Fast Low-Rank Matrix Estimation for Ill-Conditioned Matrices

Mohammadreza Soltani
Iowa State University
Email: msoltani@iastate.edu

Chinmay Hegde
Iowa State University
Email: chinmay@iastate.edu

***Abstract*—We study the general problem of optimizing a convex function of a matrix-valued variable subject to low-rank constraints. This problem has attracted significant attention; however, existing first-order methods for solving such problems either are too slow to converge, or require multiple invocations of singular value decompositions. On the other hand, factorization-based non-convex algorithms, while being much faster, require stringent assumptions on the condition number of the optimum. In this paper, we provide a novel algorithmic framework that achieves the best of both worlds: as fast as factorization methods, while requiring no spectral assumptions. We instantiate our framework for the nonlinear affine rank minimization (NLARM) problem. For this problem, we derive explicit bounds on the sample complexity as well as running time of our approach, and show that it achieves the best possible bounds for both cases. We also support our proposed algorithm via several experimental results.**

I. Introduction

We focus on the constrained optimization problem:

$$\min_{L} \quad F(L) \quad \text{s.t.} \quad \text{rank}(L) \leq r^*, \tag{1}$$

where $F(L) : \mathbb{R}^{p\times p} \to \mathbb{R}$ is a convex smooth function defined over matrices $L \in \mathbb{R}^{p\times p}$. This problem has recently received significant attention in machine learning, statistics, and signal processing [1], [2]. Several applications abound, including affine rank minimization [3], [4], matrix completion [5], robust PCA [6]–[8], covariance/precision matrix estimation using graphical models [9], [10], phase retrieval [11], and learning shallow polynomial neural networks [12], [13], to name a few.

From the computational perspective, the traditional approach is to adopt first-order optimization for solving (1). Several different approaches have been proposed in recent years. These methods suffer from one or several of the following problems: either their convergence rate is too slow (sub-linear or worse); the computational cost per iteration is too high (quadratic or worse); or they have stringent assumptions on the spectral properties (such as the condition number) of the solution to (1). Our goal in this paper is to propose an algorithm to alleviate the above problems simultaneously. Specifically, we seek an algorithm that exhibits *linearly fast convergence*, *computationally efficient per iteration*, and at the same time, *robust to ill-conditioned problems.*

A. Our Contributions

In this paper, we propose and analyze an algorithm for solving problems of the form (1) for objective functions F that satisfy the commonly-studied *Restricted Strong Convexity/Smoothness* (RSC/RSS) conditions. Our method enjoys:

Linear convergence. We propose a fast non-convex algorithm for solving of the optimization problem in (1). Specifically, we provide rigorous analysis to show that our proposed algorithm enjoy global linear convergence (no matter how it is initialized). Our algorithm enjoys fast per-iteration running time as well.

No spectral assumptions. We show that our proposed algorithm does not depend on stringent spectral assumptions (such as condition number) on the solution to (1), therefore making our method suitable for ill-conditioned problems.

No limitations on strong convexity/smoothness constants. In a departure from the majority of the matrix optimization literature, our algorithm succeeds under no particular assumptions on the *extent* to which the objective function F is strongly smooth/convex. (See below for details).

Putting together these ingredients, we get the first *condition-free*, almost-linear time algorithm for solving problems of the form (1).

B. Techniques

Our approach is an adaptation of the algorithm proposed in [14], which is a projected gradient-type algorithm. The key idea of this work is that each gradient update is projected onto the space of matrices with rank r that is *larger* than r^*, the rank parameter in (1). This trick can alleviate situations where the objective function exhibits poor restricted strong convexity/smoothness properties; more generally, the overall algorithm can be applied to ill-posed problems. However, the per-iteration cost of their algorithms is *cubic* ($\mathcal{O}(p^3)$) in the matrix dimension, due to use of multiple SVDs.[1]

Our approach resolves this issue by replacing the exact SVD with a gap-independent *approximate* low-rank projection, while still retaining the idea of projecting onto a larger space. To establish soundness of our approach, we establish a property about (approximate) singular value thresholding that extends recent new results proved in [15], [16]. In particular, we prove a new structural result for approximate projection onto the space of rank-r matrices, showing that each projection

[1] Even if standard partial SVD routines (such as the power method) are used, the running time scales as $\mathcal{O}(p^2 r/\text{gap})$, where gap represents the difference r^{th} and $(r+1)^{th}$ singular values; if the gap is small, this blows up the running time to $\mathcal{O}(p^3)$ or worse.

step in our algorithm is *nearly non-expansive*. Integrating the above result gives linear convergence of the proposed algorithm for a very broad class of objective functions $F(L)$. Since we use approximate low-rank projections, the running time of the projection step is (almost) linear in the size of the matrix if r^* is constant.

C. Stylized Application

We instantiate our framework to a practical application that we call *nonlinear affine rank minimization* (NLARM). Formally, we consider an observation model akin to the Generalized Linear Model (GLM) [17]: $y = g(\mathcal{A}(L^*)) + e$, where g denotes a nonlinear *link* function, $\mathcal{A}$ denotes a linear *measurement* (or observation) operator, which we formally define later, and $e \in \mathbb{R}^m$ denotes an additive noise vector. The goal is to reconstruct L^* from y, given that L^* is of rank at most r^*. For this application, we derive the sample complexity and the running time of MAPLE, and analyze the statistical error rate. These results show the (near) optimality of both sample complexity and running time without dependency on the condition number.

II. Prior Work

Solving (1) as efficiently as possible has attracted considerable recent interest in the learning theory community. Most solution approaches can be categorized into four groups.

In the first group, the non-convex rank constraint is relaxed into a *nuclear norm* penalty, which results in a convex problem and can be solved by off-the-shelf solvers such as SDP solvers [18], singular value thresholding and its accelerated versions [3], [19] While convex methods are well-known, their usage in the high dimensional regime is prohibitive. The second group replaces the rank constraint with a more tractable *non-convex* regularizer instead of the nuclear norm. While this reduces the computational cost per iteration, its convergence rate is sub-linear [20].

Methods in the third group try to solve the non-convex optimization problem (1) based on the factorization approach of [21]. In these algorithms, the rank-r matrix L is factorized as $L = UV^T$, where $U, V \in \mathbb{R}^{p\times r}$. Using this idea removes the difficulties caused by the non-convex rank constraint; however, the objective function is not convex anymore. Such methods have recently gained in popularity in the machine learning with provable linear-convergence guarantees [1], [22], [23]. While these methods are currently among the fastest available, their drawback is that they may require one or multiple full singular value decomposition for initialization. More crucially, their convergence rate also depends heavily on the condition number of the optimum which makes their sample complexity and running time blow up by a significant amount if the problem is somehow poorly conditioned.

Separate from factorization approaches, the fourth group of methods use non-convex low-rank projections within classical gradient descent. The earliest such approach, called singular value projection (SVP), was introduced by [4] for matrix recovery from linear measurements, and was later modified for general M-estimation problems with well-behaved objective functions [14]. These methods require multiple invocations of exact singular value decompositions (SVDs). A similar algorithm was proposed by [24] for the squared loss case, and replaced the exact SVD with an approximate one. However, careful scrutiny reveals that their theoretical guarantees are rather restrictive, and do not demonstrate the benefits of approximate SVD over exact projections. Due to lack of space, please refer the recent survey [25] and references therein for a comprehensive discussion.

All the aforementioned groups of algorithms suffer from one (or more) of the following issues: expensive computational complexity, slow convergence rate, and troublesome dependency on the matrix condition number. In this paper, we resolve these problems by a renewed analysis of approximate low-rank projection algorithms, and integrate this analysis to obtain a new algorithm for optimizing general convex loss functions with rank constraints.

III. Algorithm and Analysis

A. Preliminaries

For convenience, all our matrix variables will be of size $p\times p$, but our results extend seamlessly to rectangular matrices. We use $\|A\|_2$ and $\|A\|_F$ for the spectral and Frobenius norm of a matrix A, respectively. For any subspace $W \subset \mathbb{R}^{p\times p}$, we denote $\mathcal{P}_W$ as the orthogonal projection operator onto it. Our analysis will rely on the following definition [14], [26]:

Definition 1. *A function f satisfies the Restricted Strong Convexity (RSC) and Restricted Strongly Smoothness (RSS) conditions if for all $L_1, L_2 \in \mathbb{R}^{p\times p}$ such that rank$(L_1) \le r$, rank$(L_2) \le r$, we have:*

$$\frac{m_r}{2}\|L_2 - L_1\|_F^2 \le f(L_2) - f(L_1) - \langle \nabla f(L_1), L_2 - L_1\rangle \le \frac{M_r}{2}\|L_2 - L_1\|_F^2, \qquad (2)$$

where m_r and M_r are called the RSC and RSS constants respectively.

Let $\mathbb{U}_r$ be the set of all rank-r matrix subspaces of $\mathbb{R}^{p\times p}$. We will exclusively focus on low-rank approximation algorithms that satisfy the following two properties:

Definition 2 (Approximate tail projection)**.** *Let $\epsilon > 0$. Then, $\mathcal{T} : \mathbb{R}^{p\times p} \to \mathbb{U}_r$ is an approximate tail projection algorithm if for all $L \in \mathbb{R}^{p\times p}$, $\mathcal{T}$ returns a subspace $Z = \mathcal{T}(L)$ that satisfies: $\|L - \mathcal{P}_Z L\|_F \le (1+\epsilon)\|L - L_r\|_F$, where $\mathcal{P}_Z L = ZZ^T L$, and L_r is the optimal rank-r approximation of L in the Frobenius norm.*

Definition 3 (Per-vector approximation guarantee)**.** *Let $L \in \mathbb{R}^{p\times p}$. Suppose there is an algorithm that satisfies approximate tail projection such that it returns a subspace Z with basis vectors $z_1, z_2, \ldots, z_r$ and approximate ratio ϵ. Then, this algorithm additionally satisfies the per-vector approximation guarantee if $|u_i^T LL^T u_i - z_i LL^T z_i| \le \epsilon\sigma_{r+1}^2$, where u_i's are the eigenvectors of L.*

Algorithm 1 MAPLE

Inputs: rank r, step size η, approximate tail projection $\mathcal{T}$
Outputs: Estimates $\widehat{L}$
Initialization: $L^0 \leftarrow 0$, $t \leftarrow 0$
while $t \leq T$ **do**
  $L^{t+1} = \mathcal{T}\left(L^t - \eta \nabla F(L^t)\right)$
  $t \leftarrow t+1$
end while
Return: $\widehat{L} = L^T$

In this paper, we focus on the randomized Block Krylov SVD (BKSVD) method for implementation of $\mathcal{T}$. This algorithm has been proposed by [27] which satisfies both of these properties with probability at least $99/100$. However, one can alternately use a recent algorithm called Lazy-PCA [28] with very similar properties. For constant approximation factors ϵ, the running time of these algorithms is given by $\widetilde{\mathcal{O}}(p^2 r)$, ***independent*** of any spectral properties of the input matrix.

As we discussed above, our goal is to solve the optimization problem (1). The traditional approach is to perform projected gradient descent: $L^{t+1} = P_r\left(L^t - \eta \nabla F(L^t)\right)$, where P_r denotes an exact projection onto the space of rank-r matrices, and can be accomplished via SVD. However, for large p, this incurs cubic running time and can be very challenging. To alleviate this issue, one can instead attempt to replace the full SVD in each iteration with a tail-approximate low-rank projection; it is known that such projections can computed in $\mathcal{O}(p^2 \log p)$ time [29]. This is precisely our proposed algorithm, which we call *Matrix Approximation for Low-rank Estimation* (MAPLE), is described in pseudocode form as Algorithm 1. This algorithm is structurally very similar to [14], [24]. However, the proof of [14] requires exact low-rank projections, and [24] is specific to least-squares loss functions and with weak guarantees which scales up the running time to $\mathcal{O}(p^3)$. A key point is that our algorithm uses approximate low-rank projections with parameter r such that $r \geq r^*$. As we show in Theorem 5, the combination of using approximate projection, together with choosing a large enough rank parameter r, enables efficient solution of problems of the form (1) for *any (given) restricted convexity/smoothness constants* M, m.

In our implementation of MAPLE, we invoke the BKSVD method for low-rank approximation mentioned above[2]. Assuming BKSVD as the approximate low-rank projection of choice, we now prove a key structural result about the non-expansiveness of $\mathcal{T}$. This result, to the best of our knowledge, is novel and generalizes a recent result reported in [15], [16]. Please see [30] for the proof of all theoretical results.

Lemma 4. *For $r > (1+\frac{1}{1-\epsilon})r^*$ and for any matrices $L, L^* \in \mathbb{R}^{p\times p}$ with rank$(L^*) = r^*$, we have*

$$\|\mathcal{T}(L) - L^*\|_F^2 \leq \left(1 + \frac{2}{\sqrt{1-\epsilon}}\frac{\sqrt{r^*}}{\sqrt{r-r^*}}\right)\|L - L^*\|_F^2,$$

where $\mathcal{T} : \mathbb{R}^{p\times p} \to \mathbb{U}_r$ denotes the approximate tail projection defined in Definition 2 and $\epsilon > 0$ is the corresponding approximation ratio.

[2]We note that since the BKSVD algorithm is randomized while the definitions of approximate tail projection and per-vector approximation guarantee are deterministic. Fortunately, the running time of BKSVD depends only logarithmically on the failure probability, and therefore an additional union bound argument is required to precisely prove algorithmic correctness of our method.

TABLE I: *Comparison of MAPLE with existing methods for NLARM. κ denotes the condition number of L^*, and ϑ denotes the final optimization error. Sample complexity of all the algorithms is given by $n = \widetilde{\mathcal{O}}(pr^*)$. We have presented for each algorithm the best known running time result with bounded $\frac{M}{m}$ assumption.*

Algorithm	Running Time
Convex [3]	$\mathcal{O}(\frac{p^3}{\vartheta})$
NC-Reg [20]	$\mathcal{O}(\frac{p^2 r^*}{\vartheta})$
Factorized [23]	$\mathcal{O}((p^2 r^* + p^2 \log p)\kappa^2 \log(\frac{1}{\vartheta}) + p^3)$
SVP [14]	$\mathcal{O}(p^3 \log(\frac{1}{\vartheta}))$
MAPLE	$\mathcal{O}(p^2 r^* \log p \log(\frac{1}{\vartheta}))$

Using the above lemma, we provide our main theory supporting the statistical and computational efficiency of MAPLE.

Theorem 5 (Linear convergence of MAPLE)**.** *Assume that the objective function $F(L)$ satisfies the RSC/RSS conditions with parameters M_{2r+r^*} and m_{2r+r^*}. Define $\nu = \sqrt{1 + \frac{2}{\sqrt{1-\epsilon}}\frac{\sqrt{r^*}}{\sqrt{r-r^*}}}$. Let J_t denotes the subspace formed by the span of the column spaces of the matrices L^t, L^{t+1}, and L^*, the solution of* (1)*. In addition, assume that $r > \frac{C_1}{1-\epsilon}\left(\frac{M_{2r+r^*}}{m_{2r+r^*}}\right)^4 r^*$ for some $C_1 > 2$. Choose step size as η as $\frac{1-\sqrt{\alpha'}}{M_{2r+r^*}} \leq \eta \leq \frac{1+\sqrt{\alpha'}}{m_{2r+r^*}}$ where $\alpha' = \frac{\sqrt{\alpha}-1}{\sqrt{1-\epsilon}\sqrt{\alpha}-1+2}$ for some $\alpha = \Theta(r/r^*) > 1$. Then, MAPLE outputs a sequence of estimates L^t such that:*

$$\|L^{t+1} - L^*\|_F \leq \rho\|L^t - L^*\|_F + \nu\eta\|\mathcal{P}_{J_t}\nabla F(L^*)\|_F, \quad (3)$$

where $\rho = \nu\sqrt{1 + M_{2r+r^*}^2\eta^2 - 2m_{2r+r^*}\eta} < 1$.

The above theorem implies that no matter how large $\frac{M}{m}$ is, its effect is balanced by ν through factor ρ in (3) by choosing $r > r^*$. Theorem 5 also guarantees the linear convergence to L^* up to the statistical property of L^*, $\|\mathcal{P}_{J_t}\nabla F(L^*)\|_F$, which determines the quality of the estimation.

B. Stylized Application: Nonlinear Matrix Recovery

Consider the nonlinear observation model $y = g(\mathcal{A}(L^*)) + e$, where $\mathcal{A}$ is a linear operator, $\mathcal{A} : \mathbb{R}^{p\times p} \to \mathbb{R}^n$ parametrized by n full rank matrices, $A_i \in \mathbb{R}^{p\times p}$ such that $(\mathcal{A}(L^*))_i = \langle A_i, L^*\rangle$ for $i = 1, \ldots, n$. Also, e denotes an additive sub-gaussian noise vector with i.i.d., zero-mean entries that is also assumed to be independent of $\mathcal{A}$ (see [30] for more details). If $g(x) = x$, we have the well-known matrix sensing problem for which a large number of algorithms have been proposed.

The goal is to estimate the ground truth matrix $L^* \in \mathbb{R}^{p\times p}$ for more general nonlinear link functions. In this paper, we assume that link function $g(x)$ is a differentiable monotonic function, satisfying $0 < \mu_1 \le g'(x) \le \mu_2$ for all $x \in \mathcal{D}(g)$ (domain of g). This assumption is standard in statistical learning [17] and in nonlinear sparse recovery [26], [31], [32]. Also, as we will discuss below, this assumption will be helpful for verifying the RSC/RSS condition for the loss function that we define as follows. We estimate L^* by solving the optimization problem:

$$\begin{aligned} \min_{L} \quad & F(L) = \frac{1}{n}\sum_{i=1}^{n} \Omega(\langle A_i, L\rangle) - y_i\langle A_i, L\rangle \\ \text{s.t.} \quad & \text{rank}(L) \le r, \end{aligned} \tag{4}$$

where $\Omega : \mathbb{R} \to \mathbb{R}$ is chosen such that $\Omega'(x) = g(x)$. [3] Due assumption on the derivative of g, we see that $F(L)$ is a convex function (actually strongly convex), and can be considered as a special case of general problem in (1). We assume that the design matrices A_i's are constructed as follows. Consider a partial Fourier or partial Hadamard matrix $X' \in \mathbb{R}^{n\times p^2}$ which is multiplied from the right by a diagonal matrix, D, whose diagonal entries are uniformly distributed over $\{-1, +1\}^{p^2}$. Call the resulting matrix $X = X'D$ where each row is denoted by $X_i^T \in \mathbb{R}^{p^2}$. If we reshape each of these rows as a matrix, we obtain "measurement" (or "design") matrices $A_i \in \mathbb{R}^{p\times p}$ for $i = 1, \dots, m$. This particular choice of design matrices A_i's is because they support fast matrix-vector multiplication which takes $\mathcal{O}(p^2 \log(p))$. The following theorem gives the upper bound on the term, $\|\mathcal{P}_{J_t}\nabla F(L^*)\|_F$, as a "statistical error" term, and is zero in the absence of noise.

Theorem 6. *Consider the observation model $y = g(\mathcal{A}L^*) + e$ described above. Let the number of samples scale as $n = \mathcal{O}(pr \text{ polylog } (p))$, then with high probability, for any given subspace $J \subset \mathbb{R}^{p\times p}$: we have for $t = 1, \dots, T$:*

$$\|\mathcal{P}_J \nabla F(L^*)\|_F \le \frac{1+\delta_{2r+r^*}}{\sqrt{n}}\|e\|_2, \tag{5}$$

where $0 < \delta_{2r+r^*} < 1$ is a constant that depends on $\mathcal{A}$.

Putting together Theorem 5 and statistical bound in (5), by starting from $L^0 = 0$, MAPLE provides an estimate of L^* with an error ϑ of within $\mathcal{O}(\log(\frac{1}{\epsilon}))$ iterations. Next, we verify the assumption of RSC/RSS in Theorem 5.

Theorem 7 (RSC/RSS conditions for MAPLE)**.** *Let the number of samples scaled as $n = \mathcal{O}(pr \text{ polylog}(p))$. Also, assume that $\frac{\mu_2^4(1+\omega)^4}{\mu_1^4(1-\omega)^4} \le C_2(1-\epsilon)\frac{r}{r^*}$ for some $C_2, \omega > 0$ and $\epsilon > 0$ denotes the approximation ratio in algorithm 1. Then with high probability, the loss function $F(L)$ in (4) satisfies RSC/RSS conditions with constants $m_{2r+r^*} \ge \mu_1(1-\omega)$ and $M_{2r+r^*} \le \mu_2(1+\omega)$ in each iteration.*

Sample complexity. By Theorem 7, the sample complexity of MAPLE algorithm is given by $n = \mathcal{O}(pr \text{ polylog}(p))$ in order to achieve a specified estimation error. This sample complexity is nearly as good as the optimal rate, $\mathcal{O}(pr)$.

[3]The objective function $F(L)$ in (4) is standard; see [32] for a discussion.

Time complexity. Each iteration of MAPLE needs to compute the gradient, plus an approximate tail projection to produce a rank-r matrix. Computing the gradient involves one application of the linear operator $\mathcal{A}$ for calculating $\mathcal{A}(L)$, and one application of the adjoint operator, i.e., $\mathcal{A}^*(y - g(\mathcal{A}(L))$. Let T_{mult} and T'_{mult} denote the required time for these operations, respectively. On the other hand, approximate tail projection takes $\mathcal{O}\left(\frac{p^2 r \log p}{\sqrt{\varepsilon}}\right)$ operations for achieving the approximate ratio ϵ According to [27]. Thanks to the linear convergence of MAPLE, the total number of iterations for achieving ϑ accuracy is given by $T_{iter} = \mathcal{O}\left(\log\left(\frac{\|L^*\|_F}{\vartheta}\right)\right)$. Now define $\pi = \frac{M}{m}$. Hence, the overall running time scales as $T = \mathcal{O}\left(\left(T_{mult} + T'_{mult} + \frac{p^2 r^* \pi^4 \log p}{\sqrt{\epsilon}}\right)\left(\log\frac{\|L^*\|_F}{\vartheta}\right)\right)$ by the choice of r according to Theorem 5. If we assume that the design matrices A_i's are implemented via a Fast Fourier Transform, computing $T_{mult} = T'_{mult}$ takes $\mathcal{O}(p^2 \log p)$ operations. As a result, $T = \mathcal{O}\left(\left(p^2 \log p + \frac{p^2 r^* \pi^4 \log p}{\sqrt{\epsilon}}\right)\left(\log\frac{\|L^*\|_F}{\vartheta}\right)\right)$.

In Table I, for $g(x) = x$ and defined operator $\mathcal{A}$, we summarize the (asymptotic) running time of several algorithms with a constant ratio of M/m for all the algorithms.

IV. Experimental results

We provide some experiments that show the efficiency of MAPLE compared to existing approaches. Due to lack of space, we refer to [30] for more synthetic simulations as well as real data experiments. Here, the link function is set to $g(x) = 2x + \sin(x)$; this function satisfies the derivative conditions discussed above. We construct the ground truth low-rank matrix L^* with rank r^* by generating a random matrix $U \in \mathbb{R}^{p\times r^*}$ with entries drawn from the standard normal distribution. We ortho-normalize the columns of U, and set $L^* = UDU^T$ where $D \in \mathbb{R}^{r^*\times r^*}$ is a diagonal matrix with $D_{11} = \kappa(L^*)$, and $D_{jj} = 1$ for $j \ne 1$. After this, we apply a linear operator $\mathcal{A}$ on L^*, i.e., $\mathcal{A}(L^*)_i = \langle A_i, L^*\rangle$ where the choice of A_i has been discussed above. Finally, we obtain the measurements $y = g(\mathcal{A}(L^*))$. When reporting noise robustness, we add a Gaussian noise vector $e \in \mathbb{R}^m$ to $g(\mathcal{A}(L^*))$. In Panel (a) and (b) of Figure 1, the running time of the four algorithms are compared. For this experiment, we have chosen $p = 1000$, and the rank of the underlying matrix L^* to be 50. We also set the projected rank as $r = 50$. The number of measurements is set to $n = 4pr$. We consider a well-conditioned matrix L^* with $\kappa(L^*) = 1.1$ for (a) and $\kappa = 20$ for (b). Then we measure the relative error in estimating of L^* in Frobenius norm in log scale versus the CPU time takes for 200 iterations for all of the algorithms. We run the algorithms for 15 Monte Carlo trials. As we can see, when κ is small, FGD has comparable running time with MAPLE (plot (a)); on the other hand, when we have ill-posed L^*, FGD takes much longer to achieve the same relative error (plot (b)). Finally, we consider the noisy scenario in which the observation y is corrupted by different Gaussian noise level. The parameters are set as $p = 300$, $r = 10, 25, 40$ for MAPLE and 10 for the others, $r^* = 10$, $n = 7pr$, and $\kappa = 2$. The plot in Panel (c) shows the averaged over 50 trials of the

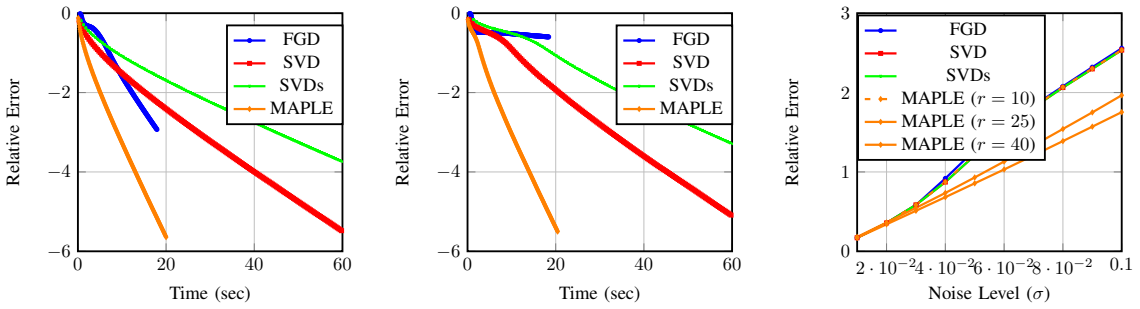

Fig. 1: *Comparisons with of algorithms with* $g(x) = 2x + sin(x)$. *(a) and (b) Average of the relative error in estimating* L^*. *Parameters:* $p = 1000$, $r^* = r = 50$, *and* $n = 4pr$. *(a):* $\kappa(L^*) = 1.1$. *(b):* $\kappa(L^*) = 20$. *(c) Average of the relative error with different noise level. Parameters:* $p = 300$, $\kappa = 2$, *and* $n = 7pr$.

relative error in L^* versus the various standard deviations. From this plot, we see that MAPLE with $r = 40$ is most robust, indicating that projection onto the larger subspace is beneficial when noise is present.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants CCF-1566281, CCF-1750920, and a Black and Veatch Faculty Fellowship.

REFERENCES

[1] Y. Chen and M. Wainwright, "Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees," *arXiv preprint arXiv:1509.03025*, 2015.

[2] M. Udell, C. Horn, R. Zadeh, and S. Boyd, "Generalized low rank models," *Foundations and Trends® in Machine Learning*, vol. 9, no. 1, pp. 1–118, 2016.

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

[4] P. Jain, R. Meka, and I. Dhillon, "Guaranteed rank minimization via singular value projection," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2010, pp. 937–945.

[5] E. Candès and B. Recht, "Exact matrix completion via convex optimization." *Found. Comput. Math.*, vol. 9, no. 6, 2009.

[6] E. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM*, vol. 58, no. 3, p. 11, 2011.

[7] V. Chandrasekaran, S. Sanghavi, P. Parrilo, and A. S. Willsky, "Sparse and low-rank matrix decompositions," in *Proc. Allerton Conf. on Comm., Contr., and Comp.*, 2009, pp. 962–967.

[8] P. Netrapalli, U. Niranjan, S. Sanghavi, A. Anandkumar, and P. Jain, "Non-convex robust pca," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2014, pp. 1107–1115.

[9] C. Hsieh, I. Dhillon, P. Ravikumar, S. Becker, and P. Olsen, "Quic & dirty: A quadratic approximation approach for dirty statistical models," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2014, pp. 2006–2014.

[10] V. Chandrasekaran, P. Parrilo, and A. S. Willsky, "Latent variable graphical model selection via convex optimization," in *Proc. Allerton Conf. on Comm., Contr., and Comp.*, 2010, pp. 1610–1613.

[11] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2013, pp. 2796–2804.

[12] R. Livni, S. Shalev-Shwartz, and O. Shamir, "On the computational efficiency of training neural networks," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2014, pp. 855–863.

[13] M. Soltani and C. Hegde, "Towards provable learning of polynomial neural networks using low-rank matrix estimation," in *Proc. Int. Conf. Art. Intell. Stat. (AISTATS)*, 2017.

[14] P. Jain, A. Tewari, and P. Kar, "On iterative hard thresholding methods for high-dimensional m-estimation," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2014, pp. 685–693.

[15] J. Shen and P. Li, "A tight bound of hard thresholding," *arXiv preprint arXiv:1605.01656*, 2016.

[16] X. Li, T. Zhao, R. Arora, H. Liu, and J. Haupt, "Nonconvex sparse learning via stochastic optimization with progressive variance reduction," *arXiv preprint arXiv:1605.02711*, 2016.

[17] S. Kakade, V. Kanade, O. Shamir, and A. Kalai, "Efficient learning of generalized linear and single index models with isotonic regression," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2011, pp. 927–935.

[18] M. G. and S. B., "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, 2014.

[19] T. Goldstein, C. Studer, and R. Baraniuk, "A field guide to forward-backward splitting with a FASTA implementation," *arXiv eprint*, vol. abs/1411.3406, 2014. [Online]. Available: http://arxiv.org/abs/1411.3406

[20] Y. Quanming, J. Kwok, T. Wang, and T. Liu, "Large-scale low-rank matrix learning with non-convex regularizers," *arXiv preprint arXiv:1708.00146*, 2017.

[21] S. Burer and R. Monteiro, "A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization," *Mathematical Programming*, vol. 95, no. 2, pp. 329–357, 2003.

[22] S. Tu, R. Boczar, M. Simchowitz, M. Soltanolkotabi, and B. Recht, "Low-rank solutions of linear matrix equations via procrustes flow," in *Proc. Int. Conf. Machine Learning*, 2016, pp. 964–973.

[23] S. Bhojanapalli, A. Kyrillidis, and S. Sanghavi, "Dropping convexity for faster semi-definite optimization," in *29th Ann. Conf. Learning Theory*, 2016, pp. 530–582.

[24] S. Becker, V. Cevher, and A. Kyrillidis, "Randomized low-memory singular value projection," in *Proc. Sampling Theory and Appl. (SampTA)*, no. EPFL-CONF-184017, 2013.

[25] M. A. Davenport and J. Romberg, "An overview of low-rank matrix recovery from incomplete observations," *IEEE J. Select. Top. Sig. Proc.*, vol. 10, no. 4, pp. 608–622, 2016.

[26] S. Negahban, B. Yu, M. Wainwright, and P. Ravikumar, "A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2011.

[27] C. Musco and C. Musco, "Randomized block krylov methods for stronger and faster approximate singular value decomposition," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2015, pp. 1396–1404.

[28] Z. Allen-Zhu and Y. Li, "Lazysvd: Even faster svd decomposition yet without agonizing pain," in *Adv. Neural Inf. Proc. Sys. (NIPS)*, 2016, pp. 974–982.

[29] K. Clarkson and D. Woodruff, "Low-rank psd approximation in input-sparsity time," in *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2017, pp. 2061–2072.

[30] M. Soltani and C. Hegde, "Fast low-rank matrix estimation without the condition number," *arXiv preprint arXiv:1712.03281*, 2017.

[31] Z. Yang, Z. Wang, H. Liu, Y. Eldar, and T. Zhang, "Sparse nonlinear regression: Parameter estimation and asymptotic inference," *J. Machine Learning Research*, 2015.

[32] M. Soltani and C. Hegde, "Fast algorithms for demixing sparse signals from nonlinear observations," *IEEE Trans. Sig. Proc.*, vol. 65, no. 16, pp. 4209–4222, Aug 2017.