Towards Provable Learning of Polynomial Neural Networks Using Low-Rank Matrix Estimation

Mohammadreza Soltani
Iowa State University
moltani@iastate.edu

Chinmay Hegde
Iowa State University
chinmay@iastate.edu

Abstract

We study the problem of (provably) learning the weights of a two-layer neural network with quadratic activations. In particular, we focus on the under-parametrized regime where the number of neurons in the hidden layer is (much) smaller than the dimension of the input. Our approach uses a lifting trick, which enables us to borrow algorithmic ideas from low-rank matrix estimation. In this context, we propose two novel, non-convex training algorithms which do not need any extra tuning parameters other than the number of hidden neurons. We support our algorithms with rigorous theoretical analysis, and show that the proposed algorithms enjoy linear convergence, fast running time per iteration, and near-optimal sample complexity. Finally, we complement our theoretical results with several numerical experiments.

1 Introduction

The re-emergence of neural networks (spurred by the advent of deep learning) has had a remarkable impact on various sub-domains of artificial intelligence (AI) including object recognition in images, natural language processing, and automated drug discovery, among many others. However, despite the successful empirical performance of neural networks for these AI tasks, provable methods for learning neural networks remain relatively mysterious. Indeed, training a network of even moderate size requires solving a very large-scale, highly non-convex optimization problem. In this paper, we (provably) resolve several algorithmic challenges that arise in the context of a special class of (shallow) neural networks by making connections to the better-studied problem of low-rank matrix estimation. Our hope is that a rigorous understanding of the fundamental limits of training shallow networks can be used as building blocks to obtain theoretical insights for more complex networks.

1.1 Setup

Consider a shallow (two-layer) neural network architecture, as illustrated in Figure 1. This network comprises $p$ input nodes, a single hidden layer with $r$ neurons with activation function $\sigma(z)$, first layer weights $\{w_j\}_{j=1}^r \subset \mathbb{R}^p$, and an output layer comprising of a single node and weights $\{\alpha_j\}_{j=1}^r \subset \mathbb{R}$. If $\sigma(z) = z^2$, then the above network is called a polynomial neural network (Livni et al., 2014). More precisely, the input-output relationship between an input, $x \in \mathbb{R}^p$, and the corresponding output, $y \in \mathbb{R}$, is given by:

$$\hat{y} = \sum_{j=1}^r \alpha_j \sigma(w_j^T x) = \sum_{j=1}^r \alpha_j (w_j, x)^2.$$
pairs \(\{(x_i, y_i)\}_{i=1}^m\). We do so by finding a set of weights \(\{\alpha_j, w_j\}_{j=1}^p\) that minimize the following empirical risk:

\[
\min_{W \in \mathbb{R}^{r \times r}, \alpha \in \mathbb{R}^r} F(W, \alpha) = \frac{1}{2m} \sum_{i=1}^m (y_i - \hat{y}_i)^2, \tag{1}
\]

where the rows of \(W\) and the entries of \(\alpha\) indicate the first-and second-layer weights, respectively. Numerous recent papers have explored (provable) algorithms to learn the weights of such a network under distributional assumptions on the input data (Livni et al., 2014; Lin & Ye, 2016; Janzamin et al., 2015; Tian, 2016; Zhong et al., 2016; Soltanolkotabi et al., 2017; Li & Yuan, 2017)\(^1\).

Clearly, the empirical risk defined in (1) is extremely nonconvex (involving fourth-powers of the entries of \(w_j\), coupled with the squares of \(\alpha_j\)). However, this can be circumvented using a clever lifting trick: if we define the matrix variable \(L_s = \sum_{j=1}^p \alpha_j w_j w_j^T\), then the input-output relationship becomes:

\[
\hat{y}_i = x_i^T L_s x_i = \langle x_i, x_i^T, L_s \rangle, \tag{2}
\]

where \(x_i \in \mathbb{R}^p\) denotes the \(i\)th training sample. Moreover, the variable \(L_s\) is a rank-\(r\) matrix of size \(p \times p\). Therefore, (1) can be viewed as an instance of learning a fixed (but unknown) rank-\(r\) symmetric matrix \(L_s \in \mathbb{R}^{p \times p}\) with \(r \ll p\), from a small number of rank-one linear observations given by \(A_i = x_i x_i^T\). While still non-convex, low-rank matrix estimation problems such as (2) are much better understood. Two specific instances in statistical learning include:

**Matrix sensing and matrix completion.** Reconstructing low-rank matrices from (noisy) linear measurements of the form \(y_i = \langle X_i, L_s \rangle\) impact several applications in control and system identification (Fazel, 2002), collaborative filtering (Candès & Recht, 2009; Recht et al., 2010), and imaging. The problem (2) specializes the matrix sensing problem to the case where the measurement vectors \(X_i\) are constrained to be themselves rank-one.

**Covariance sketching.** Estimating a high-dimensional covariance matrix, given a stream of independent samples \(\{s_i\}_{i=0}^\infty\), \(s_i \in \mathbb{R}^p\), involves maintaining the empirical estimate \(Q = E[s_i s_i^T]\), which can require quadratic \(O(p^2)\) space complexity. Alternatively, one can record a sequence of \(m \ll p^2\) linear sketches of each sample: \(z_i = x_i^T s_i\) for \(i = 1, \ldots, m\). At the conclusion of the stream, sketches corresponding to a given vector \(x_i\) are squared and aggregated to form a measurement: \(y_i = E[z_i^2] = E[(x_i^T s_i)^2] = x_i^T Q x_i\), which is nothing but a linear sketch of \(Q\) of the form (2). Again, several matrix estimation methods that “invert” such sketches exist; see Cai & Zhang (2015); Chen et al. (2015); Dasarathy et al. (2015).

### 1.2 Our contributions

In this paper, we make concrete algorithmic progress on solving low-rank matrix estimation problems of the form (2). In the context of learning polynomial neural networks, once we have estimated a rank-\(r\) symmetric matrix \(L_s\), we can always produce weights \(\{\alpha_j, w_j\}\) by an eigendecomposition of \(L_s\).

In general, a range of algorithms for solving (2) (or variants thereof) exist in the literature, and can be broadly classified into two categories:

(i) **convex** approaches, all of which involve enforcing the rank-\(r\) assumption in terms of a convex penalty term, such as the nuclear norm (Fazel, 2002; Recht et al., 2010; Cai & Zhang, 2015; Chen et al., 2015; Cai et al., 2010);

(ii) **nonconvex** approaches based on either alternating minimization (Zhong et al., 2015; Lin & Ye, 2016) or greedy approximation (Livni et al., 2014; Shalev-Shwartz et al., 2011).

Both types of approaches suffer from severe computational difficulties, particularly when the data dimension \(p\) is large. Even the most computationally efficient convex approaches require multiple invocations of singular value decomposition (SVD) of a (potentially) large \(p \times p\) matrix, which can incur cubic \(O(p^3)\) running time. Moreover, even the best available non-convex approaches require a very accurate initialization, and also require that the underlying matrix \(L_s\) is well-conditioned; if this is not the case, the running time of all available methods again inflates to \(O(p^3)\) or worse.

In this paper, we take a different approach, and show how to leverage recent results in low-rank approximation to our advantage (Musco & Musco, 2015; Hegde et al., 2016). Our algorithm is also non-convex; however, unlike all earlier works, our method does not require any full SVD calculations. Specifically, we demonstrate that a careful concatenation of **randomized, approximate** SVD methods, coupled with appropriately defined gradient steps, leads to efficient and accurate matrix estimation.

To our knowledge, this work constitutes the first **nearly-linear** time method for low-rank matrix estimation from rank-one observations. Consequently, in the context of learning two-layer polynomial networks, our method is the first to exhibit **nearly-linear** running time, is **sample-optimal** for fixed target rank \(r\), and is **unconditional** (i.e., it makes no assumptions on

\(^1\)While quadratic activation functions are rarely used in practice, stacking multiple such two-layer blocks can be used to simulate networks with higher-order polynomial and sigmoidal activations (Livni et al., 2014).
the condition number of $L_*$ or the weight matrix $W$). Numerical experiments reveal that our methods yield a very attractive tradeoff between sample complexity and running time for efficient matrix estimation.

1.3 Techniques

At a high level, our method can be viewed as a variant of the seminal algorithms proposed in Jain et al. (2010) and Jain et al. (2014), which essentially perform projected (or proximal) gradient descent with respect to the space of rank-$r$ matrices. However, since computing SVD in high dimensions can be a bottleneck, we cannot use this approach directly. To this end, we use the approximation-based matrix estimation framework proposed in Hegde et al. (2016). This work demonstrates how to carefully integrate approximate SVD methods into singular value projection (SVP)-based matrix estimation algorithms; in particular, algorithms that satisfy certain “head” and “tail” projection properties (explained below in Section 2) are sufficient to guarantee robust and fast convergence. Crucially, this framework removes the need to compute even a single SVD, as opposed to factorized methods which necessarily require one or multiple SVDs, together with stringent condition number assumptions.

However, a direct application of (projected) gradient descent does not succeed for matrix estimation problems obeying (2); two major obstacles arise:

Obstacle 1. It is well-known the measurement operator that maps $L_*$ to $y$ does not satisfy the so-called Restricted Isometry Property over rank-$r$ matrices (Cai & Zhang, 2015; Chen et al., 2015; Zhong et al., 2015); therefore, all statistical and algorithmic correctness arguments of Hegde et al. (2016) no longer apply.

Obstacle 2. The algebraic structure of the rank-one observations in (2) inflates the running time of computing even a simple gradient update to $O(p^3)$ (irrespective of the algorithmic cost of rank-$r$ projection, whether done using exact or approximate SVDs).

We resolve Obstacle 1 by carefully exploiting the rank-one structure of the observations. In particular, we develop a modification of the randomized block-Krylov SVD (or BK-SVD) algorithm of Musco & Musco (2015) to work for the case of certain “implicitly defined” matrices; specifically, we design a randomized SVD routine where the input is a linear operator that is constructed using vector-valued components. This modification, coupled with the tail-and head-projection arguments developed in Hegde et al. (2016), enables us to achieve a fast per-iteration computational complexity. In particular, our algorithm strictly improves over the (worst-case) per-iteration running time of all existing algorithms; see Table 1.

2 Main results

2.1 Preliminaries

Let us first introduce some notation. Throughout this paper, $\| \cdot \|_F$ and $\| \cdot \|_2$ denote the matrix Frobenius and spectral norm, respectively, and Tr$(\cdot)$ denotes matrix trace. The phrase “with high probability” indicates an event whose failure rate is exponentially small. We assume that the training data samples $(x, y)$ obey a generative model (2) written as:

$$y = \sum_{i=1}^{r} \alpha_i^* \sigma((w_i^*, x)) = x^T L_* x + e'$$

where $L_* \in \mathbb{R}^{p \times p}$ is the “ground-truth” matrix (with rank equal to $r$), and $e' \in \mathbb{R}$ is additive noise. Define $A : \mathbb{R}^{p \times p} \rightarrow \mathbb{R}^m$ such that:

$$A(L_*) = [x_1^T L_* x_1, x_2^T L_* x_2, \ldots, x_m^T L_* x_m]^T,$$

and each $x_i \overset{i.i.d.}{\sim} \mathcal{N}(0, I)$ is a normal random vector in $\mathbb{R}^p$ for $i = 1, \ldots, m$. The adjoint operator of $A$ is defined as $A^*(y) = \sum_{i=1}^{m} y_i x_i^T$. Also, noise vector is shown by $e \in \mathbb{R}^m$; throughout the paper (for the purpose of analysis) we assume that $e$ is zero-mean,
Table 1: Summary of our contributions and comparison with existing algorithms. Here, $\beta = \frac{a}{r}$ denotes the condition number of $L^*$.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Sample complexity ($m$)</th>
<th>Total Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convex</td>
<td>$O(pr)$</td>
<td>$O\left(\frac{p^2}{\sqrt{r}} \log\left(p \log(\frac{p}{\sqrt{r}})\right)\right)$</td>
</tr>
<tr>
<td>GECO</td>
<td>N/A</td>
<td>$O\left(\frac{p^2}{\sqrt{r}} \log\left(p \log(\frac{p}{\sqrt{r}})\right)\right)$</td>
</tr>
<tr>
<td>AltMin-LRROM</td>
<td>$O \left(\frac{pr}{2} \log^2(p) \beta^2 \log\left(\frac{1}{\epsilon}\right)\right)$</td>
<td>$O \left(\frac{mpr}{2} \log\left(\frac{1}{\epsilon}\right) + p^3\right)$</td>
</tr>
<tr>
<td>gFM</td>
<td>$O \left(\frac{pr^3}{2} \beta^2 \log\left(\frac{1}{\epsilon}\right)\right)$</td>
<td>$O \left(\frac{mpr^3}{2} \beta^2 \log\left(\frac{1}{\epsilon}\right) + p^3\right)$</td>
</tr>
<tr>
<td>EP-ROM [This paper]</td>
<td>$O \left(\frac{pr^2}{2} \log^2(p) \beta^2 \log\left(\frac{1}{\epsilon}\right)\right)$</td>
<td>$O \left(\frac{mpr^2}{2} \log\left(\frac{1}{\epsilon}\right) + p^3\right)$</td>
</tr>
<tr>
<td>AP-ROM [This paper]</td>
<td>$O \left(\frac{pr^3}{2} \log^2(p) \beta^2 \log\left(\frac{1}{\epsilon}\right)\right)$</td>
<td>$O \left(\frac{mpr^3}{2} \log\left(\frac{1}{\epsilon}\right) + p^3\right)$</td>
</tr>
</tbody>
</table>

subgaussian with i.i.d entries, and independent of $x_i$’s. The goal is to learn the rank-$r$ matrix parameter $L^*$ from as few samples as possible.

In our analysis, we require the operators $A$ and $A^*$ to satisfy the following regularity condition with respect to the set of low-rank matrices. We call this the Conditional Unbiased Restricted Isometry Property, abbreviated as CU-RIP($\rho$):

**Definition 1.** Consider fixed rank-$r$ matrices $L_1$ and $L_2$. Then, $A$ is said to satisfy CU-RIP($\rho$) if there exists $0 < \rho < 1$ such that

$$\left\|L_1 - L_2 - \frac{1}{2m} A^* A (L_1 - L_2) \right\|_2 \leq \rho \left\|L_1 - L_2\right\|_2.$$

Let $\mathcal{U}_r$ denote the set of all rank-$r$ matrix subspaces, i.e., subspaces of $\mathbb{R}^{p \times p}$ which are spanned by any $r$ atoms of the form $uv^T$ where $u, v \in \mathbb{R}^p$ are unit $\ell_2$-norm vectors. We use the idea of head and tail approximate projections with respect to $\mathcal{U}_r$ first proposed in Hegde et al. (2015), and instantiated in the context of low-rank approximation in Hegde et al. (2016).

**Definition 2** (Approximate tail projection). $T : \mathbb{R}^{p \times p} \rightarrow \mathcal{U}_r$ is a $\epsilon$-approximate tail projection algorithm if for all $L \in \mathbb{R}^{p \times p}$, $T$ returns a subspace $V = T(L)$ that satisfies:

$$\|L - P_W L\|_F \leq (1 + \epsilon) \|L - L_r\|_F, \quad \text{where } L_r \text{ is the optimal rank-$r$ approximation of } L.$$

**Definition 3** (Approximate head projection). $\mathcal{H} : \mathbb{R}^{p \times p} \rightarrow \mathcal{U}_r$ is a $\epsilon$-approximate head projection if for all $L \in \mathbb{R}^{p \times p}$, the returned subspace $V = \mathcal{H}(L)$ satisfies:

$$\|P_V L\|_F \geq (1 - \epsilon) \|L_r\|_F, \quad \text{where } L_r \text{ is the optimal rank-$r$ approximation of } L.$$

### 2.2 Algorithms and theoretical results

We now propose methods to estimate $L^*$, given knowledge of $\{x_i, y_i\}_{i=1}^m$. Our first method is somewhat computationally inefficient, but achieves very good sample complexity and serves to illustrate the overall algorithmic approach. Consider the non-convex, constrained risk minimization problem:

$$\min_{L \in \mathbb{R}^{p \times p}} F(L) = \frac{1}{2m} \sum_{i=1}^m (y_i - x_i^T L x_i)^2 \tag{4}$$

s.t. $\text{rank}(L) \leq r$.

To solve this problem, we first propose an algorithm that we call Exact Projections for Rank-One Matrix estimation, or EP-ROM, described in pseudocode form in Algorithm 1.

We now analyze this algorithm. First, we provide a theoretical result which establishes statistical and optimization convergence rates of EP-ROM. More precisely, we derive an upper bound on the estimation error (measured using the spectral norm) of recovering $L^*$. Due to space constraints, we defer all the proofs to the appendix.

**Theorem 4** (Linear convergence of EP-ROM). Consider the sequence of iterates $(L_t)$ obtained in EP-ROM. Assume that in each iteration the linear operator $A$ satisfies CU-RIP($\rho$) for some $0 < \rho < \frac{1}{2}$, then EP-ROM outputs a sequence of estimates $L_t$ such that:

$$\|L_{t+1} - L^*\|_2 \leq q \|L_t - L^*\|_2 + \frac{1}{2m} (1^T e + \|A^* e\|_2), \tag{5}$$

where $0 < q < 1$.

The contraction factor, $q$, in Equation (5) can be arbitrary small if we choose $m$ sufficiently large, and we elaborate it in Theorem (6). The second and third term in (5) represent the statistical error rate. In the next Theorem, we show that these error terms can be suitably bounded. Furthermore, Theorem 4 implies (via induction) that EP-ROM exhibits linear convergence; please see Corollary 7.

**Theorem 5** (Bounding the statistical error). Consider the generative model (3) with zero-mean subgaussian noise $e \in \mathbb{R}^m$ with i.i.d. entries (and independent of the $x_i$’s) such that $\tau = \max_{1 \leq i \leq m} \|e_i\|_{\psi_2}$ (Here,
Algorithm 1 EP-ROM

Inputs: $y$, number of iterations $K$, independent data samples $\{x_1^t, x_2^t, \ldots, x_m^t\}$ for $t = 1, \ldots, K$, rank $r$
Outputs: Estimates $\hat{L}$
Initialization: $L_0 \leftarrow 0$, $t \leftarrow 0$
Calculate: $y = \frac{1}{m} \sum_{i=1}^{m} y_i$
while $t \leq K$ do
  $L_{t+1} = P_r \left( L_t - \frac{1}{2m} \sum_{i=1}^{m} ((x_i^t)^T L_t x_i^t - y_i) x_i^t (x_i^t)^T - (\frac{1}{2m} 1^T A(L_t) - \frac{1}{2} y^T) I \right)$
  $t \leftarrow t + 1$
end while
Return: $\hat{L} = L_K$

$\| \cdot \|_{\psi_2}$ denotes the $\psi_2$-norm; see Definition 11 in the appendix. Then, with probability at least $1 - \gamma$, we have:

$$\frac{1}{m} \| 1^T e \| + \frac{1}{m} A^* e \|_2 \leq C'' \sqrt{\frac{p \log^2 p}{m} \log \frac{p}{\gamma}}.$$  \hspace{1cm} (6)

where $C'' > 0$ is constant which depends on $\tau$.

To establish linear convergence of EP-ROM, we assume that the CU-RIP holds at each iteration. The following theorem certifies this assumption.

Theorem 6 (Verifying CU-RIP). At any iteration $t$ of EP-ROM, with probability at least $1 - \xi$, CU-RIP$(\rho)$ is satisfied with parameter $\rho < \frac{1}{2}$ provided that $m = \mathcal{O} \left( \frac{1}{\tau^2} p r^2 \log^3 p \log (\frac{1}{\xi}) \right)$ for some $\delta > 0$.

Integrating the above results, together with the assumption of availability of a batch of $m$ independent samples in each iteration, we obtain the following corollary formally establishing linear convergence. We acknowledge that this assumption of “fresh samples” is somewhat unrealistic and is an artifact of our proof techniques; nonetheless, it is a standard mechanism for proofs for non-convex low-rank matrix estimation (Hardt, 2014; Zhong et al., 2016)

Corollary 7. After $K$ iterations, with high probability the output of EP-ROM satisfies:

$$\| L_K - L^* \|_2 \leq q^K \| L_1 \|_2 + \frac{C''}{1 - q} \sqrt{\frac{p \log^3 p}{m}}.$$  \hspace{1cm} (7)

where $C''$ is given in (6). Thus, under all the assumptions in theorem 4, to achieve $\epsilon$-accuracy for estimation of $L^*$ in terms of the spectral norm, EP-ROM needs $K = \mathcal{O}(\log (\frac{\| L_1 \|}{\epsilon}))$ iterations. Based on Theorems 5, 6, and Corollary 7, the sample complexity of EP-ROM scales as $m = \mathcal{O}(pr^2 \log^3 p \log (\frac{1}{\epsilon}))$.

While EP-ROM exhibits linear convergence, the per iteration complexity is still high since it requires projection onto the space of rank-$r$ matrices, which necessitates the application of SVD. In the absence of any spectral assumptions on the input to the SVD, the per-iteration running time of EP-ROM can be cubic, which can be prohibitive. Overall, we obtain a running time of $\mathcal{O}(p^3 r^2)$ in order to achieve $\epsilon$-accuracy (please see Section 5.3 in the appendix for a longer discussion).

To reduce the running time, one can instead replace a standard SVD routine with approximation heuristics such as Lanczos iterations (Lanczos, 1950); however, these may not result in algorithms with provable convergence guarantees. Instead, following Hegde et al. (2016), we can use a pair of inaccurate rank-$r$ projections (in particular, tail- and head-approximate projection operators). Based on this idea, we propose our second algorithm that we call Approximate Projection for Rank One Matrix estimation, or AP-ROM. We display the pseudocode of AP-ROM in Algorithm 2.

The specific choice of approximate SVD algorithms that simulate the operators $\mathcal{T}(\cdot)$ and $\mathcal{H}(\cdot)$ is flexible. We note that tail-approximate projections have been widely studied in the numerical linear algebra literature (Clarkson & Woodruff, 2013; Mahoney & Drineas, 2009; Rokhlin et al., 2009); however, head-approximation projection methods are less well-known. In our method, we use the randomized Block Krylov SVD (BK-SVD) method proposed by Musco & Musco (2015), which has been shown to satisfy both types of approximation guarantees (Hegde et al., 2016). One can alternatively use LazySVD, recently proposed by Allen-Zhu & Li (2016), which also satisfies both guarantees. The nice feature of these methods is that their running time is independent of the spectral gap of the matrix. We leverage this property to show asymptotic improvements over other fast SVD methods (such as the power method).

We briefly discuss the BK-SVD algorithm. In particular, BK-SVD takes an input matrix with size $p \times p$ with rank $r$ and returns a $r$-dimensional subspace which approximates the top right $r$ singular vectors of the input. Mathematically, if $A \in \mathbb{R}^{p \times p}$ is the input, $A_r$ is the best rank-$r$ approximation to it, and $Z$ is a basis matrix that spans the subspace returned by BK-SVD, then the projection of $A$ into $Z$, $B = ZZ^T A$ satisfies
the following relations:

\[
\|A - B\|_F \leq (1 + \varepsilon)\|A - A_r\|_F, \\
|u_i^T A A^T u_i - z_i A A^T z_i| \leq \varepsilon \sigma_i^2 + 1,
\]

where \( \varepsilon > 0 \) is defined as the tail and head projection approximate constant, and \( u_i \) denotes the \( i \)th right eigenvector of \( A \). In Appendix-B of Hegde et al. (2016), it has been shown that the per-vector guarantee can be used to prove the approximate head projection property, i.e., \( \|B\|_F \geq (1 - \varepsilon)\|A_r\|_F \).

We now establish that AP-ROM also exhibits linear convergence, while obeying similar statistical properties as EP-ROM. We have the following results:

**Theorem 8** (Convergence of AP-ROM). Consider the sequence of iterates \( (L_t) \) obtained in AP-ROM. Assume that in each iteration \( t \), \( A \) satisfies CU-RIP\((\rho')\) for some \( 0 < \rho' < 1 \), then AP-ROM outputs a sequence of estimates \( L_t \) such that:

\[
\|L_{t+1} - L_*\|_F \leq q_1^t \|L_t - L_*\|_F + q_2^t \left( \|1 - \varepsilon\| + \|A^*\|_2 \right),
\]

where \( q_1^t = (2 + \varepsilon)(\rho' + \sqrt{1 - \phi^2}), \quad q_2^t = \frac{\sqrt{2} \varepsilon}{2m} \left( 2 - \varepsilon + \phi (2\varepsilon - \rho' + 2\varepsilon) \right), \) and \( \phi = (1 - \varepsilon)(1 - \rho') - \rho'. \)

Similar to Theorem 6, we can show that CU-RIP is satisfied in each iteration of AP-ROM with probability at least \( 1 - \xi \), provided that \( m = \mathcal{O}\left( \frac{\varepsilon \rho^3}{\xi} \log^3 \left( \frac{p}{\varepsilon^2} \right) \right) \).

Hence, we require a factor-\( r \) increase compared to before. Overall, we have the following result:

**Corollary 9.** The output of AP-ROM satisfies the following after \( K \) iterations with high probability:

\[
\|L_K - L_*\|_F \leq (q_1^*)^K \|L_*\|_F + \frac{C r^2 q_2^*}{1 - q_1^*} \sqrt{\frac{p r^3 \log^3 p \log \left( \frac{p}{\varepsilon^2} \right)}{m}}.
\]

where \( q_1^* \) and \( q_2^* \) have been defined in Theorem 8.

Hence, under the assumptions in Theorem 8, in order to achieve \( \varepsilon \)-accuracy in the estimation of \( L_* \) in terms of Frobenius norm, AP-ROM requires \( K = \mathcal{O}(\log((L_*/\varepsilon)^2)) \) iterations. From Theorem 8 and Corollary 9, we observe that the sample-complexity of AP-ROM (i.e., the number of samples \( m \) to achieve a given accuracy) slightly increases as \( m = \mathcal{O}(pr^3 \log^2 p \log(\frac{1}{\varepsilon})) \).

### 2.3 Improving running time

The above analysis of AP-ROM shows that instead of using exact rank-\( r \) projections (as in EP-ROM), one can use instead tail and head approximate projection which is implemented by the BK-SVD method of Musco & Musco (2015). The running time for this method is given by \( \mathcal{O}(pr^2 r) \) if \( r \ll p \). While the running time of the projection step is gap-independent, the calculation of the gradient (i.e., the input to the head projection method \( \mathcal{H} \)) is itself the major bottleneck. In essence, this is related to the calculation of the adjoint operator, \( A^*(d) = \sum_{i=1}^m d^{(i)} x_i x_i^T \), which requires \( \mathcal{O}(p^2) \) operations for each sample. Coupled with the sample-complexity of \( m = \mathcal{O}(pr^3) \), this means that the running time per-iteration scaled as \( \mathcal{O}(pr^3) \), which overshadows any gains achieved during the projection step (please see Section 5.3 in the appendix).

To address this challenge, we propose a modified version of BK-SVD for head approximate projection which uses the special rank-one structures involved in the calculation of the gradients. We call this method **Modified BK-SVD**, or MBK-SVD. The basic idea is to implicitly evaluate each Krylov-subspace iteration within BK-SVD, and avoid any explicit calculation of the adjoint operator \( A^* \) applied to the current estimate. Due to space constraints, the pseudocode as well as the running time analysis of MBK-SVD is deferred to the appendix. We have:

**Theorem 10.** AP-ROM (with modified BK-SVD) runs in time \( K = \mathcal{O}(pr^2 r \log^2 (\frac{1}{\varepsilon}) \text{polylog}(p)) \).

### 3 Prior art

Due to space constraints, we only provide here a brief (and incomplete) review of related work, and describe
how our method differs from earlier techniques.

Problems involving low-rank matrix estimation have received significant attention from the machine learning community over the last few years; see Davenport & Romberg (2016) for a recent survey. In early works for matrix estimation, the observation operator $\mathcal{A}$ is assumed to be parametrized by $m$ independent full-rank $p \times p$ matrices that satisfy certain restricted isometry conditions (Recht et al., 2010; Liu, 2011). In this setup, it has been established that $m = \mathcal{O}(pr)$ observations are sufficient to recover an unknown rank-$r$ matrix $L_*$ in (3) (Candes & Plan, 2011), and this scaling of sample complexity is statistically optimal.

In the context of provable methods for learning neural networks, two-layer networks have received special attention. For instance, Livni et al. (2014) has considered a two-layer network with quadratic activation function (identical to the model proposed above), and proposed a greedy, improper learning algorithm: in each iteration, the algorithms adds one hidden neuron to the network until the risk falls below a threshold. While this algorithm is guaranteed to converge, its convergence rate is sublinear.

Recently, in Zhong et al. (2016), the authors have proposed a linearly convergent algorithm for learning two-layer networks for several classes of activation functions. They also derived an upper bound on the sample complexity of network learning which is linear in $p$, and depends polynomially on $r$ and other spectral properties of the ground-truth (planted) weights.

However, their theory does not provide convergence guarantees for quadratic functions; this paper closes this gap. Note that our focus here is not the estimation of the weights $\{a_j, w_j\}$ themselves, but rather, any network that gives the same input-relationship. As a result, our guarantees are stated in terms of the low-rank matrix $L_*$.

Other works have also studied similar two-layer setups, including Janzamin et al. (2015); Tian (2016); Soltanolkotabi et al. (2017); Li & Yuan (2017). In contrast with these results, our framework does not assume the over-parameterized setting where the number of hidden neurons $r$ is greater than $p$. In addition, we explicitly derive a sample complexity that is linear in $p$, as well as demonstrate linear time convergence. Also, observe that if we let $L_*$ to be rank-1, then Problem (2) is known as generalized phase retrieval for which several excellent algorithms are known (Candes et al., 2013, 2015; Netrapalli et al., 2013). However, our problem is more challenging as it allows $L_*$ to have arbitrary rank-$r$.

We now briefly contrast our method with other algorithmic techniques for low-rank matrix estimation. Broadly, two classes of such techniques exist. The first class of matrix estimation techniques can be categorized as approaches based on convex relaxation (Chen et al., 2015; Cai & Zhang, 2015; Kueng et al., 2017; Candes et al., 2013). For instance, the authors in Chen et al. (2015); Cai & Zhang (2015) demonstrate that the observation operator $\mathcal{A}$ satisfies a specialized mixed-norm isometry condition called the RIP-$\ell_2/\ell_1$. Further, they show that the sample complexity of matrix estimation using rank-one projections matches the optimal rate $\mathcal{O}(pr)$. However, these methods advocate using either semidefinite programming (SDP) or proximal sub-gradient algorithms (Boyd & Vandenberghe, 2004; Goldstein et al., 2014, 2015), both of which are too slow for very high-dimensional problems.

The second class of techniques can be categorized as non-convex approaches, which are all based on a factorization-based approach initially advocated by Burer & Monteiro (2003). Here, the underlying low-rank matrix variable is factorized as $L_* = UV^T$, where $U, V \in \mathbb{R}^{p \times r}$ (Zheng & Lafferty, 2015; Tu et al., 2016). In the Altmin-LRROM method proposed by Zhong et al. (2015), $U$ and $V$ are updated in alternative fashion. However, the setup in Zhong et al. (2015) is dif-
ferent from this paper, as it uses an asymmetric ob-
ervation model, in which observation $y_i$ is given by
$y_i = x_i^T L x_i z_i$, with $x_i$ and $z_i$ being independent ran-
dom vectors. Our goal is to analyze the more challeng-
ing case where the observation operator $A$ is symmet-
ric and defined according (3). In a subsequent work
(called the generalized factorization machine) by Lin et al. (2017), $U$ and $V$ are updated based on the con-
struction of certain sequences of moment estimators.

Both the approaches of (Zhong et al., 2015) and (Lin et al., 2017) require a spectral initialization which in-
volves running a rank-$r$ SVD on a given $p \times p$ ma-
trix, and therefore the running time heavily depends
on the condition number (i.e., the ratio of the maxi-
mum and the minimum nonzero singular values) of $L_*$. 
To our knowledge, only three works in the matrix es-
timation literature require no full SVDs (Bhojanapalli et al., 2016; Hegde et al., 2016; Ge et al., 2016). How-
ever, both Bhojanapalli et al. (2016) and Hegde et al. (2016) assume that the restricted isometry property is satis-
fied, which is not applicable in our setting. More-
ever, Ge et al. (2016) makes stringent assumptions on the condition number, as well as the coherence, of the unknown matrix.

Finally, we mention that a matrix estimation scheme
using approximate SVDs (based on Frank-Wolfe type
greedy approximation) has been proposed for learn-
ing polynomial neural networks (Shalev-Shwartz et al.,
2011; Livni et al., 2014). Moreover, this approach has
been shown to compare favorably to typical neural
network learning methods (such as stochastic grad-
dient descent). However, the rate of convergence is sub-
linear, and they provide no sample-complexity guaran-
tees. Indeed, the main motivating factor of our paper
was to accelerate the running time of such greedy ap-
proximation techniques. We complete this line of work
by providing (a) rigorous statistical analysis that pre-
cisely establishes upper bounds on the number of sam-
ple required for learning such networks, and (b) an
algorithm that provably exhibits linear convergence, as well as nearly-linear per iteration running time.

4 Experimental results and discussion

We illustrate some experiments to support our pro-
posed algorithms. We compare EP-ROM and AP-
ROM with convex (nuclear norm) minimization as well
as the gFM algorithm of Lin & Ye (2016). To solve
the nuclear norm minimization, we use FASTA (Gold-
stein et al., 2014, 2015) which efficiently implements an
accelerated proximal sub-gradient method. For AP-
ROM, we consider our proposed modified BK-SVD
method (MBK-SVD). In addition, SVD and SVDS de-
note the projection step being used in EP-ROM. In
all the experiments, we generate a low-rank matrix,
$L_* = UU^T$, such that $U \in \mathbb{R}^{p \times r}$ with $r = 5$ where
the entries of $U$ is randomly chosen according to the
standard normal distribution.

Figures 2(a) and 2(b) show the phase transition of suc-
cessful estimation as well as the evolution of the ob-
jective function, $\frac{1}{2} \|y - A(L_*)\|_2^2$ versus the iteration
count $t$ for five algorithms. In figure 2(a), we have
used 50 Monte Carlo trials and the phase transition
plot is generated based on the empirical probability
of success; here, success is when the relative error be-
tween $\hat{L}$ (the estimate of $L_*$) and the ground truth
$L_*$ (measured in terms of spectral norm) is less than
0.05. For solving convex nuclear norm minimization
using FASTA, we set the Lagrangian parameter, $\mu$ i.e.,
$\mu\|L\|_\star + \frac{1}{2}\|y - AL\|_2^2$ via a grid search. In Figure 2(a),
there is no additive noise. As we can see in this Figure,
the phase transition for the convex method is slightly
better than those for non-convex algorithms, which is
consistent with known theoretical results. However,
the convex method is improper, i.e., the rank of $L$ is
much higher than the target rank. In Figure 2(b) we
consider an additive standard normal noise with stan-
dard deviation equal to 0.1, and average over 10 Monte
Carlo trials. As illustrated in this plot, all non-convex
algorithm have much better performance in decreasing
the objective function compared to convex method.

Finally, in Figure 2(c), we compare the algorithms in
the high-dimensional regime where $p = 1000$, $m =
75000$, and $r = 5$ in terms of running time. We let
all the algorithms run 15 iterations, and then compute
the CPU time in seconds for each of them. The $y$-axis
denotes the logarithm of relative error in spectral norm
and we report averages over 10 Monte Carlo trials.
As we can see, convex methods are the slowest (as
expected); the non-convex methods are comparable to
each other, while MBK-SVD is the fastest. This plot
verifies that our modified head approximate projection
routine is faster than other non-convex methods, which
makes it a promising approach for high-dimensional
matrix estimation applications.

Discussion. It seems plausible that the matrix-based
techniques of this paper can be extended to learn net-
works with similar polynomial-like activation functions
(such as the squared ReLU). Moreover, similar algo-
rithms can be plausibly used to train multi-layer net-
works using a greedy (layer-by-layer) learning strategy.
Finally, it will be interesting to integrate our methods
with practical approaches such as stochastic gradient
descent (SGD).

Acknowledgement. This work was supported in part
by NSF grant CCF-1566281.
References


Towards Provable Learning of Polynomial Neural Networks


