, 2016.

[10] X. Yi, D. Park, Y. Chen, and C. Caramanis, "Fast algorithms for robust pca via gradient descent," in *Neur. Info. Proc. Sys. (NeurIPS)*, 2016.

[11] Y. Chen, Y. Chi, J. Fan, and C. Ma, "Spectral methods for data science: A statistical perspective," *arXiv preprint arXiv:2012.08496*, 2020.

[12] S. Nayer and N. Vaswani, "Sample-efficient low rank phase retrieval," *IEEE Trans. Info. Th.*, Dec. 2021.

[13] ——, "Fast and sample-efficient federated low rank matrix recovery from column-wise linear and quadratic projections," *IEEE Trans. Info. Th.*, February 2023 (on arXiv:2102.10217 since Feb. 2021).

[14] M. Hardt, "Understanding alternating minimization for matrix completion," in *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE, 2014, pp. 651–660.

[15] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff, "Fast approximation of matrix coherence and statistical leverage," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 3475–3506, 2012.

[16] S. Ubaru, A. Mazumdar, and Y. Saad, "Low rank approximation using error correcting coding matrices," in *International Conference on Machine Learning*. PMLR, 2015, pp. 702–710.

[17] Z. Li, B. Ding, C. Zhang, N. Li, and J. Zhou, "Federated matrix factorization with privacy guarantee," *Proceedings of the VLDB Endowment*, vol. 15, no. 4, 2021.

[18] P. Jain and P. Netrapalli, "Fast exact matrix completion with finite samples," in *Conf. on Learning Theory*, 2015, pp. 1007–1034.

[19] M. Fazel, "Matrix rank minimization with applications," *PhD thesis, Stanford Univ*, 2002.

[20] N. Vaswani, "Efficient federated low rank matrix recovery via alternating gd and minimization: A simple proof," *IEEE Trans. Info. Th.*, pp. 5162 – 5167, July 2024.

[21] L. W. Mackey, A. Talwalkar, and M. I. Jordan, "Distributed matrix completion and robust factorization," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 913–960, 2015.

[22] C. Teflioudi, F. Makari, and R. Gemulla, "Distributed matrix completion," in *2012 ieee 12th international conference on data mining*. IEEE, 2012, pp. 655–664.

[23] V. W. Anelli, Y. Deldjoo, T. Di Noia, A. Ferrara, and F. Narducci, "User-controlled federated matrix factorization for recommender systems," *Journal of Intelligent Information Systems*, vol. 58, no. 2, pp. 287–309, 2022.

[24] X. He, Q. Ling, and T. Chen, "Byzantine-robust stochastic gradient descent for distributed low-rank matrix completion," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 322–326.

[25] M. Pilanci, "Information-theoretic bounds on sketching," *Information-Theoretic Methods in Data Science*, p. 104, 2021.

[26] A. Ghosh, R. K. Maity, and A. Mazumdar, "Distributed newton can communicate less and resist byzantine workers," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 028–18 038, 2020.

[27] S. Wang, F. Roosta, P. Xu, and M. W. Mahoney, "Giant: Globally improved approximate newton method for distributed optimization," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[28] R. Ward and T. Kolda, "Convergence of alternating gradient descent for matrix factorization," *Advances in Neural Information Processing Systems*, vol. 36, pp. 22 369–22 382, 2023.

[29] M. Hardt and E. Price, "The noisy power method: A meta algorithm with applications," in *Neur. Info. Proc. Sys. (NeurIPS)*, 2014, pp. 2861–2869.

[30] S. Babu, S. G. Lingala, and N. Vaswani, "Fast low rank compressive sensing for accelerated dynamic MRI," *IEEE Trans. Comput. Imag*, 2023.

[31] S. Nayer, P. Narayanamurthy, and N. Vaswani, "Provable low rank phase retrieval," *IEEE Trans. Info. Th.*, March 2020.

[32] A. Khaled, K. Mishchenko, and P. Richtárik, "Tighter theory for local sgd on identical and heterogeneous data," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4519–4529.

[33] S. U. Stich, "Local SGD converges fast and communicates little," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=S1g2JnRcFX

[34] R. Vershynin, *High-dimensional probability: An introduction with applications in data science*. Cambridge University Press, 2018, vol. 47.

[35] J. A. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Trans. Info. Th.*, pp. 1030–1051, March 2006.

[36] Y. Chen, Y. Chi, J. Fan, and C. Ma, "Spectral methods for data science: A statistical perspective," *Foundations and Trends® in Machine Learning*, vol. 14, no. 5, p. 566–806, 2021. [Online]. Available: http://dx.doi.org/10.1561/2200000079

[37] M. Nashed, "A decomposition relative to convex sets," *Proceedings of the American Mathematical Society*, vol. 19, no. 4, pp. 782–786, 1968.

[38] E. H. Zarantonello, “Projections on convex sets in hilbert space and spectral theory: Part i. projections on convex sets: Part ii. spectral theory,” in *Contributions to nonlinear functional analysis*. Elsevier, 1971, pp. 237–424.

[39] Y. Chen and E. Candes, “Solving random quadratic systems of equations is nearly as easy as solving linear systems,” in *Neur. Info. Proc. Sys. (NeurIPS)*, 2015, pp. 739–747.

[40] A. P. Singh and N. Vaswani, “Noisy low rank column-wise sensing,” *arXiv preprint arXiv:2409.08384*, 2024.

[41] A. A. Abbasi, S. Moothedath, and N. Vaswani, “Fast federated low rank matrix completion,” in *2023 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2023, pp. 1–6.

Ahmed Ali Abbasi received the B.E. degree in electrical engineering from the National University of Sciences and Technology, Islamabad, Pakistan in 2017 and an M.S. degree in electrical engineering from Tufts University in 2022. At Tufts, he was awarded the Greenough Graduate Fellowship in 2018 for standing first in the PhD qualification exam, and was also one of three inaugural recipients of the Tufts Tripods Fellowship in 2019. He is currently Professor Namrata Vaswani’s PhD student at Iowa State University Ames, IA, USA.

Namrata Vaswani (Email: namrata@iastate.edu) is a Professor of Electrical and Computer Engineering, and the Anderlik Professor of Engineering at Iowa State University. She received a Ph.D. in 2004 from the University of Maryland, College Park and a B.Tech. from Indian Institute of Technology (IIT-Delhi) in India in 1999. Her research interests lie in statistical machine learning and signal processing, and their applications in medical imaging and video. Vaswani is also the founder and director of the CyMath K-12 (school) math tutoring and mentoring program at Iowa State. Vaswani has served as an Area Editor for IEEE Signal Processing Magazine, and an Associate Editor for the IEEE Transactions on Information Theory and the IEEE Transactions on Signal Processing, and has guest-edited a special issue for the Proceedings of the IEEE. She is a recipient of the IEEE Signal Processing Society (SPS) Best Paper Award (2014), the University of Maryland ECE Distinguished Alumni Award (2019) and the Iowa State Mid-Career Achievement in Research Award (2019). Vaswani is Fellow of the AAAS (2023) and a Fellow of the IEEE (2019).