J Math Imaging Vis (2014) 48:235–265
DOI 10.1007/s10851-013-0434-7

Matrix Recipes for Hard Thresholding Methods

Anastasios Kyrillidis · Volkan Cevher

Published online: 6 April 2013

Abstract In this paper, we present and analyze a new set of low-rank recovery algorithms for linear inverse problems within the class of hard thresholding methods. We provide strategies on how to set up these algorithms via basic ingredients for different configurations to achieve complexity vs. accuracy tradeoffs. Moreover, we study acceleration schemes via memory-based techniques and randomized, ϵ-approximate matrix projections to decrease the computational costs in the recovery process. For most of the configurations, we present theoretical analysis that guarantees convergence under mild problem conditions. Simulation results demonstrate notable performance improvements as compared to state-of-the-art algorithms both in terms of reconstruction accuracy and computational complexity.

Keywords Affine rank minimization · Hard thresholding · ϵ-approximation schemes · Randomized algorithms

1 Introduction

In this work, we consider the general affine rank minimization (ARM) problem, described as follows:

THE ARM PROBLEM: *Assume $\boldsymbol{X}^* \in \mathbb{R}^{m\times n}$ is a rank-k matrix of interest ($k \ll \min\{m,n\}$) and let $\boldsymbol{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ be a known linear operator. Given a set of observations as $\boldsymbol{y} = \boldsymbol{\mathcal{A}}\boldsymbol{X}^* + \boldsymbol{\varepsilon} \in \mathbb{R}^p$, we desire to recover $\boldsymbol{X}^*$ from $\boldsymbol{y}$ in a scalable and robust manner.*

The challenge in this problem is to recover the true low-rank matrix in subsampled settings where $p \ll m \cdot n$. In such cases, we typically exploit the prior information that $\boldsymbol{X}^*$ is low-rank and thus, we are interested in finding a matrix $\boldsymbol{X}$ of rank at most k that minimizes the data error $f(\boldsymbol{X}) := \|\boldsymbol{y} - \boldsymbol{\mathcal{A}}\boldsymbol{X}\|_2^2$ as follows:

$$\begin{aligned} \underset{\boldsymbol{X}\in\mathbb{R}^{m\times n}}{\text{minimize}} \quad & f(\boldsymbol{X}) \\ \text{subject to} \quad & \operatorname{rank}(\boldsymbol{X}) \le k. \end{aligned} \tag{1}$$

The ARM problem appears in many applications; low dimensional embedding [1], matrix completion [2], image compression [3], function learning [4, 5] just to name a few. We present below important ARM problem cases, as characterized by the nature of the linear operator $\boldsymbol{\mathcal{A}}$.

General linear maps: In many ARM problem cases, $\boldsymbol{\mathcal{A}}$ or $\boldsymbol{\mathcal{A}}^*$ has a dense range, satisfying specific incoherence or restricted isometry properties (discussed later in the paper); here, $\boldsymbol{\mathcal{A}}^*$ is the adjoint operator of $\boldsymbol{\mathcal{A}}$. In Quantum Tomography, [6] studies the Pauli operator, a *compressive* linear map $\boldsymbol{\mathcal{A}}$ that consists of the Kronecker product of 2×2 matrices and obeys restricted isometry properties, defined later in the paper. Furthermore, recent developments indicate connections of ridge function learning [4, 7] and phase retrieval [8] with the ARM problem where $\boldsymbol{\mathcal{A}}$ is a Bernoulli and a Fourier operator, respectively.

Matrix Completion (MC): Let Ω be the set of ordered pairs that represent the coordinates of the observable entries in $\boldsymbol{X}^*$. Then, the set of observations satisfy $\boldsymbol{y} = \boldsymbol{\mathcal{A}}_\Omega \boldsymbol{X}^* + \boldsymbol{\varepsilon}$ where $\boldsymbol{\mathcal{A}}_\Omega$ defines a linear mask over the observable entries Ω. To solve the MC problem, a potential criterion is given by (1) [2]. As a motivating example, consider the famous

A. Kyrillidis (✉) · V. Cevher
Laboratory for Information and Inference Systems, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland
e-mail: anastasios.kyrillidis@epfl.ch
url: http://lions.epfl.ch

V. Cevher
e-mail: volkan.cevher@epfl.ch

Netflix problem [9], a recommender system problem where users' movie preferences are inferred by a limited subset of entries in a database.

Principal Component Analysis: In Principal Component Analysis (PCA), we are interested in identifying a low rank subspace that best explains the data in the Euclidean sense from the observations $\boldsymbol{y} = \mathcal{A}\boldsymbol{X}^*$ where $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ is an identity linear map that stacks the columns of the matrix $\boldsymbol{X}^*$ into a single column vector with $p = m \cdot n$. We observe that the PCA problem falls under the ARM criterion in (1). While (1) is generally NP-hard to solve optimally, PCA can be solved in polynomial time using the truncated Singular Value Decomposition (SVD) of $\mathcal{A}^*\boldsymbol{y}$. As an extension to the PCA setting, [10] considers the Robust PCA problem where $\boldsymbol{y}$ is further corrupted by gross sparse noise. We extend the framework proposed in this paper for the RPCA case and its generalizations in [11].

For the rest of the paper, we consider only the low rank estimation case in (1). As running test cases to support our claims, we consider the MC setting as well as the general ARM setting where $\mathcal{A}$ is constituted by permuted subsampled noiselets [12].

1.1 Two Camps of Recovery Algorithms

Convex relaxations: In [13], the authors study the nuclear norm $\|\boldsymbol{X}\|_* := \sum_{i=1}^{\text{rank}(\boldsymbol{X})} \sigma_i$ as a convex surrogate of $\text{rank}(\boldsymbol{X})$ operator so that we can leverage convex optimization approaches, such as interior-point methods—here, σ_i denotes the i-th singular value of $\boldsymbol{X}$. Under basic incoherence properties of the sensing linear mapping $\mathcal{A}$, [13] provides provable guarantees for unique low rank matrix recovery using the nuclear norm.

Once (1) is relaxed to a convex problem, decades of knowledge on convex analysis and optimization can be leveraged. Interior point methods find a solution with fixed precision in polynomial time but their complexity might be prohibitive even for moderate-sized problems [14, 15]. More suitable for large-scale data analysis, first-order methods constitute low-complexity alternatives but most of them introduce complexity vs. accuracy tradeoffs [16–19].

Non-convex approaches: In contrast to the convex relaxation approaches, iterative greedy algorithms maintain the nonconvex nature of (1). Unfortunately, solving (1) optimally is in general NP-hard [20]. Due to this computational intractability, the algorithms in this class greedily refine a rank-k solution using only "local" information available at the current iteration [21–23].

1.2 Contributions

In this work, we study a special class of iterative greedy algorithms known as hard thresholding methods. Similar results have been derived for the vector case [24]. Note that the transition from sparse vector approximation to ARM is *non-trivial*; while s-sparse signals "live" in the union of finite number of subspaces, the set of rank-k matrices expands to infinitely many subspaces. Thus, the selection rules do not generalize in a straightforward way.

Our contributions are the following:

Ingredients of hard thresholding methods: We analyze the behaviour and performance of hard thresholding methods from a global perspective. Five building blocks are studied: (i) step size selection μ_i, (ii) gradient or least-squares updates over restricted low-rank subspaces (e.g., adaptive block coordinate descent), (iii) memory exploitation, (iv) active low-rank subspace tracking and, (v) low-rank matrix approximations (described next). We highlight the impact of these key pieces on the convergence rate and signal reconstruction performance and provide optimal and/or efficient strategies on how to set up these ingredients under different problem conditions.

Low-rank matrix approximations in hard thresholding methods: In [25], the authors show that the solution efficiency can be significantly improved by ϵ-approximation algorithms. Based on similar ideas, we analyze the impact of ϵ-approximate low rank-revealing schemes in the proposed algorithms with well-characterized time and space complexities. Moreover, we provide extensive analysis to prove convergence using ϵ-approximate low-rank projections.

Hard thresholding-based framework with improved convergence conditions: We study hard thresholding variants that provide salient computational tradeoffs for the class of greedy methods on low-rank matrix recovery. These methods, as they iterate, exploit the non-convex scaffold of low rank subspaces on which the approximation problem resides. Using simple analysis tools, we derive improved conditions that guarantee convergence, compared to state-of-the-art approaches.

The organization of the paper is as follows. In Sect. 2, we set up the notation and provide some definitions and properties, essential for the rest of the paper. In Sect. 3, we describe the basic algorithmic frameworks in a nutshell, while in Sect. 4 we provide important "ingredients" for the class of hard-thresholding methods; detailed convergence analysis proofs are provided in Sect. 5. The complexity analysis of the proposed algorithms is provided in Sect. 6. We study two acceleration schemes in Sects. 7 and 8, based on memory utilization and ϵ-approximate low-rank projections, respectively. We further improve convergence speed by exploiting randomized low rank projections in Sect. 9, based on power iteration-based subspace finder tools [26]. We provide empirical support for our claims through experimental results on synthetic and real data in Sect. 10. Finally, we conclude with future work directions in Sect. 11.

2 Elementary Definitions and Properties

We reserve lower-case and bold lower-case letters for scalar and vector variable representation, respectively. Bold upper-case letters denote matrices while bold calligraphic upper-case letters represent linear operators. We use calligraphic upper-case letters for set representations. We use $\boldsymbol{X}(i)$ to represent the matrix estimate at the i-th iteration.

The rank of $\boldsymbol{X}$ is denoted as $\text{rank}(\boldsymbol{X}) \leq \min\{m,n\}$. The empirical data error is denoted as $f(\boldsymbol{X}) := \|\boldsymbol{y} - \boldsymbol{\mathcal{A}}\boldsymbol{X}\|_2^2$ with gradient $\nabla f(\boldsymbol{X}) := -2\boldsymbol{\mathcal{A}}^*(\boldsymbol{y} - \boldsymbol{\mathcal{A}}\boldsymbol{X})$, where * is the adjoint operation over the linear mapping $\boldsymbol{\mathcal{A}}$. The inner product between matrices $\boldsymbol{A},\ \boldsymbol{B} \in \mathbb{R}^{m\times n}$ is denoted as $\langle \boldsymbol{A}, \boldsymbol{B}\rangle = \text{trace}(\boldsymbol{B}^T\boldsymbol{A})$, where ^{T} represents the transpose operation. $\mathbf{I}$ represents an identity matrix with dimensions apparent from the context.

Let $\mathcal{S}$ be a set of orthonormal, rank-1 matrices that span an arbitrary subspace in $\mathbb{R}^{m\times n}$. We reserve $\text{span}(\mathcal{S})$ to denote the subspace spanned by $\mathcal{S}$. With slight abuse of notation, we use:

$$\text{rank}\big(\text{span}(\mathcal{S})\big) \equiv \max_{\boldsymbol{X}} \big\{\text{rank}(\boldsymbol{X}) :\ \boldsymbol{X} \in \text{span}(\mathcal{S})\big\}, \tag{2}$$

to denote the *maximum* rank a matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ can have such that $\boldsymbol{X}$ lies in the subspace spanned by the set $\mathcal{S}$. Given a finite set $\mathcal{S}$, $|\mathcal{S}|$ denotes the cardinality of $\mathcal{S}$. For any matrix $\boldsymbol{X}$, we use $R(\boldsymbol{X})$ to denote its range.

We define a *minimum cardinality* set of orthonormal, rank-1 matrices that span the subspace induced by a set of rank-1 (and possibly non-orthogonal) matrices $\mathcal{S}$ as:

$$\text{ortho}(\mathcal{S}) \in \arg\min_{\mathcal{T}} \big\{|\mathcal{T}| : \mathcal{T} \subseteq \mathcal{U} \text{ s.t. } \text{span}(\mathcal{T}) = \text{span}(\mathcal{S})\big\},$$

where $\mathcal{U}$ denotes the superset that includes all the sets of *orthonormal*, rank-1 matrices in $\mathbb{R}^{m\times n}$ such that $\langle \boldsymbol{T}_i, \boldsymbol{T}_j\rangle = 0,\ i \neq j,\ \forall \boldsymbol{T}_i,\ \boldsymbol{T}_j \in \mathcal{T}$ and, $\|\boldsymbol{T}_i\|_F = 1,\ \forall i$. In general, $\text{ortho}(\mathcal{S})$ is not unique.

A well-known lemma used in the convergence rate proofs of this class of greedy hard thresholding algorithms is defined next.

Lemma 1 [27] *Let $\mathcal{J} \subseteq \mathbb{R}^{m\times n}$ be a closed convex set and $f : \mathcal{J} \to \mathbb{R}$ be a smooth objective function defined over $\mathcal{J}$. Let $\boldsymbol{X}^* \in \mathcal{J}$ be a local minimum of the objective function f over the set $\mathcal{J}$. Then*

$$\big\langle \nabla f\big(\boldsymbol{X}^*\big), \boldsymbol{X} - \boldsymbol{X}^*\big\rangle \geq 0, \quad \forall \boldsymbol{X} \in \mathcal{J}. \tag{3}$$

2.1 Singular Value Decomposition (SVD) and Its Properties

Definition 1 [SVD] Let $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ be a rank-l ($l < \min\{m,n\}$) matrix. Then, the SVD of $\boldsymbol{X}$ is given by:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \big[\boldsymbol{U}_\alpha\ \boldsymbol{U}_\beta\big]\begin{bmatrix}\widetilde{\boldsymbol{\Sigma}} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{V}_\alpha^T\\ \boldsymbol{V}_\beta^T\end{bmatrix}, \tag{4}$$

where $\boldsymbol{U}_\alpha \in \mathbb{R}^{m\times l}$, $\boldsymbol{U}_\beta \in \mathbb{R}^{m\times(m-l)}$, $\boldsymbol{V}_\alpha \in \mathbb{R}^{n\times l}$, $\boldsymbol{V}_\beta \in \mathbb{R}^{n\times(n-l)}$ and $\widetilde{\boldsymbol{\Sigma}} = \text{diag}(\sigma_1,\ldots,\sigma_l) \in \mathbb{R}^{l\times l}$ for $\sigma_1,\ldots,\sigma_l \in \mathbb{R}_+$. Here, the columns of $\boldsymbol{U}, \boldsymbol{V}$ represent the set of left and right singular vectors, respectively, and $\sigma_1,\ldots,\sigma_l$ denote the singular values.

For any matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ with arbitrary $\text{rank}(\boldsymbol{X}) \leq \min\{m,n\}$, its best orthogonal projection $\mathcal{P}_k(\boldsymbol{X})$ onto the set of rank-k ($k < \text{rank}(\boldsymbol{X})$) matrices $\mathcal{C}_k := \{\boldsymbol{A} \in \mathbb{R}^{m\times n} : \text{rank}(\boldsymbol{A}) \leq k\}$ defines the optimization problem:

$$\mathcal{P}_k(\boldsymbol{X}) \in \arg\min_{\boldsymbol{Y}\in\mathcal{C}_k} \|\boldsymbol{Y} - \boldsymbol{X}\|_F. \tag{5}$$

According to the Eckart-Young theorem [28], the best rank-k approximation of a matrix $\boldsymbol{X}$ corresponds to its truncated SVD: if $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, then $\mathcal{P}_k(\boldsymbol{X}) := \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{V}_k^T$ where $\boldsymbol{\Sigma}_k \in \mathbb{R}^{k\times k}$ is a diagonal matrix that contains the first k diagonal entries of $\boldsymbol{\Sigma}$ and $\boldsymbol{U}_k, \boldsymbol{V}_k$ contain the corresponding left and right singular vectors, respectively. Moreover, this projection is not always unique. In the case of multiple identical singular values, the lexicographic approach is used to break ties. In any case, $\|\mathcal{P}_k(\boldsymbol{X}) - \boldsymbol{X}\|_F \leq \|\boldsymbol{W} - \boldsymbol{X}\|_F$ for any rank-k $\boldsymbol{W} \in \mathbb{R}^{m\times n}$.

2.2 Subspace Projections

Given a set of orthonormal, rank-1 matrices $\mathcal{S}$, we denote the orthogonal projection operator onto the subspace induced by $\mathcal{S}$ as $\mathcal{P}_{\mathcal{S}}$[1] which is an idempotent linear transformation; furthermore, we denote the orthogonal projection operator onto the orthogonal subspace of $\mathcal{S}$ as $\mathcal{P}_{\mathcal{S}^\perp}$. We can always decompose a matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ into two matrix components, as follows:

$$\boldsymbol{X} := \mathcal{P}_{\mathcal{S}}\boldsymbol{X} + \mathcal{P}_{\mathcal{S}^\perp}\boldsymbol{X}, \quad \text{such that } \langle \mathcal{P}_{\mathcal{S}}\boldsymbol{X}, \mathcal{P}_{\mathcal{S}^\perp}\boldsymbol{X}\rangle = 0.$$

If $\boldsymbol{X} \in \text{span}(\mathcal{S})$, the best projection of $\boldsymbol{X}$ onto the subspace induced by $\mathcal{S}$ is the matrix $\boldsymbol{X}$ itself. Moreover, $\|\mathcal{P}_{\mathcal{S}}\boldsymbol{X}\|_F \leq \|\boldsymbol{X}\|_F$ for any $\mathcal{S}$ and $\boldsymbol{X}$.

Definition 2 [Orthogonal projections using SVD] Let $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ be a matrix with arbitrary rank and SVD decomposition given by (4). Then, $\mathcal{S} := \{\boldsymbol{u}_i\boldsymbol{v}_i^T : i = 1,\ldots,k\}$ ($k \leq \text{rank}(\boldsymbol{X})$) constitutes a set of orthonormal, rank-1 matrices that spans the best k-rank subspace in $R(\boldsymbol{X})$ and $R(\boldsymbol{X}^T)$; here, $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ denote the i-th left and right singular vectors, respectively. The orthogonal projection onto this subspace is given by [2]:

$$\mathcal{P}_{\mathcal{S}}\boldsymbol{X} = \mathcal{P}_{\mathcal{U}}\boldsymbol{X} + \boldsymbol{X}\mathcal{P}_{\mathcal{V}} - \mathcal{P}_{\mathcal{U}}\boldsymbol{X}\mathcal{P}_{\mathcal{V}}, \tag{6}$$

[1] The distinction between $\mathcal{P}_{\mathcal{S}}$ and $\mathcal{P}_k$ for k positive integer is apparent from context.

where $\mathcal{P}_{\mathcal{U}} = \boldsymbol{U}_{:,1:k}\boldsymbol{U}_{:,1:k}^T$ and $\mathcal{P}_{\mathcal{V}} = \boldsymbol{V}_{:,1:k}\boldsymbol{V}_{:,1:k}^T$ in MATLAB notation. Moreover, the orthogonal projection onto the $\mathcal{S}^{\perp}$ is given by:

$$\mathcal{P}_{\mathcal{S}^{\perp}}\boldsymbol{X} = \boldsymbol{X} - \mathcal{P}_{\mathcal{S}}\boldsymbol{X}. \tag{7}$$

In the algorithmic descriptions, we use $\mathcal{S} \leftarrow \mathcal{P}_k(\boldsymbol{X})$ to denote the set of rank-1, orthonormal matrices as outer products of the k left $\boldsymbol{u}_i$ and right $\boldsymbol{v}_i$ principal singular vectors of $\boldsymbol{X}$ that span the best rank-k subspace of $\boldsymbol{X}$; e.g. $\mathcal{S} = \{\boldsymbol{u}_i\boldsymbol{v}_i,\ i = 1,\ldots,k\}$. Moreover, $\widehat{\boldsymbol{X}} \leftarrow \mathcal{P}_k(\boldsymbol{X})$ denotes a/the best rank-k projection matrix of $\boldsymbol{X}$. In some cases, we use $\{\mathcal{S},\ \widehat{\boldsymbol{X}}\} \leftarrow \mathcal{P}_k(\boldsymbol{X})$ when we compute both. The distiction between these cases is apparent from the context.

2.3 Restricted Isometry Property

Many conditions have been proposed in the literature to establish solution uniqueness and recovery stability such as null space property [29], exact recovery condition [30], etc. For the matrix case, [13] proposed the *restricted isometry property* (RIP) for the ARM problem.

Definition 3 [Rank Restricted Isometry Property (R-RIP) for matrix linear operators [13]] A linear operator $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ satisfies the R-RIP with constant $\delta_k(\mathcal{A}) \in (0, 1)$ if and only if:

$$\big(1 - \delta_k(\mathcal{A})\big)\|\boldsymbol{X}\|_F^2 \le \|\mathcal{A}\boldsymbol{X}\|_2^2 \le \big(1 + \delta_k(\mathcal{A})\big)\|\boldsymbol{X}\|_F^2, \tag{8}$$

$\forall \boldsymbol{X} \in \mathbb{R}^{m\times n}$ such that $\operatorname{rank}(\boldsymbol{X}) \le k$. We write δ_k to mean $\delta_k(\mathcal{A})$, unless otherwise stated.

[6] shows that Pauli operators satisfy the rank-RIP in compressive settings while, in function learning, the linear map $\mathcal{A}$ is designed specifically to satisfy the rank-RIP [7].

2.4 Some Useful Bounds Using R-RIP

In this section, we present some lemmas that are useful in our subsequent developments—these lemmas are consequences of the R-RIP of $\mathcal{A}$.

Lemma 2 [21] *Let $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ be a linear operator that satisfies the R-RIP with constant δ_k. Then, $\forall \boldsymbol{v} \in \mathbb{R}^p$, the following holds true*:

$$\big\|\mathcal{P}_{\mathcal{S}}\big(\mathcal{A}^*\boldsymbol{v}\big)\big\|_F \le \sqrt{1+\delta_k}\|\boldsymbol{v}\|_2, \tag{9}$$

where $\mathcal{S}$ is a set of orthonormal, rank-1 matrices in $\mathbb{R}^{m\times n}$ such that $\operatorname{rank}(\mathcal{P}_{\mathcal{S}}\boldsymbol{X}) \le k$, $\forall \boldsymbol{X} \in \mathbb{R}^{m\times n}$.

Lemma 3 [21] *Let $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ be a linear operator that satisfies the R-RIP with constant δ_k. Then, $\forall \boldsymbol{X} \in \mathbb{R}^{m\times n}$, the following holds true*:

$$\begin{aligned}(1-\delta_k)\|\mathcal{P}_{\mathcal{S}}\boldsymbol{X}\|_F &\le \|\mathcal{P}_{\mathcal{S}}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}}\boldsymbol{X}\|_F \\ &\le (1+\delta_k)\|\mathcal{P}_{\mathcal{S}}\boldsymbol{X}\|_F,\end{aligned} \tag{10}$$

where $\mathcal{S}$ is a set of orthonormal, rank-1 matrices in $\mathbb{R}^{m\times n}$ such that $\operatorname{rank}(\mathcal{P}_{\mathcal{S}}\boldsymbol{X}) \le k$, $\forall \boldsymbol{X} \in \mathbb{R}^{m\times n}$.

Lemma 4 [22] *Let $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ be a linear operator that satisfies the R-RIP with constant δ_k and $\mathcal{S}$ be a set of orthonormal, rank-1 matrices in $\mathbb{R}^{m\times n}$ such that* $\operatorname{rank}(\mathcal{P}_{\mathcal{S}}\boldsymbol{X}) \le k$, $\forall \boldsymbol{X} \in \mathbb{R}^{m\times n}$. *Then, for $\mu > 0$, $\mathcal{A}$ satisfies*:

$$\lambda\big(\mu\mathcal{P}_{\mathcal{S}}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}}\big) \in \big[\mu(1-\delta_k), \mu(1+\delta_k)\big], \tag{11}$$

where $\lambda(\mathcal{B})$ represents the range of eigenvalues of the linear operator $\mathcal{B} : \mathbb{R}^p \to \mathbb{R}^{m\times n}$. Moreover, $\forall \boldsymbol{X} \in \mathbb{R}^{m\times n}$, it follows that:

$$\begin{aligned}&\big\|\big(\mathbf{I} - \mu\mathcal{P}_{\mathcal{S}}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}}\big)\mathcal{P}_{\mathcal{S}}\boldsymbol{X}\big\|_F \\ &\quad\le \max\big\{\mu(1+\delta_k) - 1, 1 - \mu(1-\delta_k)\big\}\|\mathcal{P}_{\mathcal{S}}\boldsymbol{X}\|_F.\end{aligned} \tag{12}$$

Lemma 5 [22] *Let $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ be a linear operator that satisfies the R-RIP with constant δ_k and $\mathcal{S}_1, \mathcal{S}_2$ be two sets of orthonormal, rank-1 matrices in $\mathbb{R}^{m\times n}$ such that*

$$\operatorname{rank}(\mathcal{P}_{\mathcal{S}_1\cup\mathcal{S}_2}\boldsymbol{X}) \le k, \quad \forall \boldsymbol{X} \in \mathbb{R}^{m\times n}. \tag{13}$$

Then, the following inequality holds:

$$\big\|\mathcal{P}_{\mathcal{S}_1}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_1^{\perp}}\boldsymbol{X}\big\|_F \le \delta_k\|\mathcal{P}_{\mathcal{S}_1^{\perp}}\boldsymbol{X}\|_F, \quad \forall \boldsymbol{X} \in \operatorname{span}(\mathcal{S}_2). \tag{14}$$

3 Algebraic Pursuits in a Nutshell

Explicit descriptions of the proposed algorithms are provided in Algorithms 1 and 2. Algorithm 1 follows from the ALgrebraic PursuitS (ALPS) scheme for the vector case [31]. MATRIX ALPS I provides efficient strategies for adaptive step size selection and additional signal estimate updates at each iteration (these motions are explained in detail in the next subsection). Algorithm 2 (ADMiRA) [21] further improves the performance of Algorithm 1 by introducing least squares optimization steps on restricted subspaces—this technique borrows from a series of vector reconstruction algorithms such as CoSaMP [32], Subspace Pursuit (SP) [33] and Hard Thresholding Pursuit (HTP) [34].

In a nutshell, both algorithms simply seek to improve the subspace selection by iteratively collecting an extended subspace $\mathcal{S}_i$ with $\operatorname{rank}(\operatorname{span}(\mathcal{S}_i)) \le 2k$ and then finding the rank-k matrix that fits the measurements in this restricted subspace using least squares or gradient descent motions.

Algorithm 1 MATRIX ALPS I

Input: $\boldsymbol{y}$, $\mathcal{A}$, k, Tolerance η, MaxIterations
Initialize: $\boldsymbol{X}(0) \leftarrow 0$, $\mathcal{X}_0 \leftarrow \{\emptyset\}$, $i \leftarrow 0$
repeat

1: $\mathcal{D}_i \leftarrow \mathcal{P}_k(\mathcal{P}_{\mathcal{X}_i^\perp} \nabla f(\boldsymbol{X}(i)))$ *(Best rank-k subspace orthogonal to $\mathcal{X}_i$)*
2: $\mathcal{S}_i \leftarrow \mathcal{D}_i \cup \mathcal{X}_i$ *(Active subspace expansion)*
3: $\mu_i \leftarrow \arg\min_\mu \|\boldsymbol{y} - \mathcal{A}(\boldsymbol{X}(i) - \frac{\mu}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i)))\|_2^2 = \frac{\|\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))\|_F^2}{\|\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))\|_2^2}$ *(Step size selection)*
4: $\boldsymbol{V}(i) \leftarrow \boldsymbol{X}(i) - \frac{\mu_i}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))$ *(Error norm reduction via gradient descent)*
5: $\{\mathcal{W}_i,\ \boldsymbol{W}(i)\} \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$ *(Best rank-k subspace selection)*
6: $\xi_i \leftarrow \arg\min_\xi \|\boldsymbol{y} - \mathcal{A}(\boldsymbol{W}(i) - \frac{\xi}{2}\mathcal{P}_{\mathcal{W}_i}\nabla f(\boldsymbol{W}(i)))\|_2^2 = \frac{\|\mathcal{P}_{\mathcal{W}_i}\nabla f(\boldsymbol{W}(i))\|_F^2}{\|\mathcal{A}\mathcal{P}_{\mathcal{W}_i}\nabla f(\boldsymbol{W}(i))\|_2^2}$ *(Step size selection)*
7: $\boldsymbol{X}(i+1) \leftarrow \boldsymbol{W}(i) - \frac{\xi_i}{2}\mathcal{P}_{\mathcal{W}_i}\nabla f(\boldsymbol{W}(i))$ with $\mathcal{X}_{i+1} \leftarrow \mathcal{P}_k(\boldsymbol{X}(i+1))$ *(De-bias using gradient descent)*
$i \leftarrow i+1$
until $\|\boldsymbol{X}(i) - \boldsymbol{X}(i-1)\|_2 \le \eta\|\boldsymbol{X}(i)\|_2$ or MaxIterations.

Algorithm 2 ADMiRA Instance

Input: $\boldsymbol{y}$, $\mathcal{A}$, k, Tolerance η, MaxIterations
Initialize: $\boldsymbol{X}(0) \leftarrow 0$, $\mathcal{X}_0 \leftarrow \{\emptyset\}$, $i \leftarrow 0$
repeat

1: $\mathcal{D}_i \leftarrow \mathcal{P}_k(\mathcal{P}_{\mathcal{X}_i^\perp} \nabla f(\boldsymbol{X}(i)))$ *(Best rank-k subspace orthogonal to $\mathcal{X}_i$)*
2: $\mathcal{S}_i \leftarrow \mathcal{D}_i \cup \mathcal{X}_i$ *(Active subspace expansion)*
3: $\boldsymbol{V}(i) \leftarrow \arg\min_{\boldsymbol{V}:\boldsymbol{V}\in\mathrm{span}(\mathcal{S}_i)} \|\boldsymbol{y} - \mathcal{A}\boldsymbol{V}\|_2^2$ *(Error norm reduction via least-squares optimization)*
4: $\{\mathcal{X}_{i+1},\ \boldsymbol{X}(i+1)\} \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$ *(Best rank-k subspace selection)*
$i \leftarrow i+1$
until $\|\boldsymbol{X}(i) - \boldsymbol{X}(i-1)\|_2 \le \eta\|\boldsymbol{X}(i)\|_2$ or MaxIterations.

At each iteration, the Algorithms 1 and 2 perform motions from the following list:

(1) *Best rank-k subspace orthogonal to $\mathcal{X}_i$ and active subspace expansion*: We identify the best rank-k subspace of the current gradient $\nabla f(\boldsymbol{X}(i))$, orthogonal to $\mathcal{X}_i$ and then merge this low-rank subspace with $\mathcal{X}_i$. This motion guarantees that, at each iteration, we expand the current rank-k subspace estimate with k new, rank-1 orthogonal subspaces to explore.

(2a) *Error norm reduction via greedy descent with adaptive step size selection* (*Algorithm* 1): We decrease the data error by performing a single gradient descent step. This scheme is based on a one-shot step size selection procedure (Step size selection step)—detailed description of this approach is given in Sect. 4.

(2b) *Error norm reduction via least squares optimization* (*Algorithm* 2): We decrease the data error $f(\boldsymbol{X})$ on the active $O(k)$-low rank subspace. Assuming $\mathcal{A}$ is well-conditioned over low-rank subspaces, the main complexity of this operation is dominated by the solution of a symmetric linear system of equations.

(3) *Best rank-k subspace selection*: We project the constrained solution onto the set of rank-k matrices $\mathcal{C}_k := \{\boldsymbol{A} \in \mathbb{R}^{m\times n} : \mathrm{rank}(\boldsymbol{A}) \le k\}$ to arbitrate the active support set. This step is calculated in polynomial time complexity as a function of $m \times n$ using SVD or other matrix rank-revealing decomposition algorithms—further discussions about this step and its approximations can be found in Sects. 8 and 9.

(4) *De-bias using gradient descent* (*Algorithm* 1): We debias the current estimate $\boldsymbol{W}(i)$ by performing an additional gradient descent step, decreasing the data error. The step size selection procedure follows the same motions as in (2a).

4 Ingredients for Hard Thresholding Methods

4.1 Step Size Selection

For the sparse vector approximation problem, recent works on the performance of the IHT algorithm provide strong convergence rate guarantees in terms of RIP constants [35]. However, as a prerequisite to achieve these strong isometry constant bounds, the step size is set $\mu_i = 1, \forall i$, given that the sensing matrix satisfies $\|\boldsymbol{\Phi}\|_2^2 < 1$ where $\|\cdot\|_2$ denotes the spectral norm [34]; similar analysis can be found in [3] for the matrix case. From a different perspective, [36] proposes a constant step size $\mu_i = 1/(1+\delta_{2K})$, $\forall i$, based on

a simple but intuitive convergence analysis of the gradient descent method.

Unfortunately, most of the above problem assumptions are not naturally met; the authors in [37] provide an intuitive example where IHT algorithm behaves differently under various scalings of the sensing matrix; similar counterexamples can be devised for the matrix case. Violating these assumptions usually leads to unpredictable signal recovery performance of the class of hard thresholding methods. Therefore, more sophisticated step size selection procedures should be devised to tackle these issues during actual recovery. On the other hand, the computation of R-RIP constants has exponential time complexity for the strategy of [3].

To this end, existing approaches broadly fall into two categories: constant and adaptive step size selection. In this work, we present efficient strategies to adaptively select the step size μ_i that implies fast convergence rate, for mild R-RIP assumptions on $\mathcal{A}$. Constant step size strategies easily follow from [24] and are not listed in this work.

Adaptive step size selection. There is limited work on the adaptive step size selection for hard thresholding methods. To the best of our knowledge, apart from [24, 37, 38] are the only studies that attempt this via line searching for the vector case. At the time of review process, we become aware of [39] which implements ideas presented in [37] for the matrix case.

According to Algorithm 1, let $\boldsymbol{X}(i)$ be the current rank-k matrix estimate spanned by the set of orthonormal, rank-1 matrices in $\mathcal{X}_i$. Using regular gradient descent motions, the new rank-k estimate $\boldsymbol{W}(i)$ can be calculated through:

$$\boldsymbol{V}_i = \boldsymbol{X}(i) - \frac{\mu}{2}\nabla f\big(\boldsymbol{X}(i)\big), \qquad \big\{\mathcal{W}_i, \boldsymbol{W}(i)\big\} \leftarrow \mathcal{P}_k\big(\boldsymbol{V}(i)\big).$$

We highlight that the rank-k approximate matrix may not be unique. It then holds that the subspace spanned by $\mathcal{W}_i$ originates: (i) either from the subspace of $\mathcal{X}_i$, (ii) or from the best subspace (in terms of the Frobenius norm metric) of the current gradient $\nabla f(\boldsymbol{X}(i))$, *orthogonal to* $\mathcal{X}_i$, (iii) or from the combination of orthonormal, rank-1 matrices lying on the union of the above two subspaces. The statements above can be summarized in the following expression:

$$\operatorname{span}(\mathcal{W}_i) \in \operatorname{span}(\mathcal{D}_i \cup \mathcal{X}_i) \tag{15}$$

for any step size μ_i and $\mathcal{D}_i \leftarrow \mathcal{P}_k(\mathcal{P}_{\mathcal{X}_i^{\perp}} \nabla f(\boldsymbol{X}(i)))$. Since $\operatorname{rank}(\operatorname{span}(\mathcal{W}_i)) \leq k$, we easily deduce the following key observation: let $\mathcal{S}_i \leftarrow \mathcal{D}_i \cup \mathcal{X}_i$ be a set of rank-1, orthonormal matrices where $\operatorname{rank}(\operatorname{span}(\mathcal{S}_i)) \leq 2k$. Given $\mathcal{W}_i$ is unknown before the i-th iteration, $\mathcal{S}_i$ spans the smallest subspace that contains $\mathcal{W}_i$ such that the following equality

$$\begin{aligned}&\mathcal{P}_k\bigg(\boldsymbol{X}(i) - \frac{\mu_i}{2}\nabla f\big(\boldsymbol{X}(i)\big)\bigg)\\ &\quad= \mathcal{P}_k\bigg(\boldsymbol{X}(i) - \frac{\mu_i}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f\big(\boldsymbol{X}(i)\big)\bigg)\end{aligned} \tag{16}$$

necessarily holds.[2]

To compute step-size μ_i, we use:

$$\begin{aligned}\mu_i &= \arg\min_{\mu} \bigg\| \boldsymbol{y} - \mathcal{A}\bigg(\boldsymbol{X}(i) - \frac{\mu}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f\big(\boldsymbol{X}(i)\big)\bigg)\bigg\|_2^2\\ &= \frac{\|\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))\|_F^2}{\|\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))\|_2^2},\end{aligned} \tag{17}$$

i.e., μ_i is the minimizer of the objective function, given the current gradient $\nabla f(\boldsymbol{X}(i))$. Note that:

$$1 - \delta_{2k}(\mathcal{A}) \leq \frac{1}{\mu_i} \leq 1 + \delta_{2k}(\mathcal{A}), \tag{18}$$

due to R-RIP—i.e., we select $2k$ subspaces such that μ_i satisfies (18). We can derive similar arguments for the additional step size selection ξ_i in Step 6 of Algorithm 1.

Adaptive μ_i scheme results in more restrictive worst-case isometry constants compared to [3, 34, 41], but faster convergence and better stability are empirically observed in general. In [3], the authors present the Singular Value Projection (SVP) algorithm, an iterative hard thresholding algorithm for the ARM problem. According to [3], both constant and iteration dependent (but user-defined) step sizes are considered. Adaptive strategies presented in [3] require the computation of R-RIP constants which has exponential time complexity. Figures 1(a)–(b) illustrate some characteristic examples. The performance varies for different problem configurations. For $\mu > 1$, SVP *diverges* for various test cases. We note that, for large fixed matrix dimensions m, n, adaptive step size selection becomes computationally expensive compared to constant step size selection strategies, as the rank of $\boldsymbol{X}^*$ increases.

4.2 Updates on Restricted Subspaces

In Algorithm 1, at each iteration, the new estimate $\boldsymbol{W}(i) \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$ can be further refined by applying a single or multiple gradient descent updates with line search restricted on $\mathcal{W}_i$ [34] (Step 7 in Algorithm 1):

$$\boldsymbol{X}(i+1) \leftarrow \boldsymbol{W}(i) - \frac{\xi_i}{2}\mathcal{P}_{\mathcal{W}_i}\nabla f\big(\boldsymbol{W}(i)\big),$$

[2] In the case of multiple identical singular values, any ties are lexicographically dissolved.

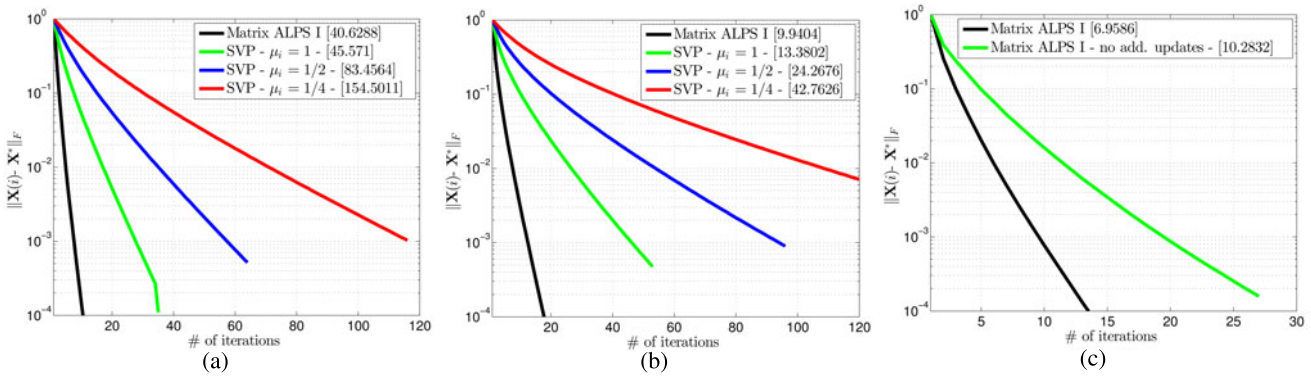

Fig. 1 Median error per iteration for various step size policies and 20 Monte-Carlo repetitions. In brackets, we present the median time consumed for convergence in seconds. (**a**) $m = n = 2048$, $p = 0.4n^2$, and rank $k = 70$—$\mathcal{A}$ is formed by permuted and subsampled noiselets [40]. (**b**) $n = 2048$, $m = 512$, $p = 0.4n^2$, and rank $k = 50$—we use underdetermined linear map $\mathcal{A}$ according to the MC problem (**c**) $n = 2048$, $m = 512$, $p = 0.4n^2$, and rank $k = 40$—we use underdetermined linear map $\mathcal{A}$ according to the MC problem

where $\xi_i = \frac{\|\mathcal{P}_{\mathcal{W}_i} \nabla f(\boldsymbol{W}(i))\|_F^2}{\|\mathcal{A}\mathcal{P}_{\mathcal{W}_i} \nabla f(\boldsymbol{W}(i))\|_2^2}$. In spirit, the gradient step above is the same as block coordinate descent in convex optimization where we find the subspaces adaptively. Figure 1(c) depicts the acceleration achieved by using additional gradient updates over restricted low-rank subspaces for a test case.

4.3 Acceleration via Memory-Based Schemes and Low-Rank Matrix Approximations

Memory-based techniques can be used to improve convergence speed. Furthermore, low-rank matrix approximation tools overcome the computational overhead of computing the best low-rank projection by inexactly solving (5). We keep the discussion on memory utilization for Sect. 7 and low-rank matrix approximations for Sects. 8 and 9 where we present new algorithmic frameworks for low-rank matrix recovery.

4.4 Active Low-Rank Subspace Tracking

Per iteration of Algorithms 1 and 2, we perform projection operations $\mathcal{P}_{\mathcal{S}}\boldsymbol{X}$ and $\mathcal{P}_{\mathcal{S}^\perp}\boldsymbol{X}$ where $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, as described by (6) and (7), respectively. Since $\mathcal{S}$ is constituted by outer products of left and right singular vectors as in Definition 2, $\mathcal{P}_{\mathcal{S}}\boldsymbol{X}$ (resp. $\mathcal{P}_{\mathcal{S}^\perp}\boldsymbol{X}$) projects onto the (resp. complement of the) best low-rank subspace in $R(\boldsymbol{X})$ and $R(\boldsymbol{X}^T)$. These operations are highly connected with the adaptive step size selection and the updates on restricted subspaces. Unfortunately, the time-complexity to compute $\mathcal{P}_{\mathcal{S}}\boldsymbol{X}$ is dominated by three matrix-matrix multiplications which decelerates the convergence of the proposed schemes in high-dimensional settings. To accelerate the convergence in many test cases, it turns out that we do not have to use the best projection $\mathcal{P}_{\mathcal{S}}$ in practice.[3] Rather, employing *inexact* projections is sufficient to converge to the optimal solution: either (i) $\mathcal{P}_{\mathcal{U}}\boldsymbol{X}$ onto the best low-rank subspace in $R(\boldsymbol{X})$ only (if $m \ll n$) or (ii) $\boldsymbol{X}\mathcal{P}_{\mathcal{V}}$ onto the best low-rank subspace in $R(\boldsymbol{X}^T)$ only (if $m \gg n$);[4] $\mathcal{P}_{\mathcal{U}}$ and $\mathcal{P}_{\mathcal{V}}$ are defined in Definition 2 and require only one matrix-matrix multiplication.

Figure 2 shows the time overhead due to the exact projection application $\mathcal{P}_{\mathcal{S}}$ compared to $\mathcal{P}_{\mathcal{U}}$ for $m \leq n$. In Fig. 2(a), we use subsampled and permuted noiselets for linear map $\mathcal{A}$ and in Figs. 2(b)–(c), we test the MC problem. While in the case $m = n$ the use of (6)–(7) has a clear advantage over inexact projections using only $\mathcal{P}_{\mathcal{U}}$, the latter case converges faster to the desired accuracy 5×10^{-4} when $m \ll n$ as shown in Figs. 2(a)–(b). In our derivations, we assume $\mathcal{P}_{\mathcal{S}}$ and $\mathcal{P}_{\mathcal{S}^\perp}$ as defined in (6) and (7).

5 Convergence Guarantees

In this section, we present the theoretical convergence guarantees of Algorithms 1 and 2 as functions of R-RIP constants. To characterize the performance of the proposed algorithms, both in terms of convergence rate and noise resilience, we use the following recursive expression:

$$\left\|\boldsymbol{X}(i+1) - \boldsymbol{X}^*\right\|_F \leq \rho \left\|\boldsymbol{X}(i) - \boldsymbol{X}^*\right\|_F + \gamma \|\boldsymbol{\varepsilon}\|_2. \tag{19}$$

In (19), γ denotes the approximation guarantee and provides insights into algorithm's reconstruction capabilities when additive noise is present; $\rho < 1$ expresses the convergence rate towards a region around $\boldsymbol{X}^*$, whose radius is determined

[3] From a different perspective and for a different problem case, similar ideas have been used in [18].

[4] We can move between these two cases by a simple transpose of the problem.

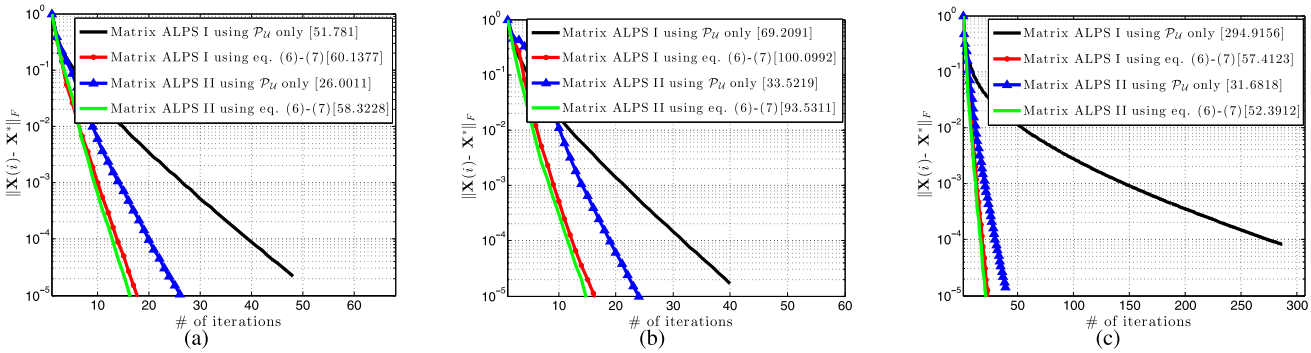

Fig. 2 Median error per iteration for MATRIX ALPS I and MATRIX ALPS II variants over 10 Monte-Carlo repetitions. In brackets, we present the median time consumed for convergene in seconds. (**a**) $n = 2048$, $m = 512$, $p = 0.25n^2$, and rank $k = 40$. (**b**) $n = 2000$, $m = 1000$, $p = 0.25n^2$, and rank $k = 50$. (**c**) $n = m = 1000$, $p = 0.25n^2$, and rank $k = 50$

by $\frac{\gamma}{1-\rho}\|\boldsymbol{\varepsilon}\|_2$. In short, (19) characterizes how the distance to the true signal $\boldsymbol{X}^*$ is decreased and how the noise level affects the accuracy of the solution, at each iteration.

5.1 MATRIX ALPS I

An important lemma for our derivations below is given next:

Lemma 6 [Active subspace expansion] *Let $\boldsymbol{X}(i)$ be the matrix estimate at the i-th iteration and let $\mathcal{X}_i$ be a set of orthonormal, rank-1 matrices such that $\mathcal{X}_i \leftarrow \mathcal{P}_k(\boldsymbol{X}(i))$. Then, at each iteration, the Active Subspace Expansion step in Algorithms 1 and 2 identifies information in $\boldsymbol{X}^*$, such that*:

$$\left\|\mathcal{P}_{\mathcal{X}^*}\mathcal{P}_{\mathcal{S}_i^\perp}\boldsymbol{X}^*\right\|_F \leq (2\delta_{2k}+2\delta_{3k})\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F + \sqrt{2(1+\delta_{2k})}\|\boldsymbol{\varepsilon}\|_2, \tag{20}$$

where $\mathcal{S}_i \leftarrow \mathcal{X}_i \cup \mathcal{D}_i$ and $\mathcal{X}^* \leftarrow \mathcal{P}_k(\boldsymbol{X}^*)$.

Lemma 6 states that, at each iteration, the active subspace expansion step identifies a $2k$ rank subspace such that the amount of unrecovered energy of $\boldsymbol{X}^*$—i.e., the projection of $\boldsymbol{X}^*$ onto the orthogonal subspace of span$(\mathcal{S}_i)$—is bounded by (20).

Then, Theorem 1 characterizes the iteration invariant of Algorithm 1 for the matrix case:

Theorem 1 [Iteration invariant for MATRIX ALPS I] *The $(i+1)$-th matrix estimate $\boldsymbol{X}(i+1)$ of* MATRIX ALPS I *satisfies the following recursion*:

$$\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F \leq \rho\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F + \gamma\|\boldsymbol{\varepsilon}\|_2, \tag{21}$$

where $\rho := (\frac{1+2\delta_{2k}}{1-\delta_{2k}})(\frac{4\delta_{2k}}{1-\delta_{2k}} + (2\delta_{2k}+2\delta_{3k})\frac{2\delta_{3k}}{1-\delta_{2k}})$ and $\gamma := (\frac{1+2\delta_{2k}}{1-\delta_{2k}})(\frac{2\sqrt{1+\delta_{2k}}}{1-\delta_{2k}} + \frac{2\delta_{3k}}{1-\delta_{2k}}\sqrt{2(1+\delta_{2k})}) + \frac{\sqrt{1+\delta_k}}{1-\delta_k}$. Moreover, when $\delta_{3k} < 0.1235$, the iterations are contractive.

To provide some intuition behind this result, assume that $\boldsymbol{X}^*$ is a rank-k matrix. Then, according to Theorem 1, for $\rho < 1$, the approximation parameter γ in (21) satisfies:

$$\gamma < 5.7624, \quad \text{for } \delta_{3k} < 0.1235.$$

Moreover, we derive the following:

$$\rho < \frac{1+2\delta_{3k}}{(1-\delta_{3k})^2}\left(4\delta_{3k}+8\delta_{3k}^2\right) < \frac{1}{2} \quad \Rightarrow \quad \delta_{3k} < 0.079,$$

which is *a stronger* R-RIP condition assumption compared to state-of-the-art approaches [21]. In the next section, we further improve this guarantee using Algorithm 2.

Unfolding the recursive formula (21), we obtain the following upper bound for $\|\boldsymbol{X}(i)-\boldsymbol{X}^*\|_F$ at the i-th iteration:

$$\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F \leq \rho^i\left\|\boldsymbol{X}(0)-\boldsymbol{X}^*\right\|_F + \frac{\gamma}{1-\rho}\|\boldsymbol{\varepsilon}\|_2. \tag{22}$$

Then, given $\boldsymbol{X}(0) = \boldsymbol{0}$, MATRIX ALPS I finds a rank-k solution $\widehat{\boldsymbol{X}} \in \mathbb{R}^{m\times n}$ such that $\|\widehat{\boldsymbol{X}}-\boldsymbol{X}^*\|_F \leq \frac{\gamma+1-\rho}{1-\rho}\|\boldsymbol{\varepsilon}\|_2$ after $i := \lceil \frac{\log(\|\boldsymbol{X}^*\|_F/\|\boldsymbol{\varepsilon}\|_2)}{\log(1/\rho)} \rceil$ iterations.

If we ignore steps 5 and 6 in Algorithm 1, we obtain another projected gradient descent variant for the affine rank minimization problem, for which we obtain the following performance guarantees—the proof follows from the proof of Theorem 1.

Corollary 1 [MATRIX ALPS I Instance] *In Algorithm 1, we ignore steps 5 and 6 and let $\{\mathcal{X}_{i+1},\ \boldsymbol{X}(i+1)\} \leftarrow \mathcal{P}_k(\boldsymbol{V}_i)$. Then, by the same analysis, we observe that the following recursion is satisfied*:

$$\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F \leq \rho\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F + \gamma\|\boldsymbol{\varepsilon}\|_2, \tag{23}$$

for $\rho := (\frac{4\delta_{2k}}{1-\delta_{2k}} + (2\delta_{2k}+2\delta_{3k})\frac{2\delta_{3k}}{1-\delta_{2k}})$ and $\gamma := (\frac{2\sqrt{1+\delta_{2k}}}{1-\delta_{2k}} + \frac{2\delta_{3k}}{1-\delta_{2k}}\sqrt{2(1+\delta_{2k})})$. Moreover, $\rho < 1$ when $\delta_{3k} < 0.1594$.

We observe that the absence of the additional estimate update over restricted support sets results in less restrictive isometry constants compared to Theorem 1. In practice, additional updates result in faster convergence, as shown in Fig. 1(c).

5.2 ADMiRA Instance

In MATRIX ALPS I, the gradient descent steps constitute a first-order approximation to least-squares minimization problems. Replacing Step 4 in Algorithm 1 with the following optimization problem:

$$\boldsymbol{V}(i) \leftarrow \underset{\boldsymbol{V}:\boldsymbol{V}\in\operatorname{span}(\mathcal{S}_i)}{\arg\min} \|\boldsymbol{y}-\boldsymbol{\mathcal{A}}\boldsymbol{V}\|_2^2, \tag{24}$$

we obtain ADMiRA (furthermore, we remove the de-bias step in Algorithm 1). Assuming that the linear operator $\boldsymbol{\mathcal{A}}$, restricted on sufficiently low-rank subspaces, is well conditioned in terms of the R-RIP assumption, the optimization problem (24) has a unique optimal minimizer. By exploiting the optimality condition in Lemma 1, ADMiRA instance in Algorithm 2 features the following guarantee:

Theorem 2 [Iteration invariant for ADMiRA instance] *The $(i+1)$-th matrix estimate $\boldsymbol{X}(i+1)$ of ADMiRA answers the following recursive expression*:

$$\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F \le \rho\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F + \gamma\|\boldsymbol{\varepsilon}\|_F,$$

$\rho := (2\delta_{2k}+2\delta_{3k})\sqrt{\frac{1+3\delta_{3k}^2}{1-\delta_{3k}^2}}$, *and* $\gamma := \sqrt{\frac{1+3\delta_{3k}^2}{1-\delta_{3k}^2}}\sqrt{2(1+\delta_{3k})}+(\frac{\sqrt{1+3\delta_{3k}^2}}{1-\delta_{3k}}+\sqrt{3})\sqrt{1+\delta_{2k}}$. *Moreover, when $\delta_{3k}<0.2267$, the iterations are contractive*.

Similarly to MATRIX ALPS I analysis, the parameter γ in Theorem 2 satisfies:

$$\gamma < 5.1848, \quad \text{for } \delta_{3k} < 0.2267.$$

Furthermore, to compare the approximation guarantees of Theorem 2 with [21], we further observe:

$$\delta_{3k} < 0.1214, \quad \text{for } \rho < 1/2.$$

We remind that [21] provides convergence guarantees for ADMiRA with $\delta_{4k} < 0.04$ for $\rho = 1/2$.

6 Complexity Analysis

In each iteration, computational requirements of the proposed hard thresholding methods mainly depend on the total number of linear mapping operations $\boldsymbol{\mathcal{A}}$, gradient descent steps, least-squares optimizations, projection operations and matrix decompositions for low rank approximation. Different algorithmic configurations (e.g. removing steps 6 and 7 in Algorithm 1) lead to hard thresholding variants with less computational complexity per iteration and better R-RIP conditions for convergence but a degraded performance in terms of stability and convergence speed is observed in practice. On the other hand, these additional processing steps increase the required time-complexity per iteration; hence, low iteration counts are desired to tradeoff these operations.

A non-exhaustive list of linear map examples includes the identity operator (Principal component analysis (PCA) problem), Fourier/Wavelets/Noiselets transformations and the famous Matrix Completion problem where $\boldsymbol{\mathcal{A}}$ is a mask operator such that only a fraction of elements in $\boldsymbol{X}$ is observed. Assuming the most demanding case where $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{A}}^*$ are dense linear maps with no structure, the computation of the gradient $\nabla f(\boldsymbol{X}(i))$ at each iteration requires $O(pkmn)$ arithmetic operations.

Given a set $\mathcal{S}$ of orthonormal, rank-1 matrices, the projection $\mathcal{P}_{\mathcal{S}}\boldsymbol{X}$ for any matrix $\boldsymbol{X}\in\mathbb{R}^{m\times n}$ requires time complexity $O(\max\{m^2n, mn^2\})$ as a sequence of matrix-matrix multiplication operations.[5] In MATRIX ALPS I, the adaptive step size selection steps require $O(\max\{pkmn, m^2n\})$ time complexity for the calculation of μ_i and ξ_i quantities. In ADMiRA solving a least-squares system restricted on rank-$2k$ and rank-k subspaces requires $O(pk^2)$ complexity; according to [21, 32], the complexity of this step can be further reduced using iterative techniques such as the Richardson method or conjugate gradients algorithm.

Using the Lanczos method, we require $O(kmn)$ arithmetic operations to compute a rank-k matrix approximation for a given constant accuracy; a prohibitive time-complexity that does not scale well for many practical applications. Sections 8 and 9 describe approximate low rank matrix projections and how they affect the convergence guarantees of the proposed algorithms.

Overall, the operation that dominates per iteration requires $O(\max\{pkmn, m^2n, mn^2\})$ time complexity in the proposed schemes.

7 Memory-Based Acceleration

Iterative algorithms can use memory to gain momentum in convergence. Based on Nesterov's optimal gradient methods [42], we propose a hard thresholding variant, described in Algorithm 3 where an additional update on $\boldsymbol{X}(i+1)$ with momentum step size τ_i is performed using previous matrix estimates.

[5] While such operation has $O(\max\{m^2n, mn^2\})$ complexity, each application of $\mathcal{P}_{\mathcal{S}}\boldsymbol{X}$ requires three matrix-matrix multiplications. To reduce such computational cost, we *relax* this operation in Sect. 10 where in practice we use only $\mathcal{P}_{\mathcal{U}}$ that needs one matrix-matrix multiplication.

Algorithm 3 MATRIX ALPS II

Input: y, $\mathcal{A}$, k, Tolerance η, MaxIterations
Initialize: $\boldsymbol{X}(0) \leftarrow 0$, $\mathcal{X}_0 \leftarrow \{\emptyset\}$, $\boldsymbol{Q}(0) \leftarrow 0$, $\mathcal{Q}_0 \leftarrow \{\emptyset\}$, $\tau_i\ \forall i$, $i \leftarrow 0$
repeat

$\mathcal{D}_i \leftarrow \mathcal{P}_k(\mathcal{P}_{\mathcal{Q}_i^{\perp}} \nabla f(\boldsymbol{Q}(i)))$ *(Best rank-k subspace orthogonal to $\mathcal{Q}_i$)*

$\mathcal{S}_i \leftarrow \mathcal{D}_i \cup \mathcal{Q}_i$ *(Active subspace expansion)*

$\mu_i \leftarrow \arg\min_{\mu} \|\boldsymbol{y} - \mathcal{A}(\boldsymbol{Q}(i) - \frac{\mu}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{Q}(i)))\|_2^2 = \frac{\|\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{Q}(i))\|_F^2}{\|\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{Q}(i))\|_2^2}$ *(Step size selection)*

$\boldsymbol{V}(i) \leftarrow \boldsymbol{Q}(i) - \frac{\mu_i}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{Q}(i))$ *(Error norm reduction via gradient descent)*

$\{\mathcal{X}_{i+1},\ \boldsymbol{X}(i+1)\} \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$ *(Best rank-k subspace selection)*

$\boldsymbol{Q}(i+1) \leftarrow \boldsymbol{X}(i+1) + \tau_i(\boldsymbol{X}(i+1) - \boldsymbol{X}(i))$ *(Momentum update)*

$\mathcal{Q}_{i+1} \leftarrow \text{ortho}(\mathcal{X}_i \cup \mathcal{X}_{i+1})$

$i \leftarrow i+1$

until $\|\boldsymbol{X}(i) - \boldsymbol{X}(i-1)\|_2 \le \eta\|\boldsymbol{X}(i)\|_2$ or MaxIterations.

Similar to μ_i strategies, τ_i can be preset as constant or adaptively computed at each iteration. Constant momentum step size selection has no additional computational cost but convergence rate acceleration is not guaranteed for some problem formulations in practice. On the other hand, empirical evidence has shown that adaptive τ_i selection strategies result in faster convergence compared to zero-memory methods with *similar complexity*.

For the case of strongly convex objective functions, Nesterov [43] proposed the following constant momentum step size selection scheme: $\tau_i = \frac{\alpha_i(1-\alpha_i)}{\alpha_i^2+\alpha_{i+1}}$, where $\alpha_0 \in (0,1)$ and α_{i+1} is computed as the root $\in (0,1)$ of

$$\alpha_{i+1}^2 = (1-\alpha_{i+1})\alpha_i^2 + q\alpha_{i+1}, \quad \text{for } q \triangleq \frac{1}{\kappa^2(\mathcal{A})}, \tag{25}$$

where $\kappa(\mathcal{A})$ denotes the condition number of $\mathcal{A}$. In this scheme, exact calculation of q parameter is computationally expensive for large-scale data problems and approximation schemes are leveraged to compensate this complexity bottleneck.

Based upon adaptive μ_i selection, we propose to select τ_i as the minimizer of the objective function:

$$\begin{aligned}\tau_i &= \arg\min_{\tau} \left\|\boldsymbol{y} - \mathcal{A}\boldsymbol{Q}(i+1)\right\|_2^2 \\ &= \frac{\langle \boldsymbol{y} - \mathcal{A}\boldsymbol{X}(i), \mathcal{A}\boldsymbol{X}(i) - \mathcal{A}\boldsymbol{X}(i-1)\rangle}{\|\mathcal{A}\boldsymbol{X}(i) - \mathcal{A}\boldsymbol{X}(i-1)\|_2^2},\end{aligned} \tag{26}$$

where $\mathcal{A}\boldsymbol{X}(i), \mathcal{A}\boldsymbol{X}(i-1)$ are already *pre-computed* at each iteration. According to (26), τ_i is dominated by the calculation of a vector inner product, a computationally cheaper process than q calculation.

Theorem 3 characterizes Algorithm 3 for *constant* momentum step size selection. To keep the main ideas simple, we ignore the additional gradient updates in Algorithm 3. In addition, we only consider the noiseless case for clarity. The convergence rate proof for these cases is provided in the Appendix.

Theorem 3 [Iteration invariant for MATRIX ALPS II] *Let $\boldsymbol{y} = \mathcal{A}\boldsymbol{X}^*$ be a noiseless set of observations. To recover $\boldsymbol{X}^*$ from $\boldsymbol{y}$ and $\mathcal{A}$, the $(i+1)$-th matrix estimate $\boldsymbol{X}(i+1)$ of* MATRIX ALPS II *satisfies the following recursion*:

$$\begin{aligned}\left\|\boldsymbol{X}(i+1) - \boldsymbol{X}^*\right\|_F \le{}& \alpha(1+\tau_i)\left\|\boldsymbol{X}(i) - \boldsymbol{X}^*\right\|_F \\ &+ \alpha\tau_i\left\|\boldsymbol{X}(i-1) - \boldsymbol{X}^*\right\|_F,\end{aligned} \tag{27}$$

where $\alpha := \frac{4\delta_{3k}}{1-\delta_{3k}} + (2\delta_{3k} + 2\delta_{4k})\frac{2\delta_{3k}}{1-\delta_{3k}}$. Moreover, solving the above second-order recurrence, the following inequality holds true:

$$\left\|\boldsymbol{X}(i+1) - \boldsymbol{X}^*\right\|_F \le \rho^{i+1}\left\|\boldsymbol{X}(0) - \boldsymbol{X}^*\right\|_F, \tag{28}$$

for $\rho := \frac{\alpha(1+\tau_i)+\sqrt{\alpha^2(1+\tau_i)^2+4\alpha\tau_i}}{2}$.

Theorem 3 provides convergence rate behaviour proof for the case where τ_i is constant $\forall i$. The more elaborate case where τ_i follows the policy described in (26) is left as an open question for future work. To provide some insight for (28), for $\tau_i = 1/4$, $\forall i$ and $\tau_i = 1/2$, $\forall i$, $\delta_{4k} < 0.1187$ and $\delta_{4k} < 0.095$ guarantee convergence in Algorithm 3, respectively. While the RIP requirements for memory-based MATRIX ALPS II are more stringent than the schemes proposed in the previous section, it outperforms Algorithms 1 and 2. Figure 2 shows the acceleration achieved in MATRIX ALPS II by using inexact projections $\mathcal{P}_{\mathcal{U}}$. Using the proper projections (6)–(7), Fig. 3 shows acceleration in practice when using the adaptive momentum step size strategy: while a wide range of constant momentum step sizes leads to convergence, providing flexibility to select an appropriate τ_i, adaptive τ_i avoids this arbitrary τ_i selection while further decreases the number of iterations needed for convergence in most cases.

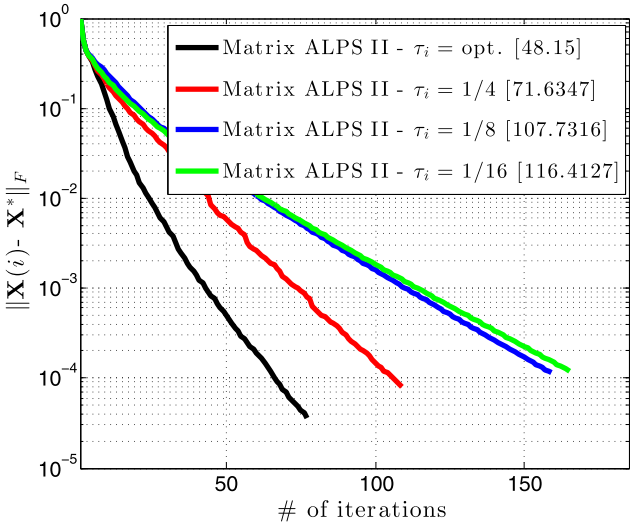

Fig. 3 Median error per iteration for various momentum step size policies and 10 Monte-Carlo repetitions. Here, $n = 1024$, $m = 256$, $p = 0.25n^2$, and rank $k = 40$. We use permuted and subsampled noiselets for the linear map $\mathcal{A}$. In brackets, we present the median time for convergence in seconds

8 Accelerating MATRIX ALPS: ϵ-Approximation of SVD via Column Subset Selection

A time-complexity bottleneck in the proposed schemes is the computation of the singular value decomposition to find subspaces that describe the unexplored information in matrix $\boldsymbol{X}^*$. Unfortunately, the computational cost of regular SVD for best subspace tracking is prohibitive for many applications.

Based on [44, 45], we can obtain randomized SVD approximations of a matrix $\boldsymbol{X}$ using *column subset selection* ideas: we compute a leverage score for each column that represents its "significance". In particular, we define a probability distribution that weights each column depending on the amount of information they contain; usually, the distribution is related to the ℓ_2-norm of the columns. The main idea of this approach is to compute a surrogate rank-k matrix $\mathcal{P}_k^{\epsilon}(\boldsymbol{X})$ by subsampling the columns according to this distribution. It turns out that the total number of sampled columns is a function of the parameter ϵ. Moreover, [46, 47] proved that, given a target rank k and an approximation parameter ϵ, we can compute an ϵ-approximate rank-k matrix $\mathcal{P}_k^{\epsilon}(\boldsymbol{X})$ according to the following definition.

Definition 4 [ϵ-approximate low-rank projection] Let $\boldsymbol{X}$ be an arbitrary matrix. Then, $\mathcal{P}_k^{\epsilon}(\boldsymbol{X})$ projection provides a rank-k matrix approximation to $\boldsymbol{X}$ such that:

$$\left\|\mathcal{P}_k^{\epsilon}(\boldsymbol{X}) - \boldsymbol{X}\right\|_F^2 \leq (1+\epsilon)\left\|\mathcal{P}_k(\boldsymbol{X}) - \boldsymbol{X}\right\|_F^2, \tag{29}$$

where $\mathcal{P}_k(\boldsymbol{X}) \in \arg\min_{\boldsymbol{Y}:\mathrm{rank}(\boldsymbol{Y})\leq k} \|\boldsymbol{X} - \boldsymbol{Y}\|_F$.

For the following theoretical results, we assume the following condition on the sensing operator $\mathcal{A}$: $\|\mathcal{A}^*\boldsymbol{\beta}\|_F \leq \lambda$, $\forall \boldsymbol{\beta} \in \mathbb{R}^p$ where $\lambda > 0$. Using ϵ-approximation schemes to perform the Active subspace selection step, the following upper bound holds. The proof is provided in the Appendix:

Lemma 7 [ϵ-approximate active subspace expansion] *Let $\boldsymbol{X}(i)$ be the matrix estimate at the i-th iteration and let $\mathcal{X}_i$ be a set of orthonormal, rank-1 matrices in $\mathbb{R}^{m\times n}$ such that $\mathcal{X}_i \leftarrow \mathcal{P}_k(\boldsymbol{X}(i))$. Furthermore, let*

$$\mathcal{D}_i^{\epsilon} \leftarrow \mathcal{P}_k^{\epsilon}\left(\mathcal{P}_{\mathcal{X}_i^{\perp}} \nabla f\left(\boldsymbol{X}(i)\right)\right),$$

be a set of orthonormal, rank-1 matrices that span rank-k subspace such that (29) is satisfied for $\boldsymbol{X} := \mathcal{P}_{\mathcal{X}_i^{\perp}} \nabla f(\boldsymbol{X}(i))$. Then, at each iteration, the Active Subspace Expansion step in Algorithms 1 and 2 captures information contained in the true matrix $\boldsymbol{X}^*$, such that:

$$\begin{aligned}&\left\|\mathcal{P}_{\mathcal{X}^*}\mathcal{P}_{\mathcal{S}_i^{\perp}}\boldsymbol{X}^*\right\|_F \\ &\quad\leq (2\delta_{2k} + 2\delta_{3k})\left\|\boldsymbol{X}(i) - \boldsymbol{X}^*\right\|_F + \sqrt{2(1+\delta_{2k})}\|\boldsymbol{\varepsilon}\|_2 \\ &\qquad + 2\lambda\sqrt{\epsilon},\end{aligned} \tag{30}$$

where $\mathcal{S}_i \leftarrow \mathcal{X}_i \cup \mathcal{D}_i^{\epsilon}$ and $\mathcal{X}^* \leftarrow \mathcal{P}_k(\boldsymbol{X}^*)$.

Furthermore, to prove the following theorems, we extend Lemma 10, provided in the Appendix, as follows. The proof easily follows from the proof of Lemma 10, using Definition 4:

Lemma 8 [*ϵ-approximation rank-k subspace selection*] *Let $\boldsymbol{V}(i)$ be a rank-$2k$ proxy matrix in the subspace spanned by $\mathcal{S}_i$ and let $\widehat{\boldsymbol{W}}(i) \leftarrow \mathcal{P}_k^{\epsilon}(\boldsymbol{V}(i))$ denote the rank-k ϵ-approximation to $\boldsymbol{V}(i)$, according to (5). Then:*

$$\begin{aligned}\left\|\widehat{\boldsymbol{W}}(i) - \boldsymbol{V}(i)\right\|_F^2 &\leq (1+\epsilon)\left\|\boldsymbol{W}(i) - \boldsymbol{V}(i)\right\|_F \\ &\leq (1+\epsilon)\left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i) - \boldsymbol{X}^*\right)\right\|_F \\ &\leq (1+\epsilon)\left\|\boldsymbol{V}(i) - \boldsymbol{X}^*\right\|_F\end{aligned} \tag{31}$$

where $\boldsymbol{W}(i) \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$.

8.1 MATRIX ALPS I Using ϵ-Approximate Low-Rank Projection via Column Subset Selection

Using ϵ-approximate SVD in MATRIX ALPS I, the following iteration invariant theorem holds:

Theorem 4 [Iteration invariant with ϵ-approximate projections for MATRIX ALPS I] *The $(i+1)$-th matrix estimate $\boldsymbol{X}(i+1)$ of* MATRIX ALPS I *with ϵ-approximate projections $\mathcal{D}_i^{\epsilon} \leftarrow \mathcal{P}_k^{\epsilon}(\mathcal{P}_{\mathcal{X}_i^{\perp}} \nabla f(\boldsymbol{X}(i)))$ and $\widehat{\boldsymbol{W}}(i) \leftarrow \mathcal{P}_k^{\epsilon}(\boldsymbol{V}(i))$ in*

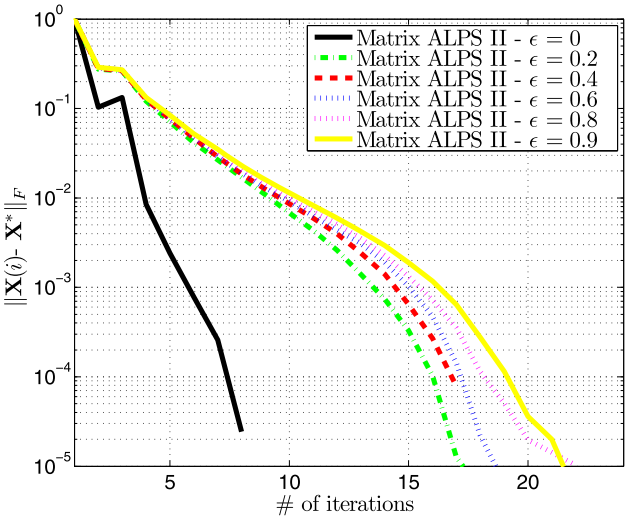

Fig. 4 Performance comparison using ϵ-approximation SVD [47] in MATRIX ALPS II. $m = n = 256$, $p = 0.4n^2$, rank of $\boldsymbol{X}^*$ equals 2 and $\mathcal{A}$ constituted by permuted noiselets. The non-smoothness in the error curves is due to the extreme low rankness of $\boldsymbol{X}^*$ for this setting

Algorithm 1 satisfies the following recursion:

$$\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F \leq \rho\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F + \gamma\|\boldsymbol{\varepsilon}\|_2 + \beta\lambda, \tag{32}$$

where $\rho := (1+\frac{3\delta_k}{1-\delta_k})(2+\epsilon)[(1+\frac{\delta_{3k}}{1-\delta_{2k}})4\delta_{3k}+\frac{2\delta_{2k}}{1-\delta_{2k}}]$, $\beta := (1+\frac{3\delta_k}{1-\delta_k})(2+\epsilon)(1+\frac{\delta_{3k}}{1-\delta_{2k}})2\sqrt{\epsilon}$, *and* $\gamma := (1+\frac{3\delta_k}{1-\delta_k})(2+\epsilon)[(1+\frac{\delta_{3k}}{1-\delta_{2k}})\sqrt{2(1+\delta_{2k})}+2\frac{\sqrt{1+\delta_{2k}}}{1-\delta_{2k}}]$.

Similar analysis can be conducted for the ADMiRA algorithm. To illustrate the impact of SVD ϵ-approximation on the signal reconstruction performance of the proposed methods, we replace the *best* rank-k projections in steps 1 and 5 of Algorithm 1 by the ϵ-approximation SVD algorithm, presented in [47]. In this paper, the column subset selection algorithm satisfies the following theorem:

Theorem 5 *Let* $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ *be a signal of interest with arbitrary* rank $< \min\{m,n\}$ *and let* $\boldsymbol{X}_k$ *represent the best rank-k approximation of* $\boldsymbol{X}$. *After* $2(k+1)(\log(k+1)+1)$ *passes over the data, the Linear Time Low-Rank Matrix Approximation algorithm in* [47] *computes a rank-k approximation* $\mathcal{P}_k^{\epsilon}(\boldsymbol{X}) \in \mathbb{R}^{m\times n}$ *such that Definition 4 is satisfied with probability at least* 3/4.

The proof is provided in [47]. In total, Linear Time Low-Rank Matrix Approximation algorithm [47] requires $O(mn(k/\epsilon + k^2\log k) + (m+n)(k^2/\epsilon^2 + k^3\log k/\epsilon + k^4\log^2 k))$ and $O(\min\{m,n\}(k/\epsilon + k^2\log k))$ time and space complexity, respectively. However, while column subset selection methods such as [47] reduce the overall complexity of low-rank projections in theory, in practice this applies only in very high-dimensional settings. To strengthen this argument, in Fig. 4 we compare SVD-based MATRIX ALPS II with MATRIX ALPS II using the ϵ-approximate column subset selection method in [47]. We observe that the total number of iterations for convergence increases due to ϵ-approximate low-rank projections, as expected. Nevertheless, we observe that, on average, the column subset selection process [47] is computationally prohibitive compared to regular SVD due to the time overhead in the column selection procedure—fewer passes over the data are desirable in practice to tradeoff the increased number of iterations for convergence. In the next section, we present alternatives based on recent trends in randomized matrix decompositions and how we can use them in low-rank recovery.

9 Accelerating MATRIX ALPS: SVD Approximation Using Randomized Matrix Decompositions

Finding low-cost SVD approximations to tackle the above complexity issues is a challenging task. Recent works on probabilistic methods for matrix approximation [26] provide a family of efficient approximate projections on the set of rank-deficient matrices with clear computational advantages over regular SVD computation in practice and attractive theoretical guarantees. In this work, we build on the low-cost, power-iteration *subspace tracking* scheme, described in Algorithms 4.3 and 4.4 in [26]. Our proposed algorithm is described in Algorithm 4.

The convergence guarantees of Algorithm 4 follow the same motions described in Sect. 8, where ϵ is a function of m, n, k and q.

10 Experiments

10.1 List of Algorithms

In the following experiments, we compare the following algorithms: (i) the Singular Value Projection (SVP) algorithm [3], a non-convex first-order projected gradient descent algorithm with *constant* step size selection (we study the case where $\mu = 1$), (ii) the inexact ALM algorithm [18] based on augmented Langrance multiplier method, (iii) the OptSpace algorithm [48], a gradient descent algorithm on the Grassmann manifold, (iv) the Grassmannian Rank-One Update Subspace Estimation (GROUSE) and the Grassmannian Robust Adaptive Subspace Tracking methods (GRASTA) [49, 50], two stochastic gradient descent algorithms that operate on the Grassmannian—moreover, to allay the impact of outliers in the subspace selection step, GRASTA incorporates the augmented Lagrangian of ℓ_1-norm loss function into the Grassmannian optimization framework, (v) the Riemannian

Algorithm 4 Randomized MATRIX ALPS II with QR Factorization

Input: y, $\mathcal{A}$, k, q, Tolerance η, MaxIterations
Initialize: $\boldsymbol{X}(0) \leftarrow 0$, $\mathcal{X}_0 \leftarrow \{\emptyset\}$, $\boldsymbol{Q}(0) \leftarrow 0$, $\mathcal{Q}_0 \leftarrow \{\emptyset\}$, $\tau_i\ \forall i$, $i \leftarrow 0$
repeat

$\mathcal{D}_i \leftarrow \text{RANDOMIZEDPOWERITERATION}(\mathcal{P}_{\mathcal{Q}_i^{\perp}} \nabla f(\mathbf{Q}(i)),\ k,\ q)$ *(Rank-k subspace via Randomized Power Iteration)*

$\mathcal{S}_i \leftarrow \mathcal{D}_i \cup \mathcal{Q}_i$ *(Active subspace expansion)*

$\mu_i \leftarrow \arg\min_{\mu} \|\boldsymbol{y} - \boldsymbol{\mathcal{A}}(\boldsymbol{Q}(i) - \frac{\mu}{2}\mathcal{P}_{\mathcal{S}_i} \nabla f(\boldsymbol{Q}(i)))\|_2^2 = \frac{\|\mathcal{P}_{\mathcal{S}_i} \nabla f(\boldsymbol{Q}(i))\|_F^2}{\|\boldsymbol{\mathcal{A}}\mathcal{P}_{\mathcal{S}_i} \nabla f(\boldsymbol{Q}(i))\|_2^2}$ *(Step size selection)*

$\boldsymbol{V}(i) \leftarrow \boldsymbol{Q}(i) - \frac{\mu_i}{2}\mathcal{P}_{\mathcal{S}_i} \nabla f(\boldsymbol{Q}(i))$ *(Error norm reduction via gradient descent)*

$\mathcal{W} \leftarrow \text{RANDOMIZEDPOWERITERATION}(\mathbf{V}(i),\ k,\ q)$ *(Rank-k subspace via Randomized Power Iteration)*

$\boldsymbol{X}(i+1) \leftarrow \mathcal{P}_{\mathcal{W}} \mathbf{V}(i)$ *(Best rank-k subspace selection)*

$\boldsymbol{Q}(i+1) \leftarrow \boldsymbol{X}(i+1) + \tau_i(\boldsymbol{X}(i+1) - \boldsymbol{X}(i))$ *(Momentum update)*

$\mathcal{Q}_{i+1} \leftarrow \text{ortho}(\mathcal{X}_i \cup \mathcal{X}_{i+1})$

$i \leftarrow i + 1$

until $\|\boldsymbol{X}(i) - \boldsymbol{X}(i-1)\|_2 \leq \eta \|\boldsymbol{X}(i)\|_2$ or MaxIterations.

Trust Region Matrix Completion algorithm (RTRMC) [51], a matrix completion method using first- and second-order Riemannian trust-region approaches, (vi) the Low rank Matrix Fitting algorithm (LMatFit) [52], a nonlinear successive over-relaxation algorithm and (vii) the algorithms MATRIX ALPS I, ADMiRA [21], MATRIX ALPS II and Randomized MATRIX ALPS II with QR Factorization (referred shortly as MATRIX ALPS II with QR) presented in this paper.

10.2 Implementation Details

To properly compare the algorithms in the above list, we preset a set of parameters that are common. We denote the ratio between the number of observed samples and the number of variables in $\boldsymbol{X}^*$ as $\text{SR} := p/(m \cdot n)$ (sampling ratio). Furthermore, we reserve FR to represent the degree of freedom in a rank-k matrix to the number of observations—this corresponds to the following definition $\text{FR} := (k(m+n-k))/p$. In most of the experiments, we fix the number of observable data $p = 0.3mn$ and vary the dimensions and the rank k of the matrix $\boldsymbol{X}^*$. This way, we create a wide range of different problem configurations with variable FR.

Most of the algorithms in comparison as well as the proposed schemes are implemented in MATLAB. We note that the LMaFit software package contains parts implemented in C that reduce the per iteration computational time. This provides insights for further time savings in our schemes; we leave a fully optimized implementation of our algorithms as future work. In this paper, we mostly test cases where $m \ll n$. Such settings can be easily found in real-world problems such as recommender systems (e.g. Netflix, Amazon, etc.) where the number of products, movies, etc. is much greater than the number of active users.

In all algorithms, we fix the maximum number of iterations to 500, unless otherwise stated. To solve a least squares problem over a restricted low-rank subspace, we use conjugate gradients with maximum number of iterations given by cg_maxiter := 500 and tolerance parameter cg_tol := 10^{-10}. We use the same stopping criteria for the majority of algorithms under consideration:

$$\frac{\|\boldsymbol{X}(i) - \boldsymbol{X}(i-1)\|_F}{\|\boldsymbol{X}(i)\|_F} \leq \text{tol}, \tag{33}$$

where $\boldsymbol{X}(i)$, $\boldsymbol{X}(i-1)$ denote the current and the previous estimate of $\boldsymbol{X}^*$ and tol := 5×10^{-5}. If this is not the case, we tweak the algorithms to minimize the total execution time and achieve similar reconstruction performance as the rest of the algorithms. For SVD calculations, we use the lansvd implementation in PROPACK package [53]—moreover, all the algorithms in comparison use the same linear operators $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{A}}^*$ for gradient and SVD calculations and conjugate-gradient least-squares minimizations. For fairness, we modified all the algorithms so that they *exploit the true rank*. Small deviations from the true rank result in relatively small degradation in terms of the reconstruction performance. In case the rank of $\boldsymbol{X}^*$ is unknown, one has to predict the dimension of the principal singular space. The authors in [3], based on ideas in [48], propose to compute singular values incrementally until a significant gap between singular values is found. Similar strategies can be found in [18] for the convex case.

In MATRIX ALPS II and MATRIX ALPS II with QR, we perform $\mathcal{Q}_i \leftarrow \text{ortho}(\mathcal{X}_i \cup \mathcal{X}_{i+1})$ to construct a set of orthonormal rank-1 matrices that span the subspace, spanned by $\mathcal{X}_i \cup \mathcal{X}_{i+1}$. While such operation can be implemented using factorization procedures (such as SVD or QR decompositions), in practice this degrades the time complexity of the algorithm substantially as the rank k and the problem dimensionality increase. In our implementations, we simply *union* the set of orthonormal rank-1 matrices,

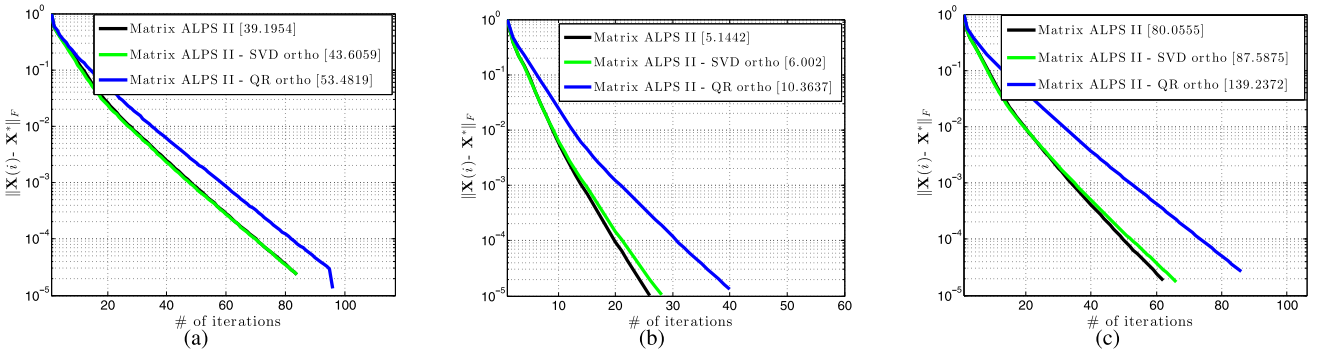

Fig. 5 Median error per iteration for MATRIX ALPS II variants over 10 Monte-Carlo repetitions. In brackets, we present the mean time consumed for convergene in seconds. (**a**) $n = 1024$, $m = 256$, $p = 0.25n^2$, and rank $k = 20$. (**b**) $n = 2048$, $m = 512$, $p = 0.25n^2$, and rank $k = 60$. (**c**) $n = 1000$, $m = 500$, $p = 0.25n^2$, and rank $k = 50$

without further orthogonalization. Thus, we employ *inexact* projections for computational efficiency which results in faster convergence. Figure 5 shows the time overhead due to the additional orthogonalization process. We compare three algorithms: MATRIX ALPS II (no orthogonalization step), MATRIX ALPS II using SVD for orthogonalization and, MATRIX ALPS II using QR for orthogonalization. In Figs. 5(a)–(b), we use subsampled and permuted noiselets for linear map $\mathcal{A}$ and in Fig. 5(c), we test the MC problem. In all the experimental cases considered in this work, we observed identical performance in terms of reconstruction accuracy for the three variants, as can be also seen in Fig. 5. To this end, for the rest of the paper, we use MATRIX ALPS II where $\mathcal{Q}_i \leftarrow \mathcal{X}_i \cup \mathcal{X}_{i+1}$.

10.3 Limitations of $\|\cdot\|_*$-Based Algorithms: A Toy Example

While nucluear norm heuristic is widely used in solving the low-rank minimization problem, [54] presents simple problem cases where convex, nuclear norm-based, algorithms *fail* in practice. Using the $\|\cdot\|_*$-norm in the objective function as the convex surrogate of the rank$(\cdot)$ metric might lead to a candidate set with multiple solutions, introducing ambiguity in the selection process. Borrowing the example in [54], we test the list of algorithms above on a toy problem setting that does not satisfy the rank-RIP. To this end, we design the following problem: let $\boldsymbol{X}^* \in \mathbb{R}^{5\times 4}$ be the matrix of interest with rank$(\boldsymbol{X}^*) = 2$, as shown in Fig. 6(a). We consider the case where we have access to $\boldsymbol{X}^*$ only through a subset of its entries, as shown in Fig. 6(b).

In Fig. 7, we present the reconstruction performance of various matrix completion solvers after 300 iterations. Although there are multiple solutions that induce the recovered matrix and have the same rank as $\boldsymbol{X}^*$, most of the algorithms in comparison reconstruct $\boldsymbol{X}^*$ successfully. We note that, in some cases, the inadequancy of an algorithm to reconstruct

Fig. 6 Matrix Completion toy example for $\boldsymbol{X}^* \in \mathbb{R}^{5\times 4}$. We use '?' to denote the unobserved entried

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix} \quad \begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ ? & ? & ? & 1 \\ 2 & ? & ? & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(a) (b)

$\boldsymbol{X}^*$ is not because of the (relaxed) problem formulation but due to its fast—but inaccurate—implementation (fast convergence versus reconstruction accuracy tradeoff).

10.4 Synthetic Data

General affine rank minimization using noiselets: In this experiment, the set of observations $\boldsymbol{y} \in \mathbb{R}^p$ satisfy:

$$\boldsymbol{y} = \mathcal{A}\boldsymbol{X}^* + \boldsymbol{\varepsilon}. \tag{34}$$

Here, we use permuted and subsampled noiselets for the linear operator $\mathcal{A}$ [12]. The signal $\boldsymbol{X}^*$ is generated as the multiplication of two low-rank matrices, $\mathbf{L} \in \mathbb{R}^{m\times k}$ and $\mathbf{R} \in \mathbb{R}^{n\times k}$, such that $\boldsymbol{X}^* = \mathbf{L}\mathbf{R}^T$ and $\|\boldsymbol{X}^*\|_F = 1$. Both $\mathbf{L}$ and $\mathbf{R}$ have random independent and identically distributed (iid) Gaussian entries with zero mean and unit variance. In the noisy case, the additive noise term $\boldsymbol{\varepsilon} \in \mathbb{R}^p$ contains entries drawn from a zero mean Gaussian distribution with $\|\boldsymbol{\varepsilon}\|_2 \in \{10^{-3}, 10^{-4}\}$.

We compare the following algorithms: SVP, ADMiRA, MATRIX ALPS I, MATRIX ALPS II and MATRIX ALPS II with QR for various problem configurations, as depicted in Table 1 (there is no available code with arbitrary sensing operators for the rest algorithms). In Table 1, we show the median values of reconstruction error, number of iterations and execution time over 50 Monte Carlo iterations. For all cases, we assume SR = 0.3 and we set the maximum number of iterations to 500. Bold font denotes the fastest execution time. Furthermore, Fig. 8 illustrates the effectiveness of the algorithms for some representative problem configurations.

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(a) SVT

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(b) FPC

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(c) SVP ($\mu = 1$)

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(d) ALM

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 0 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(e) OptSpace

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(f) SET

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(g) ADMiRA

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(h) GRASTA

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(i) LMatFit

$$\begin{pmatrix} 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

(j) MATRIX ALPS II

Fig. 7 Toy example reconstruction performance for various algorithms. We observe that $\boldsymbol{X}^*$ is an integer matrix—since the algorithms under consideration return real matrices as solutions, we round the solution elementwise

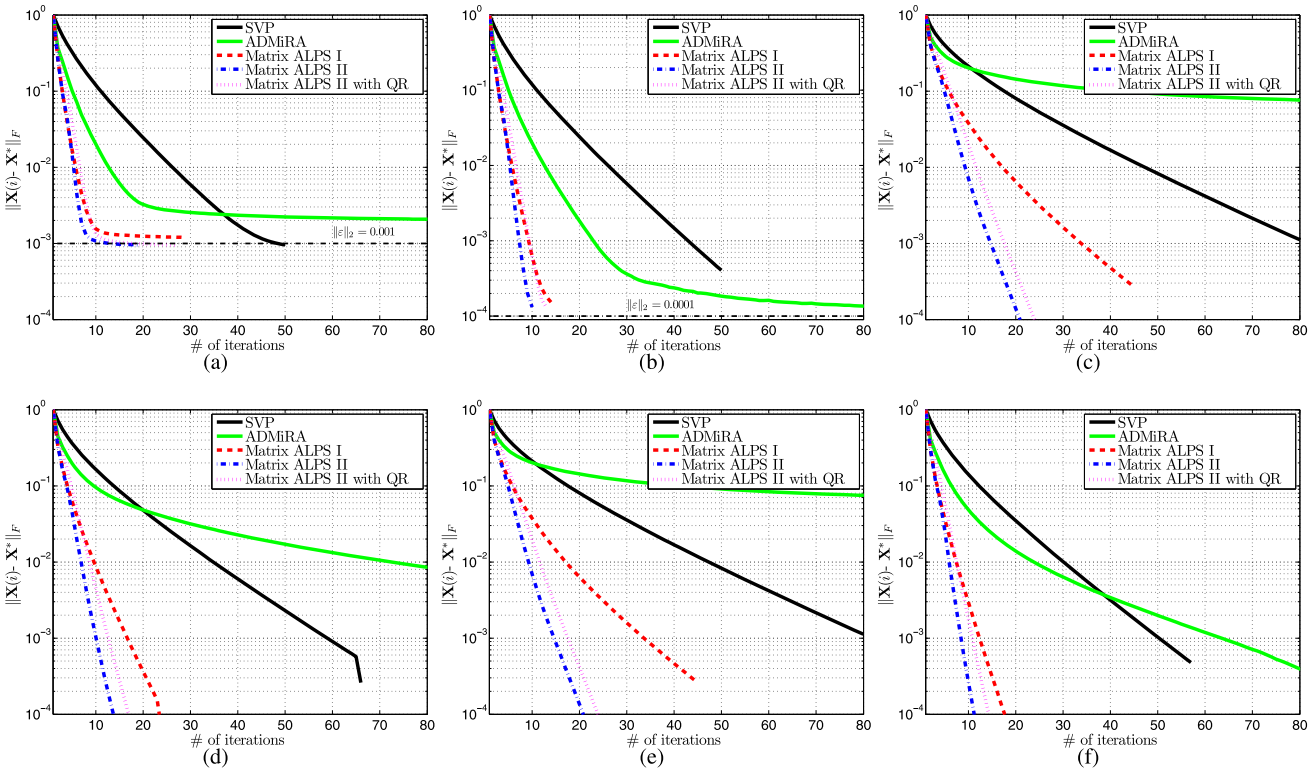

Fig. 8 Low rank signal reconstruction using noiselet linear operator. The error curves are the median values across 50 Monte-Carlo realizations over each iteration. For all cases, we assume $p = 0.3mn$. (**a**) $m = 256$, $n = 512$, $k = 10$ and $\|\boldsymbol{\varepsilon}\|_2 = 10^{-3}$. (**b**) $m = 256$, $n = 512$, $k = 10$ and $\|\boldsymbol{\varepsilon}\|_2 = 10^{-4}$. (**c**) $m = 256$, $n = 512$, $k = 20$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**d**) $m = 512$, $n = 1024$, $k = 30$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**e**) $m = 512$, $n = 1024$, $k = 40$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**f**) $m = 1024$, $n = 2048$, $k = 50$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$

In Table 1, MATRIX ALPS II and MATRIX ALPS II with QR obtain accurate low-rank solutions much faster than the rest of the algorithms in comparison. In high dimensional settings, MATRIX ALPS II with QR scales better as the problem dimensions increase, leading to faster convergence. Moreover, its execution time is at least a few orders of magnitude smaller compared to SVP, ADMiRA and MATRIX ALPS I implementations.

Robust matrix completion: We design matrix completion problems in the following way. The signal of interest $\boldsymbol{X}^* \in \mathbb{R}^{m\times n}$ is synthesized as a rank-k matrix, factorized as $\boldsymbol{X}^* := \mathbf{L}\mathbf{R}^T$ with $\|\boldsymbol{X}^*\|_F = 1$ where $\mathbf{L} \in \mathbb{R}^{m\times k}$ and

Table 1 General ARM using Noiselets

Configuration				FR	SVP			ADMiRA			MATRIX ALPS I		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
256	512	5	0	0.097	38	2.2×10^{-4}	0.78	27	4.4×10^{-5}	2.26	13.5	1×10^{-5}	0.7
256	512	5	10^{-3}	0.097	38	6×10^{-4}	0.91	700	2×10^{-3}	65.94	16	7×10^{-4}	0.92
256	512	5	10^{-4}	0.097	38	2.1×10^{-4}	0.94	700	4.1×10^{-4}	69.03	11.5	7.9×10^{-5}	0.72
256	512	10	0	0.193	50	3.4×10^{-4}	1.44	38	5×10^{-5}	4.42	13	3.9×10^{-5}	0.92
256	512	10	10^{-3}	0.193	50	9×10^{-4}	1.39	700	1.7×10^{-3}	56.94	29	1.2×10^{-3}	1.78
256	512	10	10^{-4}	0.193	50	3.5×10^{-4}	1.38	700	9.3×10^{-5}	64.69	14	1.4×10^{-4}	0.93
256	512	20	0	0.38	86	7×10^{-4}	3.32	700	4.1×10^{-5}	81.93	45	2×10^{-4}	4.09
256	512	20	10^{-3}	0.38	86	1.5×10^{-3}	3.45	700	4.2×10^{-2}	77.35	69	2.3×10^{-3}	5.05
256	512	20	10^{-4}	0.38	86	7×10^{-4}	3.26	700	4×10^{-2}	79.47	46	4×10^{-4}	4.1
512	1024	30	0	0.287	66	4.9×10^{-4}	8.79	295	5.4×10^{-5}	143.53	24	1×10^{-4}	8.01
512	1024	40	0	0.38	86	7×10^{-4}	10.09	700	4.3×10^{-2}	251.27	45	2×10^{-4}	11.08
1024	2048	50	0	0.24	57	4.3×10^{-4}	42.88	103	5.2×10^{-5}	312.62	18	5.7×10^{-5}	35.86

Configuration				FR	MATRIX ALPS II			MATRIX ALPS II with QR		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time
256	512	5	0	0.097	8	7.1×10^{-6}	0.42	10	9.1×10^{-6}	**0.39**
256	512	5	10^{-3}	0.097	9	7×10^{-4}	**0.56**	20	7×10^{-4}	0.93
256	512	5	10^{-4}	0.097	8	7×10^{-5}	0.5	10	7.8×10^{-5}	**0.46**
256	512	10	0	0.193	10	2.3×10^{-5}	0.68	13	2.4×10^{-5}	**0.64**
256	512	10	10^{-3}	0.193	19	1×10^{-3}	**1.29**	27	1×10^{-3}	1.35
256	512	10	10^{-4}	0.193	10	1.1×10^{-4}	0.68	13	1.1×10^{-4}	**0.62**
256	512	20	0	0.38	21	1×10^{-4}	1.92	24	1×10^{-4}	**1.26**
256	512	20	10^{-3}	0.38	36	1.5×10^{-3}	2.67	39	1.5×10^{-3}	**1.69**
256	512	20	10^{-4}	0.38	21	2×10^{-4}	1.87	24	2×10^{-4}	**1.22**
512	1024	30	0	0.287	14	4.5×10^{-5}	4.7	18	3.3×10^{-5}	**4.15**
512	1024	40	0	0.38	21	1×10^{-4}	6.01	24	1×10^{-4}	**4.53**
1024	2048	50	0	0.24	12	2.5×10^{-5}	22.76	15	3.3×10^{-5}	**17.94**

$\mathbf{R} \in \mathbb{R}^{n \times k}$ as defined above. In sequence, we subsample $\boldsymbol{X}^*$ by observing $p = 0.3mn$ entries, drawn uniformly at random. We denote the set of ordered pairs that represent the coordinates of the observable entries as $\Omega = \{(i, j) : [\boldsymbol{X}^*]_{ij} \text{ is known}\} \subseteq \{1, \ldots, m\} \times \{1, \ldots, n\}$ and let $\boldsymbol{\mathcal{A}}_\Omega$ denote the linear operator (mask) that samples a matrix according to Ω. Then, the set of observations satisfies:

$$\boldsymbol{y} = \boldsymbol{\mathcal{A}}_\Omega \boldsymbol{X}^* + \boldsymbol{\varepsilon}, \tag{35}$$

i.e., the known entries of $\boldsymbol{X}^*$ are structured as a vector $\boldsymbol{y} \in \mathbb{R}^p$, disturbed by a dense noise vector $\boldsymbol{\varepsilon} \in \mathbb{R}^p$ with fixed-energy, which is populated by iid zero-mean Gaussians.

To demonstrate the reconstruction accuracy and the convergence speeds, we generate various problem configurations (both noisy and noiseless settings), according to (35). The energy of the additive noise takes values $\|\boldsymbol{\varepsilon}\|_2 \in \{10^{-3}, 10^{-4}\}$. All the algorithms are tested for the same signal-matrix-noise realizations. A summary of the results can be found in Tables 2, 3 and, 4 where we present the median values of reconstruction error, number of iterations and execution time over 50 Monte Carlo iterations. For all cases, we assume SR = 0.3 and set the maximum number of iterations to 700. Bold font denotes the fastest execution time. Some convergence error curves for specific cases are illustrated in Figs. 9 and 10.

In Table 2, LMaFit [52] implementation has the fastest convergence for small scale problem configuration where $m = 300$ and $n = 600$. We note that part of LMaFit implementation uses C code for acceleration. GROUSE [49] is a competitive low-rank recovery method with attractive execution times for the *extreme low rank* problem settings due to stochastic gradient descent techniques. Nevertheless,

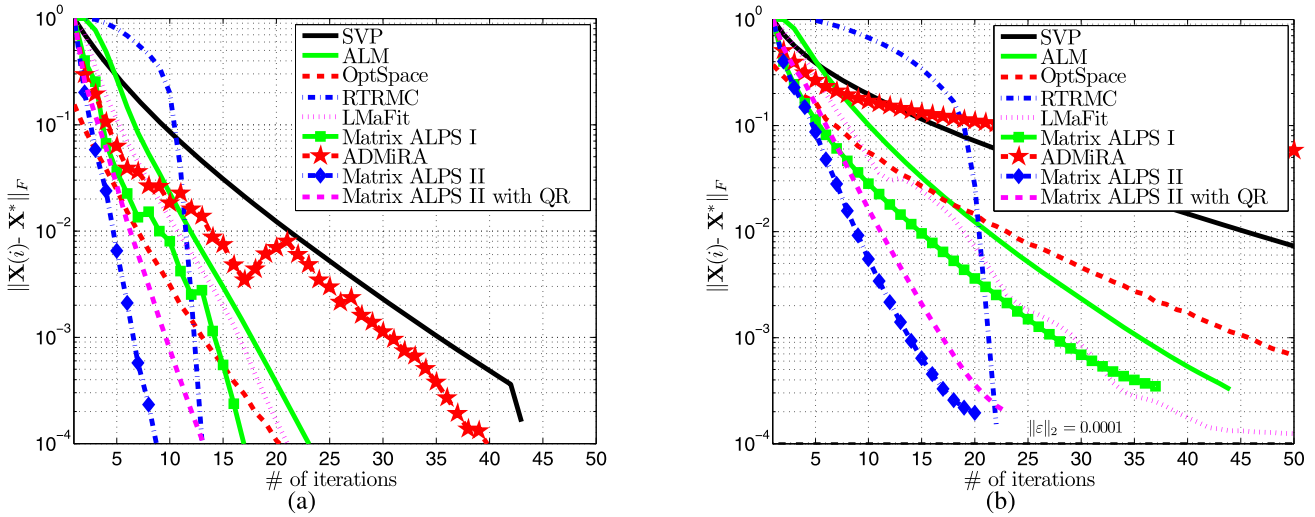

Fig. 9 Low rank matrix recovery for the matrix completion problem. The error curves are the median values across 50 Monte-Carlo realizations over each iteration. For all cases, we assume $p = 0.3mn$. (**a**) $m = 300$, $n = 600$, $k = 5$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**b**) $m = 300$, $n = 600$, $k = 20$ and $\|\boldsymbol{\varepsilon}\|_2 = 10^{-4}$

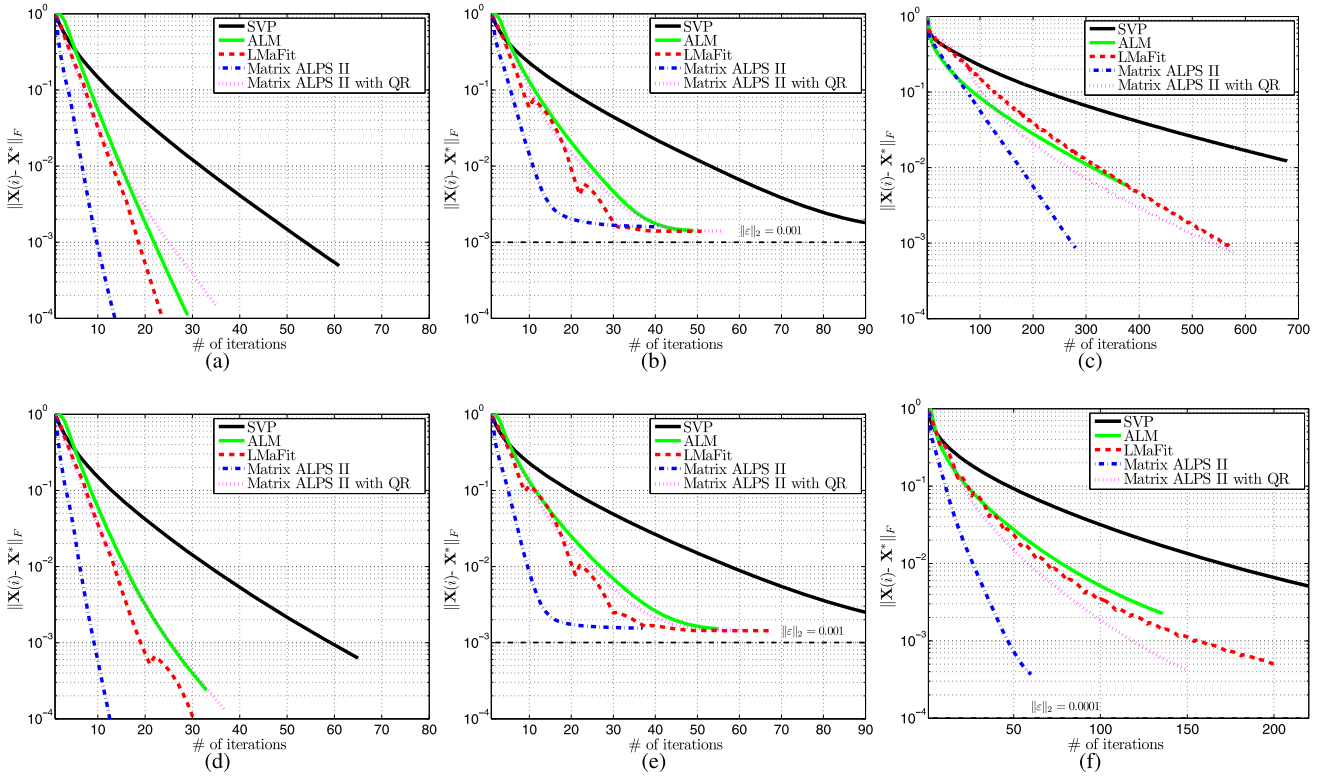

Fig. 10 Low rank matrix recovery for the matrix completion problem. The error curves are the median values across 50 Monte-Carlo realizations over each iteration. For all cases, we assume $p = 0.3mn$. (**a**) $m = 700$, $n = 1000$, $k = 30$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**b**) $m = 700$, $n = 1000$, $k = 50$ and $\|\boldsymbol{\varepsilon}\|_2 = 10^{-3}$. (**c**) $m = 700$, $n = 1000$, $k = 110$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**d**) $m = 500$, $n = 2000$, $k = 10$ and $\|\boldsymbol{\varepsilon}\|_2 = 0$. (**e**) $m = 500$, $n = 2000$, $k = 50$ and $\|\boldsymbol{\varepsilon}\|_2 = 10^{-3}$. (**f**) $m = 500$, $n = 2000$, $k = 80$ and $\|\boldsymbol{\varepsilon}\|_2 = 10^{-4}$

its execution time performance degrades significantly as we increase the rank of $\boldsymbol{X}^*$. Moreover, we observe how randomized low rank projections accelerate the convergence speed where MATRIX ALPS II with QR converges faster than MATRIX ALPS II. In Tables 3 and 4, we increase the problem dimensions. Here, MATRIX ALPS II with QR has

Table 2 Matrix Completion problem for $m = 300$ and $n = 600$. "−" depicts no information or not applicable due to time overhead

Configuration				FR	SVP			GROUSE			TFOCS		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
300	600	5	0	0.083	43	2.9×10^{-4}	0.59	–	1.52×10^{-4}	0.08	–	8.69×10^{-5}	3.36
300	600	5	10^{-3}	0.083	42	6×10^{-4}	0.65	–	2×10^{-4}	0.082	–	5×10^{-4}	3.85
300	600	5	10^{-4}	0.083	43	3×10^{-4}	0.64	–	2×10^{-4}	0.079	–	1×10^{-4}	3.5
300	600	10	0	0.165	54	4×10^{-4}	0.9	–	4.5×10^{-6}	0.22	–	2×10^{-4}	6.43
300	600	10	10^{-3}	0.165	54	9×10^{-4}	0.89	–	2×10^{-4}	0.16	–	8×10^{-4}	7.83
300	600	10	10^{-4}	0.165	54	4×10^{-4}	0.91	–	2×10^{-4}	0.16	–	1×10^{-4}	6.75
300	600	20	0	0.326	85	8×10^{-4}	2.04	–	1×10^{-4}	0.81	–	2×10^{-4}	30.04
300	600	40	0	0.637	241	3.4×10^{-3}	11.1	–	3.1×10^{-3}	13.94	–	–	–
					Inexact ALM			OptSpace			GRASTA		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
300	600	5	0	0.083	24	6.7×10^{-5}	0.47	31	2.8×10^{-6}	2.41	–	2.2×10^{-4}	2.07
300	600	5	10^{-3}	0.083	24	6×10^{-4}	0.49	297	5×10^{-4}	22.82	–	1×10^{-4}	2.07
300	600	5	10^{-4}	0.083	24	1×10^{-4}	0.49	267	1×10^{-4}	21.56	–	8×10^{-5}	2.1
300	600	10	0	0.165	26	1×10^{-4}	0.6	37	2.3×10^{-6}	8.42	–	8.6×10^{-6}	4.5
300	600	10	10^{-3}	0.165	26	8×10^{-4}	0.59	304	8×10^{-4}	66.02	–	5.5×10^{-3}	3.43
300	600	10	10^{-4}	0.165	26	1×10^{-4}	0.61	304	1×10^{-4}	65.56	–	5.3×10^{-3}	3.44
300	600	20	0	0.326	44	3×10^{-4}	1.37	–	–	–	–	5×10^{-4}	10.51
300	600	40	0	0.637	134	1.6×10^{-3}	7.08	–	–	–	–	5.2×10^{-3}	251.34
					RTRMC			LMaFit			MATRIX ALPS I		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
300	600	5	0	0.083	13	1.2×10^{-4}	0.59	20	2.2×10^{-4}	**0.054**	22	1.8×10^{-5}	0.76
300	600	5	10^{-3}	0.083	13	1×10^{-4}	0.59	19	5×10^{-4}	**0.049**	37	7×10^{-4}	1.34
300	600	5	10^{-4}	0.083	13	2×10^{-4}	0.59	21	1×10^{-4}	**0.052**	18	1×10^{-4}	0.61
300	600	10	0	0.165	16	1.1×10^{-3}	1.03	23	1×10^{-4}	**0.064**	16	1×10^{-4}	0.65
300	600	10	10^{-3}	0.165	17	1×10^{-4}	1.09	26	8×10^{-4}	**0.077**	30	1.1×10^{-3}	1.16
300	600	10	10^{-4}	0.165	17	2×10^{-4}	1.09	32	1×10^{-4}	**0.097**	16	1×10^{-4}	0.63
300	600	20	0	0.326	22	4×10^{-4}	2.99	37	2×10^{-4}	**0.12**	37	2×10^{-4}	2.05
300	600	40	0	0.637	35	3×10^{-5}	11.83	233	4.9×10^{-4}	**2.52**	500	6.5×10^{-2}	45.67
					ADMiRA			MATRIX ALPS II			MATRIX ALPS II with QR		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
300	600	5	0	0.083	59	5.2×10^{-5}	2.86	10	1.7×10^{-5}	0.34	14	3.2×10^{-5}	0.45
300	600	5	10^{-3}	0.083	700	4×10^{-3}	30.96	12	6×10^{-4}	0.44	24	6×10^{-4}	0.81
300	600	5	10^{-4}	0.083	700	4.5×10^{-3}	31.45	10	1×10^{-4}	0.36	14	1×10^{-4}	0.47
300	600	10	0	0.165	47	1×10^{-3}	2.56	12	3×10^{-5}	0.48	16	3.4×10^{-5}	0.49
300	600	10	10^{-3}	0.165	700	1.5×10^{-3}	28.49	19	9×10^{-4}	0.74	29	9×10^{-4}	0.95
300	600	10	10^{-4}	0.165	700	1×10^{-4}	31.99	12	1×10^{-4}	0.49	16	1×10^{-4}	0.54
300	600	20	0	0.326	700	1.2×10^{-3}	41.86	20	1×10^{-4}	1.16	23	1×10^{-4}	0.79
300	600	20	0	0.326	–	–	–	72	2×10^{-4}	7.21	68	2×10^{-4}	2.6

faster convergence for most of the cases and scales well as the problem size increases. We note that we do not exploit stochastic gradient descent techniques in the recovery process to accelerate convergence which is left for future work.

10.5 Real Data

We use real data images to highlight the reconstruction performance of the proposed schemes. To this end, we perform grayscale image denoising from an incomplete set of ob-

Table 3 Matrix Completion problem for $m = 700$ and $n = 1000$. "–" depicts no information or not applicable due to time overhead

Configuration				FR	SVP			Inexact ALM			GROUSE		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
700	1000	5	0	0.04	34	1.9×10^{-4}	1.77	23	6.5×10^{-5}	1.69	–	3.5×10^{-5}	**0.23**
700	1000	5	10^{-3}	0.04	34	4.2×10^{-4}	1.92	23	3.7×10^{-4}	1.87	–	3.1×10^{-4}	**0.24**
700	1000	30	0	0.239	61	4.6×10^{-4}	6.39	29	1.2×10^{-4}	3.91	–	3.2×10^{-5}	3.15
700	1000	30	10^{-3}	0.239	61	1.1×10^{-3}	6.33	29	1×10^{-3}	3.87	–	8×10^{-4}	3.14
700	1000	50	0	0.393	95	8.5×10^{-4}	14.47	49	3.2×10^{-4}	9.02	–	1.3×10^{-5}	10.31
700	1000	50	10^{-3}	0.393	95	1.6×10^{-3}	15.15	49	1.4×10^{-3}	9.11	–	8×10^{-4}	10.34
700	1000	110	0	0.833	683	1.2×10^{-2}	253.1	374	5.8×10^{-3}	152.61	–	1.2×10^{-1}	110.93
700	1000	110	10^{-3}	0.833	682	1.3×10^{-2}	256.21	374	6.8×10^{-3}	154.34	–	1.05×10^{-1}	111.05

					LMaFit			MATRIX ALPS II			MATRIX ALPS II with QR		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
700	1000	5	0	0.04	24	7.2×10^{-6}	0.67	8	1.5×10^{-5}	1.15	15	8.3×10^{-5}	1.05
700	1000	5	10^{-3}	0.04	17	3.7×10^{-4}	0.5	10	4.5×10^{-4}	1.38	15	3.8×10^{-4}	1.1
700	1000	30	0	0.239	34	9.2×10^{-6}	**1.95**	14	4.5×10^{-5}	3.69	35	1.1×10^{-4}	2.6
700	1000	30	10^{-3}	0.239	30	1×10^{-3}	**1.71**	25	1.1×10^{-3}	6.1	35	1×10^{-3}	2.61
700	1000	50	0	0.393	53	2.7×10^{-5}	4.59	25	8.6×10^{-5}	8.87	57	1.6×10^{-5}	**4.47**
700	1000	50	10^{-3}	0.393	52	1.4×10^{-3}	4.53	40	1.6×10^{-3}	14.38	57	1.4×10^{-3}	**4.49**
700	1000	110	0	0.833	584	9×10^{-4}	101.95	280	8×10^{-4}	214.93	553	7×10^{-4}	**51.72**
700	1000	110	10^{-3}	0.833	584	3.7×10^{-3}	102.15	336	4.7×10^{-3}	261.98	551	3.7×10^{-3}	**51.62**

served pixels—similar experiments can be found in [52]. Based on the matrix completion setting, we observe a limited number of pixels from the original image and perform a low rank approximation based only on the set of measurements. While the true underlying image might not be low-rank, we apply our solvers to obtain low-rank approximations.

Figures 11 and 12 depict the reconstruction results. In the first test case, we use a 512×512 grayscale image as shown in the top left corner of Fig. 11. For this case, we observe only the 35 % of the total number of pixels, randomly selected—a realization is depicted in the top right plot in Fig. 11. In sequel, we fix the desired rank to $k = 40$. The best rank-40 approximation using SVD is shown in the top middle of Fig. 11 where the full set of pixels is observed. Given a fixed common tolerance and the same stopping criteria, Fig. 11 shows the recovery performance achieved by a range of algorithms under consideration for 10 Monte-Carlo realizations. We repeat the same experiment for the second image in Fig. 12. Here, the size of the image is 256×256, the desired rank is set to $k = 30$ and we observe the 33 % of the image pixels. In contrast to the image denoising procedure above, we measure the reconstruction error of the computed solutions with respect to the *best rank*-30 *approximation* of the true image. In both cases, we note that MATRIX ALPS II has a better phase transition performance as compared to the rest of the algorithms.

11 Discussion

In this paper, we present new strategies and review existing ones for hard thresholding methods to recover low-rank matrices from dimensionality reducing, linear projections. Our discussion revolves around four basic building blocks that exploit the problem structure to reduce computational complexity without sacrificing stability.

In theory, constant μ_i selection schemes are accompanied with strong RIP constant conditions but empirical evidence reveal signal reconstruction vulnerabilities. While convergence derivations of adaptive schemes are characterized by weaker bounds, the performance gained by this choice in terms of convergence rate, is quite significant. Memory-based methods lead to convergence speed with (almost) no extra cost on the complexity of hard thresholding methods—we provide theoretical evidence for convergence for simple cases but more theoretical justification is needed to generalize this part as future work. Lastly, further estimate refinement over low rank subspaces using gradient update steps or pseudoinversion optimization techniques provides signal reconstruction efficacy, but more computational power is needed per iteration.

Table 4 Matrix Completion problem for $m = 500$ and $n = 2000$. "–" depicts no information or not applicable due to time overhead

Configuration				FR	SVP			Inexact ALM			GROUSE		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
500	2000	30	0	0.083	64	5.3×10^{-4}	10.18	32	1.9×10^{-4}	6.47	–	1.6×10^{-4}	**2.46**
500	2000	30	10^{-3}	0.083	64	1.1×10^{-3}	6.69	32	1×10^{-3}	4.51	–	6×10^{-4}	**1.94**
500	2000	30	10^{-4}	0.083	64	5.4×10^{-4}	10.14	32	2.2×10^{-4}	6.51	–	1.6×10^{-4}	**2.46**
500	2000	50	0	0.408	103	1.1×10^{-4}	15.74	54	5×10^{-4}	10.8	–	8×10^{-5}	7.32
500	2000	50	10^{-3}	0.408	103	1.8×10^{-3}	24.97	54	1.55×10^{-3}	16.14	–	9×10^{-4}	8.6
500	2000	50	10^{-4}	0.408	102	1.1×10^{-3}	24.85	54	5×10^{-4}	16.17	–	7×10^{-5}	8.59
500	2000	80	0	0.645	239	3.5×10^{-3}	92.91	134	1.7×10^{-3}	59.33	–	1×10^{-4}	79.64
500	2000	80	10^{-3}	0.645	239	4.2×10^{-3}	94.86	134	2.8×10^{-3}	60.68	–	1×10^{-4}	79.98
500	2000	80	10^{-4}	0.645	239	3.6×10^{-3}	93.95	134	1.8×10^{-3}	60.76	–	1×10^{-4}	79.48
500	2000	100	0	0.8	523	1.1×10^{-2}	259.13	307	6×10^{-3}	173.14	–	4.5×10^{-2}	143.41
500	2000	100	10^{-3}	0.8	525	1.2×10^{-2}	262.19	308	7×10^{-3}	176.04	–	5.2×10^{-2}	142.85
500	2000	100	10^{-4}	0.8	523	1.1×10^{-2}	262.11	307	6×10^{-3}	170.47	–	5.1×10^{-2}	144.78

					LMaFit			MATRIX ALPS II			MATRIX ALPS II with QR		
m	n	k	$\|\boldsymbol{\varepsilon}\|_2$		iter.	err.	time	iter.	err.	time	iter.	err.	time
500	2000	30	0	0.083	37	1.3×10^{-5}	3.05	13	3.1×10^{-5}	4.84	37	1.2×10^{-5}	4.04
500	2000	30	10^{-3}	0.083	37	1×10^{-3}	2.52	22	1.1×10^{-3}	5.35	37	1×10^{-3}	3.32
500	2000	30	10^{-4}	0.083	35	1×10^{-4}	2.86	13	1.3×10^{-4}	4.85	37	1.6×10^{-4}	4.05
500	2000	50	0	0.408	60	6×10^{-5}	6.06	22	1×10^{-4}	7.6	60	2×10^{-4}	**5.67**
500	2000	50	10^{-3}	0.408	60	1.4×10^{-3}	7.26	36	1.6×10^{-3}	19.64	59	1.6×10^{-3}	**6.91**
500	2000	50	10^{-4}	0.408	60	2×10^{-4}	7.29	22	2×10^{-4}	11.87	59	2×10^{-4}	**6.75**
500	2000	80	0	0.645	183	3×10^{-4}	33.65	61	2×10^{-4}	49.53	151	3×10^{-4}	**18.66**
500	2000	80	10^{-3}	0.645	183	2.3×10^{-3}	33.48	92	2.4×10^{-3}	75.51	151	2.3×10^{-3}	**18.87**
500	2000	80	10^{-4}	0.645	183	3×10^{-4}	33.47	61	4×10^{-4}	49.52	151	3×10^{-4}	**18.92**
500	2000	100	0	0.8	519	1.5×10^{-3}	115.11	148	4×10^{-4}	153.74	429	7×10^{-4}	**55.1**
500	2000	100	10^{-3}	0.8	529	3.6×10^{-3}	117.7	228	3.7×10^{-3}	239.92	427	3.4×10^{-3}	**55.7**
500	2000	100	10^{-3}	0.8	520	1.6×10^{-3}	116.66	148	6×10^{-4}	154.46	428	8×10^{-4}	**55.07**

We connect ϵ-approximation low-rank revealing schemes with first-order gradient descent algorithms to solve general affine rank minimization problems; to the best of our knowledge, this is the first attempt to theoretically characterize the performance of iterative greedy algorithms with ϵ-approximation schemes. In all cases, experimental results illustrate the effectiveness of the proposed schemes on different problem configurations.

Acknowledgements This work was supported in part by the European Commission under Grant MIRG-268398, ERC Future Proof, SNF 200021-132548 and DARPA KeCoM program #11-DARPA-1055. VC also would like to acknowledge Rice University for his Faculty Fellowship.

Appendix

Remark 1 Let $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ with SVD: $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, and $\boldsymbol{Y} \in \mathbb{R}^{m \times n}$ with SVD: $\boldsymbol{Y} = \widetilde{\boldsymbol{U}}\widetilde{\boldsymbol{\Sigma}}\widetilde{\boldsymbol{V}}^T$. Assume two sets: (i) $\mathcal{S}_1 = \{\boldsymbol{u}_i\boldsymbol{u}_i^T : i \in \mathcal{I}_1\}$ where $\boldsymbol{u}_i$ is the i-th singular vector of $\boldsymbol{X}$ and $\mathcal{I}_1 \subseteq \{1, \ldots, \text{rank}(\boldsymbol{X})\}$ and, (ii) $\mathcal{S}_2 = \{\boldsymbol{u}_i\boldsymbol{u}_i^T, \tilde{\boldsymbol{u}}_j\tilde{\boldsymbol{u}}_j{}^T : i \in \mathcal{I}_2,\ j \in \mathcal{I}_3\}$ where $\tilde{\boldsymbol{u}}_i$ is the i-th singular vector of $\boldsymbol{Y}$, $\mathcal{I}_1 \subseteq \mathcal{I}_2 \subseteq \{1, \ldots, \text{rank}(\boldsymbol{X})\}$ and, $\mathcal{I}_3 \subseteq \{1, \ldots, \text{rank}(\boldsymbol{Y})\}$. We observe that the subspaces defined by $\boldsymbol{u}_i\boldsymbol{u}_i^T$ and $\tilde{\boldsymbol{u}}_j\tilde{\boldsymbol{u}}_j{}^T$ are not necessarily orthogonal.

To this end, let $\widehat{\mathcal{S}}_2 = \text{ortho}(\mathcal{S}_2)$; this operation can be easily computed via SVD. Then, the following commutativity property holds true for any matrix $\boldsymbol{W} \in \mathbb{R}^{m \times n}$:

$$\mathcal{P}_{\mathcal{S}_1}\mathcal{P}_{\widehat{\mathcal{S}}_2}\boldsymbol{W} = \mathcal{P}_{\widehat{\mathcal{S}}_2}\mathcal{P}_{\mathcal{S}_1}\boldsymbol{W}. \tag{36}$$

A.1 Proof of Lemma 6

Given $\mathcal{X}^* \leftarrow \mathcal{P}_k(\boldsymbol{X}^*)$ using SVD factorization, we define the following quantities: $\mathcal{S}_i \leftarrow \mathcal{X}_i \cup \mathcal{D}_i$, $\mathcal{S}_i^* \leftarrow \text{ortho}(\mathcal{X}_i \cup \mathcal{X}^*)$. Then, given the structure of the sets $\mathcal{S}_i$ and $\mathcal{S}_i^*$

$$\mathcal{P}_{\mathcal{S}_i}\mathcal{P}_{(\mathcal{S}_i^*)^\perp} = \mathcal{P}_{\mathcal{D}_i}\mathcal{P}_{(\mathcal{X}^* \cup \mathcal{X}_i)^\perp}, \tag{37}$$

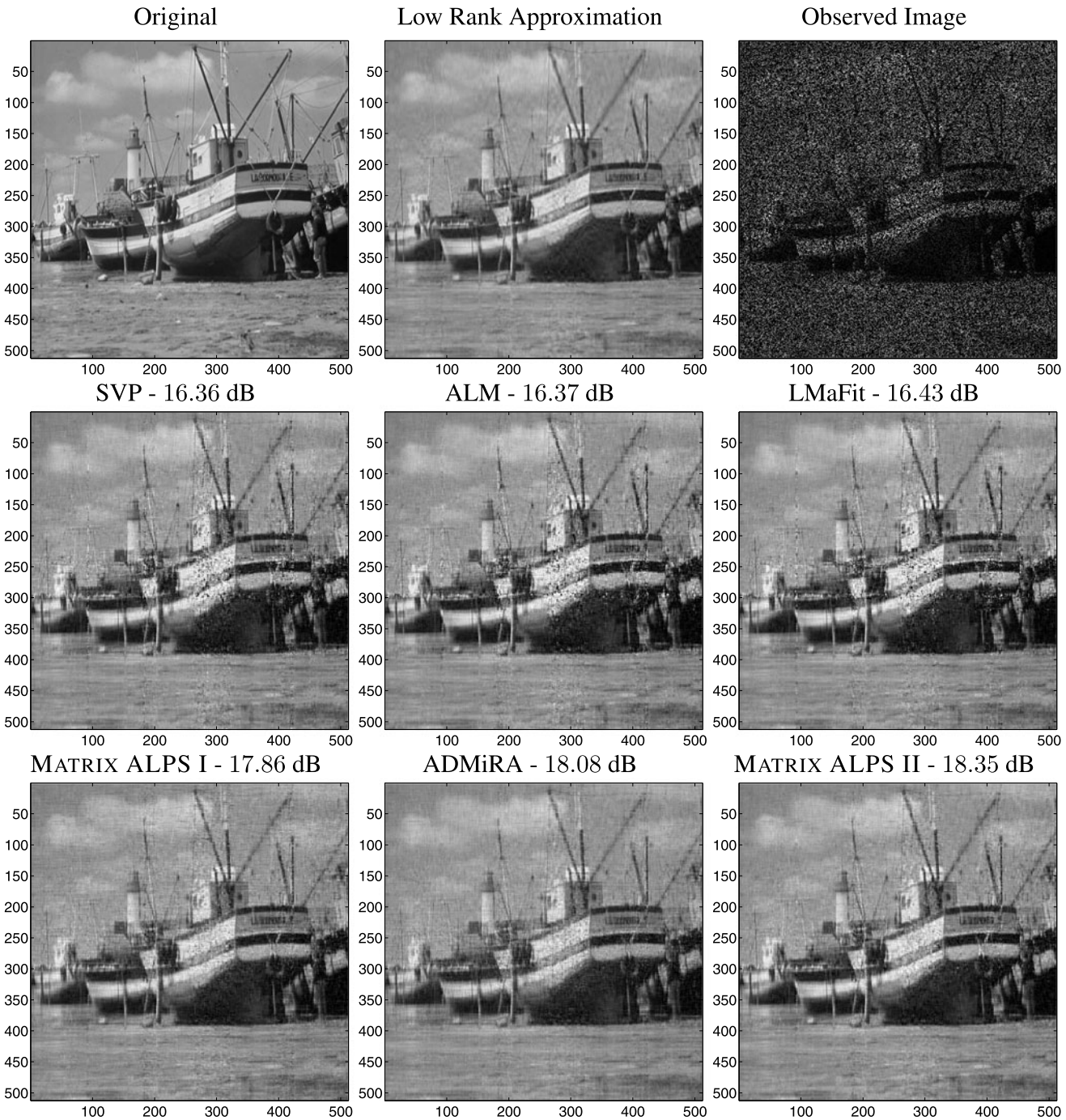

Fig. 11 Reconstruction performance in image denoising settings. The image size is 512×512 and the desired rank is preset to $k = 40$. We observe 35 % of the pixels of the true image. We depict the median reconstruction error with respect to the true image in dB over 10 Monte Carlo realizations

and

$$\mathcal{P}_{\mathcal{S}_i^*}\mathcal{P}_{\mathcal{S}_i^\perp} = \mathcal{P}_{\mathcal{X}^*}\mathcal{P}_{(\mathcal{D}_i \cup \mathcal{X}_i)^\perp}. \tag{38}$$

Since the subspace defined in $\mathcal{D}_i$ is the best rank-k subspace, orthogonal to the subspace spanned by $\mathcal{X}_i$, the following holds true:

$$\begin{aligned} &\left\|\mathcal{P}_{\mathcal{D}_i}\mathcal{P}_{\mathcal{X}_i^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \geq \left\|\mathcal{P}_{\mathcal{X}^*}\mathcal{P}_{\mathcal{X}_i^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\ \Rightarrow \quad &\left\|\mathcal{P}_{\mathcal{S}_i}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \geq \left\|\mathcal{P}_{\mathcal{S}_i^*}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2. \end{aligned}$$

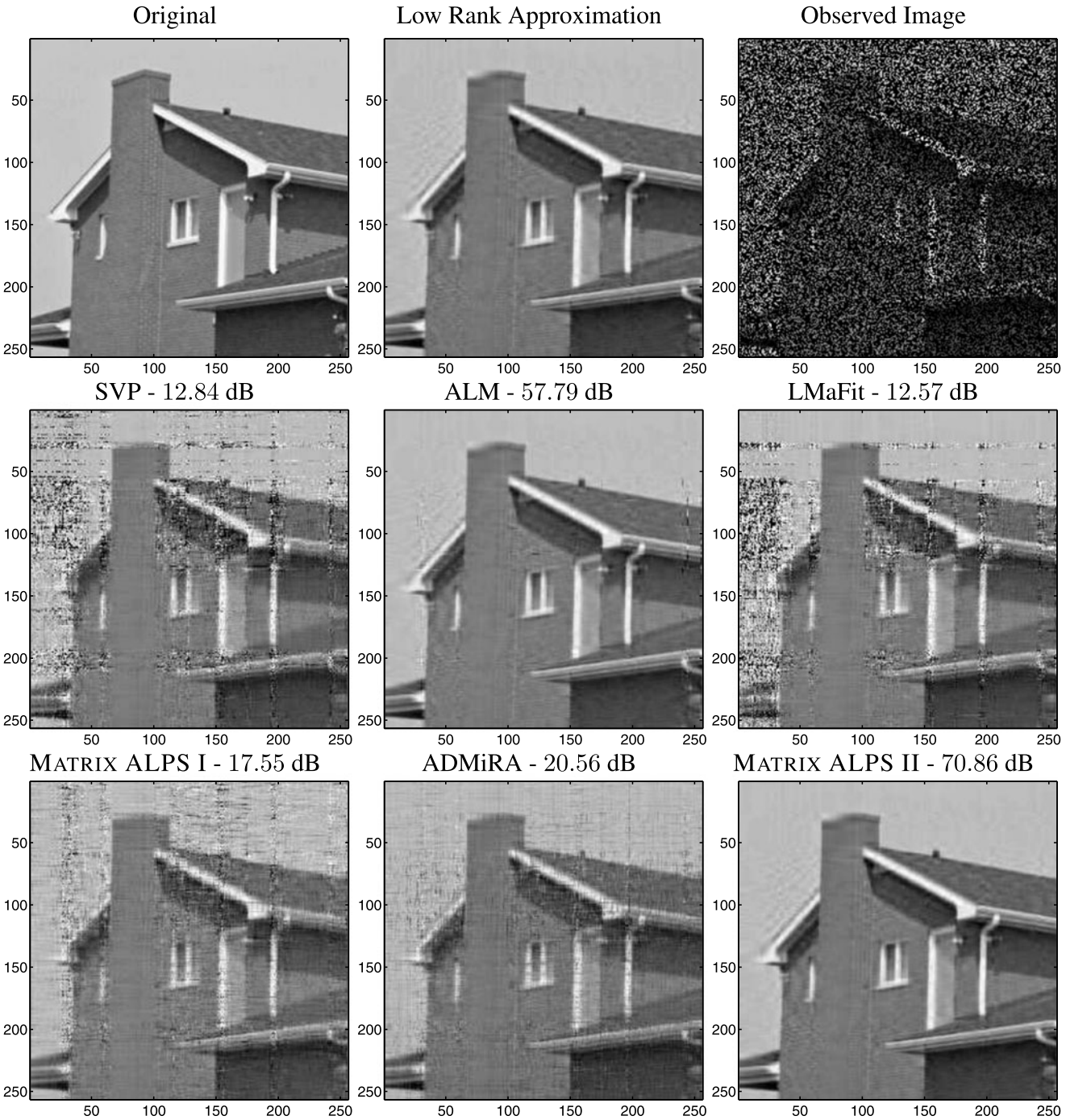

Fig. 12 Reconstruction performance in image denoising settings. The image size is 256×256 and the desired rank is preset to $k = 30$. We observe 33 % of the pixels of the best rank-30 approximation of the image. We depict the median reconstruction with respect to the best rank-30 approximation in dB over 10 Monte Carlo realizations

Removing the common subspaces in $\mathcal{S}_i$ and $\mathcal{S}_i^*$ by the commutativity property of the projection operation and using the shortcut $\mathcal{P}_{\mathcal{A}\setminus\mathcal{B}} \equiv \mathcal{P}_{\mathcal{A}}\mathcal{P}_{\mathcal{B}^\perp}$ for sets $\mathcal{A}$, $\mathcal{B}$, we get:

$$\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \geq \left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2$$

$$\Rightarrow \quad \left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\boldsymbol{\mathcal{A}}^*\boldsymbol{\mathcal{A}}\big(\boldsymbol{X}^* - \boldsymbol{X}(i)\big) + \mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\boldsymbol{\mathcal{A}}^*\boldsymbol{\varepsilon}\right\|_F$$
$$\geq \left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\boldsymbol{\mathcal{A}}^*\boldsymbol{\mathcal{A}}\big(\boldsymbol{X}^* - \boldsymbol{X}(i)\big) + \mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\boldsymbol{\mathcal{A}}^*\boldsymbol{\varepsilon}\right\|_F. \qquad (39)$$

Next, we assume that $\mathcal{P}_{(\mathcal{A}\setminus\mathcal{B})^\perp}$ denotes the orthogonal projection onto the subspace spanned by $\mathcal{P}_{\mathcal{A}}\mathcal{P}_{\mathcal{B}^\perp}$. Then, on the

left hand side of (39), we have:

$$\begin{aligned}
&\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)+\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\overset{\text{(i)}}{\leq}\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\overset{\text{(ii)}}{=}\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)+\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\overset{\text{(iii)}}{=}\left\|\big(\mathbf{I}-\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\big)\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right.\\
&\quad\left.+\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{(\mathcal{S}_i\setminus\mathcal{S}_i^*)^\perp}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\leq\left\|\big(\mathbf{I}-\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\big)\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{(\mathcal{S}_i\setminus\mathcal{S}_i^*)^\perp}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\overset{\text{(iv)}}{\leq}\delta_{3k}\left\|\boldsymbol{X}^*-\boldsymbol{X}(i)\right\|_F+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\quad+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{(\mathcal{S}_i\setminus\mathcal{S}_i^*)^\perp}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\overset{\text{(v)}}{\leq}\delta_{3k}\left\|\boldsymbol{X}^*-\boldsymbol{X}(i)\right\|_F+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\quad+\delta_{3k}\left\|\mathcal{P}_{(\mathcal{S}_i\setminus\mathcal{S}_i^*)^\perp}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\overset{\text{(vi)}}{\leq}2\delta_{3k}\left\|\boldsymbol{X}^*-\boldsymbol{X}(i)\right\|_F+\left\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F, \qquad (40)
\end{aligned}$$

where (i) due to triangle inequality over Frobenius metric norm, (ii) since $\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}(\boldsymbol{X}(i)-\boldsymbol{X}^*)=\mathbf{0}$, (iii) by using the fact that $\boldsymbol{X}(i)-\boldsymbol{X}^*:=\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}(\boldsymbol{X}(i)-\boldsymbol{X}^*)+\mathcal{P}_{(\mathcal{S}_i\setminus\mathcal{S}_i^*)^\perp}(\boldsymbol{X}(i)-\boldsymbol{X}^*)$, (iv) due to Lemma 4, (v) due to Lemma 5 and (vi) since $\|\mathcal{P}_{(\mathcal{S}_i\setminus\mathcal{S}_i^*)^\perp}(\boldsymbol{X}^*-\boldsymbol{X}(i))\|_F\leq\|\boldsymbol{X}(i)-\boldsymbol{X}^*\|_F$.

For the right hand side of (39), we calculate:

$$\begin{aligned}
&\left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)+\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\geq\left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad-\left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{(\mathcal{S}_i^*\setminus\mathcal{S}_i)^\perp}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad-\left\|\big(\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}-\mathbf{I}\big)\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F\\
&\quad-\left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F\\
&\geq\left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\big(\boldsymbol{X}^*-\boldsymbol{X}(i)\big)\right\|_F-2\delta_{2k}\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F\\
&\quad-\left\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F \qquad (41)
\end{aligned}$$

by using Lemmas 4 and 5. Combining (40) and (41) in (39), we get:

$$\begin{aligned}
\left\|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{S}_i}\boldsymbol{X}^*\right\|_F&\leq(2\delta_{2k}+2\delta_{3k})\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F\\
&\quad+\sqrt{2(1+\delta_{2k})}\|\boldsymbol{\varepsilon}\|_2.
\end{aligned}$$

A.2 Proof of Theorem 1

Let $\mathcal{X}^*\leftarrow\mathcal{P}_k(\boldsymbol{X}^*)$ be a set of orthonormal, rank-1 matrices that span the range of $\boldsymbol{X}^*$. In Algorithm 1, $\boldsymbol{W}(i)\leftarrow\mathcal{P}_k(\boldsymbol{V}(i))$. Thus:

$$\begin{aligned}
&\left\|\boldsymbol{W}(i)-\boldsymbol{V}(i)\right\|_F^2\leq\left\|\boldsymbol{X}^*-\boldsymbol{V}(i)\right\|_F^2\\
&\quad\Rightarrow\quad\left\|\boldsymbol{W}(i)-\boldsymbol{X}^*+\boldsymbol{X}^*-\boldsymbol{V}(i)\right\|_F^2\leq\left\|\boldsymbol{X}^*-\boldsymbol{V}(i)\right\|_F^2\\
&\quad\Rightarrow\quad\left\|\boldsymbol{W}(i)-\boldsymbol{X}^*\right\|_F^2\leq2\big\langle\boldsymbol{W}(i)-\boldsymbol{X}^*,\boldsymbol{V}(i)-\boldsymbol{X}^*\big\rangle. \qquad (42)
\end{aligned}$$

From Algorithm 1, (i) $\boldsymbol{V}(i)\in\operatorname{span}(\mathcal{S}_i)$, (ii) $\boldsymbol{X}(i)\in\operatorname{span}(\mathcal{S}_i)$ and (iii) $\boldsymbol{W}(i)\in\operatorname{span}(\mathcal{S}_i)$. We define $\mathcal{E}\leftarrow\operatorname{ortho}(\mathcal{S}_i\cup\mathcal{X}^*)$ where $\operatorname{rank}(\operatorname{span}(\mathcal{E}))\leq3k$ and let $\mathcal{P}_{\mathcal{E}}$ be the orthogonal projection onto the subspace defined by $\mathcal{E}$.

Since $\boldsymbol{W}(i)-\boldsymbol{X}^*\in\operatorname{span}(\mathcal{E})$ and $\boldsymbol{V}(i)-\boldsymbol{X}^*\in\operatorname{span}(\mathcal{E})$, the following hold true:

$$\boldsymbol{W}(i)-\boldsymbol{X}^*=\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{W}(i)-\boldsymbol{X}^*\big)\quad\text{and}$$
$$\boldsymbol{V}(i)-\boldsymbol{X}^*=\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big).$$

Then, (42) can be written as:

$$\begin{aligned}
&\left\|\boldsymbol{W}(i)-\boldsymbol{X}^*\right\|_F^2\leq2\big\langle\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{W}(i)-\boldsymbol{X}^*\big),\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\rangle\quad\Rightarrow\\
&=\underbrace{2\big\langle\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{W}(i)-\boldsymbol{X}^*\big),\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i)-\boldsymbol{X}^*-\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\big(\boldsymbol{X}(i)-\boldsymbol{X}^*\big)\big)\big\rangle}_{\doteq A}+\underbrace{2\mu_i\big\langle\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{W}(i)-\boldsymbol{X}^*\big),\mathcal{P}_{\mathcal{E}}\mathcal{P}_{\mathcal{S}_i}\big(\mathcal{A}^*\boldsymbol{\varepsilon}\big)\big\rangle}_{\doteq B}. \qquad (43)
\end{aligned}$$

In B, we observe:

$$\begin{aligned}
B&:=2\mu_i\big\langle\mathcal{P}_{\mathcal{E}}\big(\boldsymbol{W}(i)-\boldsymbol{X}^*\big),\mathcal{P}_{\mathcal{E}}\mathcal{P}_{\mathcal{S}_i}\big(\mathcal{A}^*\boldsymbol{\varepsilon}\big)\big\rangle\\
&\overset{\text{(i)}}{=}2\mu_i\big\langle\boldsymbol{W}(i)-\boldsymbol{X}^*,\mathcal{P}_{\mathcal{S}_i}\big(\mathcal{A}^*\boldsymbol{\varepsilon}\big)\big\rangle\\
&\overset{\text{(ii)}}{\leq}2\mu_i\left\|\boldsymbol{W}(i)-\boldsymbol{X}^*\right\|_F\left\|\mathcal{P}_{\mathcal{S}_i}\big(\mathcal{A}^*\boldsymbol{\varepsilon}\big)\right\|_F\\
&\overset{\text{(iii)}}{\leq}2\mu_i\sqrt{1+\delta_{2k}}\left\|\boldsymbol{W}(i)-\boldsymbol{X}^*\right\|_F\|\boldsymbol{\varepsilon}\|_2, \qquad (44)
\end{aligned}$$

where (i) holds since $\mathcal{P}_{\mathcal{S}_i}\mathcal{P}_{\mathcal{E}}=\mathcal{P}_{\mathcal{E}}\mathcal{P}_{\mathcal{S}_i}=\mathcal{P}_{\mathcal{S}_i}$ for $\operatorname{span}(\mathcal{S}_i)\in\operatorname{span}(\mathcal{E})$, (ii) is due to Cauchy-Schwarz inequality and, (iii) is easily derived using Lemma 2.

In A, we perform the following motions:

$$\begin{aligned}
A &:= 2\big\langle \boldsymbol{W}(i) - \boldsymbol{X}^*, \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \\
&\quad - \mu_i \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big)\big\rangle \\
&\overset{\text{(i)}}{=} 2\big\langle \boldsymbol{W}(i) - \boldsymbol{X}^*, \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \\
&\quad - \mu_i \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} [\mathcal{P}_{\mathcal{S}_i} + \mathcal{P}_{\mathcal{S}_i^\perp}] \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big)\big\rangle \\
&= 2\big\langle \boldsymbol{W}(i) - \boldsymbol{X}^*, \big(\mathbf{I} - \mu_i \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i}\big) \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big)\big\rangle \\
&\quad - 2\mu_i \big\langle \boldsymbol{W}(i) - \boldsymbol{X}^*, \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big)\big\rangle \\
&\overset{\text{(ii)}}{\leq} 2\big\| \boldsymbol{W}(i) - \boldsymbol{X}^* \big\|_F \\
&\quad \times \big\| \big(\mathbf{I} - \mu_i \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i}\big) \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F \\
&\quad + 2\mu_i \big\| \boldsymbol{W}(i) - \boldsymbol{X}^* \big\|_F \\
&\quad \times \big\| \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F, \qquad (45)
\end{aligned}$$

where (i) is due to $\mathcal{P}_{\mathcal{E}}(\boldsymbol{X}(i) - \boldsymbol{X}^*) := \mathcal{P}_{\mathcal{S}_i} \mathcal{P}_{\mathcal{E}}(\boldsymbol{X}(i) - \boldsymbol{X}^*) + \mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}(\boldsymbol{X}(i) - \boldsymbol{X}^*)$ and (ii) follows from Cauchy-Schwarz inequality. Since $\frac{1}{1+\delta_{2k}} \leq \mu_i \leq \frac{1}{1-\delta_{2k}}$, Lemma 4 implies:

$$\begin{aligned}
\lambda\big(\mathbf{I} - \mu_i \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i}\big) &\in \left[1 - \frac{1-\delta_{2k}}{1+\delta_{2k}}, \frac{1+\delta_{2k}}{1-\delta_{2k}} - 1\right] \\
&\leq \frac{2\delta_{2k}}{1-\delta_{2k}},
\end{aligned}$$

and thus:

$$\begin{aligned}
&\big\| \big(\mathbf{I} - \mu_i \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i}\big) \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F \\
&\quad \leq \frac{2\delta_{2k}}{1-\delta_{2k}} \big\| \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F.
\end{aligned}$$

Furthermore, according to Lemma 5:

$$\begin{aligned}
&\big\| \mathcal{P}_{\mathcal{S}_i} \boldsymbol{\mathcal{A}}^* \boldsymbol{\mathcal{A}} \mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F \\
&\quad \leq \delta_{3k} \big\| \mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F
\end{aligned}$$

since $\operatorname{rank}(\mathcal{P}_{\mathcal{K}} \boldsymbol{X}) \leq 3k, \forall \boldsymbol{X} \in \mathbb{R}^{m \times n}$ for $\mathcal{K} \leftarrow \operatorname{ortho}(\mathcal{E} \cup \mathcal{S}_i)$. Since $\mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}(\boldsymbol{X}(i) - \boldsymbol{X}^*) = \mathcal{P}_{\mathcal{X}^* \setminus (\mathcal{D}_i \cup \mathcal{X}_i)} \boldsymbol{X}^*$ where

$$\mathcal{D}_i \leftarrow \mathcal{P}_k\big(\mathcal{P}_{\mathcal{X}_i^\perp} \nabla f\big(\boldsymbol{X}(i)\big)\big),$$

then:

$$\begin{aligned}
&\big\| \mathcal{P}_{\mathcal{S}_i^\perp} \mathcal{P}_{\mathcal{E}}\big(\boldsymbol{X}(i) - \boldsymbol{X}^*\big) \big\|_F \\
&\quad = \big\| \mathcal{P}_{\mathcal{X}^* \setminus (\mathcal{D}_i \cup \mathcal{X}_i)} \boldsymbol{X}^* \big\|_F \\
&\quad \leq (2\delta_{2k} + 2\delta_{3k}) \big\| \boldsymbol{X}(i) - \boldsymbol{X}^* \big\|_F + \sqrt{2(1+\delta_{2k})} \|\boldsymbol{\varepsilon}\|_2,
\end{aligned}$$

using Lemma 6. Combining the above in (45), we compute:

$$\begin{aligned}
A &\leq \left(\frac{4\delta_{2k}}{1-\delta_{2k}} + (2\delta_{2k} + 2\delta_{3k}) \frac{2\delta_{3k}}{1-\delta_{2k}} \right) \big\| \boldsymbol{W}(i) - \boldsymbol{X}^* \big\|_F \\
&\quad \times \big\| \boldsymbol{X}(i) - \boldsymbol{X}^* \big\|_F \\
&\quad + \frac{2\delta_{3k}}{1-\delta_{2k}} \big\| \boldsymbol{W}(i) - \boldsymbol{X}^* \big\|_F \sqrt{2(1+\delta_{2k})} \|\boldsymbol{\varepsilon}\|_2. \qquad (46)
\end{aligned}$$

Combining (44) and (46) in (43), we get:

$$\begin{aligned}
&\big\| \boldsymbol{W}(i) - \boldsymbol{X}^* \big\|_F \\
&\quad \leq \left(\frac{4\delta_{2k}}{1-\delta_{2k}} + (2\delta_{2k} + 2\delta_{3k}) \frac{2\delta_{3k}}{1-\delta_{2k}} \right) \big\| \boldsymbol{X}(i) - \boldsymbol{X}^* \big\|_F \\
&\qquad + \left(\frac{2\sqrt{1+\delta_{2k}}}{1-\delta_{2k}} + \frac{2\delta_{3k}}{1-\delta_{2k}} \sqrt{2(1+\delta_{2k})} \right) \|\boldsymbol{\varepsilon}\|_2. \qquad (47)
\end{aligned}$$

Focusing on steps 5 and 6 of Algorithm 1, we perform similar motions to obtain:

$$\begin{aligned}
\big\| \boldsymbol{X}(i+1) - \boldsymbol{X}^* \big\|_F &\leq \left(\frac{1+2\delta_{2k}}{1-\delta_{2k}} \right) \big\| \boldsymbol{W}(i) - \boldsymbol{X}^* \big\|_F \\
&\quad + \frac{\sqrt{1+\delta_k}}{1-\delta_k} \|\boldsymbol{\varepsilon}\|_2. \qquad (48)
\end{aligned}$$

Combining the recursions in (47) and (48), we finally compute:

$$\big\| \boldsymbol{X}(i+1) - \boldsymbol{X}^* \big\|_F \leq \rho \big\| \boldsymbol{X}(i) - \boldsymbol{X}^* \big\|_F + \gamma \|\boldsymbol{\varepsilon}\|_2,$$

for $\rho := (\frac{1+2\delta_{2k}}{1-\delta_{2k}})(\frac{4\delta_{2k}}{1-\delta_{2k}} + (2\delta_{2k} + 2\delta_{3k}) \frac{2\delta_{3k}}{1-\delta_{2k}})$ and

$$\begin{aligned}
\gamma := &\left(\left(\frac{1+2\delta_{2k}}{1-\delta_{2k}} \right) \left(\frac{2\sqrt{1+\delta_{2k}}}{1-\delta_{2k}} + \frac{2\delta_{3k}}{1-\delta_{2k}} \sqrt{2(1+\delta_{2k})} \right) \right. \\
&\left. + \frac{\sqrt{1+\delta_k}}{1-\delta_k} \right).
\end{aligned}$$

For the convergence parameter ρ, further compute:

$$\begin{aligned}
&\left(\frac{1+2\delta_{2k}}{1-\delta_{2k}} \right) \left(\frac{4\delta_{2k}}{1-\delta_{2k}} + (2\delta_{2k} + 2\delta_{3k}) \frac{2\delta_{3k}}{1-\delta_{2k}} \right) \\
&\quad \leq \frac{1+2\delta_{3k}}{(1-\delta_{3k})^2} \big(4\delta_{3k} + 8\delta_{3k}^2\big) =: \hat{\rho} \qquad (49)
\end{aligned}$$

for $\delta_k \leq \delta_{2k} \leq \delta_{3k}$. Calculating the roots of this expression, we easily observe that $\rho < \hat{\rho} < 1$ for $\delta_{3k} < 0.1235$.

A.3 Proof of Theorem 2

Before we present the proof of Theorem 2, we list a series of lemmas that correspond to the motions Algorithm 2 performs.

Lemma 9 [Error norm reduction via least-squares optimization] *Let $\mathcal{S}_i$ be a set of orthonormal, rank-1 matrices that span a rank-2k subspace in $\mathbb{R}^{m\times n}$. Then, the least squares solution $\boldsymbol{V}(i)$ given by*:

$$\boldsymbol{V}(i) \leftarrow \underset{\boldsymbol{V}:\boldsymbol{V}\in\text{span}(\mathcal{S}_i)}{\arg\min} \|\boldsymbol{y}-\mathcal{A}\boldsymbol{V}\|_2^2, \tag{50}$$

satisfies:

$$\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \le \frac{1}{\sqrt{1-\delta_{3k}^2(\mathcal{A})}}\left\|\mathcal{P}_{\mathcal{S}_i^\perp}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F + \frac{\sqrt{1+\delta_{2k}}}{1-\delta_{3k}}\|\boldsymbol{\varepsilon}\|_2. \tag{51}$$

Proof We observe that $\|\boldsymbol{V}(i)-\boldsymbol{X}^*\|_F^2$ is decomposed as follows:

$$\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F^2 = \left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F^2 + \left\|\mathcal{P}_{\mathcal{S}_i^\perp}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F^2. \tag{52}$$

In (50), $\boldsymbol{V}(i)$ is the minimizer over the low-rank subspace spanned by $\mathcal{S}_i$ with $\text{rank}(\text{span}(\mathcal{S}_i)) \le 2k$. Using the optimality condition (Lemma 1) over the convex set $\Theta = \{\boldsymbol{X} : \text{span}(\boldsymbol{X}) \in \mathcal{S}_i\}$, we have:

$$\left\langle \nabla f\left(\boldsymbol{V}(i)\right), \mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{X}^*-\boldsymbol{V}(i)\right)\right\rangle \ge 0 \quad \Rightarrow \quad \left\langle \mathcal{A}\boldsymbol{V}(i)-\boldsymbol{y}, \mathcal{A}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle \le 0 \tag{53}$$

for $\mathcal{P}_{\mathcal{S}_i}\boldsymbol{X}^* \in \text{span}(\mathcal{S}_i)$. Given condition (53), the first term on the right hand side of (52) becomes:

$$\begin{aligned}
&\left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F^2 \\
&= \left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle \\
&\overset{(53)}{\le} \left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle \\
&\quad - \left\langle \mathcal{A}\boldsymbol{V}(i)-\boldsymbol{y}, \mathcal{A}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle \\
&\le \left|\left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \left(\mathbf{I}-\mathcal{A}^*\mathcal{A}\right)\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right| \\
&\quad + \left\langle \boldsymbol{\varepsilon}, \mathcal{A}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle.
\end{aligned} \tag{54}$$

Focusing on the term $|\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, (\mathbf{I}-\mathcal{A}^*\mathcal{A})\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{V}(i)-\boldsymbol{X}^*)\rangle|$, we derive the following:

$$\begin{aligned}
&\left|\left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \left(\mathbf{I}-\mathcal{A}^*\mathcal{A}\right)\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right| \\
&= \left|\left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle \right. \\
&\quad \left. - \left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right| \\
&\overset{(i)}{=} \left|\left\langle \mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right), \mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right. \\
&\quad \left. - \left\langle \mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right), \mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right|
\end{aligned}$$

$$\begin{aligned}
&\overset{(ii)}{=} \left|\left\langle \mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right), \mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right. \\
&\quad \left. - \left\langle \mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right), \mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right| \\
&= \left|\left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \right.\right. \\
&\quad \left.\left. \left(\mathbf{I}-\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\right)\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right|,
\end{aligned}$$

where (i) follows from the facts that $\boldsymbol{V}(i)-\boldsymbol{X}^* \in \text{span}(\text{ortho}(\mathcal{S}_i \cup \mathcal{X}^*))$ and thus $\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}(\boldsymbol{V}(i)-\boldsymbol{X}^*) = \boldsymbol{V}(i)-\boldsymbol{X}^*$ and (ii) is due to $\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{P}_{\mathcal{S}_i} = \mathcal{P}_{\mathcal{S}_i}$ since $\text{span}(\mathcal{S}_i) \subseteq \text{span}(\text{ortho}(\mathcal{S}_i \cup \mathcal{X}^*))$. Then, (54) becomes:

$$\begin{aligned}
&\left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F^2 \\
&\le \left|\left\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \right.\right. \\
&\quad \left.\left. \left(\mathbf{I}-\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\right)\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle\right| \\
&\quad + \left\langle \boldsymbol{\varepsilon}, \mathcal{A}\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\rangle \\
&\overset{(i)}{\le} \left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \\
&\quad \times \left\|\left(\mathbf{I}-\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i\cup\mathcal{X}^*}\right)\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F \\
&\quad + \left\|\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F \left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F \\
&\overset{(ii)}{\le} \delta_{3k}\left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F \left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \\
&\quad + \sqrt{1+\delta_{2k}}\left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F \|\boldsymbol{\varepsilon}\|_2,
\end{aligned} \tag{55}$$

where (i) comes from Cauchy-Swartz inequality and (ii) is due to Lemmas 2 and 4. Simplifying the above quadratic expression, we obtain:

$$\left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F \le \delta_{3k}\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F + \sqrt{1+\delta_{2k}}\|\boldsymbol{\varepsilon}\|_2. \tag{56}$$

As a consequence, (52) can be upper bounded by:

$$\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F^2 \le \left(\delta_{3k}\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F + \sqrt{1+\delta_{2k}}\|\boldsymbol{\varepsilon}\|_2\right)^2 + \left\|\mathcal{P}_{\mathcal{S}_i^\perp}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F^2. \tag{57}$$

We form the quadratic polynomial for this inequality assuming as unknown variable the quantity $\|\boldsymbol{V}(i)-\boldsymbol{X}^*\|_F$. Bounding by the largest root of the resulting polynomial, we get:

$$\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \le \frac{1}{\sqrt{1-\delta_{3k}^2(\mathcal{A})}}\left\|\mathcal{P}_{\mathcal{S}_i^\perp}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F + \frac{\sqrt{1+\delta_{2k}}}{1-\delta_{3k}}\|\boldsymbol{\varepsilon}\|_2. \tag{58}$$

□

The following lemma characterizes how subspace *pruning* affects the recovered energy:

Lemma 10 [Best rank-k subspace selection] *Let $\boldsymbol{V}(i) \in \mathbb{R}^{m\times n}$ be a rank-$2k$ proxy matrix in the subspace spanned by $\mathcal{S}_i$ and let $\boldsymbol{X}(i+1) \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$ denote the best rank-k approximation to $\boldsymbol{V}(i)$, according to (5). Then*:

$$\begin{aligned}\big\|\boldsymbol{X}(i+1)-\boldsymbol{V}(i)\big\|_F &\le \big\|\mathcal{P}_{\mathcal{S}_i}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\|_F \\ &\le \big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F. \end{aligned} \tag{59}$$

Proof Since $\boldsymbol{X}(i+1)$ denotes the best rank-k approximation to $\boldsymbol{V}(i)$, the following inequality holds for any rank-k matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ in the subspace spanned by $\mathcal{S}_i$, i.e. $\forall \boldsymbol{X} \in \operatorname{span}(\mathcal{S}_i)$:

$$\big\|\boldsymbol{X}(i+1)-\boldsymbol{V}(i)\big\|_F \le \big\|\boldsymbol{X}-\boldsymbol{V}(i)\big\|_F. \tag{60}$$

Since $\mathcal{P}_{\mathcal{S}_i}\boldsymbol{V}(i) = \boldsymbol{V}(i)$, the left inequality in (59) is satisfied for $\boldsymbol{X} := \mathcal{P}_{\mathcal{S}_i}\boldsymbol{X}^*$ in (60). □

Lemma 11 *Let $\boldsymbol{V}(i)$ be the least squares solution in Step 2 of the ADMiRA algorithm and let $\boldsymbol{X}(i+1)$ be a proxy, rank-k matrix to $\boldsymbol{V}(i)$ according to*: $\boldsymbol{X}(i+1) \leftarrow \mathcal{P}_k(\boldsymbol{V}(i))$. *Then, $\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\|_F$ can be expressed in terms of the distance from $\boldsymbol{V}(i)$ to $\boldsymbol{X}^*$ as follows*:

$$\begin{aligned}\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F &\le \sqrt{1+3\delta_{3k}^2}\big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F \\ &\quad+\sqrt{1+3\delta_{3k}^2}\sqrt{\frac{3(1+\delta_{2k})}{1+3\delta_{3k}^2}}\|\boldsymbol{\varepsilon}\|_2. \end{aligned} \tag{61}$$

Proof We observe the following

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F^2 \\ &\quad= \big\|\boldsymbol{X}(i+1)-\boldsymbol{V}(i)+\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F^2 \\ &\quad= \big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F^2+\big\|\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\|_F^2 \\ &\qquad-2\big\langle\boldsymbol{V}(i)-\boldsymbol{X}^*,\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\rangle. \end{aligned} \tag{62}$$

Focusing on the right hand side of expression (62), $\langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \boldsymbol{V}(i)-\boldsymbol{X}(i+1)\rangle = \langle \boldsymbol{V}(i)-\boldsymbol{X}^*, \mathcal{P}_{\mathcal{S}_i}(\boldsymbol{V}(i)-\boldsymbol{X}(i+1))\rangle$ can be similarly analysed as in Lemma 10 where we obtain the following expression:

$$\begin{aligned}&\big|\big\langle\boldsymbol{V}(i)-\boldsymbol{X}^*,\mathcal{P}_{\mathcal{S}_i}\big(\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big)\big\rangle\big| \\ &\quad\le \delta_{3k}\big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F\big\|\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\|_F \\ &\qquad+\sqrt{1+\delta_{2k}}\big\|\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\|_F\|\boldsymbol{\varepsilon}\|_2. \end{aligned} \tag{63}$$

Now, expression (62) can be further transformed as:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F^2 \\ &\overset{\text{(i)}}{\le} \big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F^2+\big\|\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\|_F^2 \\ &\qquad+2\big(\delta_{3k}\big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F\big\|\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\|_F \\ &\qquad+\sqrt{1+\delta_{2k}}\big\|\boldsymbol{V}(i)-\boldsymbol{X}(i+1)\big\|_F\|\boldsymbol{\varepsilon}\|_2\big), \end{aligned} \tag{64}$$

where (i) is due to (63). Using Lemma 10, we further have:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F^2 \\ &\quad\le \big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F^2+\big\|\mathcal{P}_{\mathcal{S}_i}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\|_F^2 \\ &\qquad+2\big(\delta_{3k}\big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_F\big\|\mathcal{P}_{\mathcal{S}_i}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\|_F \\ &\qquad+\sqrt{1+\delta_{2k}}\big\|\mathcal{P}_{\mathcal{S}_i}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\|_F\|\boldsymbol{\varepsilon}\|_2\big). \end{aligned} \tag{65}$$

Furthermore, replacing $\|\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{X}^*-\boldsymbol{V}(i))\|_F$ with its upper bound defined in (56), we get:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_2^2 \\ &\overset{\text{(i)}}{\le} \big(1+3\delta_{3k}^2\big)\left(\big\|\boldsymbol{V}(i)-\boldsymbol{X}^*\big\|_2+\sqrt{\frac{3(1+\delta_{2k})}{1+3\delta_{3k}^2}}\|\boldsymbol{\varepsilon}\|\right)^2, \end{aligned} \tag{66}$$

where (i) is obtained by completing the squares and eliminating negative terms. □

Applying basic algebra tools in (61) and (51), we get:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F \\ &\quad\le \sqrt{\frac{1+3\delta_{3k}^2}{1-\delta_{3k}^2}}\big\|\mathcal{P}_{\mathcal{S}_i^\perp}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\|_F \\ &\qquad+\left(\frac{\sqrt{1+3\delta_{3k}^2}}{1-\delta_{3k}}+\sqrt{3}\right)\sqrt{1+\delta_{2k}}\|\boldsymbol{\varepsilon}\|_2. \end{aligned}$$

Since $\boldsymbol{V}(i) \in \operatorname{span}(\mathcal{S}_i)$, we observe $\mathcal{P}_{\mathcal{S}_i^\perp}(\boldsymbol{V}(i)-\boldsymbol{X}^*) = -\mathcal{P}_{\mathcal{S}_i^\perp}\boldsymbol{X}^* = -\mathcal{P}_{\mathcal{X}^*\setminus(\mathcal{D}_i\cup\mathcal{X}_i)}\boldsymbol{X}^*$. Then, using Lemma 6, we obtain:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F \\ &\quad\le (2\delta_{2k}+2\delta_{3k})\sqrt{\frac{1+3\delta_{3k}^2}{1-\delta_{3k}^2}}\big\|\boldsymbol{X}^*-\boldsymbol{X}(i)\big\|_F \\ &\qquad+\left[\sqrt{\frac{1+3\delta_{3k}^2}{1-\delta_{3k}^2}}\sqrt{2(1+\delta_{3k})}\right. \\ &\qquad\left.+\left(\frac{\sqrt{1+3\delta_{3k}^2}}{1-\delta_{3k}}+\sqrt{3}\right)\sqrt{1+\delta_{2k}}\right]\|\boldsymbol{\varepsilon}\|_2. \end{aligned} \tag{67}$$

Given $\delta_{2k} \le \delta_{3k}$, ρ is upper bounded by $\rho < 4\delta_{3k}\sqrt{\frac{1+3\delta_{3k}}{1-\delta_{3k}^2}}$. Then, $4\delta_{3k}\sqrt{\frac{1+3\delta_{3k}}{1-\delta_{3k}^2}} < 1 \Leftrightarrow \delta_{3k} < 0.2267$.

A.4 Proof of Theorem 3

Let $\mathcal{X}^* \leftarrow \mathcal{P}_k(\boldsymbol{X}^*)$ be a set of orthonormal, rank-1 matrices that span the range of $\boldsymbol{X}^*$. In Algorithm 3, $\boldsymbol{X}(i+1)$ is the best rank-k approximation of $\boldsymbol{V}(i)$. Thus:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{V}(i)\big\|_F^2 \le \big\|\boldsymbol{X}^*-\boldsymbol{V}(i)\big\|_F^2\\ &\quad\Rightarrow\quad \big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F^2\\ &\quad\le 2\big\langle \boldsymbol{X}(i+1)-\boldsymbol{X}^*, \boldsymbol{V}(i)-\boldsymbol{X}^*\big\rangle. \end{aligned} \tag{68}$$

From Algorithm 3, (i) $\boldsymbol{V}(i) \in \text{span}(\mathcal{S}_i)$, (ii) $\boldsymbol{Q}_i \in \text{span}(\mathcal{S}_i)$ and (iii) $\boldsymbol{W}(i) \in \text{span}(\mathcal{S}_i)$. We define $\mathcal{E} \leftarrow \text{ortho}(\mathcal{S}_i \cup \mathcal{X}^*)$ where we observe $\text{rank}(\text{span}(\mathcal{E})) \le 4k$ and let $\mathcal{P}_\mathcal{E}$ be the orthogonal projection onto the subspace defined by $\mathcal{E}$.

Since $\boldsymbol{X}(i+1)-\boldsymbol{X}^* \in \text{span}(\mathcal{E})$ and $\boldsymbol{V}(i)-\boldsymbol{X}^* \in \text{span}(\mathcal{E})$, the following hold true:

$$\boldsymbol{X}(i+1)-\boldsymbol{X}^* = \mathcal{P}_\mathcal{E}\big(\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big),$$

and,

$$\boldsymbol{V}(i)-\boldsymbol{X}^* = \mathcal{P}_\mathcal{E}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big).$$

$$\begin{aligned} g(i+1) &\le \left[b_1\left(\frac{\alpha(1+\tau_i)+\sqrt{\Delta}}{2}\right)^{i+1}\right.\\ &\quad\left.+\, b_2\left(\frac{\alpha(1+\tau_i)-\sqrt{\Delta}}{2}\right)^{i+1}\right]\big\|\boldsymbol{X}(0)-\boldsymbol{X}^*\big\|_F\\ &\le \left[(b_1+b_2)\left(\frac{\alpha(1+\tau_i)+\sqrt{\Delta}}{2}\right)^{i+1}\right]\\ &\quad\times\big\|\boldsymbol{X}(0)-\boldsymbol{X}^*\big\|_F. \end{aligned} \tag{69}$$

Then, (68) can be written as:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F^2\\ &\le 2\big\langle \mathcal{P}_\mathcal{E}\big(\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big), \mathcal{P}_\mathcal{E}\big(\boldsymbol{V}(i)-\boldsymbol{X}^*\big)\big\rangle\\ &= 2\big\langle \mathcal{P}_\mathcal{E}\big(\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big),\\ &\qquad \mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i+\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\big(\boldsymbol{X}^*-\boldsymbol{Q}_i\big)-\boldsymbol{X}^*\big)\big\rangle\\ &\overset{\text{(i)}}{=} 2\big\langle \boldsymbol{X}(i+1)-\boldsymbol{X}^*, \mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\\ &\qquad -\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}[\mathcal{P}_{\mathcal{S}_i}+\mathcal{P}_{\mathcal{S}_i^\perp}]\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\rangle\\ &= 2\big\langle \boldsymbol{X}(i+1)-\boldsymbol{X}^*,\\ &\qquad \big(\mathbf{I}-\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\big)\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\rangle\\ &\qquad -2\mu_i\big\langle \boldsymbol{X}(i+1)-\boldsymbol{X}^*, \mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\rangle\\ &\overset{\text{(ii)}}{\le} 2\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F\\ &\qquad\times\big\|\big(\mathbf{I}-\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\big)\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F\\ &\qquad+2\mu_i\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F\\ &\qquad\times\big\|\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F, \end{aligned} \tag{70}$$

where (i) is due to $\mathcal{P}_\mathcal{E}(\boldsymbol{Q}_i-\boldsymbol{X}^*) := \mathcal{P}_{\mathcal{S}_i}\mathcal{P}_\mathcal{E}(\boldsymbol{Q}_i-\boldsymbol{X}^*) + \mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}(\boldsymbol{Q}_i-\boldsymbol{X}^*)$ and (ii) follows from Cauchy-Schwarz inequality. Since $\frac{1}{1+\delta_{3k}} \le \mu_i \le \frac{1}{1-\delta_{3k}}$, Lemma 4 implies:

$$\begin{aligned}\lambda\big(\mathbf{I}-\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\big) &\in \left[1-\frac{1-\delta_{3k}}{1+\delta_{3k}}, \frac{1+\delta_{3k}}{1-\delta_{3k}}-1\right]\\ &\le \frac{2\delta_{3k}}{1-\delta_{3k}},\end{aligned}$$

and thus:

$$\begin{aligned}&\big\|\big(\mathbf{I}-\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\big)\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F\\ &\quad\le \frac{2\delta_{3k}}{1-\delta_{3k}}\big\|\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F.\end{aligned}$$

Furthermore, according to Lemma 5:

$$\begin{aligned}&\big\|\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F\\ &\quad\le \delta_{4k}\big\|\mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F\end{aligned}$$

since $\text{rank}(\mathcal{P}_\mathcal{K}\boldsymbol{Q}) \le 4k, \forall \boldsymbol{Q} \in \mathbb{R}^{m\times n}$ where $\mathcal{K} \leftarrow \text{ortho}(\mathcal{E} \cup \mathcal{S}_i)$. Since $\mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}(\boldsymbol{Q}_i-\boldsymbol{X}^*) = \mathcal{P}_{\mathcal{X}^*\setminus(\mathcal{D}_i\cup\mathcal{X}_i)}\boldsymbol{X}^*$ where

$$\mathcal{D}_i \leftarrow \mathcal{P}_k\big(\mathcal{P}_{\mathcal{Q}_i^\perp}\nabla f(\boldsymbol{Q}_i)\big),$$

then:

$$\begin{aligned}\big\|\mathcal{P}_{\mathcal{S}_i^\perp}\mathcal{P}_\mathcal{E}\big(\boldsymbol{Q}_i-\boldsymbol{X}^*\big)\big\|_F &= \big\|\mathcal{P}_{\mathcal{X}^*\setminus(\mathcal{D}_i\cup\mathcal{X}_i)}\boldsymbol{X}^*\big\|_F \le (2\delta_{3k}\\ &\quad+2\delta_{4k})\big\|\boldsymbol{Q}_i-\boldsymbol{X}^*\big\|_F,\end{aligned} \tag{71}$$

using Lemma 6. Using the above in (70), we compute:

$$\begin{aligned}&\big\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\big\|_F\\ &\quad\le\left(\frac{4\delta_{3k}}{1-\delta_{3k}}+(2\delta_{3k}+2\delta_{4k})\frac{2\delta_{3k}}{1-\delta_{3k}}\right)\big\|\boldsymbol{Q}_i-\boldsymbol{X}^*\big\|_F.\end{aligned} \tag{72}$$

Furthermore:

$$\begin{aligned}&\big\|\boldsymbol{Q}_i-\boldsymbol{X}^*\big\|_F\\ &\quad=\big\|\boldsymbol{X}(i)+\tau_i\big(\boldsymbol{X}(i)-\boldsymbol{X}(i-1)\big)\big\|_F\\ &\quad=\big\|(1+\tau_i)\big(\boldsymbol{X}(i)-\boldsymbol{X}^*\big)+\tau_i\big(\boldsymbol{X}^*-\boldsymbol{X}(i-1)\big)\big\|_F\\ &\quad\le(1+\tau_i)\big\|\boldsymbol{X}(i)-\boldsymbol{X}^*\big\|_F+\tau_i\big\|\boldsymbol{X}(i-1)-\boldsymbol{X}^*\big\|_F.\end{aligned} \tag{73}$$

Combining (72) and (73), we get:

$$\begin{aligned}
&\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F \\
&\quad\leq (1+\tau_i)\left(\frac{4\delta_{3k}}{1-\delta_{3k}}+(2\delta_{3k}+2\delta_{4k})\frac{2\delta_{3k}}{1-\delta_{3k}}\right) \\
&\qquad\times\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F \\
&\qquad+\tau_i\left(\frac{4\delta_{3k}}{1-\delta_{3k}}+(2\delta_{3k}+2\delta_{4k})\frac{2\delta_{3k}}{1-\delta_{3k}}\right) \\
&\qquad\times\left\|\boldsymbol{X}(i-1)-\boldsymbol{X}^*\right\|_F. \qquad (74)
\end{aligned}$$

Let $\alpha := \frac{4\delta_{3k}}{1-\delta_{3k}} + (2\delta_{3k}+2\delta_{4k})\frac{2\delta_{3k}}{1-\delta_{3k}}$ and $g(i) := \|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\|_F$. Then, (74) defines the following homogeneous recurrence:

$$g(i+1)-\alpha(1+\tau_i)g(i)+\alpha\tau_i g(i-1)\leq 0. \qquad (75)$$

Using the *method of characteristic roots* to solve the above recurrence, we assume that the homogeneous linear recursion has solution of the form $g(i)=r^i$ for $r\in\mathbb{R}$. Thus, replacing $g(i)=r^i$ in (75) and factoring out $r^{(i-2)}$, we form the following characteristic polynomial:

$$r^2-\alpha(1+\tau_i)r-\alpha\tau_i\leq 0. \qquad (76)$$

Focusing on the worst case where (76) is satisfied with equality, we compute the roots $r_{1,2}$ of the quadratic characteristic polynomial as:

$$r_{1,2}=\frac{\alpha(1+\tau_i)\pm\sqrt{\Delta}}{2}, \quad \text{where } \Delta:=\alpha^2(1+\tau_i)^2+4\alpha\tau_i.$$

Then, as a general solution, we combine the above roots with unknown coefficients b_1, b_2 to obtain (69). Using the initial condition $g(0) := \|\boldsymbol{X}(0)-\boldsymbol{X}^*\|_F \overset{\boldsymbol{X}(0)=\boldsymbol{0}}{=} \|\boldsymbol{X}^*\|_F = 1$, we get $b_1+b_2=1$. Thus, we conclude to the following recurrence:

$$\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F\leq\left(\frac{\alpha(1+\tau_i)+\sqrt{\Delta}}{2}\right)^{i+1}.$$

A.5 Proof of Lemma 7

Let $\mathcal{D}_i^\epsilon \leftarrow \mathcal{P}_k^\epsilon(\mathcal{P}_{\mathcal{X}_i^\perp}\nabla f(\boldsymbol{X}(i)))$ and $\mathcal{D}_i \leftarrow \mathcal{P}_k(\mathcal{P}_{\mathcal{X}_i^\perp}\nabla f(\boldsymbol{X}(i)))$. Using Definition 4, the following holds true:

$$\begin{aligned}
&\left\|\mathcal{P}_{\mathcal{D}_i^\epsilon}\nabla f\big(\boldsymbol{X}(i)\big)-\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\leq(1+\epsilon)\left\|\mathcal{P}_{\mathcal{D}_i}\nabla f\big(\boldsymbol{X}(i)\big)-\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2. \qquad (77)
\end{aligned}$$

Furthermore, we observe:

$$\begin{aligned}
&\left\|\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad=\left\|\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \quad\Leftrightarrow\quad \left\|\mathcal{P}_{\mathcal{D}_i^\epsilon}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\qquad+\left\|\mathcal{P}_{(\mathcal{D}_i^\epsilon)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad=\left\|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{X}_i}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2+\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2. \qquad (78)
\end{aligned}$$

Here, we use the notation defined in the proof of Lemma 6. Since $\mathcal{P}_{\mathcal{D}_i}\nabla f(\boldsymbol{X}(i))$ is the best rank-k approximation to $\nabla f(\boldsymbol{X}(i))$, we have:

$$\begin{aligned}
&\left\|\mathcal{P}_{\mathcal{D}_i}\nabla f\big(\boldsymbol{X}(i)\big)-\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\leq\left\|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{X}_i}\nabla f\big(\boldsymbol{X}(i)\big)-\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\Leftrightarrow\quad \left\|\mathcal{P}_{\mathcal{D}_i^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2\leq\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\Leftrightarrow\quad (1+\epsilon)\left\|\mathcal{P}_{\mathcal{D}_i^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\leq(1+\epsilon)\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2, \qquad (79)
\end{aligned}$$

where $\operatorname{rank}(\operatorname{span}(\operatorname{ortho}(\mathcal{X}^*\setminus\mathcal{X}_i)))\leq k$. Using (77) in (79), the following series of inequalities are observed:

$$\begin{aligned}
&\left\|\mathcal{P}_{(\mathcal{D}_i^\epsilon)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\leq(1+\epsilon)\left\|\mathcal{P}_{\mathcal{D}_i^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad\leq(1+\epsilon)\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2. \qquad (80)
\end{aligned}$$

Now, in (78), we compute the series of inequalities in (81)-(82).

$$\begin{aligned}
\left\|\mathcal{P}_{\mathcal{D}_i^\epsilon}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2+\left\|\mathcal{P}_{(\mathcal{D}_i^\epsilon)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2&=\left\|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{X}_i}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad+\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \overset{(79)}{\Leftrightarrow} \qquad (81)\\
\left\|\mathcal{P}_{\mathcal{D}_i^\epsilon}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2+(1+\epsilon)\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2&\geq\left\|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{X}_i}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \\
&\quad+\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \quad\Leftrightarrow \\
\left\|\mathcal{P}_{\mathcal{D}_i^\epsilon}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2+\epsilon\left\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2&\geq\left\|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{X}_i}\nabla f\big(\boldsymbol{X}(i)\big)\right\|_F^2 \quad\Leftrightarrow
\end{aligned}$$

$$\begin{aligned}
&\|\mathcal{P}_{\mathcal{D}_i^\epsilon}\nabla f(\boldsymbol{X}(i))\|_F^2 + \|\mathcal{P}_{\mathcal{X}_i}\nabla f(\boldsymbol{X}(i))\|_F^2 + \epsilon\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f(\boldsymbol{X}(i))\|_F^2 \geq \|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{X}_i}\nabla f(\boldsymbol{X}(i))\|_F^2 + \|\mathcal{P}_{\mathcal{X}_i}\nabla f(\boldsymbol{X}(i))\|_F^2 \quad \Leftrightarrow \\
&\|\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))\|_F^2 + \epsilon\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f(\boldsymbol{X}(i))\|_F^2 \geq \|\mathcal{P}_{\mathcal{S}_i^*}\nabla f(\boldsymbol{X}(i))\|_F^2 \quad \Leftrightarrow \\
&\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\nabla f(\boldsymbol{X}(i))\|_F^2 + \epsilon\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\nabla f(\boldsymbol{X}(i))\|_F^2 \geq \|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))\|_F^2 \quad \Leftrightarrow \\
&\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F^2 + \epsilon\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F^2 \geq \|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F^2 \quad \Leftrightarrow \\
&\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F + \sqrt{\epsilon}\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F \geq \|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F. \qquad (82)
\end{aligned}$$

Focusing on $\|\mathcal{P}^\perp_{\mathcal{X}^*\setminus\mathcal{X}_i}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F$, we observe:

$$\begin{aligned}
&\|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\mathcal{A}^*(\boldsymbol{y}-\mathcal{A}\boldsymbol{X}(i))\|_F \\
&\quad= \|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\mathcal{A}^*(\mathcal{A}\boldsymbol{X}^*+\boldsymbol{\varepsilon}-\mathcal{A}\boldsymbol{X}(i))\|_F \\
&\quad\leq \|\mathcal{P}_{(\mathcal{X}^*\setminus\mathcal{X}_i)^\perp}\mathcal{A}^*\mathcal{A}(\boldsymbol{X}^*-\boldsymbol{X}(i))\|_F + \|\mathcal{P}^\perp_{\mathcal{X}^*\setminus\mathcal{X}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \\
&\quad\leq \|\mathcal{A}^*\mathcal{A}(\boldsymbol{X}^*-\boldsymbol{X}(i))\|_F + \|\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \leq 2\lambda. \qquad (83)
\end{aligned}$$

Moreover, we know the following hold true from Lemma 6:

$$\begin{aligned}
&\|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\mathcal{A}(\boldsymbol{X}^*-\boldsymbol{X}(i)) + \mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \\
&\quad\leq 2\delta_{3k}\|\boldsymbol{X}^*-\boldsymbol{X}(i)\|_F + \|\mathcal{P}_{\mathcal{S}_i\setminus\mathcal{S}_i^*}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \qquad (84)
\end{aligned}$$

and

$$\begin{aligned}
&\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\mathcal{A}(\boldsymbol{X}^*-\boldsymbol{X}(i)) + \mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \\
&\quad\geq \|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}(\boldsymbol{X}^*-\boldsymbol{X}(i))\|_F - 2\delta_{2k}\|\boldsymbol{X}(i)-\boldsymbol{X}^*\|_F \\
&\quad\quad - \|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F. \qquad (85)
\end{aligned}$$

Combining (83)–(85) in (82), we obtain:

$$\begin{aligned}
\|\mathcal{P}_{\mathcal{S}_i^*\setminus\mathcal{S}_i}\boldsymbol{X}^*\|_F &= \|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{S}_i}\boldsymbol{X}^*\|_F \\
&\leq (2\delta_{2k}+2\delta_{3k})\|\boldsymbol{X}(i)-\boldsymbol{X}^*\|_F \\
&\quad + \sqrt{2(1+\delta_{2k})}\|\boldsymbol{\varepsilon}\|_2 + 2\lambda\sqrt{\epsilon}.
\end{aligned}$$

A.6 Proof of Theorem 4

To prove Theorem 4, we combine the following series of lemmas for each step of Algorithm 1.

Lemma 12 [Error norm reduction via gradient descent] *Let $\mathcal{S}_i \leftarrow ortho(\mathcal{X}_i \cup \mathcal{D}_i^\epsilon)$ be a set of orthonormal, rank-1 matrices that span a rank-2k subspace in $\mathbb{R}^{m\times n}$. Then (86) holds.*

$$\begin{aligned}
\|\boldsymbol{V}(i)-\boldsymbol{X}^*\|_F \leq &\left[\left(1+\frac{\delta_{3k}}{1-\delta_{2k}}\right)(2\delta_{2k}+2\delta_{3k}+\delta_k)+\frac{2\delta_{2k}}{1-\delta_{2k}}\right]\|\boldsymbol{X}(i)-\boldsymbol{X}^*\|_F \\
&+\left[\left(1+\frac{\delta_{3k}}{1-\delta_{2k}}\right)\sqrt{2(1+\delta_{2k})}+\frac{\sqrt{1+\delta_{2k}}}{1-\delta_{2k}}\right]\|\boldsymbol{\varepsilon}\|_2 + \left(1+\frac{\delta_{3k}}{1-\delta_{2k}}\right)2\lambda\sqrt{\epsilon}. \qquad (86)
\end{aligned}$$

Proof We observe the following:

$$\begin{aligned}
\|\boldsymbol{V}(i)-\boldsymbol{X}^*\|_F^2 &= \|\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{V}(i)-\boldsymbol{X}^*)\|_F^2 \\
&\quad + \|\mathcal{P}_{\mathcal{S}_i^\perp}(\boldsymbol{V}(i)-\boldsymbol{X}^*)\|_F^2. \qquad (87)
\end{aligned}$$

The following equations hold true:

$$\|\mathcal{P}_{\mathcal{S}_i^\perp}(\boldsymbol{V}(i)-\boldsymbol{X}^*)\|_F^2 = \|\mathcal{P}_{\mathcal{S}_i^\perp}\boldsymbol{X}^*\|_F^2 = \|\mathcal{P}_{\mathcal{X}^*\setminus\mathcal{S}_i}\boldsymbol{X}^*\|_F^2.$$

Furthermore, we compute:

$$\begin{aligned}
&\|\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{V}(i)-\boldsymbol{X}^*)\|_F \\
&\quad= \left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{X}(i)-\frac{\mu_i}{2}\mathcal{P}_{\mathcal{S}_i}\nabla f(\boldsymbol{X}(i))-\boldsymbol{X}^*\right)\right\|_F \\
&\quad= \|\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{X}(i)-\boldsymbol{X}^*) - \mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}(\boldsymbol{X}(i)-\boldsymbol{X}^*) \\
&\quad\quad + \mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \\
&\quad\leq \|(\mathbf{I}-\mu_i\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i}\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{X}(i)-\boldsymbol{X}^*))\|_F \\
&\quad\quad + \mu_i\|\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\mathcal{A}\mathcal{P}_{\mathcal{S}_i^\perp}(\boldsymbol{X}(i)-\boldsymbol{X}^*)\|_F \\
&\quad\quad + \mu_i\|\mathcal{P}_{\mathcal{S}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\|_F \\
&\quad\overset{(i)}{\leq} \frac{2\delta_{2k}}{1-\delta_{2k}}\|\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{X}(i)-\boldsymbol{X}^*)\|_F \\
&\quad\quad + \frac{\delta_{3k}}{1-\delta_{2k}}\|\mathcal{P}_{\mathcal{S}_i^\perp}(\boldsymbol{X}(i)-\boldsymbol{X}^*)\|_F
\end{aligned}$$

$$+\frac{\sqrt{1+\delta_{2k}}}{1-\delta_{2k}}\|\boldsymbol{\varepsilon}\|_2, \tag{88}$$

where (i) is due to Lemmas 2, 4, 5 and $\frac{1}{1+\delta_{2k}} \le \mu_i \le \frac{1}{1-\delta_{2k}}$.

Using the subadditivity property of the square root in (87), (88), Lemma 7 and the fact that $\|\mathcal{P}_{\mathcal{S}_i}(\boldsymbol{X}(i)-\boldsymbol{X}^*)\|_F \le \|\boldsymbol{X}(i)-\boldsymbol{X}^*\|_F$, we obtain:

$$\begin{aligned}
&\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \\
&\quad\le \left\|\mathcal{P}_{\mathcal{S}_i}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F + \left\|\mathcal{P}_{\mathcal{S}_i^{\perp}}\left(\boldsymbol{V}(i)-\boldsymbol{X}^*\right)\right\|_F \\
&\quad\le \hat{\rho}\left\|\boldsymbol{X}(i)-\boldsymbol{X}^*\right\|_F \\
&\qquad+\left(1+\frac{\delta_{3k}}{1-\delta_{2k}}\right)\sqrt{\epsilon}\left\|\mathcal{P}^{\perp}_{\mathcal{X}^*\setminus\mathcal{X}_i}\mathcal{A}^*\boldsymbol{\varepsilon}\right\|_F \\
&\qquad+\left[\left(1+\frac{\delta_{3k}}{1-\delta_{2k}}\right)\sqrt{2(1+\delta_{2k})}+\frac{\sqrt{1+\delta_{2k}}}{1-\delta_{2k}}\right]\|\boldsymbol{\varepsilon}\|_2,
\end{aligned} \tag{89}$$

where $\hat{\rho} := (1+\frac{\delta_{3k}}{1-\delta_{2k}})(2\delta_{2k}+2\delta_{3k})+\frac{2\delta_{2k}}{1-\delta_{2k}}$. □

We exploit Lemma 8 to obtain the following inequalities:

$$\begin{aligned}
\left\|\widehat{\boldsymbol{W}}_i-\boldsymbol{X}^*\right\|_F &= \left\|\widehat{\boldsymbol{W}}_i-\boldsymbol{V}(i)+\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \\
&\le \left\|\widehat{\boldsymbol{W}}_i-\boldsymbol{V}(i)\right\|_F + \left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \\
&\le (1+\epsilon)\left\|\boldsymbol{W}(i)-\boldsymbol{V}(i)\right\|_F + \left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F \\
&\le (2+\epsilon)\left\|\boldsymbol{V}(i)-\boldsymbol{X}^*\right\|_F,
\end{aligned} \tag{90}$$

where the last inequality holds since $\boldsymbol{W}(i)$ is the best rank-k matrix estimate of $\boldsymbol{V}(i)$ and, thus, $\|\boldsymbol{W}(i)-\boldsymbol{V}(i)\|_F \le \|\boldsymbol{V}(i)-\boldsymbol{X}^*\|_F$.

Following similar motions for steps 6 and 7 in Matrix ALPS I, we obtain:

$$\begin{aligned}
&\left\|\boldsymbol{X}(i+1)-\boldsymbol{X}^*\right\|_F \\
&\quad\le \left(1+\frac{2\delta_k}{1-\delta_k}+\frac{\delta_{2k}}{1-\delta_k}\right)\left\|\widehat{\boldsymbol{W}}_i-\boldsymbol{X}^*\right\|_F \\
&\qquad+\frac{\sqrt{1+\delta_k}}{1-\delta_k}\|\boldsymbol{\varepsilon}\|_2.
\end{aligned} \tag{91}$$

Combining (91), (90) and (89), we obtain the desired inequality.

References

1. Baraniuk, R.G., Cevher, V., Wakin, M.B.: Low-dimensional models for dimensionality reduction and signal recovery: a geometric perspective. Proc. IEEE **98**(6), 959–971 (2010)
2. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Found. Comput. Math. **9**(6), 717–772 (2009)
3. Meka, R., Jain, P., Dhillon, I.S.: Guaranteed rank minimization via singular value projection. In: NIPS Workshop on Discrete Optimization in Machine Learning (2010)
4. Tyagi, H., Cevher, V.: Learning ridge functions with randomized sampling in high dimensions. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2025–2028. IEEE Press, New York (2012)
5. Tyagi, H., Cevher, V.: Learning non-parametric basis independent models from point queries via low-rank methods. Technical report, EPFL (2012)
6. Liu, Y.K.: Universal low-rank matrix recovery from Pauli measurements (2011)
7. Tyagi, H., Cevher, V.: Active learning of multi-index function models. In: Advances in Neural Information Processing Systems, vol. 25, pp. 1475–1483 (2012)
8. Candes, E.J., Li, X.: Solving quadratic equations via phaselift when there are about as many equations as unknowns (2012). Preprint arXiv:1208.6247
9. Bennett, J., Lanning, S.: The netflix prize. In: KDD Cup and Workshop in Conjunction with KDD (2007)
10. Candes, E.J., Li, X., Ma, Y., Wright, J.: Robust principal component analysis? J. ACM **58**(3) (2011)
11. Kyrillidis, A., Cevher, V.: Matrix alps: Accelerated low rank and sparse matrix reconstruction. Technical report, EPFL (2012)
12. Waters, A.E., Sankaranarayanan, A.C., Baraniuk, R.G.: Sparcs: recovering low-rank and sparse matrices from compressive measurements. In: NIPS (2011)
13. Fazel, M., Recht, B., Parrilo, P.A.: Guaranteed minimum rank solutions to linear matrix equations via nuclear norm minimization. SIAM Rev. **52**(3), 471–501 (2010)
14. Liu, Z., Vandenberghe, L.: Interior-point method for nuclear norm approximation with application to system identification. SIAM J. Matrix Anal. Appl. **31**, 1235–1256 (2009)
15. Mohan, K., Fazel, M.: Reweighted nuclear norm minimization with application to system identification. In: American Control Conference (ACC). IEEE Press, New York (2010)
16. Cai, J.-F., Candès, E.J., Shen, Z.: A singular value thresholding algorithm for matrix completion. SIAM J. Optim. **20**, 1956–1982 (2010)
17. Recht, B., Re, C.: Parallel stochastic gradient algorithms for large-scale matrix completion. Preprint (2011)
18. Lin, Z., Chen, M., Ma, Y.: The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices (2010). preprint arXiv:1009.5055
19. Wright, J., Wu, L., Chen, M., Lin, Z., Ganesh, A., Ma, Y.: Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. UIUC Technical Report UILU-ENG-09-2214
20. Natarajan, B.K.: Sparse approximate solutions to linear systems. SIAM J. Comput. **24**(2), 227–234 (1995)
21. Lee, K., Bresler, Y.: Admira: atomic decomposition for minimum rank approximation. IEEE Trans. Inf. Theory **56**(9), 4402–4416 (2010)
22. Goldfarb, D., Ma, S.: Convergence of fixed-point continuation algorithms for matrix rank minimization. Found. Comput. Math. **11**, 183–210 (2011)
23. Beck, A., Teboulle, M.: A linearly convergent algorithm for solving a class of nonconvex/affine feasibility problems. In: Fixed-Point Algorithms for Inverse Problems in Science and Engineering, pp. 33–48 (2011)
24. Kyrillidis, A., Cevher, V.: Recipes on hard thresholding methods. In: Computational Advances in Multi-Sensor Adaptive Processing, Dec. 2011
25. Kyrillidis, A., Cevher, V.: Combinatorial selection and least absolute shrinkage via the Clash algorithm. In: IEEE International Symposium on Information Theory, July 2012
26. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev. **53**, 217–288 (2011)
27. Bertsekas, D.: Nonlinear Programming. Athena Scientific, Nashua (1995)

28. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge Univ. Press, Cambridge (1990)
29. Cohen, A., Dahmen, W., DeVore, R.: Compressed sensing and best k-term approximation. J. Am. Math. Soc. **22**(1), 211–231 (2009)
30. Tropp, J.A.: Greed is good: algorithmic results for sparse approximation. IEEE Trans. Inf. Theory **50**(10), 2231–2242 (2004)
31. Cevher, V.: An alps view of sparse recovery. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5808–5811. IEEE Press, New York (2011)
32. Needell, D., Tropp, J.A.: Cosamp: iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmon. Anal. **26**(3), 301–321 (2009)
33. Dai, W., Milenkovic, O.: Subspace pursuit for compressive sensing signal reconstruction. IEEE Trans. Inf. Theory **55**, 2230–2249 (2009)
34. Foucart, S.: Hard thresholding pursuit: an algorithm for compressed sensing. SIAM J. Numer. Anal. **49**(6), 2543–2563 (2011)
35. Blumensath, T., Davies, M.E.: Iterative hard thresholding for compressed sensing. Appl. Comput. Harmon. Anal. **27**(3), 265–274 (2009)
36. Garg, R., Khandekar, R.: Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property. In: ICML. ACM Press, New York (2009)
37. Blumensath, T., Davies, M.E.: Normalized iterative hard thresholding: guaranteed stability and performance. IEEE J. Sel. Top. Signal Process. **4**(2), 298–309 (2010)
38. Blumensath, T.: Accelerated iterative hard thresholding. Signal Process. **92**, 752–756 (2012)
39. Tanner, J., Wei, K.: Normalized iterative hard thresholding for matrix completion. Preprint (2012)
40. Coifman, R., Geshwind, F., Meyer, Y.: Noiselets. Appl. Comput. Harmon. Anal. **10**(1), 27–44 (2001)
41. Foucart, S.: Sparse recovery algorithms: sufficient conditions in terms of restricted isometry constants. In: Proceedings of the 13th International Conference on Approximation Theory (2010)
42. Nesterov, Y.: Gradient methods for minimizing composite objective function. core discussion papers 2007076, université catholique de louvain. Center for Operations Research and Econometrics (CORE) (2007)
43. Nesterov, Y.: Introductory Lectures on Convex Optimization. Kluwer Academic, Dordrecht (1996)
44. Drineas, P., Frieze, A., Kannan, R., Vempala, S., Vinay, V.: Clustering large graphs via the singular value decomposition. Mach. Learn. **56**(1), 9–33 (2004)
45. Drineas, P., Kannan, R., Mahoney, M.W.: Fast Monte Carlo algorithms for matrices ii: computing a low-rank approximation to a matrix. SIAM J. Comput. **36**, 158–183 (2006)
46. Deshpande, A., Rademacher, L., Vempala, S., Wang, G.: Matrix approximation and projective clustering via volume sampling. In: Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm, SODA '06, New York, NY, USA, pp. 1117–1126. ACM Press, New York (2006)
47. Deshpande, A., Vempala, S.: Adaptive sampling and fast low-rank matrix approximation. Electron. Colloq. Comput. Complex. **13**, 042 (2006)
48. Keshavan, R.H., Montanari, A., Oh, S.: Matrix completion from a few entries. IEEE Trans. Inf. Theory **56**(6), 2980–2998 (2010)
49. Balzano, L., Nowak, R., Recht, B.: Online identification and tracking of subspaces from highly incomplete information. In: 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 704–711. IEEE Press, New York (2010)
50. He, J., Balzano, L., Lui, J.C.S.: Online robust subspace tracking from partial information (2011). arXiv:1109.3827
51. Boumal, N., Absil, P.A.: Rtrmc: a Riemannian trust-region method for low-rank matrix completion. In: NIPS (2011)
52. Wen, Z., Yin, W., Zhang, Y.: Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. Rice University CAAM Technical Report TR10-07 (2010) Submitted
53. Larsen, R.M.: Propack: Software for large and sparse svd calculations. http://soi.stanford.edu/~rmunk/PROPACK
54. Shi, X., Yu, P.S.: Limitations of matrix completion via trace norm minimization. ACM SIGKDD Explor. Newsl. **12**(2), 16–20 (2011)