Matrix Recipes for Hard Thresholding Methods

Anastasios Kyrillidis · Volkan Cevher

Received: date / Accepted: date

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, $\epsilon$-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 · $\epsilon$-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 $X^* \in \mathbb{R}^{m \times n}$ is a rank-$k$ matrix of interest ($k \ll \min\{m, n\}$) and let $\mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p$ be a known linear operator. Given a set of observations $y = \mathcal{A}X^* + \varepsilon \in \mathbb{R}^p$, we desire to recover $X^*$ from $y$ 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 $X^*$ is low-rank and thus, we are interested in finding a matrix $X$ of rank at most $k$ that minimizes the data error $f(X) := \|y - \mathcal{A}X\|_2^2$ as follows:

$$\min_{X \in \mathbb{R}^{m \times n}} f(X) \quad \text{subject to} \quad \text{rank}(X) \leq k. \quad (1)$$

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 $\mathcal{A}$.

**General linear maps:** In many ARM problem cases, $\mathcal{A}$ or $\mathcal{A}^*$ has a dense range, satisfying specific incoherence or restricted isometry properties (discussed later in the paper); here, $\mathcal{A}^*$ is the adjoint operator of $\mathcal{A}$. In Quantum Tomography, [6] studies the Pauli operator, a compressive linear map $\mathcal{A}$ that consists of the kronecker product of $2 \times 2$ 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 $\mathcal{A}$ is a Bernoulli and a Fourier operator, respectively.

**Matrix Completion (MC):** Let $\Omega$ be the set of ordered pairs that represent the coordinates of the observable entries in $X^*$. Then, the set of observations satisfy $y = \mathcal{A}\Omega X^* + \varepsilon$ where $\mathcal{A}_\Omega$ defines a linear mask over the observable entries $\Omega$. To solve the MC problem, a potential criterion is given by (1) [2]. As a motivating example, consider the famous 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 $y = \mathcal{A}X^*$ where $\mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p$ is an identity linear map that stacks the columns of the matrix $X^*$ into a single column vector with $p = m \cdot n$. 

A. Kyrillidis
Laboratory for Information and Inference Systems, Ecole Polytechnique Federale de Lausanne
Tel.: +41 21 69 31154
E-mail: anastasios.kyrillidis@epfl.ch

V. Cevher
Laboratory for Information and Inference Systems, Ecole Polytechnique Federale de Lausanne
Tel.: +41 21 69 31101
E-mail: volkan.cevher@epfl.ch
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 $A^* y$. As an extension to the PCA setting, [10] considers the Robust PCA problem where $y$ 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 $A$ is constituted by permuted subsampled noiselets [12].

1.1 Two camps of recovery algorithms

**Convex relaxations:** In [13], the authors study the nuclear norm $\|X\|_* := \sum_{i=1}^{\text{rank}(X)} \sigma_i$ as a convex surrogate of $\text{rank}(X)$ operator so that we can leverage convex optimization approaches, such as interior-point methods—here, $\sigma_i$ denotes the $i$-th singular value of $X$. Under basic incoherence properties of the sensing linear mapping $A$, [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 behavior and performance of hard thresholding methods from a global perspective. Five building blocks are studied: $i)$ step size selection $\mu_i$, $ii)$ 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 $\epsilon$-approximation algorithms. Based on similar ideas, we analyze the impact of $\epsilon$-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 $\epsilon$-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 Section 2, we set up the notation and provide some definitions and properties, essential for the rest of the paper. In Section 3, we describe the basic algorithmic frameworks in a nutshell, while in Section 4 we provide important “ingredients” for the class of hard-thresholding methods; detailed convergence analysis proofs are provided in Section 5. The complexity analysis of the proposed algorithms is provided in Section 6. We study two acceleration schemes in Sections 7 and 8, based on memory utilization and $\epsilon$-approximate low-rank projections, respectively. We further improve convergence speed by exploiting randomized low rank projections in Section 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 Section 10. Finally, we conclude with future work directions in Section 11.

2 Elementary Definitions and Properties

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

The rank of $X$ is denoted as $\text{rank}(X) \leq \min\{m, n\}$.

The empirical data error is denoted as $f(X) := \|y - AX\|_2^2$ with gradient $\nabla f(X) : = -2A^*(y - AX)$, where $*$ is the adjoint operation over the linear mapping $A$. The inner product between matrices $A, B \in \mathbb{R}^{m \times n}$ is denoted as $\langle A, B \rangle = \text{trace}(B^T A)$, where $T$ represents the transpose operation. $I$ represents an identity matrix with dimensions apparent from the context.
Let $S$ be a set of orthonormal, rank-1 matrices that span an arbitrary subspace in $\mathbb{R}^{m \times n}$. We reserve $\text{span}(S)$ to denote the subspace spanned by $S$. With slight abuse of notation, we use:

$$\text{rank}(\text{span}(S)) = \max_x \{\text{rank}(X) : X \in \text{span}(S)\}, \tag{2}$$

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

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

where $U$ denotes the superset that includes all the sets of orthonormal, rank-1 matrices in $\mathbb{R}^{m \times n}$ such that $(T_i, T_j) = 0, i \neq j, \forall T_i, T_j \in T$ and, $||T_i||_2 = 1, \forall T_i$. In general, $\text{ortho}(S)$ 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 $J \subseteq \mathbb{R}^{m \times n}$ be a closed convex set and $f : J \to \mathbb{R}$ be a smooth objective function defined over $J$. Let $X^* \in J$ be a local minimum of the objective function $f$ over the set $J$. Then

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

### 2.1 Singular Value Decomposition (SVD) and its properties

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

$$X = U \Sigma V^T = \begin{bmatrix} U_\alpha & U_\beta \end{bmatrix} \begin{bmatrix} \Sigma_0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_0^T \\ V_\beta^T \end{bmatrix}, \tag{4}$$

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

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

$$P_k(X) = \arg \min_{Y \in C_k} \|Y - X\|_F. \tag{5}$$

According to the Eckart-Young theorem [28], the best rank-$k$ approximation of a matrix $X$ corresponds to its truncated SVD: if $X = U \Sigma V^T$, then $P_k(X) := U_k \Sigma_k V_k^T$ where $\Sigma_k \in \mathbb{R}^{k \times k}$ is a diagonal matrix that contains the first $k$ diagonal entries of $\Sigma$ and $U_k, V_k$ 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, $\|P_k(X) - X\|_F \leq \|W - X\|_F$ for any rank-$k$ $W \in \mathbb{R}^{m \times n}$.

### 2.2 Subspace projections

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

$$X := P_S X + P_{S^\perp} X, \text{ such that } \langle P_S X, P_{S^\perp} X \rangle = 0.$$ 

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

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

$$P_S X = P_{U_{1:k} \Sigma_{1:k}} + X P_{V_{1:k}}, \tag{6}$$

where $P_{U_{1:k} \Sigma_{1:k}} = U_{1:k} \Sigma_{1:k} U_{1:k}^T$ and $P_{V_{1:k}} = V_{1:k} V_{1:k}^T$ in MATLAB notation. Moreover, the orthogonal projection onto the $S^\perp$ is given by:

$$P_{S^\perp} X = X - P_S X. \tag{7}$$

In the algorithmic descriptions, we use $S \leftarrow P_k(X)$ to denote the set of rank-$k$, orthonormal matrices as outer products of the $k$ left $u_i$ and right $v_i$ principal singular vectors of $X$ that span the best rank-$k$ subspace of $X$; e.g. $S = \{u_i v_i, i = 1, \ldots, k\}$. Moreover, $\tilde{X} \leftarrow P_k(X)$ denotes a/the best rank-$k$ projection matrix of $X$. In some cases, we use $\{S, \tilde{X}\} \leftarrow P_k(X)$ when we compute both. The distinction 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.

\footnote{The distinction between $P_S$ and $P_k$ for $k$ positive integer is apparent from context.}
Definition 3 [Rank Restricted Isometry Property (R-RIP)] for matrix linear operators [13] A linear operator \( \mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p \) satisfies the R-RIP with constant \( \delta_k(\mathcal{A}) \in (0, 1) \) if and only if:
\[
(1 - \delta_k(\mathcal{A})) \|X\|_F^2 \leq \|\mathcal{A}X\|_2^2 \leq (1 + \delta_k(\mathcal{A})) \|X\|_F^2, \tag{8}
\]
\( \forall X \in \mathbb{R}^{m \times n} \) such that rank\( (X) \leq k \). We write \( \delta_k \) 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 \( \mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p \) be a linear operator that satisfies the R-RIP with constant \( \delta_k \). Then, \( \forall v \in \mathbb{R}^p \), the following holds true:
\[
\|\mathcal{P}_S(\mathcal{A}^*v)\|_F \leq \sqrt{1 + \delta_k} \|v\|_2, \tag{9}
\]
where \( S \) is a set of orthonormal, rank-1 matrices in \( \mathbb{R}^{m \times n} \) such that rank\( (\mathcal{P}_S X) \leq k, \forall X \in \mathbb{R}^{m \times n} \).

Lemma 3 [21] Let \( \mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p \) be a linear operator that satisfies the R-RIP with constant \( \delta_k \). Then, \( \forall X \in \mathbb{R}^{m \times n} \), the following holds true:
\[
(1 - \delta_k) \|\mathcal{P}_S X\|_F \leq \|\mathcal{P}_S \mathcal{A}^* \mathcal{A} \mathcal{P}_S X\|_F \leq (1 + \delta_k) \|\mathcal{P}_S X\|_F, \tag{10}
\]
where \( S \) is a set of orthonormal, rank-1 matrices in \( \mathbb{R}^{m \times n} \) such that rank\( (\mathcal{P}_S X) \leq k, \forall X \in \mathbb{R}^{m \times n} \).

Lemma 4 [22] Let \( \mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p \) be a linear operator that satisfies the R-RIP and \( \delta_k \) and \( S \) be a set of orthonormal, rank-1 matrices in \( \mathbb{R}^{m \times n} \) such that rank\( (\mathcal{P}_S X) \leq k, \forall X \in \mathbb{R}^{m \times n} \). Then, for \( \mu > 0 \), \( \mathcal{A} \) satisfies:
\[
\lambda(\mu \mathcal{P}_S \mathcal{A}^* \mathcal{A} \mathcal{P}_S) \in [\mu(1 - \delta_k), \mu(1 + \delta_k)], \tag{11}
\]
where \( \lambda(B) \) represents the range of eigenvalues of the linear operator \( B : \mathbb{R}^p \rightarrow \mathbb{R}^{m \times n} \). Moreover, \( \forall X \in \mathbb{R}^{m \times n} \), it follows that:
\[
\|\mathcal{P}_S \mathcal{A}^* \mathcal{A} \mathcal{P}_S X\|_F \leq \max \{\mu(1 + \delta_k) - 1, 1 - \mu(1 - \delta_k)\} \|\mathcal{P}_S X\|_F. \tag{12}
\]

Lemma 5 [22] Let \( \mathcal{A} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^p \) be a linear operator that satisfies the R-RIP with constant \( \delta_k \) and \( S_1, S_2 \) be two sets of orthonormal, rank-1 matrices in \( \mathbb{R}^{m \times n} \) such that rank\( (\mathcal{P}_{S_1} \mathcal{P}_{S_2} X) \leq k, \forall X \in \mathbb{R}^{m \times n} \).

Then, the following inequality holds:
\[
\|\mathcal{P}_{S_1} \mathcal{A}^* \mathcal{A} \mathcal{P}_{S_2} X\|_F \leq \delta_k \|\mathcal{P}_{S_2} X\|_F, \forall X \in \text{span}(S_2). \tag{13}
\]

3 Algebraic Pursuits in a nutshell

Explicit descriptions of the proposed algorithms are provided in Algorithms 1 and 2. Algorithm 1 follows from the Algebraic Pursuits (ALPS) scheme for the vector case [31]. Matrix ALPS 1 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 \( S \), with rank\( (\text{span}(S_i)) \leq 2k \) and then finding the rank-\( k \) matrix that fits the measurements in this restricted subspace using least squares or gradient descent motions.

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

1) Best rank-\( k \) subspace orthogonal to \( X_0 \) and active subspace expansion: We identify the best rank-\( k \) subspace of the current gradient \( \nabla J(X(i)) \), orthogonal to \( X_0 \), and then merge this low-rank subspace with \( X_0 \). 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 Section 4.

2b) Error norm reduction via least squares optimization (Algorithm 2): We decrease the data error \( f(X) \) 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 \( C_k := \{ A \in \mathbb{R}^{m \times n} : \text{rank}(A) \leq k \} \) 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 Sections 8 and 9.

4) De-bias using gradient descent (Algorithm 1): We de-bias the current estimate \( W(i) \) 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 $\|\Phi\|^2_F < 1$ 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 $\mu_i$ that implies fast convergence rate, for mild 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 $X(i)$ be the current rank-$k$ matrix estimate spanned by the set of orthonormal, rank-$1$ matrices in $X_i$. Using regular gradient descent motions, the new rank-$k$ estimate $W(i)$ can be calculated through:

$$V_i = X(i) - \mu \nabla f(X(i)), \quad \{W_i, W(i)\} \leftarrow P_k(V(i)).$$

We highlight that the rank-$k$ approximate matrix may not be unique. It then holds that the subspace spanned by $W_i$ originates: $i$ either from the subspace of $X_i$, $ii$ or from the best subspace (in terms of the Frobenius norm metric) of the current gradient $\nabla f(X(i))$, orthogonal to $X_i$, $iii$ 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:

$$\text{span}(W_i) \in \text{span}(D_i \cup X_i) \quad (15)$$

for any step size $\mu_i$ and $D_i \leftarrow P_k(\mathcal{A}_i \nabla f(X(i)))$. Since $\text{rank}(\text{span}(W_i)) \leq k$, we easily deduce the following key observation: let $S_i \leftarrow D_i \cup X_i$ be a set of rank-$1$, orthonormal matrices where $\text{rank}(\text{span}(S_i)) \leq 2k$. Given $W_i$ is unknown
before the $i$-th iteration, $S_i$ spans the smallest subspace that contains $W_i$ such that the following equality
\begin{equation}
P_k \left( X(i) - \frac{\mu_i}{2} \nabla f(X(i)) \right) = P_k \left( X(i) - \frac{\mu_i}{2} P_{S_i} \nabla f(X(i)) \right)
\end{equation}

necessarily holds. To compute step-size $\mu_i$, we use:
\begin{align}
\mu_i &= \arg \min \mu \left\| y - A \left( X(i) - \frac{\mu}{2} P_{S_i} \nabla f(X(i)) \right) \right\|_2^2 \\
&= \left\| P_{S_i} \nabla f(X(i)) \right\|_F^2 \left/ \left\| A P_{S_i} \nabla f(X(i)) \right\|_F^2 \right.
\end{align}
i.e., $\mu_i$ is the minimizer of the objective function, given the current gradient $\nabla f(X(i))$. Note that:
\begin{equation}
1 - \delta_{2k}(A) \leq \frac{1}{\mu_i} \leq 1 + \delta_{2k}(A),
\end{equation}
due to R-RIP—i.e., we select $2k$ subspaces such that $\mu_i$ satisfies (18). We can derive similar arguments for the additional step size selection $\xi_i$ in Step 6 of Algorithm 1.

Adaptive $\mu_i$ 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 $X^*$ increases.

4.2 Updates on restricted subspaces

In Algorithm 1, at each iteration, the new estimate $W(i) \leftarrow P_k(V(i))$ can be further refined by applying a single or multiple gradient descent updates with line search restricted on $W_i$ [34] (Step 7 in Algorithm 1):
\begin{equation}
X(i+1) \leftarrow W(i) - \xi_i \frac{\mu_i}{2} P_{W(i)} \nabla f(W(i)),
\end{equation}
where $\xi_i = \left\| P_{W(i)} \nabla f(W(i)) \right\|_F^2 \left/ \left\| A P_{W(i)} \nabla f(W(i)) \right\|_F^2 \right.$. 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 Section 7 and low-rank matrix approximations for Sections 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 $P_{S}X$ and $P_{A}X$ where $X \in \mathbb{R}^{m \times n}$, as described by (6) and (7), respectively. Since $S$ is constituted
by outer products of left and right singular vectors as in Definition 2, $\mathcal{P}_S X$ (resp. $\mathcal{P}_{S^\perp} X$) projects onto the (resp. complement of the) best low-rank subspace in $R(X)$ and $R(X^T)$. These operations are highly connected with the adaptive step size selection and the updates on restricted subspaces. Unfortunately, the time-complexity to compute $\mathcal{P}_S X$ 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 $\mathcal{P}_S$ in practice.\(^3\) Rather, employing inexact projections is sufficient to converge to the optimal solution: either $i)$ $\mathcal{P}_U X$ onto the best low-rank subspace in $R(X)$ only (if $m \ll n$) or ii) $X \mathcal{P}_V$ onto the best low-rank subspace in $R(X^T)$ only (if $m \gg n$); $\mathcal{P}_U$ and $\mathcal{P}_V$ are defined in Definition 2 and require only one matrix-matrix multiplication.

Figure 2 shows the time overhead due to the exact projection application $\mathcal{P}_S$ compared to $\mathcal{P}_U$ for $m \leq n$. In Figure 2(a), we use subsampled and permuted noisefors for linear map $\mathcal{A}$ and in Figures 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 $\mathcal{P}_U$, the latter case converges faster to the desired accuracy $5 \cdot 10^{-4}$ when $m \ll n$ as shown in Figures 2(a)-(b). In our derivations, we assume $\mathcal{P}_S$ and $\mathcal{P}_{S^\perp}$ 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:

$$
\|X(i + 1) - X^*\|_F \leq \rho \|X(i) - X^*\|_F + \gamma \|\epsilon\|_2.
$$

(19)

\(^3\) 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.

In (19), $\gamma$ denotes the approximation guarantee and provides insights into algorithm’s reconstruction capabilities when additive noise is present; $\gamma < 1$ expresses the convergence rate towards a region around $X^*$, whose radius is determined by $\frac{1}{\sqrt{\gamma}} \|\epsilon\|_2$. In short, (19) characterizes how the distance to the true signal $X^*$ 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 $X(i)$ be the matrix estimate at the $i$-th iteration and let $\mathcal{X}_i$ be a set of orthonormal, rank-1 matrices such that $\mathcal{X}_i \leftarrow \mathcal{P}_k(X(i))$. Then, at each iteration, the Active Subspace Expansion step in Algorithms 1 and 2 identifies information in $X(i)$, such that:

$$
\|P_{X^*}P_{S_i^*}X^*\|_F \leq (2\delta_{2k} + 2\delta_{3k})\|X(i) - X^*\|_F + \sqrt{2(1 + \delta_{2k})}\|\epsilon\|_2,
$$

(20)

where $S_i \leftarrow \mathcal{X}_i \cup D_i$ and $X^* \leftarrow \mathcal{P}_k(X^*)$.

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 $X^*—i.e., the projection of $X^*$ onto the orthogonal subspace of span$(S_i)$—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 $X(i + 1)$ of MATRIX ALPS I satisfies the following recursion:

$$
\|X(i + 1) - X^*\|_F \leq \rho \|X(i) - X^*\|_F + \gamma \|\epsilon\|_2,
$$

(21)

where $\rho := \left(1 + \frac{2\delta_{2k}}{1 - \delta_{2k}}\right)\left(\frac{4\delta_{2k}}{1 - \delta_{2k}} + (2\delta_{2k} + 2\delta_{3k})\frac{2\delta_{2k}}{1 - \delta_{2k}}\right)$ and $\gamma := \frac{2\delta_{2k}}{1 - \delta_{2k}}\sqrt{2(1 + \delta_{2k})} + \frac{\sqrt{1 + 2\delta_{2k}}}{1 - \delta_{2k}}\|\epsilon\|_2$.

Moreover, when $\delta_{3k} < 0.1253$, the iterations are contractive.
To provide some intuition behind this result, assume that \( X^* \) is a rank-\( k \) matrix. Then, according to Theorem 1, for \( \rho < 1 \), the approximation parameter \( \gamma \) in (21) satisfies:
\[
\gamma < 5.7624, \text{ for } \delta_{3k} < 0.1235.
\]
Moreover, we derive the following:
\[
\rho < \frac{1 + 2\delta_{3k}}{(1 - \delta_{3k})^2} \left( 4\delta_{3k} + 8S^2_{3k} \right) < \frac{1}{2} \Rightarrow \delta_{3k} < 0.079,
\]
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 \( \|X(i) - X^*\|_F \) at the \( i \)-th iteration:
\[
\|X(i) - X^*\|_F \leq \rho^i \|X(0) - X^*\|_F + \frac{\gamma}{1 - \rho} \|\varepsilon\|_2^2. \tag{22}
\]

Then, given \( X(0) = 0 \), MATRIX ALPS I finds a rank-\( k \) solution \( \hat{X} \in \mathbb{R}^{m \times n} \) such that \( \|\hat{X} - X^*\|_F \leq \frac{2(1 - \rho)}{1 - \rho} \|\varepsilon\|_2 \) after \( i := \frac{\log(\|X(0)\|_F)}{\log(1/\rho)} \) 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 \( \{X_{+1}, X(i + 1)\} \leftarrow \mathcal{P}_k(V_i) \). Then, by the same analysis, we observe that the following recursion is satisfied:
\[
\|X(i + 1) - X^*\|_F \leq \rho \|X(i) - X^*\|_F + \frac{\gamma}{1 - \rho} \|\varepsilon\|_2^2, \tag{23}
\]
\[
\text{for } \rho := \frac{4\delta_{3k}}{1 - 8\delta_{3k}} + \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}}
\]
\[
\text{and } \gamma := \frac{2\sqrt{1 + 2\delta_{3k}}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}}
\]
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 Figure 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:
\[
V(i) \leftarrow \arg\min_{V} \|y - AV\|_2^2, \tag{24}
\]

we obtain ADMiRA (furthermore, we remove the de-bias step in Algorithm 1). Assuming that the linear operator \( \mathcal{A} \) 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 \( X(i + 1) \) of ADMiRA answers the following recursive expression:
\[
\|X(i + 1) - X^*\|_F \leq \rho \|X(i) - X^*\|_F + \frac{\gamma}{1 - \rho} \|\varepsilon\|_2^2,
\]
\[
\rho := \frac{2\delta_{3k} + 2\delta_{4k}}{1 - 8\delta_{3k}}, \text{and } \gamma := \frac{\sqrt{1 + 8\delta_{3k}}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}} \frac{2\delta_{4k}}{1 - 8\delta_{3k}}
\]
Moreover, when \( \delta_{3k} < 0.2267 \), the iterations are contractive.

Similarly, to MATRIX ALPS I analysis, the parameter \( \gamma \) in Theorem 2 satisfies:
\[
\gamma < 5.1848, \text{ for } \delta_{3k} < 0.2267.
\]

Furthermore, to compare the approximation guarantees of Theorem 2 with [21], we further observe:
\[
\delta_{3k} < 0.1214, \text{ for } \rho < 1/2.
\]

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 \( \mathcal{A} \), 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/Noiselines tranformations and the famous Matrix Completion problem where \( \mathcal{A} \) is a mask operator such that only a fraction of elements in \( X \) is observed. Assuming the most demanding case where \( \mathcal{A} \) and \( \mathcal{A}^T \) are dense linear maps with no structure, the computation of the gradient \( \nabla f(X(i)) \) at each iteration requires \( O(pkmn) \) arithmetic operations.

Given a set \( S \) of orthonormal, rank-1 matrices, the projection \( \mathcal{P}_S X \) for any matrix \( X \in \mathbb{R}^{m \times n} \) 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 \( \mu_i \) and \( \xi_i \) quantities. In

\(^5\) While such operation has \( O(\max\{m^2n, mn^2\}) \) complexity, each application of \( \mathcal{P}_S X \) requires three matrix-matrix multiplications. To reduce such computational cost, we relax this operation in Section 10 where in practice we use only \( \mathcal{P}_d \) that needs one matrix-matrix multiplication.
ADMIRA solving a least-squares system restricted on rank-
2k and rank-k subspaces requires $O(pk^2)$ complexity; ac-
according to [32], [21], the complexity of this step can be fur-
ther reduced using iterative techniques such as the Richard-
son method or conjugate gradients algorithm.

Using the Lanczos method, we require $O(kmn)$ arith-
metic 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. Sec-
tions 8 and 9 describe approximate low rank matrix projec-
tions and how they affect the convergence guarantees of the
proposed algorithms.

Overall, the operation that dominates per iteration re-
quires $O(\max\{pkmn, m^2n, mn^2\})$ 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 $X(i+1)$ with
momentum step size $\tau_i$ is performed using previous matrix
estimates.

Similar to $\mu_i$ strategies, $\tau_i$ 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, em-
pirical evidence has shown that adaptive $\tau_i$ selection stra-
tegies result in faster convergence compared to zero-memory
methods with similar complexity.

For the case of strongly convex objective functions, Nes-
terov [43] proposed the following constant momentum step
size selection scheme: $\tau_i = \frac{\alpha_i(1-\alpha_i)}{\alpha_i + \alpha_{i+1}}$, where $\alpha_0 \in (0, 1)$
and $\alpha_{i+1}$ is computed as the root in $(0, 1)$ of
\[
\alpha_{i+1}^2 = (1 - \alpha_{i+1})^2 + q \alpha_{i+1}, \quad \text{for } q = \frac{1}{\kappa^2(\mathcal{A})},
\]
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 bot-
tleneck.

Based upon adaptive $\mu_i$ selection, we propose to select
$\tau_i$ as the minimizer of the objective function:
\[
\tau_i = \arg \min_{\tau} \|y - \mathcal{A}Q(i+1)\|_2^2 = \frac{\langle y - \mathcal{A}X(i), \mathcal{A}X(i) - \mathcal{A}X(i-1) \rangle}{\|\mathcal{A}X(i) - \mathcal{A}X(i-1)\|_2^2},
\]
where $\mathcal{A}X(i), \mathcal{A}X(i-1)$ are already pre-computed at each
iteration. According to (26), $\tau_i$ is dominated by the calcu-
lation of a vector inner product, a computationally cheaper
process than $q$ calculation.

Theorem 3 characterizes Algorithm 3 for constant mo-
mentum 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.

\textbf{Theorem 3} [Iteration invariant for MATRIX ALPS II] Let
$y = \mathcal{A}X^*$ be a noiseless set of observations. To recover
$X^*$ from $y$ and $\mathcal{A}$, the $(i+1)$-th matrix estimate $X(i+1)$
of MATRIX ALPS II satisfies the following recursion:
\[
\|X(i+1) - X^*\|_F \leq \alpha(1+\tau_i)\|X(i) - X^*\|_F + \alpha\tau_i\|X(i-1) - X^*\|_F,
\]
where $\alpha := \frac{4\kappa^2}{1-\delta_{ik}} + (2\delta_{ik} + 2\delta_{ik}) \frac{2\kappa^2}{1-\delta_{ik}}$. Moreover, solving
the above second-order recurrence, the following inequality
holds true:
\[
\|X(i+1) - X^*\|_F \leq \rho^{i+1}\|X(0) - X^*\|_F,
\]
for $\rho := \alpha(1+\tau_i) + \sqrt{\alpha^2(1+\tau_i)^2 + 4\alpha\tau_i} - 2\alpha\tau_i$.  

Theorem 3 provides convergence rate behaviour proof
for the case where $\tau_i$ is constant $\forall i$. The more elaborate
case where $\tau_i$ 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 $\tau_i = 1/2$, $\forall i$, $\delta_{ik} < 0.1187$
and $\delta_{ik} < 0.095$ guarantee convergence in Algorithm 3, re-
spectively. While the RIP requirements for memory-based
MATRIX ALPS II are more stringent than the schemes pro-
posed in the previous section, it outperforms Algorithms 1 and
2. Figure 2 shows the acceleration achieved in MATRIX
ALPS II by using inexact projections $P_{\tau_i}$. Using the pro-
per projections (6)-(7), Figure 3 shows acceleration in practice
when using the adaptive momentum step size strategy: while
a wide range of constant momentum step sizes leads to con-
vergence, providing flexibility to select an appropriate $\tau_i$,
adaptive $\tau_i$ avoids this arbitrary $\tau_i$ selection while further
decreases the number of iterations needed for convergence
in most cases.
where

\[ \forall D \\exists X \quad \text{be a set of orthonormal, rank-1 matrices in} \]

Lemma 7.

\[ \text{be a set of orthonormal, rank-1 matrices in} \]

Lemma 10.

\[ \text{be a set of orthonormal, rank-1 matrices in} \]

Algorithm 3: MATRIX ALPS II

8 Accelerating MATRIX ALPS: \( \epsilon \)-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 \( \mathbf{X}^* \). 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 \( \mathbf{X} \) 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 it contains; usually, the distribution is related to the \( \ell_2 \)-norm of the columns. The main idea of this approach is to compute a surrogate rank-\( k \) matrix \( \mathcal{P}_k(\mathbf{X}) \) by subsampling the columns according to this distribution. It turns out that the total number of sampled columns is a function of the parameter \( \epsilon \). Moreover, [46, 47] proved that, given a target rank \( k \) and an approximation parameter \( \epsilon \), we can compute an \( \epsilon \)-approximate rank-\( k \) matrix \( \mathcal{P}_k(\mathbf{X}) \) according to the following definition.

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

\[ \| \mathcal{P}_k(\mathbf{X}) - \mathbf{X} \|_F^2 \leq (1 + \epsilon) \| \mathcal{P}_k(\mathbf{X}) - \mathbf{X} \|_F^2, \tag{29} \]

where \( \mathcal{P}_k(\mathbf{X}) \in \arg \min_{\mathbf{Y} \in \mathbb{R}^{m \times \text{rank}(\mathbf{Y}) \leq k}} \| \mathbf{X} - \mathbf{Y} \|_F \).

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

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

\[ \mathcal{D}_i \leftarrow \mathcal{P}_k(\mathbf{X}_i \nabla f(\mathbf{X}(i))). \]

be a set of orthonormal, rank-1 matrices that span rank-\( k \) subspaces such that (29) is satisfied for \( \mathbf{X} = \mathcal{P}_k(\mathbf{X}_i \nabla f(\mathbf{X}(i))) \).

Then, at each iteration, the Active Subspace Expansion step in Algorithms 1 and 2 captures information contained in the true matrix \( \mathbf{X}^* \), such that:

\[ \| \mathcal{P}_k(\mathbf{X}_i \nabla f(\mathbf{X}(i))) - \mathbf{X}^* \|_F \leq \left( 2\delta_{2k} + 2\delta_{2k} \right) \| \mathbf{X}(i) - \mathbf{X}^* \|_F + \sqrt{2(1 + \delta_{2k})} \| \epsilon \|_2 + 2\lambda \sqrt{\epsilon}, \tag{30} \]

where \( \beta \leq \mathbf{X}(i) \) \( \epsilon \)-approximate projection of \( \mathbf{X}(i) \).

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

\[ \| \mathbf{W}(i) - \mathbf{V}(i) \|_F^2 \leq (1 + \epsilon) \| \mathbf{W}(i) - \mathbf{V}(i) \|_F^2 \leq (1 + \epsilon) \| \mathbf{P}_k(\mathbf{V}(i) - \mathbf{X}^*) \|_F \leq (1 + \epsilon) \| \mathbf{V}(i) - \mathbf{X}^* \|_F \tag{31} \]

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

8.1 MATRIX ALPS I using \( \epsilon \)-approximate low-rank projection via column subset selection

Using \( \epsilon \)-approximate SVD in MATRIX ALPS I, the following iteration invariant theorem holds:

Theorem 4 [Iteration invariant with \( \epsilon \)-approximate projections for MATRIX ALPS I] The (\( i+1 \))-th matrix estimate \( \mathbf{X}(i+1) \) of MATRIX ALPS I with \( \epsilon \)-approximate projections \( \mathcal{P}_k(\mathbf{X}_i \nabla f(\mathbf{X}(i))) \) and \( \mathbf{W}(i) \leftarrow \mathcal{P}_k(\mathbf{V}(i)) \) in Algorithm 1 satisfies the following recursion:

\[ \| \mathbf{X}(i+1) - \mathbf{X}^* \|_F \leq \rho \| \mathbf{X}(i) - \mathbf{X}^* \|_F + \gamma \| \epsilon \|_2 + \beta \lambda, \tag{32} \]
where \( \rho := \left(1 + \frac{3\delta}{4 \kappa}ight) (2 + \epsilon) \left(1 + \frac{\delta_2}{2 \delta_2} \right) 4 \delta_3 k + \frac{2 \delta_3}{1 - 2 \delta_2}, \)
\( \beta := \left(1 + \frac{3\delta}{4 \kappa}ight) (2 + \epsilon) \left(1 + \frac{\delta_2}{2 \delta_2} \right) 2 \sqrt{\epsilon}, \text{ and} \)
\( \gamma := \left(1 + \frac{3\delta}{4 \kappa}ight) (2 + \epsilon) \left(1 + \frac{\delta_2}{2 \delta_2} \right) \sqrt{2(1 + \delta_2)} + \frac{2 \sqrt{1 + \frac{3 \delta}{4 \kappa} \delta_2}}{1 - 2 \delta_2} \).

![Graph](image)

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

Similar analysis can be conducted for the ADMiRA algorithm. To illustrate the impact of SVD \( \epsilon \)-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 \( \epsilon \)-approximation SVD algorithm, presented in [47]. In this paper, the column subset selection algorithm satisfies the following theorem:

**Theorem 5** Let \( X \in \mathbb{R}^{m \times n} \) be a signal of interest with arbitrary rank \( \leq \min\{m, n\} \) and let \( X_k \) represent the best rank-\( k \) approximation of \( X \). 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 \( \mathcal{P}_k(X) \in \mathbb{R}^{m \times n} \) 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 \( O(mn(k/\epsilon + k^2 \log k) + (m + n)(k^2/\epsilon + k^3 \log k) \) 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 Figure 4 we compare SVD-based MATRIX ALPS II with MATRIX ALPS II using the \( \epsilon \)-approximate column subset selection method in [47]. We observe that the total number of iterations for convergence increases due to \( \epsilon \)-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 Section 8, where \( \epsilon \) 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 Lagrangian 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 \( \ell_1 \)-norm loss function into the Grassmannian optimization framework, (v) the Riemannian 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 present a set of parameters that are common. We denote the ratio between the number of observed samples and the number of variables in \( X^* \) as \( \text{SR} := p/(m \cdot n) \) (sampling ratio). Furthermore, we reserve \( \text{FR} \) 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 \( X^* \). This way, we create a wide range of different problem configurations with variable \( \text{FR} \).

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 easily be 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 \( cg\_max\_iter := 500 \) and tolerance parameter \( cg\_tol := 10^{-10} \). We use the same stopping criteria for the majority of algorithms under consideration:

\[
\frac{\|X(i) - X(i-1)\|_F}{\|X(i)\|_F} \leq tol,
\]

where \( X(i), X(i-1) \) denote the current and the previous estimate of \( X^* \) and \( tol := 5 \times 10^{-5} \). 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 \( \mathcal{A} \) and \( \mathcal{A}^* \) 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 \( X^* \) 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 \( Q_i \leftarrow \text{orth}(X_i \cup X_{i+1}) \) to construct a set of orthonormal rank-1 matrices that span the subspace, spanned by \( X_i \cup X_{i+1} \). 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, 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 Figures 5(a)-(b), we use subsampled and permuted noiselets for linear map \( A \) and in Figure 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 Figure 5. To this end, for the rest of the paper, we use **Matrix ALPS II** where \( Q_i \leftarrow X_i \cup X_{i+1} \).

### 10.3 Limitations of \( \| \cdot \|_r \)-based algorithms: a toy example

While nuclear 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 \( \| \cdot \|_r \)-norm in the objective function as the convex surrogate of the rank(.) metric would 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 \( X^* \in \mathbb{R}^{5 \times 4} \) be the matrix of interest with rank(\( X^* \)) = 2, as shown in Figure 6(a). We consider the case where we have access to \( X^* \) only through a subset of its entries, as shown in Figure 6(b).

\[
\begin{bmatrix}
2 & 2 & 1 & 1 \\
2 & 2 & 1 & 1 \\
2 & 2 & 1 & 1 \\
2 & 1 & 2 & 1 \\
1 & 1 & 2 & 1 \\
\end{bmatrix}
\quad \begin{bmatrix}
2 & 2 & 1 & 1 \\
2 & 2 & 1 & 1 \\
? & ? & ? & ? \\
2 & 2 & 1 & 1 \\
? & ? & ? & ? \\
\end{bmatrix}
\]

(a) \quad (b)

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

In Figure 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 \( X^* \), most of the algorithms in comparison reconstruct \( X^* \) successfully. We note that, in some cases, the inadequacy of an algorithm to reconstruct \( X^* \) 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 \( y \in \mathbb{R}^p \) satisfy:

\[
y = AX^* + \varepsilon
\]

(34)

Here, we use permuted and subsampled noiselets for the linear operator \( A \) [12]. The signal \( X^* \) is generated as the multiplication of two low-rank matrices, \( L \in \mathbb{R}^{m \times k} \) and \( R \in \mathbb{R}^{n \times k} \), such that \( X^* = LR^T \) and \( \| X^* \|_F = 1 \). Both \( L \) and \( R \) have random independent and identically distributed (iid) Gaussian entries with zero mean and unit variance. In the noisy case, the additive noise term \( \varepsilon \in \mathbb{R}^p \) contains entries drawn from a zero mean Gaussian distribution with \( \| \varepsilon \| \in (10^{-3}, 10^{-2}) \).

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, Figure 8 illustrates the effectiveness of the algorithms for some representative problem configurations.

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 \( X^* \in \mathbb{R}^{m \times n} \) is synthesized as a rank-\( k \) matrix, factorized as \( X^* := LR^T \) with \( \| X^* \|_F = 1 \) where \( L \in \mathbb{R}^{m \times k} \) and \( R \in \mathbb{R}^{n \times k} \) as defined above. In sequence, we subsample \( X^* \) 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 \( \Omega = \{ i,j \} : [X^*]_{ij} \) is known \( \subseteq \{1,\ldots,m\} \times \{1,\ldots,n\} \) and let \( A_{\Omega} \) denote the linear operator (mask) that samples a matrix according to \( \Omega \). Then, the set of observations satisfies:

\[
y = A_{\Omega}X^* + \varepsilon,
\]

(35)

i.e., the known entries of \( X^* \) are structured as a vector \( y \in \mathbb{R}^p \), disturbed by a dense noise vector \( \varepsilon \in \mathbb{R}^p \) 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 \( \| \epsilon \|_2 \in \{10^{-3}, 10^{-4}\} \). 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 \( \text{SR} = 0.3 \) 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 Figures 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, its execution time performance degrades significantly as we increase the rank of \( X^* \). 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 faster convergence for most of the cases and scales well as the problem size increases. We note that we do not exploit
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 0.3mn. (a) $m = 256$, $n = 512$, $k = 10$ and $\|\epsilon\|_2 = 10^{-3}$. (b) $m = 256$, $n = 512$, $k = 10$ and $\|\epsilon\|_2 = 10^{-4}$. (c) $m = 256$, $n = 512$, $k = 20$ and $\|\epsilon\|_2 = 0$. (d) $m = 512$, $n = 1024$, $k = 30$ and $\|\epsilon\|_2 = 0$. (e) $m = 512$, $n = 1024$, $k = 40$ and $\|\epsilon\|_2 = 0$. (f) $m = 1024$, $n = 2048$, $k = 50$ and $\|\epsilon\|_2 = 0$.

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) $m = 600$, $n = 600$, $k = 5$ and $\|\epsilon\|_2 = 0$. (b) $m = 600$, $n = 600$, $k = 20$ and $\|\epsilon\|_2 = 10^{-4}$. 
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 observed 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 Figure 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 Figure 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 Figure 11 where the full set of pixels is observed. Given a fixed common tolerance and the same stopping criteria, Figure 11 shows the recovery performance achieved by a range of algorithms under consideration for 10 Monte-Carlo realizations. We repeat the same experiment of the true image. In both cases,

| Configuration | FR | SVP | GROUSE | TFOCS |
|---------------|----|-----|--------|-------|
| \( m n k \)   |   | iter. | err. | time | iter. | err. | time | iter. | err. | time |
| 300 600 5    | 0.083 | 42 | 2.9 \times 10^{-4} | 0.59 | - | 1.52 \times 10^{-4} | 0.08 | - | 8.69 \times 10^{-7} | 3.36 |
| 300 600 5 \times 10^{-4} | 0.083 | 42 | 6 \times 10^{-4} | 0.65 | - | 2 \times 10^{-4} | 0.082 | - | 5 \times 10^{-7} | 3.85 |
| 300 600 5 \times 10^{-4} | 0.083 | 42 | 3 \times 10^{-4} | 0.64 | - | 2 \times 10^{-4} | 0.079 | - | 1 \times 10^{-4} | 3.5 |
| 300 600 5 \times 10^{-4} | 0.165 | 54 | 4 \times 10^{-4} | 0.9 | - | 4.5 \times 10^{-6} | 0.22 | - | 2 \times 10^{-4} | 6.43 |
| 300 600 5 \times 10^{-4} | 0.165 | 54 | 9 \times 10^{-4} | 0.89 | - | 2 \times 10^{-4} | 0.16 | - | 8 \times 10^{-4} | 7.83 |
| 300 600 5 \times 10^{-4} | 0.165 | 54 | 4 \times 10^{-4} | 0.91 | - | 2 \times 10^{-4} | 0.16 | - | 1 \times 10^{-4} | 6.75 |
| 300 600 5 \times 10^{-4} | 0.320 | 85 | 8 \times 10^{-4} | 2.04 | - | 1 \times 10^{-4} | 0.81 | - | 2 \times 10^{-4} | 30.04 |
| 300 600 5 \times 10^{-4} | 0.037 | 241 | 3.4 \times 10^{-5} | 11.1 | - | 3 \times 10^{-3} | 13.94 | - | - | - |

Table 2 Matrix Completion problem for \( m = 300 \) and \( n = 600 \). “-” depicts no information or not applicable due to time overhead.
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 |
|---------------|----|-----|-------------|--------|
| \( m \) \( n \) \( k \) | \( ||e||_2 \) | iter. | err. | time | iter. | err. | time | iter. | err. | time |
| 700 1000 5 0 | 0.04 | 34 | 1.9 \( \cdot 10^{-4} \) | 1.77 | 23 | 6.5 \( \cdot 10^{-6} \) | 1.69 | - | 3.5 \( \cdot 10^{-5} \) | 0.23 |
| 700 1000 10 \(-3\) | 0.04 | 34 | 4.2 \( \cdot 10^{-4} \) | 1.92 | 23 | 3.7 \( \cdot 10^{-4} \) | 1.87 | - | 3.1 \( \cdot 10^{-5} \) | 0.24 |
| 700 1000 30 0 | 0.239 | 61 | 4.6 \( \cdot 10^{-4} \) | 6.39 | 29 | 1.2 \( \cdot 10^{-4} \) | 3.91 | - | 3.2 \( \cdot 10^{-5} \) | 3.15 |
| 700 1000 30 \(-3\) | 0.239 | 61 | 1.1 \( \cdot 10^{-3} \) | 6.43 | 29 | 1 \( \cdot 10^{-5} \) | 3.87 | - | 8 \( \cdot 10^{-6} \) | 3.14 |
| 700 1000 50 0 | 0.393 | 95 | 8.5 \( \cdot 10^{-4} \) | 14.47 | 49 | 3.2 \( \cdot 10^{-4} \) | 9.02 | - | 1.3 \( \cdot 10^{-5} \) | 10.31 |
| 700 1000 50 \(-3\) | 0.393 | 95 | 1.6 \( \cdot 10^{-3} \) | 15.15 | 49 | 1.4 \( \cdot 10^{-3} \) | 9.11 | - | 8 \( \cdot 10^{-4} \) | 10.34 |
| 700 1000 110 0 | 0.833 | 683 | 1.2 \( \cdot 10^{-2} \) | 253.1 | 574 | 5.8 \( \cdot 10^{-3} \) | 102.61 | - | 1.2 \( \cdot 10^{-2} \) | 110.93 |
| 700 1000 110 \(-3\) | 0.833 | 682 | 1.4 \( \cdot 10^{-2} \) | 256.21 | 574 | 4.8 \( \cdot 10^{-3} \) | 154.34 | - | 1.6 \( \cdot 10^{-2} \) | 111.95 |

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 |
|---------------|----|-----|-------------|--------|
| \( m \) \( n \) \( k \) | \( ||e||_2 \) | iter. | err. | time | iter. | err. | time | iter. | err. | time |
| 500 2000 30 0 | 0.083 | 64 | 5.3 \( \cdot 10^{-4} \) | 10.18 | 32 | 1.9 \( \cdot 10^{-4} \) | 6.47 | - | 1.6 \( \cdot 10^{-4} \) | 2.46 |
| 500 2000 30 \(-3\) | 0.083 | 64 | 1.1 \( \cdot 10^{-3} \) | 6.69 | 32 | 1 \( \cdot 10^{-5} \) | 4.51 | - | 6 \( \cdot 10^{-5} \) | 1.94 |
| 500 2000 30 \(-4\) | 0.083 | 64 | 5.4 \( \cdot 10^{-4} \) | 10.14 | 32 | 2.2 \( \cdot 10^{-4} \) | 6.51 | - | 1.6 \( \cdot 10^{-4} \) | 2.46 |
| 500 2000 30 \(-5\) | 0.083 | 164 | 1.1 \( \cdot 10^{-2} \) | 15.5 | 54 | 5 \( \cdot 10^{-4} \) | 19.8 | - | 8 \( \cdot 10^{-5} \) | 7.32 |
| 500 2000 50 0 | 0.408 | 165 | 1.8 \( \cdot 10^{-3} \) | 24.97 | 54 | 1.8 \( \cdot 10^{-3} \) | 16.14 | - | 9 \( \cdot 10^{-4} \) | 8.56 |
| 500 2000 50 \(-3\) | 0.408 | 382 | 1.1 \( \cdot 10^{-3} \) | 24.85 | 54 | 5 \( \cdot 10^{-4} \) | 16.17 | - | 7 \( \cdot 10^{-5} \) | 8.59 |
| 500 2000 80 0 | 0.645 | 179 | 3.5 \( \cdot 10^{-3} \) | 92.91 | 134 | 1.7 \( \cdot 10^{-3} \) | 59.33 | - | 1 \( \cdot 10^{-4} \) | 79.64 |
| 500 2000 80 \(-3\) | 0.645 | 299 | 4.2 \( \cdot 10^{-3} \) | 94.86 | 134 | 2.8 \( \cdot 10^{-3} \) | 60.68 | - | 1 \( \cdot 10^{-4} \) | 79.98 |
| 500 2000 80 \(-4\) | 0.645 | 179 | 3.6 \( \cdot 10^{-3} \) | 93.95 | 134 | 1.8 \( \cdot 10^{-3} \) | 60.76 | - | 1 \( \cdot 10^{-4} \) | 79.48 |
| 500 2000 100 0 | 0.8 | 523 | 1.1 \( \cdot 10^{-2} \) | 259.13 | 307 | 6 \( \cdot 10^{-3} \) | 173.14 | - | 4 \( \cdot 10^{-3} \) | 143.41 |
| 500 2000 100 \(-3\) | 0.8 | 525 | 1.2 \( \cdot 10^{-2} \) | 262.19 | 308 | 7 \( \cdot 10^{-3} \) | 176.04 | - | 5 \( \cdot 10^{-3} \) | 142.85 |
| 500 2000 100 \(-4\) | 0.8 | 522 | 1.2 \( \cdot 10^{-2} \) | 262.11 | 307 | 6 \( \cdot 10^{-3} \) | 170.41 | - | 4 \( \cdot 10^{-3} \) | 142.85 |

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

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

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 \( \mu \) selection schemes are accompanied with strong RIP constant conditions but empirical evidence reveal signal reconstruction vulnerabilities. While con-
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 $\rho = 0.3nn$. (a) $m = 700$, $n = 1000$, $k = 30$ and $\|\epsilon\|_2 = 0$ (b) $m = 700$, $n = 1000$, $k = 50$ and $\|\epsilon\|_2 = 10^{-3}$. (c) $m = 700$, $n = 1000$, $k = 110$ and $\|\epsilon\|_2 = 0$. (d) $m = 500$, $n = 2000$, $k = 10$ and $\|\epsilon\|_2 = 0$. (e) $m = 500$, $n = 2000$, $k = 50$ and $\|\epsilon\|_2 = 10^{-3}$. (f) $m = 500$, $n = 2000$, $k = 80$ and $\|\epsilon\|_2 = 10^{-4}$.

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.

We connect $\epsilon$-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 $\epsilon$-approximation schemes. In all cases, experimental results illustrate the effectiveness of the proposed schemes on different problem configurations.

Acknowledgments

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.

A Appendix

Remark 1 Let $X \in \mathbb{R}^{m \times n}$ with SVD: $X = U \Sigma V^T$, and $Y \in \mathbb{R}^{m \times n}$ with SVD: $Y = \tilde{U} \tilde{\Sigma} V^T$. Assume two sets: i) $S_1 = \{u_i u_j^T : i \in I_1\}$ where $u_i$ is the $i$-th singular vector of $X$ and $I_1 \subseteq \{1, \ldots, \text{rank}(X)\}$ and, ii) $S_2 = \{u_i u_j^T, \tilde{u}_i, \tilde{u}_j^T : i \in I_2, j \in I_3\}$ where $u_i$ is the $i$-th singular vector of $Y$, $I_1 \subseteq I_2 \subseteq \{1, \ldots, \text{rank}(X)\}$ and, $I_3 \subseteq \{1, \ldots, \text{rank}(Y)\}$. We observe that the subspaces defined by $u_i u_i^T$ and $\tilde{u}_i \tilde{u}_j^T$ are not necessarily orthogonal.

To this end, let $S_2 = \text{ortho}(S_2)$; this operation can be easily computed via SVD. Then, the following commutativity property holds true for any matrix $W \in \mathbb{R}^{m \times n}$:

$$P_{S_1} P_{S_2} W = P_{S_2} P_{S_1} W.$$  

(36)

A.1 Proof of Lemma 6

Given $X^* \leftarrow P_{k}(X^{*})$ using SVD factorization, we define the following quantities: $S_1 \leftarrow X^* \cup D_1, S_2^{\perp} \leftarrow \text{ortho} (A_i \cup A^*)$. Then, given the structure of the sets $S_1$ and $S_2^{\perp}$.

$$P_{S_1} P_{(S_2^{\perp})} W = P_{D_1} P_{(X^* \cup A_i^{\perp})}.$$  

(37)
and
\[ P_{S_1^\perp} P_{S_1^\perp} = P_{X_1^\perp} P_{(D_1 \cup X_1)^\perp} \] (38)

Since the subspace defined in \( D_i \) is the best rank-\( k \) subspace, orthogonal to the subspace spanned by \( X_i \), the following holds true:
\[ \left\| P_{D_i} X_1^\perp \nabla f(X(i)) \right\|_F^2 \geq \left\| P_{X_1^\perp} X_1^\perp \nabla f(X(i)) \right\|_F^2 \Rightarrow \left\| P_{D_i} \nabla f(X(i)) \right\|_F^2 \geq \left\| P_{S_1^\perp} \nabla f(X(i)) \right\|_F^2 \]

Removing the common subspaces in \( S_i \) and \( S_1^\perp \) by the commutativity property of the projection operation and using the shortcut \( P_{A \cup B} \equiv P_A P_B \) for sets \( A, B \), we get:
\[ \left\| P_{S_1 \setminus S_1^\perp} \nabla f(X(i)) \right\|_F^2 \geq \left\| P_{S_1^\perp \setminus S_1} \nabla f(X(i)) \right\|_F^2 \Rightarrow \left\| P_{S_1 \setminus S_1^\perp} A^* A (X^* - X(i)) + P_{S_1 \setminus S_1^\perp} e \right\|_F \geq \left\| P_{S_1^\perp \setminus S_1} A^* A (X^* - X(i)) + P_{S_1^\perp \setminus S_1} e \right\|_F \] (39)

Next, we assume that \( P_{(A \cup B)^\perp} \) denotes the orthogonal projection onto the subspace spanned by \( P_{A \cup B} \). Then, on the left hand side of (39), we have:
\[ \left\| P_{S_1 \setminus S_1^\perp} A^* A (X^* - X(i)) + P_{S_1 \setminus S_1^\perp} e \right\|_F \]
\[ \leq \left\| P_{S_1 \setminus S_1^\perp} A^* A (X^* - X(i)) \right\|_F + \left\| P_{S_1 \setminus S_1^\perp} e \right\|_F \]
\[ = \left\| P_{S_1 \setminus S_1^\perp} (X^* - X(i)) + P_{S_1 \setminus S_1^\perp} A^* A (X^* - X(i)) \right\|_F \]
\[ + \left\| P_{S_1 \setminus S_1^\perp} e \right\|_F \]
\[ \leq \left\| (I - P_{S_1 \setminus S_1^\perp} A^* A P_{S_1 \setminus S_1^\perp}) (X^* - X(i)) \right\|_F + \left\| P_{S_1 \setminus S_1^\perp} A^* A P_{S_1 \setminus S_1^\perp} (X^* - X(i)) \right\|_F \]
\[ + \left\| P_{S_1 \setminus S_1^\perp} e \right\|_F \]
Reconstruction performance in image denoising settings. The image size is $256 \times 256$ 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.

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

$$\|P_{S_1 \setminus S_i} A^\ast (X^\ast - X(i))\|_F \geq \|P_{S_1 \setminus S_i} (X^\ast - X(i))\|_F - \|P_{S_1 \setminus S_i} A^\ast A (X^\ast - X(i))\|_F$$

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

$$\|P_{X^\ast \setminus S_i} X^\ast\|_F \leq (2\delta_{2k} + 2\delta_{3k}) \|X(i) - X^\ast\|_F + \sqrt{2(1 + \delta_{2k})}\|\varepsilon\|_2.$$
A.2 Proof of Theorem 1

Let \( X^* \leftarrow P_S(X^*) \) be a set of orthonormal, rank-1 matrices that span the range of \( X^* \). In Algorithm 1, \( W(i) \leftarrow P_S(V(i)) \). Thus:

\[
\|W(i) - V(i)\|_F^2 \leq \|X^* - V(i)\|_F^2 \Rightarrow \\
\|W(i) - X^* + X^* - V(i)\|_F^2 \leq \|X^* - V(i)\|_F^2 \Rightarrow \\
\|W(i) - X^*\|_F^2 \leq 2\|W(i) - X^*, V(i) - X^*\|_F^2.
\]

From Algorithm 1, (i) \( V(i) \in \text{span}(S_i) \), (ii) \( X(i) \in \text{span}(S_i) \) and (iii) \( W(i) \in \text{span}(S_i) \). We define \( \mathcal{E} \leftarrow \text{orth}(S_i \cup X^*) \) where rank(span(\( \mathcal{E} \))) \leq 3k and let \( P_\mathcal{E} \) be the orthogonal projection onto the subspace defined by \( \mathcal{E} \).

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

\[
W(i) - X^* = P_\mathcal{E}(W(i) - X^*) \quad \text{and} \quad V(i) - X^* = P_\mathcal{E}(V(i) - X^*).
\]

Then, (42) can be written as:

\[
\|W(i) - X^*\|_F^2 \leq 2\|P_\mathcal{E}(W(i) - X^*), P_\mathcal{E}(V(i) - X^*)\|_F \Rightarrow \\
= 2\|P_\mathcal{E}(W(i) - X^*), P_\mathcal{E}(X(i) - X^*) - \mu_i P_S, A^* A P_S(X(i) - X^*)\|_F \Rightarrow \\
+ 2\mu_i \|P_\mathcal{E}(W(i) - X^*), P_\mathcal{E}(P_S, A^* A P_S(X(i) - X^*))\|_F. (43)
\]

In B, we observe:

\[
B := 2\mu_i \|P_\mathcal{E}(W(i) - X^*), P_\mathcal{E}(P_S, A^* A P_S(X(i) - X^*))\|_F \leq \\
(\delta_k^2 + 2\mu_i \|W(i) - X^*\|_F \|P_S, A^* A P_S\|_F) \leq \\
2\mu_i \|W(i) - X^*\|_F \|P_S, A^* A P_S\|_F \leq 2\mu_i \|W(i) - X^*\|_F \|P_S, A^* A P_S\|_F. (44)
\]

where (i) holds since \( P_S P_P = P_P P_S = P_S \), for 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:

\[
A := 2\|W(i) - X^*), P_\mathcal{E}(X(i) - X^*) - \mu_i P_S, A^* A P_\mathcal{E}(X(i) - X^*)\|_F \leq \\
(\delta_k^2 + 2\mu_i \|W(i) - X^*\|_F \|P_S, A^* A P_S\|_F) \leq \\
2\% \|W(i) - X^*), P_\mathcal{E}(X(i) - X^*)\|_F \leq \\
2\mu_i \|W(i) - X^*\|_F \|P_S, A^* A P_S\|_F. (45)
\]

where (i) is due to \( P_S P_\mathcal{E}(X(i) - X^*) := P_\mathcal{E}(P_S(X(i) - X^*) + P_\mathcal{E}(P_\mathcal{E}(X(i) - X^*)) \) and (ii) follows from Cauchy-Schwarz inequality. Since \( \frac{1}{1 + \delta_k} \leq \mu_i \leq \frac{1}{1 - \delta_k} \), Lemma 4 implies:

\[
2\mu_i \|W(i) - X^*\|_F \leq 2\|W(i) - X^*\|_F \leq 2\|X^*\|_F.
\]

and thus:

\[
\|(I - \mu_i P_S, A^* A P_S P_\mathcal{E}(X(i) - X^*)\|_F \leq \\
\leq 2\delta_k \|P_\mathcal{E}(X(i) - X^*)\|_F.
\]

Furthermore, according to Lemma 5:

\[
\|P_S, A^* A P_\mathcal{E}(X(i) - X^*)\|_F \leq \delta_k \|P_\mathcal{E}(X(i) - X^*)\|_F
\]

since rank(\( P_\mathcal{E}X \)) \leq 3k, \( \forall X \in R^{m \times n} \) for \( \mathcal{E} \leftarrow \text{orth}(S_i \cup X^*) \). Since \( P_S, A^* A P_\mathcal{E}(X(i) - X^*) = P_\mathcal{E}(X(i) - X^*), \quad \mathcal{E} \)

where \( D_i \leftarrow P_\mathcal{E} \left( P_\mathcal{E} X \right) \), then:

\[
\|P_\mathcal{E}(X(i) - X^*)\|_F \leq \|P_\mathcal{E}(X(i) - X^*)\|_F \leq \|P_\mathcal{E}(X(i) - X^*)\|_F \leq \sqrt{2(1 + \delta_k)} \|X^*\|_2,
\]

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

\[
A \leq \left( \frac{4\delta_k}{1 - \delta_k} + (2\delta_k + 2\delta_k) \frac{2\delta_k}{1 - \delta_k} \right) \|W(i) - X^*\|_F \leq \\
\|X^*(i + 1) - X^*\|_F \leq \|X^*(i + 1) - X^*\|_F \leq \sqrt{2(1 + \delta_k)} \|X^*\|_2 \|
\]

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

\[
\|W(i) - X^*\|_F \leq \\
\leq \left( \frac{4\delta_k}{1 - \delta_k} + (2\delta_k + 2\delta_k) \frac{2\delta_k}{1 - \delta_k} \right) \|X^*(i + 1) - X^*\|_F \leq \\
\|X^*(i + 1) - X^*\|_F \leq \|X^*(i + 1) - X^*\|_F \leq \sqrt{2(1 + \delta_k)} \|X^*\|_2 \|
\]

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

\[
\|X(i + 1) - X^*\|_F \leq \rho \|X(i) - X^*\|_F + \gamma \|X^*\|_2,
\]

for \( \rho := \left( \frac{1 + 2\delta_k}{1 - \delta_k} \right) \left( \frac{4\delta_k}{1 - \delta_k} + (2\delta_k + 2\delta_k) \frac{2\delta_k}{1 - \delta_k} \right) \) and

\[
\gamma := \left( \frac{1 + 2\delta_k}{1 - \delta_k} \right) \left( \frac{2\delta_k}{1 - \delta_k} + \frac{2\delta_k}{1 - \delta_k} \right) \sqrt{2(1 + \delta_k)} \right) + \frac{\sqrt{1 + \delta_k}}{1 - \delta_k} \]

For the convergence parameter \( \rho \), further compute:

\[
(1 + 2\delta_k) \left( \frac{4\delta_k}{1 - \delta_k} + (2\delta_k + 2\delta_k) \frac{2\delta_k}{1 - \delta_k} \right) \leq \\
\frac{1 + 2\delta_k}{1 - \delta_k} \left( \frac{4\delta_k}{1 - \delta_k} + (2\delta_k + 2\delta_k) \frac{2\delta_k}{1 - \delta_k} \right)
\]

for \( \delta_k \leq \delta_k \leq \delta_k \). Calculating the roots of this expression, we easily observe that \( \rho < 1 \) for \( \delta_k < 0.1235. \)
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 $S_i$ be a set of orthonormal, rank-1 matrices that span a rank-2 subspace in $\mathbb{R}^{m \times n}$. Then, the least squares solution $V(i)$ given by:

$$V(i) \leftarrow \arg \min_{V \in \text{span}(S_i)} \|y - AV\|^2_F$$

satisfies:

$$\|V(i) - X^*\|^2_F \leq \frac{1}{\sqrt{1 - \delta_{3k}(A)}} \|P_{S_i^F}(V(i) - X^*)\|_F$$

$$+ \frac{\sqrt{1 + \delta_{3k}}}{1 - \delta_{3k}} \|\epsilon\|^2_F.$$  \hfill (50)

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

$$\|V(i) - X^*\|^2_F = \|P_{S_i}(V(i) - X^*)\|^2_F + \|P_{S_i^F}(V(i) - X^*)\|^2_F.$$  \hfill (52)

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

$$\langle \nabla \|V(i)\|_F, P_{S_i}(X^* - V(i)) \rangle \geq 0 \Rightarrow \langle AV(i) - y, A P_{S_i}(V(i) - X^*) \rangle \leq 0.$$  \hfill (53)

for $P_{S_i} X^* \in \text{span}(S_i)$. Given condition (53), the first term on the right hand side of (52) becomes:

$$\|P_{S_i}(V(i) - X^*)\|^2_F = \|V(i) - X^*, P_{S_i}(V(i) - X^*)\|^2_F$$

$$\leq \|V(i) - X^*, P_{S_i}(V(i) - X^*)\|_F$$

$$\leq \langle AV(i) - y, A P_{S_i}(V(i) - X^*) \rangle$$

$$\leq (I - \mathcal{A}^* \mathcal{A}) P_{S_i}(V(i) - X^*)$$

$$\|e, P_{S_i}(V(i) - X^*)\|_F$$

(54)

Focus on the term $\|[(V(i) - X^*, (I - \mathcal{A}^* \mathcal{A}) P_{S_i}(V(i) - X^*))]$, we derive the following:

$$\|P_{S_i}(V(i) - X^*)\|^2_F$$

$$\leq \|[(V(i) - X^*, (I - \mathcal{A}^* \mathcal{A}) P_{S_i}(V(i) - X^*))]$$

$$\|e, P_{S_i}(V(i) - X^*)\|_F$$

(55)

and thus $P_{S_i^F}(V(i) - X^*) = V(i) - X^*$ and (ii) is due to $F_{S_i^F \cup X^*} = P_{S_i}$ since span($S_i$) $\subseteq$ span(orth(Si $\cup X^*$)). Then, (54) becomes:

$$\|P_{S_i}(V(i) - X^*)\|^2_F$$

$$\leq \|[(V(i) - X^*, (I - \mathcal{A}^* \mathcal{A}) P_{S_i}(V(i) - X^*))]$$

$$\|e, P_{S_i}(V(i) - X^*)\|_F$$

(56)

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

$$\|V(i) - X^*\|^2_F \leq \frac{1}{\sqrt{1 - \delta_{3k}(A)}} \|P_{S_i^F}(V(i) - X^*)\|_F$$

$$+ \frac{\sqrt{1 + \delta_{3k}}}{1 - \delta_{3k}} \|\epsilon\|^2_F.$$  \hfill (57)

The following Lemma characterizes how subspace pruning affects the recovered energy:

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

$$\|X(i + 1) - V(i)\|_F \leq \|P_{S_i}(V(i) - X(i))\|_F$$

$$\leq \|V(i) - X^*\|_F.$$  \hfill (59)

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

$$\|X(i + 1) - V(i)\|_F \leq \|X - V(i)\|_F.$$  \hfill (60)

Since $P_{S_i} V(i) = V(i)$, the left inequality in (59) is satisfied for $X := P_{S_i} X^*$ in (60).

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

$$\|X(i + 1) - X^*\|_F \leq \sqrt{1 + \frac{3\delta_{3k}^2}{1 - \delta_{3k}}} \|V(i) - X^*\|_F$$

$$+ \frac{\sqrt{1 + \delta_{3k}}}{1 - \delta_{3k}} \|\epsilon\|_2.$$  \hfill (61)
Proof. We observe the following

\[
\|X(i+1) - X^*\|_F^2 = \|X(i+1) - V(i) + V(i) - X^*\|_F^2 = \|V(i) - X^*\|_F^2 + \|V(i) - X(i+1)\|_F^2 - 2 \langle V(i) - X^*, V(i) - X(i+1) \rangle.
\]

(62)

Focusing on the right hand side of expression (62), \(\langle V(i) - X^*, P_S(i) (V(i) - X(i+1)) \rangle\) can be similarly analysed as in Lemma 10 where we obtain the following expression:

\[
\|V(i) - X^*, P_S(i) (V(i) - X(i+1))\|_F = \delta_{3k} \|V(i) - X^*\|_F \|V(i) - X(i+1)\|_F + \sqrt{1 + \delta_{2k}} \|V(i) - X(i+1)\|_F \|\epsilon\|_2^2.
\]

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

\[
\|X(i+1) - X^*\|_F^2 \leq \|V(i) - X^*\|_F^2 + \|P_S(i) (V(i) - X^*)\|_F^2 + 2 \delta_{3k} \|V(i) - X^*\|_F \|V(i) - X(i+1)\|_F + \sqrt{1 + \delta_{2k}} \|V(i) - X(i+1)\|_F \|\epsilon\|_2^2.
\]

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

\[
\|X(i+1) - X^*\|_F^2 \leq \left(1 + 3\delta_{2k}\right) \left(\|V(i) - X^*\|_F^2 + \|P_S(i) (V(i) - X^*)\|_F^2\right) + 2 \delta_{3k} \|V(i) - X^*\|_F \|V(i) - X(i+1)\|_F + \sqrt{1 + \delta_{2k}} \|P_S(i) (V(i) - X^*)\|_F \|\epsilon\|_2^2.
\]

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

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

\[
\|X(i+1) - X^*\|_F \leq \left[1 + 3\delta_{2k}\right] \|P_S(i) (V(i) - X^*)\|_F + \sqrt{1 + \delta_{2k}} \|\epsilon\|_2.
\]

Since \(V(i) \in \text{span}(S_i)\), we observe \(P_S(i) (V(i) - X^*) = -P_{S(i) \perp (\text{span}(S_i) \cup X^*)} X^*\). Then, using Lemma 6, we obtain:

\[
\|X(i+1) - X^*\|_F \leq \left(2\delta_{2k} + 2\delta_{3k}\right) \sqrt{\left(1 + \frac{2\delta_{2k}}{1 - \delta_{3k}}\right) \|X^* - X(i)\|_F} + \sqrt{2\left(1 + \delta_{2k}\right) \|\epsilon\|_2} + \sqrt{\left(1 + \frac{2\delta_{2k}}{1 - \delta_{3k}}\right) \|\epsilon\|_2}.
\]

(67)

Given \(\delta_{2k} \leq \delta_{3k}\), \(\rho\) is upper bounded by \(\rho \leq 4\delta_{3k} \sqrt{\frac{1 + 2\delta_{2k}}{1 - \delta_{3k}}} < 1\) \(\Rightarrow \delta_{3k} < 0.2267\).

A.4 Proof of Theorem 3

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

\[
\|\hat{X}(i+1) - V(i)\|_F \leq \|X^* - V(i)\|_F \Rightarrow \|\hat{X}(i+1) - X^*\|_F \leq 2 \|\hat{X}(i+1) - X^*\|_F \|V(i) - X^*\|_F.
\]

From Algorithm 3, \(V(i) \in \text{span}(S_i), Q_i \in \text{span}(S_i)\) and \(W(i) \in \text{span}(S_i)\). We define \(\tilde{E} \leftarrow \text{ortho}(S_i \cup X^*)\) where we observe \(\text{rank}(\tilde{E}) \leq 4k\) and let \(P_{\tilde{S}}\) be the orthogonal projection onto the subspace defined by \(\tilde{E}\).

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

\[
\hat{X}(i+1) - X^* = P_{\tilde{S}}(\hat{X}(i+1) - X^*),
\]

and,

\[
V(i) - X^* = P_{\tilde{S}}(V(i) - X^*).
\]

Then, (68) can be written as:

\[
\|\hat{X}(i+1) - X^*\|_F^2 \leq 2 \|P_{\tilde{S}}(\hat{X}(i+1) - X^*)\|_F^2 \leq 2 \|P_{\tilde{S}}(V(i) - X^*)\|_F^2.
\]

(68)

Furthermore, replacing \(\|P_S(i) (V(i) - V(i))\|_F\) with its upper bound defined in (66), we get:

\[
\|\hat{X}(i+1) - X^*\|_F^2 \leq \left(1 + 3\delta_{2k}\right) \left(\|V(i) - X^*\|_F^2 + \|P_S(i) (V(i) - X^*)\|_F^2\right) + 2 \delta_{3k} \|V(i) - X^*\|_F \|V(i) - X(i+1)\|_F + \sqrt{1 + \delta_{2k}} \|P_S(i) (V(i) - X^*)\|_F \|\epsilon\|_2^2.
\]

(65)

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

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

\[
\|\hat{X}(i+1) - X^*\|_F \leq \left[1 + 3\delta_{2k}\right] \|P_S(i) (V(i) - X^*)\|_F + \sqrt{1 + \delta_{2k}} \|\epsilon\|_2.
\]

Since \(V(i) \in \text{span}(S_i)\), we observe \(P_S(i) (V(i) - X^*) = -P_{S(i) \perp (\text{span}(S_i) \cup X^*)} X^*\). Then, using Lemma 6, we obtain:

\[
\|\hat{X}(i+1) - X^*\|_F \leq \left(2\delta_{2k} + 2\delta_{3k}\right) \sqrt{\left(1 + \frac{2\delta_{2k}}{1 - \delta_{3k}}\right) \|X^* - X(i)\|_F} + \sqrt{2\left(1 + \delta_{2k}\right) \|\epsilon\|_2}.
\]

(67)

Given \(\delta_{2k} \leq \delta_{3k}\), \(\rho\) is upper bounded by \(\rho \leq 4\delta_{3k} \sqrt{\frac{1 + 2\delta_{2k}}{1 - \delta_{3k}}} < 1\) \(\Rightarrow \delta_{3k} < 0.2267\).
Combining (72) and (73), we get:

\[ g(i + 1) \leq \left[ b_1 \left( \frac{\alpha(1 + \tau_i) + \sqrt{\Delta}}{2} \right)^{i+1} + b_2 \left( \frac{\alpha(1 + \tau_i) - \sqrt{\Delta}}{2} \right)^{i+1} \right] \|X(0) - X^*\|_F \]

\[ \leq \left( b_1 + b_2 \right) \left( \frac{\alpha(1 + \tau_i) + \sqrt{\Delta}}{2} \right)^{i+1} \|X(0) - X^*\|_F \]  

(69)

Furthermore:

\[ \|Q_i - X^*\|_F = \|X(i) + \tau_i(X(i) - X(i - 1))\|_F \]

\[ = \left\| (1 + \tau_i)\left[ X(i) - X^* \right] + \tau_i(X^* - X(i - 1)) \right\|_F \]

\[ \leq (1 + \tau_i)\|X(i) - X^*\|_F + \tau_i\|X(i - 1) - X^*\|_F \]  

(72)

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

\[ \|X(i + 1) - X^*\|_F \]

\[ \leq \left( 1 + \tau_i \right) \left( \frac{4\delta_{3k}}{1 - \delta_{3k}} + (2\delta_{3k} + 2\delta_{4k}) \frac{2\delta_{3k}}{1 - \delta_{3k}} \right) \|X(i) - X^*\|_F \]

\[ + \tau_i \left( \frac{4\delta_{3k}}{1 - \delta_{3k}} + (2\delta_{3k} + 2\delta_{4k}) \frac{2\delta_{3k}}{1 - \delta_{3k}} \right) \|X(i - 1) - X^*\|_F \]  

(74)

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

\[ g(i + 1) = \alpha(1 + \tau_i)g(i) + \alpha \tau_i g(i - 1) \leq 0 \]  

(75)

Using the method of characteristic roots to solve the above recurrence, we assume that the homogeneous linear recursion has solution of the form \( g(r) = r^i \) 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:

\[ r^2 - \alpha(1 + \tau_i)r - \alpha \tau_i \leq 0 \]  

(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:

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

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

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

A.5 Proof of Lemma 7

Let \( D_i \leftarrow \frac{\partial}{\partial x} \left( \mathcal{F}_i(X(i)) \right) \) and \( D_k \leftarrow \mathcal{A} \mathcal{F}_i \). Using Definition 4, the following holds true:

\[ \|\mathcal{P}_{D_i} \nabla f(X(i)) - \nabla f(X(i))\|_F^2 \]

\[ \leq (1 + \epsilon) \|\mathcal{P}_{D_i} \nabla f(X(i)) - \nabla f(X(i))\|_F^2. \]  

(77)

Furthermore, we observe:

\[ \|\nabla f(X(i))\|_F^2 = \|\nabla f(X(i))\|_F^2 \iff \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F^2 = \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F^2 + \|\nabla f(X(i))\|_F^2 \]

(78)

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

\[ \|\mathcal{P}_{D_i} \nabla f(X(i)) - \nabla f(X(i))\|_F^2 \leq \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F^2 \]

\[ \leq \|\mathcal{P}_{D_i} \nabla f(X(i)) - \nabla f(X(i))\|_F^2 \]

(79)

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

\[ \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F^2 \leq (1 + \epsilon) \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F^2 \]

\[ \leq (1 + \epsilon) \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F^2 \]  

(80)

Now, in (78), we compute the series of inequalities in (81)-(82). Focusing on \( \|\mathcal{P}_{D_i} \nabla f(X(i))\|_F \), we observe:

\[ \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

\[ \leq \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

\[ + \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

\[ \leq \|\mathcal{A}^* \mathcal{A} X^* - X(i)\|_F + \|\mathcal{A}^* \mathcal{A} X^* - X(i)\|_F \]

(83)

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

\[ \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

\[ \leq 2\delta_{3k} \|X(i) - X^*\|_F + \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

(84)

and

\[ \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

\[ \geq \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}(\mathcal{Y} - \mathcal{AX}(i))\|_F \]

\[ - 2\delta_{2k} \|X(i) - X^*\|_F \]

(85)

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

\[ \|\mathcal{P}_{\mathcal{A}^* X \setminus \mathcal{A} X}\|_F \]

\[ \leq 2\delta_{2k} + 2\delta_{3k} \|X(i) - X^*\|_F + \sqrt{2(1 + \delta_{2k})}\|\mathcal{E}\|_F \]

+ 2\lambda\sqrt{7}
\[
\begin{align*}
\|P_{D_i'} \nabla f(X(i))\|_F^2 + \|P_{(D_i')^c} \nabla f(X(i))\|_F^2 &= \|P_{X^+ \cup D_i'} \nabla f(X(i))\|_F^2 + \|P_{(X^+ \cup D_i')} \nabla f(X(i))\|_F^2 \\
\|P_{D_i'} \nabla f(X(i))\|_F^2 + (1 + \epsilon) \|P_{X^+ \cup D_i'} \nabla f(X(i))\|_F^2 &\geq \|P_{X^+ \cup D_i'} \nabla f(X(i))\|_F^2 + \|P_{(X^+ \cup D_i')} \nabla f(X(i))\|_F^2 \quad \text{(81)}
\end{align*}
\]

We exploit Lemma 8 to obtain the following inequalities:

\[
\begin{align*}
\|\tilde{W}_i - X^*\|_F^2 &= \|\tilde{W}_i - V(i) + V(i) - X^*\|_F^2 \\
&\leq \|\tilde{W}_i - V(i)\|_F + \|V(i) - X^*\|_F \\
&\leq (1 + \epsilon) \|W(i) - V(i)\|_F + \|V(i) - X^*\|_F \\
&\leq (2 + \epsilon) \|V(i) - X^*\|_F \quad \text{(90)}
\end{align*}
\]

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

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

\[
\|X(i + 1) - X^*\|_F \leq (1 + \frac{\delta_k}{1 - \delta_k}) \|\tilde{W}_i - X^*\|_F \\
+ \sqrt{1 + \frac{\delta_k}{\delta - \delta_k}} \|e\|_2 \quad \text{(91)}
\]

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

### References

1. R.G. Baraniuk, V. Cevher, and M.B. Wakin. Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective. *Proceedings of the IEEE*, 98(6):959–971, 2010.
2. E.J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):771–772, 2009.
3. R. Meka, P. Jain, and I. S. Dhillon. Guaranteed rank minimization via singular value projection. In *NIPS Workshop on Discrete Optimization in Machine Learning*, 2010.
4. H. Tyagi and V. Cevher. Learning ridge functions with randomized sampling in high dimensions. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 2025–2028. IEEE, 2012.
5. H. Tyagi and V. Cevher. Learning non-parametric basis independent models from point queries via low-rank methods. *Technical report, EPFL*, 2012.
6. Y.K. Liu. Universal low-rank matrix recovery from pauli measurements. 2011.
23. A. Beck and M. Teboulle. A linearly convergent algorithm for

22. D. Goldfarb and S. Ma. Convergence of fixed-point continuation

26. N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure

20. B.K. Natarajan. Sparse approximate solutions to linear systems.  

32. D. Needell and J.A. Tropp. CoSaMP: Iterative signal recovery from

31. V . Cevher. An alps view of sparse recovery. In

30. J. A. Tropp. Greed is good: Algorithmic results for sparse approx-

29. A. Cohen, W. Dahmen, and R. DeVore. Compressed sensing and

27. D. Bertsekas.  

19. J. Wright L. Wu M. Chen Z. Lin, A. Ganesh and Y . Ma. Fast

18. Z. Lin, M. Chen, and Y . Ma. The augmented lagrange multiplier 

16. Jian-Feng Cai, Emmanuel J. Cand`es, and Zuowei Shen. A singular 

15. K. Mohan and M. Fazel. Reweighted nuclear norm minimization 

13. M. Fazel, B. Recht, and P. A. Parrilo. Guaranteed minimum rank 

10. E.J. Candes, X. Li, Y . Ma, and J. Wright. Robust principal com- 

9. J. Bennett and S. Lanning. The netflix prize. In KDD Cup and 

7. H. Tyagi and V. Cevher. Active learning of multi-index function models. In Advances in Neural Information Processing Systems 25, pages 1475–1483, 2012.

6. E.J. Candes and X. Li. Solving quadratic equations via phaselift when there are about as many equations as unknowns. arXiv preprint arXiv:1208.6247, 2012.

5. J. Bennett and S. Lanning. The Netflix prize. In *In KDD Cup and Workshop in conjunction with KDD*, 2007.

4. E.J. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3), 2011.

3. M. Fazel, B. Recht, and P. A. Parrilo. Guaranteed minimum rank solutions to linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

2. A. C. Sankaranarayanan, and R.G. Baraniuk. Sparsc: Recovering low-rank and sparse matrices from compressive measurements. In *NIPS*, 2011.

1. A. Kyrillidis and V. Cevher. Matrix alps: Accelerated low rank and sparse matrix reconstruction. *Technical report*, EPFL, 2012.

35. T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *Appl. Comp. Harm. Anal.*, 27(3):265–274, 2009.

34. S. Foucart. Hard thresholding pursuit: an algorithm for compressed sensing. *SIAM Journal on Numerical Analysis*, 49(6):2543–2563, 2011.

33. T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *IEEE Trans. on Information Theory*, 55(2):2230–2249, May 2009.