Randomized Low-Memory Singular Value Projection

Stephen Becker* Volkan Cevher† Anastasios Kyrillidis†‡

May 17, 2013

Abstract

Affine rank minimization algorithms typically rely on calculating the gradient of a data error followed by a singular value decomposition at every iteration. Because these two steps are expensive, heuristic approximations are often used to reduce computational burden. To this end, we propose a recovery scheme that merges the two steps with randomized approximations, and as a result, operates on space proportional to the degrees of freedom in the problem. We theoretically establish the estimation guarantees of the algorithm as a function of approximation tolerance. While the theoretical approximation requirements are overly pessimistic, we demonstrate that in practice the algorithm performs well on the quantum tomography recovery problem.

1 Introduction

In many signal processing and machine learning applications, we are given a set of observations $\mathbf{y} \in \mathbb{R}^p$ of a rank-r matrix $\mathbf{X}^\star \in \mathbb{R}^{m\times n}$ as $\mathbf{y} = \boldsymbol{\mathcal{A}}\mathbf{X}^\star + \boldsymbol{\varepsilon}$ via the linear operator $\boldsymbol{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$, where $r \ll \min\{m, n\}$ and $\boldsymbol{\varepsilon} \in \mathbb{R}^p$ is additive noise. As a result, we are interested in the solution of

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

where $f(\mathbf{X}) := \|\mathbf{y} - \boldsymbol{\mathcal{A}}\mathbf{X}\|_2^2$ is the data error. While the optimization problem in (1) is non-convex, it is possible to obtain robust recovery with provable guarantees via iterative greedy algorithms (SVP) [MJD10, KC12] or convex relaxations [RFP10, CR09] from measurements as few as $p = \mathcal{O}(r(m + n - r))$.

Currently, there is a great interest in designing algorithms to handle large scale versions of (1) and its variants. As a concrete example, consider quantum tomography (QT), where we need to recover low-rank density matrices from dimensionality reducing Pauli measurements [FGLE12]. In this problem, the size of these density matrices grows exponentially with the number of quantum bits. Other collaborative filtering problems, such as the Netflix challenge, also require huge dimensional optimization. Without careful implementations or non-conventional algorithmic designs, existing algorithms quickly run into time and memory bottlenecks.

These computational difficulties typically revolve around two critical issues. First, virtually all recovery algorithms require calculating the gradient $\nabla f(\mathbf{X}) = 2\boldsymbol{\mathcal{A}}^*(\boldsymbol{\mathcal{A}}(\mathbf{X}) - \mathbf{y})$ at an intermediate iterate $\mathbf{X}$, where $\boldsymbol{\mathcal{A}}^*$ is the adjoint of $\boldsymbol{\mathcal{A}}$. When the range of $\boldsymbol{\mathcal{A}}^*$ contains dense matrices, this forces algorithms to use memory proportional to $\mathcal{O}(mn)$. Second, after the iterate is updated with the gradient, projecting onto the low-rank space requires a partial singular value decomposition (SVD). This is usually problematic for the initial iterations of convex algorithms, where they may have to perform full SVD's. In contrast, greedy algorithms [KC12] fend off the complexity of full SVD's, since they need fixed rank projections, which can be approximated via Lanczos or randomized SVD's [HMT11].

Algorithms that avoid these two issues do exist, such as [WYZ10, RR13, LRS+11, Lau12], and are typically based on the Burer-Monteiro splitting [BM03]. The main idea in Burer-Monteiro splitting is to remove the non-convex rank constraint by directly embedding it into the objective: as opposed to optimizing $\mathbf{X}$, splitting algorithms directly work with its fixed factors $\mathbf{U}\mathbf{V}^T = \mathbf{X}$ in an alternating fashion, where $\mathbf{U} \in \mathbb{R}^{m\times\hat{r}}$ and $\mathbf{V} \in \mathbb{R}^{n\times\hat{r}}$ for some

*stephen.becker@upmc.fr, Laboratoire JLL, UPMC Paris 6, Paris

†{volkan.cevher, anastasios.kyrillidis}@epfl.ch, LIONS, École polytechnique Fédérale de Lausanne

‡Authors are listed in alphabetical order

$\hat{r} \geq r$. Unfortunately, rigorous guarantees are difficult.[1] The work [JNS12] has shown approximation guarantees if $\boldsymbol{\mathcal{A}}$ satisfies a restricted isometry property with constant $\delta_{2r} \leq \kappa^2/(100r)$ (in the noiseless case), where $\kappa = \sigma_1(\mathbf{X}^\star)/\sigma_r(\mathbf{X}^\star)$, or $\delta_{2r} \leq 1/(3200r^2)$ for a bound independent of κ. The authors suggest that these bounds may be tightened, and that practical performance is better than the bound suggests.

In this paper, we merge the gradient calculation and the singular value projection steps into one and show that this not only removes a huge computational burden, but suffers only a minor convergence speed drawback in practice. Our contribution is a natural but non-trivial fusion of the Singular Value Projection (SVP) algorithm in [MJD10] and the approximate projection ideas in [KC12]. The SVP algorithm is an iterative hard-thresholding algorithm that has been considered in [MJD10, GM11]. Inexact steps in SVP have been considered as a heuristic [GM11] but have not been incorporated into an overall convergence result.[2] A non-convex framework for affine rank minimization (including variants of the SVP algorithm) that utilizes inexact projection operations with provable signal approximation and convergence guarantees is proposed in [KC12]. Neither [MJD10,KC12] considers splitting techniques in the proposed schemes.

This work, departing from [MJD10, KC12], engineers the SVP algorithm to operate like splitting algorithms that *directly work with the factors*; this added twist decreases the per iteration requirements in terms of storage and computational complexity. Using this new formulation, each iteration is nearly as fast as in the splitting method, hence removing a drawback to SVP in relation to splitting methods. Furthermore, we prove that, under some conditions, it is still possible to obtain perfect recovery even if the projections are inexact. In particular, our assumption is that the linear map $\boldsymbol{\mathcal{A}}$ satisfies the rank restricted isometry property, and in section 5.1 we give an application that satisfies this assumption, allowing perfect recovery (in the noiseless case) or stable recovery (in the presence of noise) from measurements $p \ll mn$. This approach has been used for convex [RFP10] and non-convex [MJD10, KC12] algorithms to obtain approximation guarantees.

2 Preliminary material

Notation: we write $\mathcal{P}_\Omega$ to be an orthogonal projection onto the closed set Ω when it exists. For shorthand we write $\mathcal{P}_r$ to mean $\mathcal{P}_{\{\mathbf{X}:\mathrm{rank}(\mathbf{X})\leq r\}}$ (which does exist by the Eckart-Young theorem). Computer routine names are typeset with a `typewriter font`.

2.1 R-RIP

The Rank Restricted Isometry Property (R-RIP) is a common tool used in matrix recovery [RFP10, MJD10, KC12]:

Definition 1 (R-RIP for linear operators on matrices [RFP10])**.** *A linear operator $\boldsymbol{\mathcal{A}} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ satisfies the R-RIP with constant $\delta_r(\boldsymbol{\mathcal{A}}) \in (0,1)$ if, $\forall \mathbf{X} \in \mathbb{R}^{m\times n}$ with $\mathrm{rank}(\mathbf{X}) \leq r$,*

$$(1-\delta_r(\boldsymbol{\mathcal{A}}))\big\|\mathbf{X}\big\|_F^2 \leq \big\|\boldsymbol{\mathcal{A}}\mathbf{X}\big\|_2^2 \leq (1+\delta_r(\boldsymbol{\mathcal{A}}))\big\|\mathbf{X}\big\|_F^2, \tag{2}$$

We write δ_r to mean $\delta_r(\boldsymbol{\mathcal{A}})$.

2.2 Additional convex constraints

Consider the variant

$$\begin{array}{ll} \underset{\mathbf{X}\in\mathbb{R}^{m\times n}}{\text{minimize}} & f(\mathbf{X}) \\ \text{subject to} & \mathrm{rank}(\mathbf{X}) \leq r,\ \mathbf{X} \in \mathcal{C}, \end{array} \tag{3}$$

for a convex set $\mathcal{C}$. Our main interests are $\mathcal{C}_+ = \{\mathbf{X} : \mathbf{X} \succeq 0\}$ and the matrix simplex $\mathcal{C}_\Delta = \{\mathbf{X} : \mathbf{X} \succeq 0,\ \mathrm{trace}(\mathbf{X}) = 1\}$. In both cases the constraints are unitarily invariant and the projection onto these sets can be done by taking

[1] If $\hat{r} \gtrsim \sqrt{p}$, then [BM03] shows their method obtains a global solution, but this is impractical for large p. Moreover, it is shown that the explicit rank $\hat{r}$ splitting method solves a non-convex problem that has the same local minima as (1) (if $\hat{r} = r$). However, the non-convex problems are not *equivalent* (e.g. $\mathbf{U} = \mathbf{0}$, $\mathbf{V} = \mathbf{0}$ is a stationary point for the splitting problem whereas $\mathbf{X} = \mathbf{0}$ is generally not a stationary point for (1)). Furthermore, recovery bounds for non-convex algorithms, as in [GK09] and the present paper, are statements about a sequence of iterates of the algorithm, and say nothing about the local minima.

[2] Inexact steps are often incorporated into analysis of algorithms for convex problems. Of particular note, [Lau12] allows inexact eigenvalue computations in a modified Frank-Wolfe algorithm that has applications to (1).

Algorithm 1 `RandomizedSVD`

Finds Q such that $X \approx \mathcal{P}_Q X$ where $\mathcal{P}_Q = QQ^*$.

Require: Function $\mathtt{h} : \widetilde{Z} \mapsto X\widetilde{Z}$
Require: Function $\mathtt{h}^* : \widetilde{Q} \mapsto X^*\widetilde{Q}$
Require: $r \in \mathbb{N}$ // Rank of output
Require: $q \in \mathbb{N}$ // Number of power iterations to perform
$\ell = r + \rho$ // Typical value of ρ is 5
Ω a $n \times \ell$ standard Gaussian matrix
$W \leftarrow \mathtt{h}(\Omega)$
$Q \leftarrow \mathtt{QR}(W)$ // The QR algorithm to orthogonalize W
for $j = 1, 2, \ldots, q$ **do**
  $Z \leftarrow \mathtt{QR}(\mathtt{h}^*(Q))$
  $Q \leftarrow \mathtt{QR}(\mathtt{h}(Z))$
end for
$Z \leftarrow \mathtt{h}^*(Q)$
$(U, \Sigma, V) \leftarrow \mathtt{FactoredSVD}(Q, I_\ell, Z)$ // $\widetilde{\mathbf{X}}_{i+1} = U\Sigma V^*$ in the appendix
Let Σ_r be the best rank r approximation of Σ
return (U, Σ_r, V) // $\mathbf{X}_{i+1} = U\Sigma_r V^*$ in the appendix

Algorithm 2 $\mathtt{FactoredSVD}(\widetilde{U}, \widetilde{D}, \widetilde{V})$

Computes the SVD $U\Sigma V^*$ of the matrix X implicitly given by $X = \widetilde{U}\widetilde{D}\widetilde{V}^*$

$(U, R_U) \leftarrow \mathtt{QR}(\widetilde{U})$
$(V, R_V) \leftarrow \mathtt{QR}(\widetilde{V})$
$(u, \Sigma, v) \leftarrow \mathtt{DenseSVD}(R_U \widetilde{D} R_V^*)$
return $(U, \Sigma, V) \leftarrow (Uu, \Sigma, Vv)$

the eigenvalue decomposition and projecting the eigenvalues. Furthermore, for these specific $\mathcal{C}$, $\mathcal{P}_{\{\mathbf{X}:\mathrm{rank}(\mathbf{X}) \le r\} \cap \mathcal{C}} = \mathcal{P}_{\mathcal{C}} \circ \mathcal{P}_r$ (this is not obvious; see [BCKK13]).[3]

In general, any convex set $\mathcal{C}$ satisfying the above property is compatible with our algorithm, as long as $\mathbf{X}^\star \in \mathcal{C}$. We overload notation to use $\mathcal{P}_{\mathcal{C}}$ to denote both the projection of $\mathbf{X}$ onto the set as well as the projection of its eigenvalues onto the analogous set.

2.3 Approximate singular value computations

The standard method to compute a partial SVD is the Lanczos method. By itself it is not numerically stable and requires re-orthogonalization and implicit restarts. Excellent implementations are available, but it is a sequential algorithm that calls matrix-vector products. This makes it more difficult to parallelize, which is an issue on modern multi-processor computers. The matrix-vector multiplies are also slower than grouping into matrix-matrix multiplies since it is harder to predict memory usage and this will lead to cache misses; it also precludes the use of theoretically faster algorithms such as Strassen's. Theoretically, there are no known relative error bounds in norm (à la Theorem 1).

As an alternative, we turn to randomized linear algebra. On this front, we restrict ourselves to algorithms that require only multiplications, as opposed to sub-sampling entries/rows/columns, as sub-sampling is not efficient for the application we present. The randomized approach presented in Algorithm 1 has been rediscovered many times, but has seen a recent resurgence of interest due to theoretical analysis [HMT11]:

Theorem 1 (Average Frobenius error)**.** *Suppose $\mathbf{X} \in \mathbb{R}^{m \times n}$, and choose a target rank r and oversampling parameter $\rho \ge 2$ where $\ell := r + \rho \le \min\{m, n\}$. Calculate Q and $\mathcal{P}_Q$ via `RandomizedSVD` using $q = 0$ and set $\widetilde{\mathbf{X}} = \mathcal{P}_Q \mathbf{X}$ (which is rank ℓ). Then*

$$\mathbb{E}\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F^2 \le (1 + \epsilon)\, \|\mathbf{X} - \mathbf{X}_r\|_F^2$$

[3] This formula is literally true for $\mathcal{C}_+$ and $\{\mathbf{X} : \mathbf{X} \succeq 0,\ \mathrm{trace}(\mathbf{X}) \le 1\}$. For $\mathcal{C} = \{\mathbf{X} : \mathbf{X} \succeq 0,\ \mathrm{trace}(\mathbf{X}) = 1\}$ constraints, $\mathcal{P}_{\mathcal{C}}$ can increase the rank, so formally we must work on a restricted subspace and then embed back in the larger space, but this poses no theoretical issues.

Algorithm 3 Efficient implementation of SVP, $\mathcal{K} = \{\mathbb{R}, \mathbb{C}\}$

Require: step-size $\mu > 0$, measurements $\mathbf{y}$, initial points $u_0 \in \mathcal{K}^{m\times r},\ v_0 \in \mathcal{K}^{n\times r},\ d_0 \in \mathcal{K}^r$
Require: (optional) unitarily invariant convex set $\mathcal{C}$
Require: Function $\mathtt{A} : (u, d, v) \mapsto \boldsymbol{\mathcal{A}}(u\,\mathrm{diag}(d)v^*)$
Require: Function $\mathtt{At} : (\mathbf{z}, w) \mapsto \boldsymbol{\mathcal{A}}^*(\mathbf{z})w$
Require: Function $\mathtt{At}^* : (\mathbf{z}, w) \mapsto (\boldsymbol{\mathcal{A}}^*(\mathbf{z}))^* w$

$v_{-1} \leftarrow 0$, $u_{-1} \leftarrow 0$, $d_{-1} \leftarrow 0$
for $i = 0, 1, \ldots$ **do**
  Compute β_i // See text
  $u_y \leftarrow [u_i, u_{i-1}]$, $v_y \leftarrow [v_i, v_{i-1}]$
  $d_y \leftarrow [(1 + \beta_i)d_i, -\beta_i d_{i-1}]$
  $\mathbf{z} \leftarrow A(u_y, d_y, v_y) - \mathbf{y}$ // Compute the residual
  Define the functions
  $\mathtt{h} : w \mapsto u_y\,\mathrm{diag}(d_y)v_y^* w - \mu\mathtt{At}(\mathbf{z}, w)$
  $\mathtt{h}^* : w \mapsto v_y\,\mathrm{diag}(d_y)u_y^* w - \mu\mathtt{At}^*(\mathbf{z}, w)$
  $(u_{i+1}, d_{i+1}, v_{i+1}) \leftarrow \mathtt{RandomizedSVD}(\mathtt{h}, \mathtt{h}^*, r)$ or $(u_{i+1}, d_{i+1}, u_{i+1}) \leftarrow \mathtt{RandomizedEIG}(\mathtt{h}, \mathtt{h}^*, r)$
  $d_{i+1} \leftarrow \mathcal{P}_{\mathcal{C}}(d_{i+1})$ // Optional
end for
return $X \leftarrow u_i d_i v_i^*$ // If desired

where $\mathbf{X}_r$ is the best rank r approximation in the Frobenius norm of $\mathbf{X}$ and $\epsilon = \frac{r}{\rho - 1}$.

The theorem follows from the proof of Thm. 10.5 in [HMT11] (note that Thm. 10.5 is stated in terms of $\mathbb{E}\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F$ which is not the same as $\sqrt{\mathbb{E}\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F^2}$). The expectation is with respect to the Gaussian r.v. in `RandomizedSVD`. For the sake of our analysis, we cannot immediately truncate $\widetilde{\mathbf{X}}$ to rank r since then the error bound in [HMT11] is not tight enough. Thus, since $\widetilde{X}$ is rank ℓ, in practice we even observe that $\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F^2 < \|\mathbf{X} - \mathbf{X}_r\|_F^2$, especially for small r, as shown in Figure 3. The figure also shows that using $q > 0$ power iterations is extremely helpful, though this is not taken into account in our analysis since there are no useful theoretical bounds (in the Frobenius norm). Note that variants for eigenvalues also exist; we refer to the equivalent of `RandomizedSVD` as `RandomizedEIG`, which has the property that $U = V$ and Σ need not be positive (cf., [HMT11, **?**])

3 Algorithm

3.1 Projected gradient descent

Our approach is based on the projected gradient descent algorithm:

$$\mathbf{X}_{i+1} = \mathcal{P}_r^{\epsilon}(\mathbf{X}_{i+1} - \mu_i \nabla f(\mathbf{X}_i)), \tag{4}$$

where $\mathbf{X}_i$ is the i-th iterate, $\nabla f(\cdot)$ is the gradient of the loss function, μ_i is a step-size, and $\mathcal{P}_r^{\epsilon}(\cdot)$ is the approximate projector onto rank r matrices given by `RandomizedSVD`. If we include a convex constraint $\mathcal{C}$, then the iteration is

$$\mathbf{X}_{i+1} = \mathcal{P}_{\mathcal{C}}(\mathcal{P}_r^{\epsilon}(\mathbf{X}_{i+1} - \mu_i \nabla f(\mathbf{X}_i))). \tag{5}$$

In practice, Nesterov acceleration improves performance:

$$\mathbf{Y}_{i+1} = (1 + \beta_i)\mathbf{X}_i - \beta_i \mathbf{X}_{i-1} \tag{6}$$
$$\mathbf{X}_{i+1} = \mathcal{P}(\mathbf{Y}_i - \mu_i \nabla f(\mathbf{Y}_i)), \tag{7}$$

where β_i is chosen $\beta_i = (\alpha_{i-1} - 1)/\alpha_i$ and $\alpha_0 = 1$, $2\alpha_{i+1} = 1 + \sqrt{4\alpha_i^2 + 1}$ [Nes83] (see [KC12]). Theorem 2 holds for a stepsize μ_i based on the RIP constant, which is unknown. In practice, the algorithm consistently converges as long as $\mu_i \lesssim \frac{2}{\|\boldsymbol{\mathcal{A}}\|^2}$.

Algorithm 3 shows implementation details that are important for keeping low-memory requirements. The implementation of maps like `A` and `At` depends on the structure of $\boldsymbol{\mathcal{A}}$; see section 5.1 for explicit examples.

4 Convergence

We assume the observations are generated by $\mathbf{y} = \boldsymbol{\mathcal{A}}\mathbf{X}^\star + \boldsymbol{\varepsilon}$ where $\boldsymbol{\varepsilon}$ is a noise term, not to be confused with the approximation error ϵ. In the following theorem, we will assume that $\|\boldsymbol{\mathcal{A}}\|^2 \leq mn/p$, which is true for the quantum tomography example [Liu11]; if $\boldsymbol{\mathcal{A}}$ is a normalized Gaussian, then this assumption holds in expectation.

Theorem 2. *(Iteration invariant) Pick an accuracy $\epsilon = \frac{r}{\rho - 1}$, where ρ is defined as in Theorem 1. Define $\ell = r + \rho$ and let c be an integer such that $\ell = (c-1)r$. Let $\mu_i = \frac{1}{2(1+\delta_{cr})}$ in (4) and assume $\|\boldsymbol{\mathcal{A}}\|^2 \leq mn/p$ and $f(\mathbf{X}_i) > C^2\|\boldsymbol{\varepsilon}\|^2$, where $C \geq 4$ is a constant. Then the descent scheme (4) or (5) has the following iteration invariant*

$$\mathbb{E}f(\mathbf{X}_{i+1}) \leq \theta f(\mathbf{X}_i) + \tau\|\boldsymbol{\varepsilon}\|^2, \tag{8}$$

in expectation, where

$$\theta \leq 12 \cdot \frac{1+\delta_{2r}}{1-\delta_{cr}} \cdot \left(\frac{\epsilon}{1+\delta_{cr}} \cdot \frac{mn}{p} + (1+\epsilon)\frac{3\delta_{cr}}{1-\delta_{2r}} \right),$$

and

$$\tau \leq \frac{1+\delta_{2r}}{1-\delta_{cr}} \cdot \left(12 \cdot (1+\epsilon)\left(1 + \frac{2\delta_{cr}}{1-\delta_{2r}}\right) + 8 \right).$$

The expectation is taken with respect to Gaussian random designs in `RandomizedSVD`*. If $\theta \leq \theta_\infty < 1$ for all iterations, then $\lim_{i\to\infty} \mathbb{E}f(\mathbf{X}_i) \leq \max\{C^2, \frac{\tau}{1-\theta_\infty}\}\|\boldsymbol{\varepsilon}\|^2$.*

Each call to `RandomizedSVD` draws a new Gaussian r.v., so the expected value does not depend on previous iterations. By Corollary 3.4 in [NT09], $\delta_{cr} \leq c \cdot \delta_{2r}$, which allows us to put θ and τ in terms of δ_{2r} if desired, at a slight expense in sharpness.

The expected value of the function converges linearly at rate θ to within a constant of the noise level, and in particular, it converges to zero when there is no noise since C and τ are finite. Note that convergence of the iterates follows from convergence of the function f:

Corollary 1. *If $f(\mathbf{X}_i) \leq \gamma$, then $\|\mathbf{X}_i - \mathbf{X}^\star\|_F^2 \leq \frac{(\sqrt{\gamma} + \|\boldsymbol{\varepsilon}\|_2)^2}{1-\delta_{2r}}$.*

Proof. By the R-RIP and the triangle inequality,

$$\begin{aligned}
\sqrt{1+\delta_{2r}(\boldsymbol{\mathcal{A}})}\|\mathbf{X}_i - \mathbf{X}^\star\|_F &\leq \|\boldsymbol{\mathcal{A}}(\mathbf{X}_i) - \boldsymbol{\mathcal{A}}(\mathbf{X}^\star)\|_2 \\
&= \|(\boldsymbol{\mathcal{A}}(\mathbf{X}_i) - \mathbf{y}) - (\boldsymbol{\mathcal{A}}(\mathbf{X}^\star) - \mathbf{y})\|_2 \\
&\leq \|(\boldsymbol{\mathcal{A}}(\mathbf{X}_i) - \mathbf{y})\|_2 + \|\boldsymbol{\varepsilon}\|_2 \\
&\leq \sqrt{\gamma} + \|\boldsymbol{\varepsilon}\|_2
\end{aligned}$$

□

Corollary 2 (Exact computation)**.** *If $\epsilon = 0$ and there is no additional convex constraint $\mathcal{C}$, then $\theta = \frac{2\delta_{2r}}{1-\delta_{2r}}(1 + \frac{2}{C})$ and $\tau = 1 + \frac{2\delta_{2r}}{1-\delta_{2r}}$, hence $\theta < 1$ if $\delta_{2r} < \frac{1}{3+4/C}$.*

Corollary 2 shows that without the approximate SVD, the R-RIP constants are quite reasonable. For example, with exact computation and no noise, any value of $\delta_{2r} < 1/3$ implies that $\lim_{i\to\infty} \mathbf{X}_i = \mathbf{X}^\star$. With noise, choosing $C = 4$ gives $\delta_{2r} = 1/5$ and $\theta = 3/4$, $\tau = 3/2$ and thus $\lim_{i\to\infty} f(\mathbf{X}_i) \leq \max\{16, 6\}\|\boldsymbol{\varepsilon}\|^2$.

Note that the theorem gives pessimistic values for ϵ. We want the bound on θ to be less than 1 in order to have a contraction, so we need

$$\underbrace{12 \cdot \frac{1+\delta_{2r}}{1-\delta_{cr}} \cdot \frac{\epsilon}{1+\delta_{cr}} \cdot \frac{mn}{p}}_{\text{I}} + \underbrace{12(1+\epsilon) \cdot \frac{1+\delta_{2r}}{1-\delta_{cr}} \cdot \frac{3\delta_{cr}}{1-\delta_{2r}}}_{\text{II}} < 1$$

For a rough analysis, we will give approximate conditions so that each of the I and II terms is less than 0.5. It is clear that the terms blow up if $\delta_{cr} \to 1$, so we will assume $\delta_{cr} \ll 1$ (and hence $\delta_{2r} \ll 1$). Then setting $1 + \delta_{2r} \approx 1$ in the numerator of I, we require that

$$\frac{12}{1-\delta_{cr}^2} \cdot \frac{\epsilon\, mn}{p} < \frac{1}{2} \tag{9}$$

which means that we need $\epsilon \lesssim \frac{p}{24mn}$. For quantum tomography, $m = n$ and $p = \mathcal{O}(rn)$, so we require $\epsilon \lesssim \mathcal{O}(r/n)$. From Theorem 1, our bound on ϵ is $r/(\rho - 1)$, so we require $\rho \simeq n$, which defeats the purpose of the randomized algorithm (in this case, one would just do a dense SVD). Numerical examples in the next section will show that ρ can be nearly a small constant, so the theory is not sharp.

For the II term, again approximate $1 + \delta_{2r} \approx 1$ and then multiply the denominators and ignore the $\delta_{cr}\delta_{2r}$ term to get

$$72\delta_{cr}(1 + \epsilon) \lesssim 1 - \delta_{2r} - \delta_{cr}. \tag{10}$$

Since certainly $\epsilon \leq 0.5$ and $\delta_{2r} + \delta_{cr} \leq 0.5$, a sufficient condition is $\delta_{cr} < 1/216$, which is reasonable (cf. [JNS12]).

5 Numerical experiments

5.1 Application: quantum tomography

As a concrete example, we apply the algorithm to the quantum tomography problem, which is a particular instance of (1). For details, we refer to [GLF+10,FGLE12]. The salient features are that the variable $\mathbf{X} \in \mathbb{C}^{n\times n}$ is constrained to be Hermitian positive-definite, and that, unlike many low-rank recovery problems, the linear operator $\boldsymbol{\mathcal{A}}$ satisfies the R-RIP: [Liu11] establishes that Pauli measurements (which comprise $\boldsymbol{\mathcal{A}}$) have R-RIP with overwhelming probability when $p = \mathcal{O}(rn \log^6 n)$. In the ideal case, $\mathbf{X}^\star$ is exactly rank 1, but it may have larger rank due to some (non-Gaussian) noise processes, in addition to AWGN $\boldsymbol{\varepsilon}$. Furthermore, it is known that the true solution $\mathbf{X}^\star$ has trace 1, which is also possible to exploit in our algorithmic framework.

Since $\mathbf{X}$ is Hermitian, the u and v terms in the algorithm are identical. Several computations can be simplified and there is a version of Algorithm 1 which exploits the positive-definiteness to incorporate a Nyström approximation (and also forces the approximation to be positive-definite); see [HMT11, **?**]. Here, we focus on showing how the functions `A` and `At` can be computed (due to the complex symmetry, $\texttt{At}^* = \texttt{At}$).

In quantum tomography, the linear operator has the form $(\boldsymbol{\mathcal{A}}(\mathbf{X}))_j = \langle \mathbf{E}_j, \mathbf{X}\rangle$ where $\mathbf{E}_j = \mathbf{E}_j^*$ is the Kronecker product of 2×2 Pauli matrices. There are four possible Pauli matrices $\sigma_{x,y,z}$ if we define σ_I to be the 2×2 identity matrix. For a q_b-qubit system, $\mathbf{E}_j = \sigma_{j1} \otimes \sigma_{j2} \otimes \ldots \otimes \sigma_{jq_b}$. For roughly 12 qubits and fewer, it is simple to calculate $\boldsymbol{\mathcal{A}}(\mathbf{X})$ by explicitly forming $\mathbf{E}_j$ and then creating a sparse matrix $\mathbf{A}$ with the j^{th} row of $\mathbf{A}$ equal to $\text{vec}(\mathbf{E}_j)$ so that $\boldsymbol{\mathcal{A}}(\mathbf{X}) = \mathbf{A}\,\text{vec}(\mathbf{X})$. For larger systems, storing this sparse matrix is impractical since there are $p \geq n$ rows and each row has exactly n non-zero entries, so there are over n^2 entries in $\mathbf{A}$.

To keep memory low, we exploit the Kronecker-product nature of $\mathbf{E}_j$ and store it with only q_b numbers. When $\mathbf{X} = \mathbf{x}\mathbf{x}^*$, we compute $\langle \mathbf{E}_j, \mathbf{X}\rangle = \text{trace}(\mathbf{E}_j\mathbf{x}\mathbf{x}^*) = \text{trace}(\mathbf{x}^*\mathbf{E}_j\mathbf{x})$, and $\mathbf{E}_j\mathbf{x}$ can be computed in $\mathcal{O}(q_b n)$ time. This gives us `A`. The output of `A` is real even when $\mathbf{X}$ is complex.

To compute $\texttt{At}(\mathbf{z}, \mathbf{w})$ when the dimensions are small, we just explicitly form the matrix $\mathbf{M} = \boldsymbol{\mathcal{A}}(\mathbf{z})$ and then multiply $\mathbf{M}\mathbf{w}$. To form $\mathbf{M}$, we use the same sparse matrix $\mathbf{A}$ as above and reshape the n^2 vector $\mathbf{A}^*\mathbf{z}$ into a $n \times n$ matrix. For larger dimensions, when it is impractical to store $\mathbf{A}$, we implicitly represent $\mathbf{M} = \sum_{j=1}^p \mathbf{z}_j\mathbf{E}_j$ and thus $\mathbf{M}\mathbf{w} = \sum_{j=1}^p \mathbf{z}_j\mathbf{E}_j\mathbf{w}$. In general, the output is complex. However, if it is known *a priori* that $\mathbf{X}$ is real-valued, this can be exploited by taking the real part of $\mathbf{M}$. This leads to a considerable time savings ($2\times$ to $4\times$), and all experiments shown below make this assumption.

In our numerical implementation, we code both `A` and `At` in C and parallelize the code since this is the most computationally expensive calculation. Our parallelization implementation uses both `pthreads` on local cores as well as message passing among different computers. There are two approaches to parallelization: divide the indices $j = 1, \ldots, p$ among different cores, or, when $\mathbf{x}$ or $\mathbf{w}$ has several columns, send different columns to the different cores. Both approaches are efficient in terms of message passing since $\boldsymbol{\mathcal{A}}$ is parameterized and static. The latter approach only works when $\mathbf{x}$ or $\mathbf{w}$ has a significant number of columns, and so it does not apply to Lanczos methods that perform only matrix-vector multiplies.

Recording error metrics can be costly if not done correctly. Let $\mathbf{X} = \mathbf{x}\mathbf{x}^*$ and $\mathbf{Y} = \mathbf{y}\mathbf{y}^*$ be rank-r factorizations. For the Frobenius norm error $\|\mathbf{X} - \mathbf{Y}\|_F$ which requires n^2 operations naively, we expand the term and use the cyclic invariance of trace to get $\|\mathbf{X} - \mathbf{Y}\|_F^2 = \text{trace}(\mathbf{x}^*\mathbf{x}\mathbf{x}^*\mathbf{x}) + \text{trace}(\mathbf{y}^*\mathbf{y}\mathbf{y}^*\mathbf{y}) - 2\,\text{trace}(\mathbf{x}^*\mathbf{y}\mathbf{y}^*\mathbf{x})$, which requires only $\mathcal{O}(nr^2)$ flops. In quantum information, another common metric is the trace distance [NC10] $\|\mathbf{X} - \mathbf{Y}\|_*$, where $\|\cdot\|_*$ is the nuclear norm. This calculation requires $\mathcal{O}(n^3)$ flops if calculated directly but can also be calculated cheaply via `FactoredSVD` on $\mathbf{U} = \mathbf{V} = [\mathbf{x}, \mathbf{y}]$ and $\mathbf{D} = [\mathbb{I}, \mathbf{0}; \mathbf{0}, -\mathbb{I}]$. The third common metric is the fidelity [NC10] given by $\|\mathbf{X}^{1/2}\mathbf{Y}^{1/2}\|_*$. If either $\mathbf{X}$ or $\mathbf{Y}$ is rank-1, this can be calculated cheaply as well.

5.2 Results

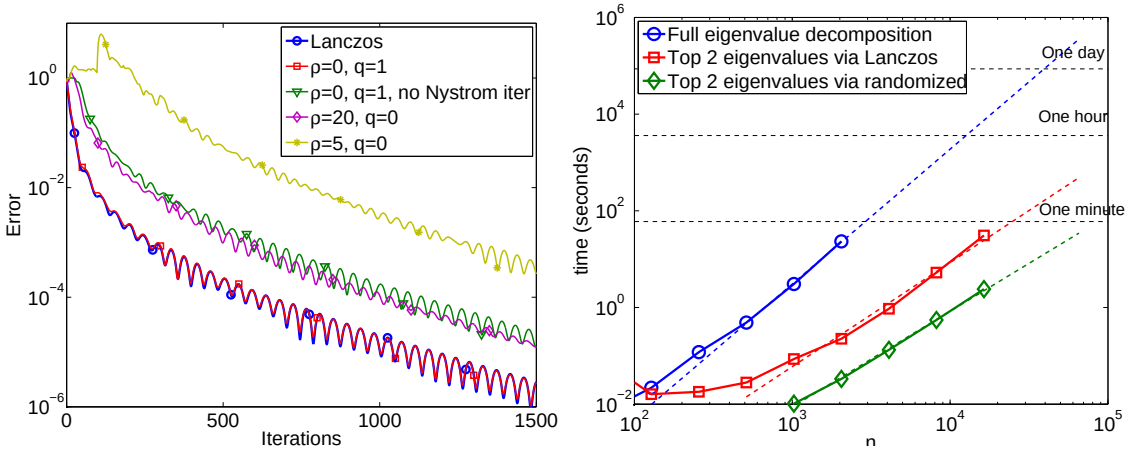

Figure 1: (Left) Convergence rate as a function of parameters to `RandomizedSVD`/`RandomizedEIG`. (Right) Comparison of just eigenvalue computation times via three methods.

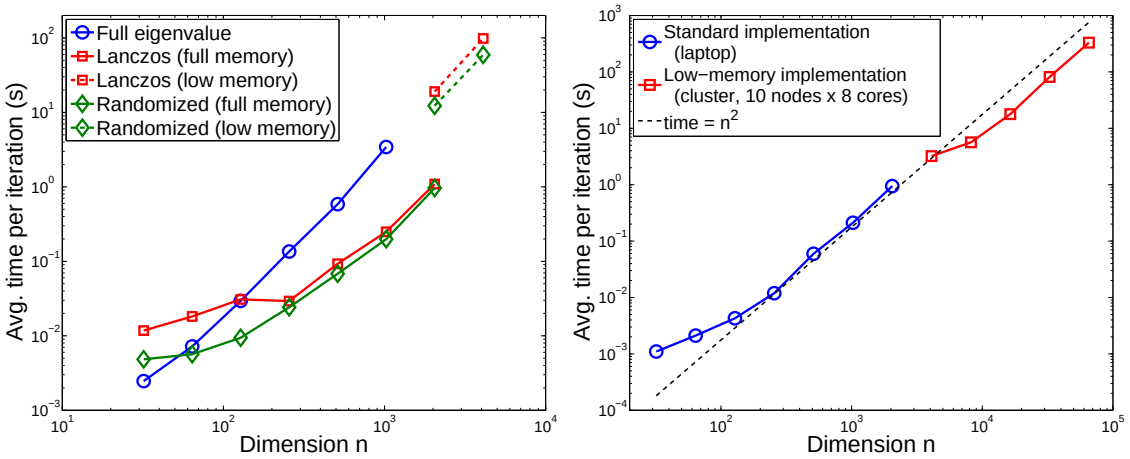

Figure 2: Mean time of 10 iterations: this includes the matrix multiplications as well as eigenvalue computations. (Left) shows times for a complete iteration of our method on a single computer using sparse matrix multiplies ("full memory") and, above 11 qubits, the custom low-memory implementation as well (not multi-threaded) on the same computer. (Right) shows times for just the `RandomizedSVD`/`RandomizedEIG`.

Figure 1 (left) plots convergence and accuracy results for a quantum tomography problem with 8 qubits and $p = 4rn$ with $r = 1$. The SVP algorithm works well on noisy problems but we focus here on a noiseless (and truly low-rank) problem in order to examine the effects of approximate SVD/eigenvalue computations. The figure shows that the power method with $q \geq 1$ is extremely effective even though it lacks theoretical guarantees; without the power method, take $\rho \simeq 20$ and we see convergence, albeit slower. When p is smaller and the R-RIP is not satisfied, taking ρ or q too small can lead to non-convergence.

Figure 1 (right) is a direct comparison of `RandomizedEIG` (with $\rho = 5$ and $q = 3$) and the Lanczos method for multiplies of the type encountered in the algorithm. The `RandomizedEIG` has the same asymptotic complexity but much better constants.

Figure 2 shows that because the eigenvalue decomposition is a significant portion of the computational cost, using `RandomizedEIG` instead of Lanczos makes a difference. The difference is not pronounced in the small-scale full-memory implementation because the variable $\mathbf{X}$ is explicitly formed and matrix multiplies are relatively cheap compared to other operations in the code. For larger dimensions with the low-memory code, $\mathbf{X}$ is never explicitly formed and multiplying with the gradient is quite costly. The randomized method requires fewer multiplies, explaining its benefit. For 12 qubits, the Lanczos method averages 98.4 seconds/iteration, whereas the randomized

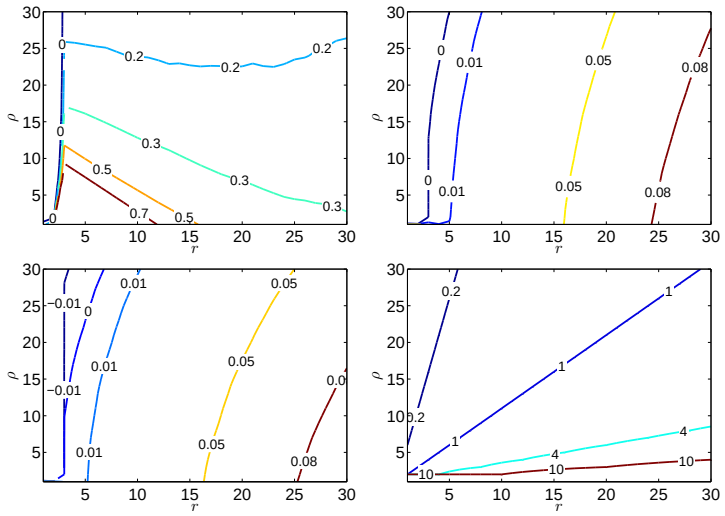

Figure 3: Top row: $\widetilde{\epsilon}$ for (left) $q = 0$ and (right) $q = 1$ power iterations. Bottom row: $\widetilde{\epsilon}$ for $q = 2$ power iterations (left), and (right) shows the bound ϵ.

$\|\mathbf{X} - \mathbf{X}^\star\|_F$	Trace distance $\|\mathbf{X} - \mathbf{X}^\star\|_*$	Fidelity $F(\mathbf{X}, \mathbf{X}^\star)$	$F(\mathbf{X}, \mathbf{X}^\star)^2$
0.0256	0.0363	0.9998	0.9997

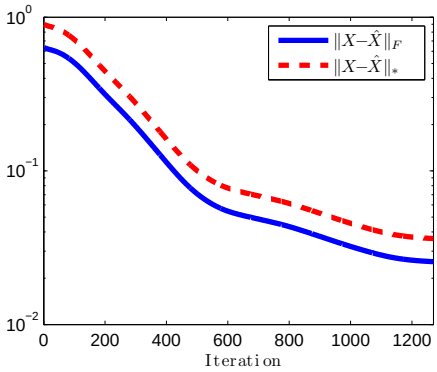

Figure 4: The table (left) shows error metrics for the noisy rank-1 16-qubit recovery. The figure (right) shows the convergence rate for the same simulation.

method averages just 59.2 seconds. The right subfigure shows that the low-memory implementation (which has memory requirement $\mathcal{O}(rn)$) still has only $\mathcal{O}(n^2)$ time complexity per iteration.

Figure 3 tests Theorem 1 by plotting the value of

$$\widetilde{\epsilon} = \|\mathbf{X} - \widetilde{\mathbf{X}}\|_F^2 / \|\mathbf{X} - \mathbf{X}_r\|_F^2 - 1$$

(which is bounded by ϵ) for matrices $\mathbf{X}$ that are generated by the iterates of the algorithm. The algorithm is set for $r = 1$ (so $\mathbf{X}$ is the sum of a rank 2 term, which includes the Nesterov term, and the full rank gradient), but the plots consider a range of r and a range of oversampling parameters ρ. The plots use $q = 0, 1$ (top row, left to right) and $q = 2$ (bottom row, left) power iterations. Because $\widetilde{\mathbf{X}}$ has rank $\ell = r + \rho$, it is possible for $\widetilde{\epsilon} < 0$, as we observe in the plots when r is small and ρ is large. For two power iterations, the error is excellent. In all cases, the observed error $\widetilde{\epsilon}$ is much better than the bound ϵ (shown bottom row, right) from Theorem 1, suggesting that it may be possible to have a more refined analysis.

Finally, to test scaling to very large data, we compute a 16 qubit state ($n = 65536$), using a known quantum state as input, with realistic quantum mechanical perturbations (global depolarizing noise of level $\gamma = 0.01$; see [FGLE12]) as well as AWGN to give a SNR of 30 dB, and $p = 5n = 327680$ measurements. The first iteration uses Lanczos and all subsequent iterations use `RandomizedEIG` using $\rho = 5$ and $q = 3$ power iterations. On a cluster with 10 computers, the mean time per iteration is 401 seconds. The table in Fig. 4 (left) shows the error metrics of the recovered matrix, and Fig. 4 (right) plots the convergence rate of the Frobenius-norm error and trace distance.

Figure 5 reports the median error on 20 test problems across a range of p. Here, $\mathbf{X}^\star$ is only approximately

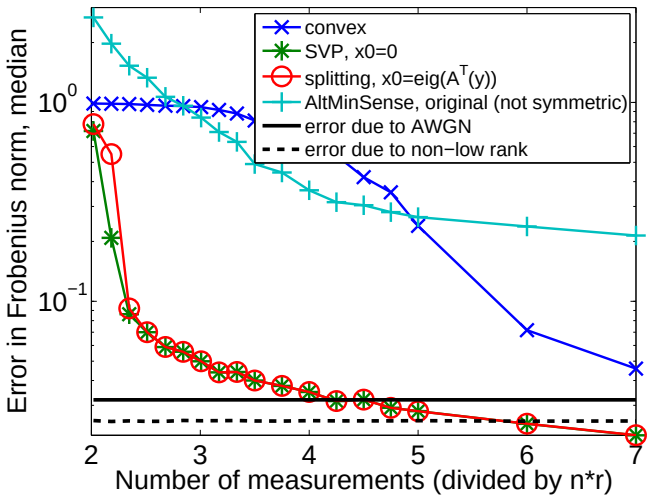

Figure 5: Accuracy comparison of several algorithms, as a function of number of samples p. Each point is the median of the results of 20 simulations.

low rank and y is contaminated with noise. We compare the convex approach [FGLE12], the "AltMinSense" approach [JNS12], and a standard splitting approach. AltMinSense and the convex approach have poor accuracy; the accuracy of AltMinSense can be improved by incorporating symmetry, but this changes the algorithm fundamentally and the theoretical guarantees are lost. The splitting approach, if initialized correctly, is accurate, but lacks guarantees. Furthermore, it is slower in practice due to slower convergence, though for some simple problems (i.e., no convex constraints $\mathcal{C}$) it is possible to accelerate using L-BFGS [Lau12].

6 Conclusion

Randomization is a powerful tool to accelerate and scale optimization algorithms, and it can be rigorously included in algorithms that are robust to small errors. In this paper, we leverage randomized approximations to remove memory bottlenecks by merging the two-key steps of most recovery algorithms in affine rank minimization problems: gradient calculation and low-rank projection. Unfortunately, the current black-box approximation guarantees, such as Theorem 1, are too pessimistic to be directly used in theoretical characterizations of our approach. For future work, motivated by the overwhelming empirical evidence of the good performance of our approach, we plan to directly analyze the impact of randomization in characterizing the algorithmic performance.

Acknowledgment

VC and AK's work was supported in part by the European Commission under Grant MIRG-268398, ERC Future Proof, SNF 200021-132548, and ARO MURI W911NF0910383. SRB is supported by the Fondation Sciences Mathématiques de Paris. The authors thank Alex Gittens for his insightful comments and Yi-Kai Liu and Steve Flammia for helpful discussions.

A Proofs

Proof of Theorem 2. There are three aspects to the proof. Even without approximate SVD calculations, the problem is non-convex, so we must leverage the R-RIP to prove that iterates converge. Mixed in with this calculation is the approximate nature of our rank ℓ point $\widetilde{\mathbf{X}}_{i+1}$, where we will apply the bounds from Theorem 1. Finally, we relate $\widetilde{\mathbf{X}}_{i+1}$ to its rank r version $\mathbf{X}_{i+1}$.

An important definition for our subsequent developments is the following:

Definition 2 (ϵ-approximate low-rank projection)**.** *Let* $\mathbf{X}$ *be an arbitrary matrix. For any* $\epsilon > 0$, $\mathcal{P}^{\epsilon}_{r',\ell'}(\mathbf{X})$ *provides a rank-*ℓ' *matrix approximation to* $\mathbf{X}$ *such that*

$$\mathbb{E}\big\|\mathcal{P}^{\epsilon}_{r',\ell'}(\mathbf{X}) - \mathbf{X}\big\|_F^2 \leq (1+\epsilon)\big\|\mathcal{P}_{r'}(\mathbf{X}) - \mathbf{X}\big\|_F^2, \tag{11}$$

where $\mathcal{P}_{r'}(\mathbf{X}) \in \operatorname{argmin}_{\mathbf{Y}:\operatorname{rank}(\mathbf{Y})\leq r'} \|\mathbf{X}-\mathbf{Y}\|_F$.

Let $\mathbf{X}_i$ be the putative rank r solution at the i-th iteration, $\mathbf{X}^\star$ be the rank r matrix we are looking for and $\widetilde{\mathbf{X}}_{i+1}$ be the rank l matrix, obtained using approximate SVD calculations. Define $L := 2(1+\delta_{r+\ell})$ and $M := 2(1-\delta_{2r})$. Then, we have:

$$\begin{aligned}
f(\widetilde{\mathbf{X}}_{i+1}) &= f(\mathbf{X}_i) + \langle \nabla f(\mathbf{X}_i),\ \widetilde{\mathbf{X}}_{i+1} - \mathbf{X}_i\rangle + \|\boldsymbol{\mathcal{A}}(\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}_i)\|_F^2 \\
&\leq f(\mathbf{X}_i) + \langle \nabla f(\mathbf{X}_i),\ \widetilde{\mathbf{X}}_{i+1} - \mathbf{X}_i\rangle + \frac{L}{2}\|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}_i\|_F^2 \\
&= f(\mathbf{X}_i) - \frac{1}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 + \frac{L}{2}\left(\|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}_i\|_F^2 + 2\langle \frac{1}{L}\nabla f(\mathbf{X}_i),\ \widetilde{\mathbf{X}}_{i+1} - \mathbf{X}_i\rangle + \frac{1}{L^2}\|\nabla f(\mathbf{X}_i)\|_F^2\right) \\
&= f(\mathbf{X}_i) - \frac{1}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 + \frac{L}{2}\|\widetilde{\mathbf{X}}_{i+1} - \left(\mathbf{X}_i - \frac{1}{L}\nabla f(\mathbf{X}_i)\right)\|_F^2.
\end{aligned} \tag{12}$$

By construction $\widetilde{\mathbf{X}}_{i+1} \in \mathcal{P}^{\epsilon}_{r,\ell}\left(\mathbf{X}_i - \frac{1}{L}\nabla f(\mathbf{X}_i)\right)$ (since the step-size is $\mu = 1/L$), so, for $\overline{\mathbf{X}}_{i+1} \in \mathcal{P}_r\left(\mathbf{X}_i - \frac{1}{L}\nabla f(\mathbf{X}_i)\right)$,

$$\begin{aligned}
\mathbb{E}\|\widetilde{\mathbf{X}}_{i+1} - (\mathbf{X}_i - \frac{1}{L}\nabla f(\mathbf{X}_i))\|_F^2 &\leq (1+\epsilon)\|\overline{\mathbf{X}}_{i+1} - (\mathbf{X}_i - \frac{1}{L}\nabla f(\mathbf{X}_i))\|_F^2 \\
&\leq (1+\epsilon)\|\mathbf{X}^\star - (\mathbf{X}_i - \frac{1}{L}\nabla f(\mathbf{X}_i))\|_F^2
\end{aligned} \tag{13}$$

by the definition of $\mathcal{P}_r(\cdot)$ (since $\operatorname{rank}(\mathbf{X}^\star) = r$). Combining (13) with (12), we obtain:

$$\begin{aligned}
\mathbb{E}f(\widetilde{\mathbf{X}}_{i+1}) &\leq f(\mathbf{X}_i) - \frac{1}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 + \frac{L}{2}(1+\epsilon)\|\mathbf{X}^\star - \mathbf{X}_i + \frac{1}{L}\nabla f(\mathbf{X}_i)\|_F^2 \\
&= f(\mathbf{X}_i) - \frac{1}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 + (1+\epsilon)\left(\frac{1}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 + \langle \nabla f(\mathbf{X}_i),\ \mathbf{X}^\star - \mathbf{X}_i\rangle + \frac{L}{2}\|\mathbf{X}^\star - \mathbf{X}_i\|_F^2\right) \\
&\leq (1+\epsilon)\left[f(\mathbf{X}_i) + \langle \nabla f(\mathbf{X}_i),\ \mathbf{X}^\star - \mathbf{X}_i\rangle + \frac{L}{2}\|\mathbf{X}^\star - \mathbf{X}_i\|_F^2\right] + \frac{\epsilon}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2
\end{aligned} \tag{14}$$

where we use the fact that $f(\mathbf{X}_i) \geq 0$ in the last inequality. Due to the restricted strong convexity of f that follows from the restricted isometry property, we have:

$$\begin{aligned}
f(\mathbf{X}^\star) &\geq f(\mathbf{X}_i) + \langle \nabla f(\mathbf{X}_i), \mathbf{X}^\star - \mathbf{X}_i\rangle + \frac{M}{2}\|\mathbf{X}^\star - \mathbf{X}_i\|_F^2 \\
f(\mathbf{X}^\star) - \frac{M}{2}\|\mathbf{X}^\star - \mathbf{X}_i\|_F^2 &\geq f(\mathbf{X}_i) + \langle \nabla f(\mathbf{X}_i), \mathbf{X}^\star - \mathbf{X}_i\rangle
\end{aligned}$$

which, combined with (14), leads to:

$$\mathbb{E}f(\widetilde{\mathbf{X}}_{i+1}) \leq (1+\epsilon)\left[f(\mathbf{X}^\star) + \frac{L-M}{2}\|\mathbf{X}^\star - \mathbf{X}_i\|_F^2\right] + \frac{\epsilon}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 \tag{15}$$

Due to the R-RIP,

$$\|\mathbf{X}^\star - \mathbf{X}_i\|_F^2 \leq \frac{\|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_i)\|_2^2}{1-\delta_{2r}} \tag{16}$$

Now define a constant C and assume $f(\mathbf{X}_i) = \|\mathbf{y} - \boldsymbol{\mathcal{A}}\mathbf{X}_i\|_2^2 > C^2\|\boldsymbol{\varepsilon}\|_2^2$ (if the assumption fails, it means $\mathbf{X}_i$ is already close to $\mathbf{X}^\star$). In particular, in the noiseless case $\|\boldsymbol{\varepsilon}\| = 0$, we may pick C arbitrarily large and set all $1/C$ terms to zero.

$$\begin{aligned}
\|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_i)\|_F^2 &= \|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_i) - \boldsymbol{\varepsilon}\|_2^2 \\
&= \|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_i)\|_2^2 + \|\boldsymbol{\varepsilon}\|_2^2 - 2\langle \boldsymbol{\varepsilon}, \mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_i)\rangle \\
&\leq f(\mathbf{X}_i) + \|\boldsymbol{\varepsilon}\|_2^2 + 2\|\boldsymbol{\varepsilon}\|_2\|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_i)\|_2 \\
&\leq f(\mathbf{X}_i) + \|\boldsymbol{\varepsilon}\|_2^2 + \frac{2}{C}f(\mathbf{X}_i)
\end{aligned} \tag{17}$$

Substituting (17) and (16) into (15), expanding the values of L and M, and noting that $f(\mathbf{X}^\star) = \|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}^\star)\|_2^2 = \|\boldsymbol{\varepsilon}\|_2^2$, gives

$$\mathbb{E}f(\widetilde{\mathbf{X}}_{i+1}) \leq (1+\epsilon)\left[\|\boldsymbol{\varepsilon}\|_2^2 + \frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\left(f(\mathbf{X}_i) + \|\boldsymbol{\varepsilon}\|_2^2 + \frac{2}{C}f(\mathbf{X}_i)\right)\right] + \frac{\epsilon}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 \tag{18}$$

$$\leq (1+\epsilon)\left[\frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\left(1+\frac{2}{C}\right)f(\mathbf{X}_i) + \left(1+\frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\right)\|\boldsymbol{\varepsilon}\|_2^2\right] + \frac{\epsilon}{2L}\|\nabla f(\mathbf{X}_i)\|_F^2 \tag{19}$$

We bound $\|\nabla f(\mathbf{X}_i)\|$ using our assumption on the magnitude of $\|\boldsymbol{\mathcal{A}}\|$:

$$\|\nabla f(\mathbf{X}_i)\|_F^2 = 4\|\boldsymbol{\mathcal{A}}^*\left(\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_i)\right)\|_F^2 \leq 4\|\boldsymbol{\mathcal{A}}^*\|^2\|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_i)\|_2^2 = 4\|\boldsymbol{\mathcal{A}}\|^2 f(\mathbf{X}_i) \leq 4\frac{mn}{p}f(\mathbf{X}_i) \tag{20}$$

For quantum tomography, we even have $\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{A}}^* = \frac{mn}{p}\mathcal{I}$, so the inequality holds with equality (and $m = n$).

Combining (19) with (20) and by the definition of L, we obtain:

$$\mathbb{E}f(\widetilde{\mathbf{X}}_{i+1}) \leq (1+\epsilon)\left[\frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\left(1+\frac{2}{C}\right)f(\mathbf{X}_i) + \left(1+\frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\right)\|\boldsymbol{\varepsilon}\|_2^2\right] + \frac{\epsilon}{1+\delta_{r+\ell}}\cdot\frac{mn}{p}f(\mathbf{X}_i) \tag{21}$$

$$= \underbrace{\left(\frac{\epsilon}{1+\delta_{r+\ell}}\cdot\frac{mn}{p} + (1+\epsilon)\frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\left(1+\frac{2}{C}\right)\right)}_{\theta'}f(\mathbf{X}_i) + \underbrace{(1+\epsilon)\left(1+\frac{\delta_{r+\ell}+\delta_{2r}}{1-\delta_{2r}}\right)}_{\tau'}\|\boldsymbol{\varepsilon}\|_2^2 \tag{22}$$

Note that if an exact SVD computation is used, then not only is $\epsilon = 0$ but also $\widetilde{\mathbf{X}}_{i+1}$ is rank r, so we are done and can use $\theta = \theta'$ and $\tau = \tau'$. To finish the proof, we now relate $\mathbb{E}f(\mathbf{X}_{i+1})$ to $\mathbb{E}f(\widetilde{\mathbf{X}}_{i+1})$. In the algorithm, $\mathbf{X}_{i+1}$ is the output of `RandomizedSVD`, and $\widetilde{\mathbf{X}}_{i+1}$ is the intermediate value $U\Sigma V^*$ on line 10 of Algo. 1. Given $\widetilde{\mathbf{X}}_{i+1}$ with $\operatorname{rank}(\widetilde{\mathbf{X}}_{i+1}) = \ell > r$, $\mathbf{X}_{i+1}$ is defined as the best rank-r approximation to $\widetilde{\mathbf{X}}_{i+1}$.[4] Thus, the following inequality holds true:

$$\begin{aligned}\|\mathbf{X}_{i+1} - \mathbf{X}^\star\|_F &= \|\mathbf{X}_{i+1} - \widetilde{\mathbf{X}}_{i+1} + \widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star\|_F \\ &\leq \|\mathbf{X}_{i+1} - \widetilde{\mathbf{X}}_{i+1}\|_F + \|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star\|_F \\ &\leq 2\|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star\|_F\end{aligned} \tag{23}$$

since $\|\mathbf{X}_{i+1} - \widetilde{\mathbf{X}}_{i+1}\|_F \leq \|\mathbf{X}^\star - \widetilde{\mathbf{X}}_{i+1}\|_F$. In particular, since the above is valid for any value of the random variable $\widetilde{\mathbf{X}}_{i+1}$, $\mathbb{E}\,\|\mathbf{X}_{i+1} - \mathbf{X}^\star\|_F^2 \leq \mathbb{E}\,4\|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star\|_F^2$. This bound is pessimistic and in practice the constant is close to 1 rather than 4.

We will again assume that $f(\widetilde{\mathbf{X}}_{i+1}), f(\mathbf{X}_{i+1}) \geq C^2\|\boldsymbol{\varepsilon}\|_2^2$, and $C > 2$, since otherwise the current point is a good-enough solution. We have:

$$\begin{aligned}f(\mathbf{X}_{i+1}) = \|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_{i+1})\|_2^2 &= \|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1}) + \boldsymbol{\varepsilon}\|_2^2 \\ &= \|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 + \|\boldsymbol{\varepsilon}\|_2^2 + 2\langle\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1}), \boldsymbol{\varepsilon}\rangle \\ &= \|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 + \|\boldsymbol{\varepsilon}\|_2^2 + 2\langle\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_{i+1}) - \boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}\rangle \\ &= \|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 + \|\boldsymbol{\varepsilon}\|_2^2 + 2\langle\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_{i+1}), \boldsymbol{\varepsilon}\rangle + 2\langle-\boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}\rangle \\ &\leq \|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 + \|\boldsymbol{\varepsilon}\|_2^2 + 2\|\mathbf{y} - \boldsymbol{\mathcal{A}}(\mathbf{X}_{i+1})\|_2\|\boldsymbol{\varepsilon}\|_2 - 2\|\boldsymbol{\varepsilon}\|_2^2 \\ &\leq \|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 - \|\boldsymbol{\varepsilon}\|_2^2 + \frac{2}{C}f(\mathbf{X}_{i+1})\end{aligned}$$

which, if $1 - 2/C \geq 0$, implies

$$f(\mathbf{X}_{i+1}) \leq \frac{1}{1-2/C}\|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 - \frac{1}{1-2/C}\|\boldsymbol{\varepsilon}\|_2^2 \tag{24}$$

[4] If we include a convex constraint $\mathcal{C}$ then instead of defining $\mathbf{X}_{i+1} = \mathcal{P}_r(\widetilde{\mathbf{X}}_{i+1})$ we have $\mathbf{X}_{i+1} = \mathcal{P}_C(\mathcal{P}_r(\widetilde{\mathbf{X}}_{i+1}))$. In this case,

$$\|\mathcal{P}_C(\mathcal{P}_r(\widetilde{\mathbf{X}}_{i+1})) - \mathbf{X}^\star\|_F = \|\mathcal{P}_C(\mathcal{P}_r(\widetilde{\mathbf{X}}_{i+1}) - \mathbf{X}^\star)\|_F \leq \|\mathcal{P}_r(\widetilde{\mathbf{X}}_{i+1}) - \mathbf{X}^\star\|_F.$$

The first equality follows from $\mathbf{X}^\star \in \mathcal{C}$ and the second is true since the projection onto a non-empty closed convex set is non-expansive. Hence the result in (23) still applies when we include the $\mathcal{C}$ constraints.

By the R-RIP assumption, we have:

$$\|\boldsymbol{\mathcal{A}}(\mathbf{X}^\star - \mathbf{X}_{i+1})\|_2^2 \leq (1+\delta_{2r})\|\mathbf{X}^\star - \mathbf{X}_{i+1}\|_F^2. \tag{25}$$

Using (23) and (25) in (24), we obtain:

$$f(\mathbf{X}_{i+1}) \leq \frac{4(1+\delta_{2r})}{1-2/C}\|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star\|_F^2 - \frac{1}{1-2/C}\|\boldsymbol{\varepsilon}\|_2^2 \tag{26}$$

Using the R-RIP property again, the following sequence of inequalities holds:

$$\begin{aligned}\|\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star\|_F^2 &\leq \frac{\|\boldsymbol{\mathcal{A}}(\widetilde{\mathbf{X}}_{i+1} - \mathbf{X}^\star)\|_F^2}{1-\delta_{r+\ell}} \\ &\leq \frac{1+2/C}{1-\delta_{r+\ell}} f(\widetilde{\mathbf{X}}_{i+1}) + \frac{1}{1-\delta_{r+\ell}}\|\boldsymbol{\varepsilon}\|_2^2 \end{aligned} \tag{27}$$

where the second inequality is obtained following same motions as (17). Combining (26)-(27) with (22), we obtain:

$$\mathbb{E}f(\mathbf{X}_{i+1}) \leq \underbrace{\frac{4(1+\delta_{2r})}{1-2/C}\cdot\frac{1+2/C}{1-\delta_{r+\ell}}\cdot\theta'}_{\theta}\cdot f(\mathbf{X}_i) + \underbrace{\left(\frac{4(1+\delta_{2r})}{1-2/C}\cdot\frac{1+2/C}{1-\delta_{r+\ell}}\cdot\tau' + \frac{4(1+\delta_{2r})}{1-2/C}\cdot\frac{1}{1-\delta_{r+\ell}} - \frac{1}{1-2/C}\right)}_{\tau}\|\boldsymbol{\varepsilon}\|_2^2$$

Now we simplify the result to make it more interpretable. Define $\rho = \ell - r$. Let c be the smallest integer such that $\ell \geq (c-1)r$ (and for simplicity, assume $\ell = (c-1)r$) so that $\delta_{r+\ell} = \delta_{cr}$ and $\delta_{r+\ell} + \delta_{2r} \leq 2\delta_{cr}$. By Theorem 1, $\epsilon \leq \frac{r}{\rho-1} = \frac{r}{(c-2)r-1}$. For concreteness, take $C \geq 4$ so that $1 + 2/C \leq 3/2$ and $(1-2/C)^{-1} \leq 2$. Then

$$\theta \leq 12\cdot\frac{1+\delta_{2r}}{1-\delta_{cr}}\cdot\left(\frac{\epsilon}{1+\delta_{cr}}\cdot\frac{mn}{p} + (1+\epsilon)\frac{3\delta_{cr}}{1-\delta_{2r}}\right) \tag{28}$$

and

$$\begin{aligned}\tau &\leq \left(12\cdot\frac{1+\delta_{2r}}{1-\delta_{cr}}\cdot(1+\epsilon)\left(1+\frac{\delta_{2r}+\delta_{cr}}{1-\delta_{2r}}\right) + \frac{8(1+\delta_{2r})}{1-\delta_{cr}}\right) \\ &\leq \frac{1+\delta_{2r}}{1-\delta_{cr}}\cdot\left(12\cdot(1+\epsilon)\left(1+\frac{2\delta_{cr}}{1-\delta_{2r}}\right)+8\right)\end{aligned} \tag{29}$$

□

References

[BCKK13] S. Becker, V. Cevher, C. Koch, and A. Kyrillidis, *Sparse projections onto the simplex*, ICML, *to appear*, 2013.

[BM03] S. Burer and R.D.C. Monteiro, *A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization*, Math. Prog. (series B) **95** (2003), no. 2, 329–357.

[CR09] E.J. Candes and B. Recht, *Exact matrix completion via convex optimization*, Found. Comput. Math. **9** (2009), 717–772.

[FGLE12] S.T. Flammia, D. Gross, Y.K. Liu, and J. Eisert, *Quantum tomography via compressed sensing: error bounds, sample complexity, and efficient estimators*, New J. Phys. **14** (2012), no. 9, 095022.

[GK09] R. Garg and R. Khandekar, *Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property*, ICML, ACM, 2009.

[GLF+10] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, *Quantum state tomography via compressed sensing*, Phys. Rev. Lett. **105** (2010), no. 15, 150401.

[GM11] D. Goldfarb and S. Ma, *Convergence of fixed-point continuation algorithms for matrix rank minimization*, Foundations of Computational Mathematics **11** (2011), no. 2, 183–210.

[HMT11] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions*, SIAM Rev. **53** (2011), no. 2, 217–288.

[JNS12] P. Jain, P. Netrapalli, and S. Sanghavi, *Low-rank matrix completion using alternating minimization*, ACM Symp. Theory Comput., 2012.

[KC12] Anastasios Kyrillidis and Volkan Cevher, *Matrix recipes for hard thresholding methods*, arXiv preprint arXiv:1203.4481 (2012).

[Lau12] S. Laue, *A hybrid algorithm for convex semidefinite optimization*, ICML, 2012.

[Liu11] Y. K. Liu, *Universal low-rank matrix recovery from Pauli measurements*, NIPS, 2011, pp. 1638–1646.

[LRS+11] J. Lee, B. Recht, R. Salakhutdinov, N. Srebro, and J. A. Tropp, *Practical large-scale optimization for max-norm regularization*, NIPS, 2011.

[MJD10] R. Meka, P. Jain, and I. S. Dhillon, *Guaranteed rank minimization via singular value projection*, NIPS, 2010.

[NC10] M.A. Nielsen and I.L. Chuang, *Quantum computation and quantum information*, Cambridge university press, 2010.

[Nes83] Y. Nesterov, *A method for unconstrained convex minimization problem with the rate of convergence* $\mathcal{O}(1/k^2)$, Doklady AN SSSR, *translated as* Soviet Math. Docl. **269** (1983), 543–547.

[NT09] D. Needell and J. Tropp, *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples*, Appl. Comput. Harmon. Anal **26** (2009), 301–321.

[RFP10] B. Recht, M. Fazel, and P.A. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM review **52** (2010), no. 3, 471–501.

[RR13] B. Recht and C. Ré, *Parallel stochastic gradient algorithms for large-scale matrix completion*, Math. Prog. Comput., *to appear* (2013).

[WYZ10] Z. Wen, W. Yin, and Y. Zhang, *Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm*, Mathematical Programming Computation (2010), 1–29.