A General Algorithm for Solving Rank-one Matrix Sensing

Lianke Qin
UC Santa Barbara

Zhao Song
Adobe Research

Ruizhe Zhang
Simons Institute, UC Berkeley

Abstract

Matrix sensing has many real-world applications in science and engineering, such as system control, distance embedding, and computer vision. The goal of matrix sensing is to recover a matrix $A_\star \in \mathbb{R}^{n \times n}$, based on a sequence of measurements $(u_i, b_i) \in \mathbb{R}^n \times \mathbb{R}$ such that $u_i^\top A_\star u_i = b_i$. Previous work (Zhong et al., 2015) focused on the scenario where matrix $A_\star$ has a small rank, e.g. rank-k. Their analysis heavily relies on the RIP assumption, making it unclear how to generalize to high-rank matrices. In this paper, we relax that rank-k assumption and solve a much more general matrix sensing problem. Given an accuracy parameter $\delta \in (0, 1)$, we can compute $A \in \mathbb{R}^{n \times n}$ in $\widetilde{O}(m^{3/2} n^2 \delta^{-1})$, such that $|u_i^\top A u_i - b_i| \leq \delta$ for all $i \in [m]$. We design an efficient algorithm with provable convergence guarantees using stochastic gradient descent for this problem.

1 INTRODUCTION

Matrix sensing is a generalization of the famous compressed sensing problem. Informally, the goal of matrix sensing is to reconstruct a matrix $A \in \mathbb{R}^{n \times n}$ using a small number of quadratic measurements (i.e., $u^\top A u$). It has many real-world applications, including image processing (Candès et al., 2011; Waters et al., 2011), quantum computing (Aaronson, 2007; Flammia et al., 2012; Kalev et al., 2015), systems (Liu and Vandenberghe, 2010) and sensor localization (Javanmard and Montanari, 2013) problems. For this problem, there are two important *theoretical* questions:

- **Q1. Compression:** How to design the sensing vectors $u \in \mathbb{R}^n$ so that the matrix can be recovered with a small number of measurements?
- **Q2. Reconstruction:** How fast can we recover the matrix given the measurements?

the study of the rank-one matrix sensing problem was initialized by (Zhong et al., 2015) , where the ground-truth matrix $A_\star$ has only rank-k, and the measurements are of the form $u_i^\top A_\star u_i$. They want to know the smallest number of measurements m to recover the matrix $A_\star$. In our setting, we assume m is a fixed input parameter and we're not allowed to choose. We show that for any m and n, how to design a faster algorithm for solving an optimization problem which is finding $A \approx A_\star$. Thus, in some sense, previous work (Zhong et al., 2015; Deng et al., 2023) mainly focuses on problem **Q1** with a low-rank assumption on $A_\star$. Our work is focusing on **Q2** without the low-rank assumption.

We observe that in many applications, the ground-truth matrix $A_\star$ does not need to be recovered *exactly* (i.e., $\|A - A_\star\| \leq n^{-c}$). For example, for distance embedding, we would like to learn an embedding matrix between all the data points in a high-dimensional space. The embedding matrix is then used for calculating data points' pairwise distances for a higher-level machine learning algorithm, such as k-nearest neighbor clustering. As long as we can recover a good approximation of the embedding matrix, the clustering algorithm can deliver the desired results. As we relax the accuracy constraints of the matrix sensing, we have the opportunity to speed up the matrix sensing time.

We formulate our problem in the following way:

Problem 1.1 (Approximate matrix sensing)**.** *Given a ground-truth positive definite matrix $A_\star \in \mathbb{R}^{n \times n}$ and m samples $(u_i, b_i) \in \mathbb{R}^n \times \mathbb{R}$ such that $u_i^\top A_\star u_i = b_i$. Let $R = \max_{i \in [m]} |b_i|$. For any accuracy parameter $\delta \in (0, 1)$, find a matrix $A \in \mathbb{R}^{n \times n}$ such that*

$$(u_i^\top A u_i - u_i A_\star u_i)^2 \leq \delta, \quad \forall i \in [m] \tag{1}$$

or

$$(1 - \delta) A_\star \preceq A \preceq (1 + \delta) A_\star. \tag{2}$$

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

We make a few remarks about Problem 1.1. First, our formulation doesn't require the matrix $A_\star$ to be low-rank as literature (Zhong et al., 2015; Deng et al., 2023). Second, we need the measurement vectors u_i to be "approximately orthogonal" (i.e., $|u_i^\top u_j|$ are small), while (Zhong et al., 2015; Deng et al., 2023) make much stronger assumptions for exact reconstruction. Third, the *measure approximation* guarantee (Eq. (1)) does not imply the *spectral approximation* guarantee (Eq. (2)). We mainly focus on achieving the first guarantee and discuss the second one in the appendix.

This problem is interesting for two reasons. First, speeding up matrix sensing is salient for a wide range of applications, where exact matrix recovery is not required. Second, we would like to understand the fundamental tradeoff between the accuracy constraint ϵ and the running time. This tradeoff can give us insights into the fundamental computation complexity for matrix sensing.

This paper makes the following contributions:

- We design a potential function to measure the distance between the approximate solution and the ground-truth matrix.
- Based on the potential function, we show that gradient descent can efficiently find an approximate solution to the matrix sensing problem. We also prove the convergence rate of our algorithm.
- Furthermore, we show that cost-per-iteration can be improved by using stochastic gradient descent with a provable convergence guarantee, which is proved by generalizing the potential function to a randomized potential function.

Technically, our potential function applies a cosh function to each "training loss" (i.e., $u_i^\top A u_i - b_i$), which is inspired by the potential function for linear programming (Cohen et al., 2019). We prove that the potential is decreasing for each iteration of gradient descent, and a small potential implies a good approximation. In this way, we can upper bound the number of iterations needed for the gradient descent algorithm.

To reduce the cost-per-iteration, we follow the idea of stochastic gradient descent and evaluate the gradient of potential function on a subset of measurements. However, we still need to know the full gradient's norm for normalization, which is a function of the training losses. It is too slow to naively compute each training loss. Instead, we use the idea of maintenance (Cohen et al., 2019; Lee et al., 2019; Brand, 2020; Brand et al., 2020; Jiang et al., 2021; Huang et al., 2022; Song et al., 2021a,b; Hu et al., 2022; Qin et al., 2023) and show that the training loss at the $(t+1)$-th iteration (i.e., $u_i^\top A_{t+1} u_i - b_i$) can be very efficiently obtained from those at the t-th iteration (i.e., $u_i^\top A_t u_i - b_i$). Therefore, we first preprocess the initial full gradient's norm, and in the following iterations, we can update this quantity based on the previous iteration's result.

We state our main result as follows:

Theorem 1.2 (Informal of Theorem 6.1)**.** *Given m measurements of matrix sensing problems, there is an algorithm that outputs a $n \times n$ matrix A in $\widetilde{O}(m^{3/2} n^2 R \delta^{-1})$ time such that $|u_i^\top A u_i - b_i| \leq \delta$, $\forall i \in [m]$.*

2 RELATED WORK

Linear Progamming Linear programming is one of foundations of the algorithm design and convex optimization. many problems can be modeled as linear programs to take advantage of fast algorithms. There are many works in accelerating linear programming runtime complexity (Lee and Sidford, 2014, 2015; Cohen et al., 2019; Lee et al., 2019; Brand, 2020; Brand et al., 2020; Song and Yu, 2021; Dong et al., 2021; Jiang et al., 2021; Gu and Song, 2022).

Semi-definite Programming Semidefinite programming optimizes a linear objective function over the intersection of the positive semidefinite cone with an affine space. Semidefinite programming is a fundamental class of optimization problems and many problems in machine learning, and theoretical computer science can be modeled or approximated as semidefinite programming problems. There are many studies to speedup the running time of Semidefinite programming (Nesterov and Nemirovskii, 1994; Helmberg et al., 1996; Lee et al., 2015; Jiang et al., 2020b,a; Huang et al., 2022; Gu and Song, 2022).

Matrix Sensing Matrix sensing (Lee and Bresler, 2009; Recht et al., 2010; Jain et al., 2010; Zhong et al., 2015; Deng et al., 2023) is a generalization of the popular compressive sensing problem for the sparse vectors and has applications in several domains such as control, vision etc. a set of universal Pauli measurements, used in quantum state tomography, have been shown to satisfy the RIP condition (Liu, 2011). These measurement operators are Kronecker products of 2×2 matrices, thus, they have appealing computation and memory efficiency. Rank-one measurement using nuclear norm minimization is also used in other work (Cai and Zhang, 2015; Kueng et al., 2017). There is also previous work working on low-rank matrix sensing to reconstruct a matrix exactly using a small number of linear measurements. ProcrustesFlow (Tu et al., 2016) designs an algorithm to recover a low-rank matrix from linear mea-

surements. There are other low-rank matrix recovering algorithms based on non-convex optimizations (Wang et al., 2017; Li et al., 2019).

3 PRELIMINARY

Notations. For a positive integer, we use $[n]$ to denote set $\{1, 2, \cdots, n\}$. We use $\cosh(x) = \frac{1}{2}(e^x + e^{-x})$ and $\sinh(x) = \frac{1}{2}(e^x - e^{-x})$. For a square matrix, we use $\mathrm{tr}[A]$ to denote the trace of A. An $n \times n$ symmetric real matrix A is said to be positive-definite if $x^\top A x > 0$ for all non-zero $x \in \mathbb{R}^n$. An $n \times n$ symmetric real matrix A is said to be positive-semidefinite if $x^\top A x \geq 0$ for all non-zero $x \in \mathbb{R}^n$. For any function f, we use $\widetilde{O}(f) = f \cdot \mathrm{poly}(\log f)$.

3.1 Matrix Hyperbolic Functions

Definition 3.1 (Matrix function)**.** *Let $f : \mathbb{R} \to \mathbb{R}$ be a real function and $A \in \mathbb{R}^{n \times n}$ be a real symmetric function with eigendecomposition*

$$A = Q \Lambda Q^{-1}$$

where $\Lambda \in \mathbb{R}^{n \times n}$ is a diagonal matrix. Then, we have

$$f(A) := Q f(\Lambda) Q^{-1},$$

where $f(\Lambda) \in \mathbb{R}^{n \times n}$ is the matrix obtained by applying f to each diagonal entry of Λ.

We have the following lemma to bound $\cosh(A)$ and delay the proof to Appendix A.3.

Lemma 3.2. *Let A be a real symmetric matrix, then we have*

$$\|\cosh(A)\| = \cosh(\|A\|) \leq \mathrm{tr}[\cosh(A)].$$

We also have

$$\|A\| \leq 1 + \log(\mathrm{tr}[\cosh(A)]).$$

3.2 Properties of sinh and cosh

We have the following lemma for properties of sinh and cosh.

Lemma 3.3 (Scalar version)**.** *Given a list of numbers $x_1, \cdots x_n$, we have*

- $(\sum_{i=1}^n \cosh^2(x_i))^{1/2} \leq \sqrt{n} + (\sum_{i=1}^n \sinh^2(x_i))^{1/2}$,
- $(\sum_{i=1}^n \sinh^2(x_i))^{1/2} \geq \frac{1}{\sqrt{n}}(\sum_{i=1}^n \cosh(x_i) - n)$.

We also have a lemma for the matrix version.

Lemma 3.4 (Matrix version)**.** *For any real symmetric matrix A, we have*

- $(\mathrm{tr}[\cosh^2(A)])^{1/2} \leq \sqrt{n} + \mathrm{tr}[\sinh^2(A)]^{1/2}$,
- $(\mathrm{tr}[\sinh^2(A)])^{1/2} \geq \frac{1}{\sqrt{n}}(\mathrm{tr}[\cosh(A)] - n)$.

We delay all the related proofs to Section A.

4 TECHNIQUE OVERVIEW

We first analyze the convergence guarantee of our matrix sensing algorithm based on gradient descent and improve its time complexity with stochastic gradient descent under the assumption where $\{u_i\}_{i \in [m]}$ are orthogonal vectors. We then analyze the convergence guarantee of our matrix sensing algorithm under a more general assumption where $\{u_i\}_{i \in [m]}$ are non-orthogonal vectors and $|u_i^\top u_j| \leq \rho$.

Gradient descent. We begin from the case where $\{u_i\}_{i \in [m]}$ are orthogonal vectors in $\mathbb{R}^n$. Hyperbolic functions such as cosh is very popular in the area of optimization (Cohen et al., 2019; Lee et al., 2019; Brand, 2020; Song and Yu, 2021; Jiang et al., 2021; Dong et al., 2021; Gu and Song, 2022; Li et al., 2023). Inspire by that, we consider the following entry-wise potential function:

$$\Phi_\lambda(A) := \sum_{i=1}^m \cosh(\lambda(u_i^\top A u_i - b_i))$$

and analyze its progress during the gradient descent according to the update formula defined in Eq. (4) for each iteration. We split the gradient of the potential function into diagonal and off-diagonal terms. We can upper bound the diagonal term and prove that the off-diagonal term is zero. Combining the two terms together, we can upper bound the progress of update per iteration in Lemma 5.3 by:

$$\Phi_\lambda(A_{t+1}) \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}.$$

By accumulating the progress of update for the entry-wise potential function over $T = \widetilde{\Omega}(\sqrt{m} R \delta^{-1})$ iterations, we have $\Phi(A_{T+1}) \leq O(m)$. This implies that our Algorithm 1 can output a matrix $A_T \in \mathbb{R}^{n \times n}$ satisfying guarantee in Eq. (23), and the corresponding time complexity is $O(mn^2)$.

We then analyze the gradient descent under the assumption where $\{u_i\}_{i \in [m]}$ are non-orthogonal vectors in $\mathbb{R}^n$, $|u_i^\top u_j| \leq \rho$ and $\rho \leq \frac{1}{10m}$. We can upper bound the diagonal entries and off-diagonal entries respectively and obtain the same progress of update per iteration in Lemma D.1. Accumulating in $T = \widetilde{\Omega}(\sqrt{m} R \delta^{-1})$ iterations, we can prove the approximation guarantee of the output matrix of our matrix sensing algorithm.

Stochastic gradient descent. To further improve the time cost per iteration of our approximate matrix sensing, by uniformly sampling a subset $\mathcal{B} \subset [m]$ of size B, we compute the gradient of the stochastic potential function:

$$\nabla \Phi_\lambda(A, \mathcal{B}) := \frac{m}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} u_i u_i^\top \lambda \sinh(\lambda(u_i^\top A u_i - b_i)),$$

and update the potential function based on the update formula defined in Eq. (8). We upper bound the diagonal and off-diagonal terms respectively and obtain the expected progress on the potential function in Lemma 6.3.

Over $T = \widetilde{\Omega}(m^{3/2} B^{-1} R \delta^{-1})$ iterations, we can upper bound $\Phi(A_{T+1}) \leq O(m)$ with high probability. With a similar argument to the gradient descent section, we can prove that the SGD matrix sensing algorithm can output a solution matrix satisfying the same approximation guarantees with high success probability in Lemma 6.4. The optimized time complexity is $O(Bn^2)$ where B is the SGD batch size.

For the more general assumption where $\{u_i\}_{i \in [m]}$ are non-orthogonal vectors in $\mathbb{R}^n$ and $|u_i^\top u_j|$ has an upper bound, We also provide the cost-per-iteration analysis for stochastic gradient descent by bounding the diagonal entries and off-diagonal entries of the gradient matrix respectively. Then we prove that the progress on the expected potential satisfies the same guarantee as the gradient descent in Lemma E.2. Therefore, our SGD matrix sensing algorithm can output a matrix satisfying the approximation guarantee after

$$T = \widetilde{\Omega}(m^{3/2} B^{-1} R \delta^{-1})$$

iterations under the general assumption.

5 GRADIENT DESCENT FOR ENTRY-WISE POTENTIAL FUNCTION

In this section, we show how to obtain an approximate solution of matrix sensing via gradient descent. For simplicity, we start from a case that $\{u_i\}_{i \in [m]}$ are orthogonal vectors in $\mathbb{R}^n$[1], which already conveys the key idea of our algorithm and analysis and we generalize the solution to the non-orthogonal case (see Appendix D). We show that $\widetilde{\Omega}(\sqrt{m}/\delta)$ iterations of gradient descent can output a δ-approximate solution, where each iteration takes $O(mn^2)$-time. Below is the main theorem of this section:

[1]We note that $A' := \sum_{i=1}^m b_i u_i u_i^\top$ is a solution satisfying $u_i^\top A' u_i = b_i$ for all $i \in [m]$. However, we pretend that we do not know this solution in this section.

Theorem 5.1 (Gradient descent for orthogonal measurements)**.** *Suppose $u_1, \ldots, u_m \in \mathbb{R}^n$ are orthogonal unit vectors, and suppose $|b_i| \leq R$ for all $i \in [m]$. There exists an algorithm such that for any $\delta \in (0, 1)$, performs $\widetilde{\Omega}(\sqrt{m} R \delta^{-1})$ iterations of gradient descent with $O(mn^2)$-time per iteration and outputs a matrix $A \in \mathbb{R}^{n \times n}$ satisfies:*

$$|u_i^\top A u_i - b_i| \leq \delta \quad \forall i \in [m].$$

In Section 5.1, we introduce the algorithm and prove the time complexity. In Section 5.2 - 5.4, we analyze the convergence of our algorithm.

5.1 Algorithm

The key idea of the gradient descent matrix sensing algorithm (Algorithm 1) is to follow the gradient of the entry-wise potential function defined as follows:

$$\Phi_\lambda(A) := \sum_{i=1}^m \cosh(\lambda(u_i^\top A u_i - b_i)). \tag{3}$$

Then, we have the following solution update formula:

$$A_{t+1} \leftarrow A_t - \epsilon \cdot \nabla \Phi_\lambda(A_t) / \|\nabla \Phi_\lambda(A_t)\|_F. \tag{4}$$

Lemma 5.2 (Cost-per-iteration of gradient descent)**.** *Each iteration of Algorithm 1 takes $O(mn^2)$-time.*

Proof. In each iteration, we first evaluate $u_i^\top A_t u_i$ for all $i \in [m]$, which takes $O(mn^2)$-time. Then, $\nabla \Phi_\lambda(A_t)$ can be computed by summing m rank-1 matrices, which takes $O(mn^2)$-time. Finally, at Line 6, the solution can be updated in $O(n^2)$-time. Thus, the total running time for each iteration is $O(mn^2)$. □

Algorithm 1 Matrix Sensing by Gradient Descent.

1: **procedure** GRADIENTDESCENT($\{u_i, b_i\}_{i \in [m]}$) ▷ Theorem 5.1
2: $\tau \leftarrow \max_{i \in [m]} b_i$
3: $A_1 \leftarrow \tau \cdot I$
4: **for** $t = 1 \to T$ **do**
5: $\nabla \Phi_\lambda(A_t) \leftarrow \sum_{i=1}^m u_i u_i^\top \lambda \sinh(\lambda(u_i^\top A_t u_i - b_i))$ ▷ Compute the gradient
6: $A_{t+1} \leftarrow A_t - \epsilon \cdot \nabla \Phi_\lambda(A_t) / \|\nabla \Phi_\lambda(A_t)\|_F$
7: **end for**
8: **return** A_{T+1}
9: **end procedure**

5.2 Analysis of One Iteration

Throughout this section, we suppose $A \in \mathbb{R}^{n \times n}$ is a symmetric matrix.

We can compute the gradient of $\Phi_\lambda(A)$ with respect to A as follows:

$$\nabla \Phi_\lambda(A) = \sum_{i=1}^m u_i u_i^\top \lambda \sinh\left(\lambda(u_i^\top A u_i - b_i)\right) \in \mathbb{R}^{n\times n}. \tag{5}$$

We can compute the Hessian of $\Phi_\lambda(A)$ with respect to A as follows

$$\nabla^2 \Phi_\lambda(A) = \sum_{i=1}^m (u_i u_i^\top) \otimes (u_i u_i^\top) \lambda^2 \cosh(\lambda(u_i^\top A u_i - b_i)).$$

The Hessian $\nabla^2 \Phi_\lambda(A) \in \mathbb{R}^{n^2 \times n^2}$ and $\otimes$ is the Kronecker product.

Lemma 5.3 (Progress on entry-wise potential)**.** *Assume that $u_i \perp u_j = 0$ for any $i, j \in [m]$ and $\|u_i\|^2 = 1$. Let $c \in (0,1)$ denote a sufficiently small positive constant. Then, for any $\epsilon, \lambda > 0$ such that $\epsilon\lambda \leq c$,*

we have for any $t > 0$,

$$\Phi_\lambda(A_{t+1}) \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}$$

Proof. We defer the proof to Appendix B.1. □

5.3 Technical Claims

We prove some technical claims in below.

Claim 5.4. *For Q_1 defined in Eq. (18), we have*

$$Q_1 \leq \left(\sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F\right) \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2.$$

Proof. For simplicity, we define $z_{t,i}$ to be

$$z_{t,i} := \lambda(u_i^\top A_t u_i - b_i).$$

Recall that

$$\nabla^2 \Phi_\lambda(A_t) = \lambda^2 \cdot \sum_{i=1}^m (u_i u_i^\top) \otimes (u_i u_i^\top) \cosh(z_{t,i}).$$

For Q_1, we have

$$\begin{aligned}
Q_1 = & \ \mathrm{tr}[\nabla^2 \Phi_\lambda(A_t) \sum_{i=1}^m \sinh^2(z_{t,i})(u_i u_i^\top \otimes u_i u_i^\top))] \\
= & \ \lambda^2 \cdot \mathrm{tr}[\sum_{i=1}^m \cosh(z_{t,i})(u_i u_i^\top) \otimes (u_i u_i^\top) \cdot \\
& \ \sum_{i=1}^m \sinh^2(z_{t,i})(u_i u_i^\top) \otimes (u_i u_i^\top)] \\
= & \ \lambda^2 \cdot \sum_{i=1}^m \mathrm{tr}[\cosh(z_{t,i}) \cdot \sinh^2(z_{t,i})(u_i u_i^\top u_i u_i^\top) \otimes \\
& \ (u_i u_i^\top u_i u_i^\top)] \\
= & \ \lambda^2 \cdot \sum_{i=1}^m \cosh(z_{t,i}) \sinh^2(z_{t,i}) \\
\leq & \ \lambda^2 \cdot (\sum_{i=1}^m \cosh^2(z_{t,i}))^{1/2} \cdot (\sum_{i=1}^m \sinh^4(z_{t,i}))^{1/2} \\
\leq & \ \lambda^2 \cdot B_1 \cdot B_2,
\end{aligned} \tag{6}$$

where the first step comes from the definition of Q_1, the second step comes from the definition of $\nabla^2\Phi_\lambda(A_t)$, the third step follows from $(A\otimes B)\cdot(C\otimes D) = (AC)\otimes(BD)$ and $u_i^\top u_j = 0$, the fourth step comes from $\|u_i\| = 1$ and $\mathrm{tr}[(u_i u_i^\top) \otimes (u_i u_i^\top)] = 1$.

For the term B_1, we have

$$\begin{aligned}
B_1 = & \ (\sum_{i=1}^m \cosh^2(\lambda(u_i^\top A_t u_i - b_i)))^{1/2} \\
\leq & \ \sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F,
\end{aligned}$$

where the second step follows Part 1 of Lemma 3.3.

For the term B_2, we have

$$\begin{aligned}
B_2 = & \ (\sum_{i=1}^m \sinh^4(\lambda(u_i^\top A_t u_i - b_i)))^{1/2} \\
\leq & \ \frac{1}{\lambda^2}\|\nabla\Phi_\lambda(A_t)\|_F^2,
\end{aligned}$$

where the second step follows from $\|x\|_4^2 \leq \|x\|_2^2$. This implies that

$$\begin{aligned}
Q_1 \leq & \ \lambda^2 \cdot B_1 \cdot B_2 \\
\leq & \ \lambda^2 \cdot (\sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F) \cdot \frac{1}{\lambda^2}\|\nabla\Phi_\lambda(A_t)\|_F^2 \\
= & \ (\sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F) \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2.
\end{aligned}$$

□

Claim 5.5. *For Q_2 defined in Eq. (19), we have $Q_2 = 0$.*

Proof. Because in Q_2 we have :

$$\begin{aligned}
& \sum_{\ell=1}^m (u_\ell u_\ell^\top \otimes u_\ell u_\ell^\top) \sum_{i\neq j} (u_i u_i^\top \otimes u_j u_j^\top) \\
= & \ \sum_{\ell=1}^m \sum_{i\neq j} (u_\ell u_\ell^\top u_i u_i^\top) \otimes (u_\ell u_\ell^\top u_j u_j^\top) \\
= & \ 0,
\end{aligned} \tag{7}$$

where the first step follows from $(A\otimes B)\cdot(C\otimes D) = (AC)\otimes(BD)$, the second step follows that $u_i^\top u_j = 0$ if $i \neq j$ and $\ell \neq i$ or $\ell \neq j$ always holds in Eq. (7).

Therefore, we get that $Q_2 = 0$. □

5.4 Convergence For Multiple Iterations

The goal of this section is to prove the convergence of Algorithm 1:

Lemma 5.6 (Convergence of gradient descent)**.** *Suppose the measurement vectors $\{u_i\}_{i\in[m]}$ are orthogonal unit vectors, and suppose $|b_i|$ is bounded by R for $i \in [m]$. Then, for any $\delta \in (0,1)$, if we take $\lambda = \Omega(\delta^{-1}\log m)$ and $\epsilon = O(\lambda^{-1})$ in Algorithm 1, then for $T = \widetilde{\Omega}(\sqrt{m}R\delta^{-1})$ iterations, the solution matrix A_T satisfies:*

$$|u_i^\top A_T u_i - b_i| \leq \delta \quad \forall i \in [m].$$

Proof. We defer the proof to Appendix B.2 □

Theorem 5.1 follows immediately from Lemma 5.2 and Lemma 5.6.

6 STOCHASTIC GRADIENT DESCENT

In this section, we show that the cost-per-iteration of the approximate matrix sensing algorithm can be improved by using a stochastic gradient descent (SGD). More specifically, SGD can obtain a δ-approximate solution with $O(Bn^2)$, where $0 < B < m$ is the size of the mini batch in SGD. Below is the main theorem of this section:

Theorem 6.1 (Stochastic gradient descent for orthogonal measurements)**.** *Suppose $u_1, \ldots, u_m \in \mathbb{R}^n$ are orthogonal unit vectors, and suppose $|b_i| \leq R$ for all $i \in [m]$. There exists an algorithm such that for any $\delta \in (0,1)$, performs*

$$\widetilde{O}(m^{3/2}B^{-1}R\delta^{-1})$$

iterations of gradient descent with

$$O(Bn^2)$$

-time per iteration and outputs a matrix $A \in \mathbb{R}^{n\times n}$ satisfies:

$$|u_i^\top A u_i - b_i| \leq \delta \quad \forall i \in [m].$$

The algorithm and its time complexity are provided in Section 6.1. The convergence is proved in Section 6.2 and 6.3. The SGD algorithm for the general measurement without the assumption that the $\{u_i\}_{i\in[m]}$ are orthogonal vectors is deferred to Appendix E.

6.1 Algorithm

We can use the stochastic gradient descent algorithm (Algorithm 2) for matrix sensing. More specifically, in each iteration, we will uniformly sample a subset $\mathcal{B} \subset [m]$ of size B, and then compute the gradient of the stochastic potential function:

$$\nabla\Phi_\lambda(A,\mathcal{B}) := \frac{m}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} u_i u_i^\top \lambda \sinh(\lambda(u_i^\top A u_i - b_i)), \tag{8}$$

which is an n-by-n matrix. Then, we do the following gradient step:

$$A_{t+1} \leftarrow A_t - \epsilon \cdot \nabla\Phi_\lambda(A_t,\mathcal{B}_t)/\|\nabla\Phi_\lambda(A_t)\|_F. \tag{9}$$

Lemma 6.2 (Running time of stochastic gradient descent)**.** *Algorithm 2 takes $O(mn^2)$-time for preprocessing and each iteration takes $O(Bn^2)$-time.*

Proof. The time-consuming step is to compute $\|\nabla\Phi_\lambda(A_t)\|_F$. Since

$$\nabla\Phi_\lambda(A_t) = \sum_{i=1}^m u_i u_i^\top \lambda \sinh\left(\lambda(u_i^\top A_t u_i - b_i)\right),$$

and $u_i \perp u_j$ for $i \neq j \in [m]$, we know that u_i is an eigenvector of $\nabla\Phi_\lambda(A)$ with eigenvalue $\lambda\sinh\left(\lambda(u_i^\top A_t u_i - b_i)\right)$ for each $i \in [m]$. Thus, we have

$$\begin{aligned}
&\|\nabla\Phi_\lambda(A_t)\|_F^2 \\
=& \sum_{i=1}^m \lambda^2 \sinh^2\left(\lambda(u_i^\top A_t u_i - b_i)\right) \\
=& \sum_{i=1}^m \lambda^2 \sinh^2(\lambda z_{t,i}),
\end{aligned}$$

where $z_{t,i} := u_i^\top A_t u_i - b_i$ for $i \in [m]$. Then, if we know $z_{t,i}{}_{i\in[m]}$, we can compute $\|\nabla\Phi_\lambda(A_t)\|_F$ in $O(m)$-time.

Consider the change $z_{t+1,i} - z_{t,i}$:

$$\begin{aligned}
& z_{t+1,i} - z_{t,i} \\
=& u_i^\top (A_{t+1} - A_t) u_i \\
=& -\frac{\epsilon}{\|\nabla\Phi_\lambda(A_t)\|_F} \cdot u_i^\top \nabla\Phi_\lambda(A_t,\mathcal{B}_t) u_i \\
=& -\frac{\epsilon\lambda m}{\|\nabla\Phi_\lambda(A_t)\|_F B} \sum_{j\in\mathcal{B}_t} u_i^\top u_j u_j^\top u_i \cdot \sinh(\lambda z_{t,j}) \\
=& -\frac{\epsilon\lambda m \sinh(\lambda z_{t,i})}{\|\nabla\Phi_\lambda(A_t)\|_F B} \cdot \mathbf{1}_{i\in\mathcal{B}_t},
\end{aligned}$$

where the last step follows from $u_i \perp u_j$ for $i \neq j$. Hence, if we have already computed $\{z_{t,i}\}_{i\in[m]}$ and $\|\nabla\Phi_\lambda(A_t)\|_F$, $\{z_{t+1,i}\}_{i\in[m]}$ can be obtained in $O(B)$-time.

Therefore, we preprocess $z_{1,i} = u_i^\top A_1 u_i - b_i$ for all $i \in [m]$ in $O(mn^2)$-time. Then, in the t-th iteration

Algorithm 2 Matrix Sensing by Stochastic Gradient Descent.

procedure SGD($\{u_i, b_i\}_{i\in[m]}$) ▷ Theorem 6.1
  $\tau \leftarrow \max_{i\in[m]} b_i$
  $A_1 \leftarrow \tau \cdot I$
  $z_i \leftarrow u_i^\top A_1 u_i - b_i$ for $i \in [m]$
  for $t = 1 \to T$ **do**
    Sample $\mathcal{B}_t \subset [m]$ of size B uniformly at random
    $\nabla\Phi_\lambda(A_t, \mathcal{B}_t) \leftarrow \frac{m}{B}\sum_{i\in\mathcal{B}_t} u_i u_i^\top \lambda \sinh(\lambda z_i)$
    $\|\nabla\Phi_\lambda(A_t)\|_F \leftarrow \big(\sum_{i=1}^m \lambda^2 \sinh^2(\lambda z_i)\big)^{1/2}$
    $A_{t+1} \leftarrow A_t - \epsilon \cdot \nabla\Phi_\lambda(A_t, \mathcal{B}_t)/\|\nabla\Phi_\lambda(A_t)\|_F$
    for $i \in \mathcal{B}_t$ **do**
      $z_i \leftarrow z_i - \epsilon\lambda m \sinh(\lambda z_i)/(\|\nabla\Phi_\lambda(A_t)\|_F B)$
    end for
  end for
  return A_{T+1}
end procedure

$(t > 0)$, we first compute

$$\nabla\Phi_\lambda(A_t, \mathcal{B}_t) = \frac{m}{B}\sum_{i\in\mathcal{B}_t} u_i u_i^\top \lambda \sinh(\lambda z_{t,i})$$

in $O(Bn^2)$-time. Next, we compute $\|\nabla\Phi_\lambda(A_t)\|_F$ using $z_{t,i}$ in $O(m)$-time. A_{t+1} can be obtained in $O(n^2)$-time. Finally, we use $O(B)$-time to update $\{z_{t+1,i}\}_{i\in[m]}$.

Hence, the total running time per iteration is

$$O(Bn^2 + m + n^2 + B) = O(Bn^2).$$

□

6.2 Analysis of One Iteration

Suppose $A \in \mathbb{R}^{n\times n}$. Let $\mathcal{B}_t$ be a uniformly random B-subset of $[m]$ at the t-th iteration, where B is a parameter.

We can compute the gradient of $\Phi_\lambda(A, \mathcal{B})$ with respect to A as follows:

$$\nabla\Phi_\lambda(A, \mathcal{B}) = \frac{m}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} u_i u_i^\top \lambda \sinh(\lambda(u_i^\top A u_i - b_i)),$$

where $\nabla\Phi_\lambda(A, \mathcal{B}) \in \mathbb{R}^{n\times n}$.

We can also compute the Hessian of $\Phi_\lambda(A, \mathcal{B})$ with respect to A as follows:

$$\nabla^2\Phi_\lambda(A, \mathcal{B}) = \frac{m}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} (u_i u_i^\top) \otimes (u_i u_i^\top)\lambda^2 \cdot \cosh(\lambda(u_i^\top A u_i - b_i))$$

where $\nabla^2\Phi_\lambda(A, \mathcal{B}) \in \mathbb{R}^{n^2\times n^2}$ and $\otimes$ is the Kronecker product.

It is easy to see the expectations of the gradient and Hessian of $\Phi_\lambda(A, \mathcal{B})$ over a random set $\mathcal{B}$:

$$\mathop{\mathbb{E}}_{\mathcal{B}\sim[m]}[\nabla\Phi_\lambda(A, \mathcal{B})] = \nabla\Phi_\lambda(A),$$
$$\mathop{\mathbb{E}}_{\mathcal{B}\sim[m]}[\nabla^2\Phi_\lambda(A, \mathcal{B})] = \nabla^2\Phi_\lambda(A)$$

Lemma 6.3 (Expected progress on potential)**.** *Given m vectors $u_1, u_2, \cdots, u_m \in \mathbb{R}^n$. Assume $\langle u_i, u_j\rangle = 0$ for any $i \neq j \in [m]$ and $\|u_i\|^2 = 1$, for all $i \in [m]$. Let $\epsilon\lambda \leq 0.01\frac{|\mathcal{B}_t|}{m}$, for all $t > 0$.*

Then, we have

$$\mathbb{E}[\Phi_\lambda(A_{t+1})] \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}.$$

Proof. We first express the expectation as follows:

$$\begin{aligned} &\mathop{\mathbb{E}}_{A_{t+1}}[\Phi_\lambda(A_{t+1})] - \Phi_\lambda(A_t) \\ \leq\ &\mathop{\mathbb{E}}_{A_{t+1}}[\langle\nabla\Phi_\lambda(A_t), (A_{t+1} - A_t)\rangle] + O(1)\cdot \\ &\mathop{\mathbb{E}}_{A_{t+1}}[\langle\nabla^2\Phi_\lambda(A_t), (A_{t+1} - A_t)\otimes(A_{t+1} - A_t)\rangle], \end{aligned} \tag{10}$$

which follows from Corollary A.2.

We choose

$$A_{t+1} = A_t - \epsilon \cdot \nabla\Phi_\lambda(A_t, \mathcal{B}_t)/\|\nabla\Phi_\lambda(A_t)\|_F.$$

Then, we can bound

$$\begin{aligned} &\mathop{\mathbb{E}}_{A_{t+1}}[-\operatorname{tr}[\nabla\Phi_\lambda(A_t)\cdot(A_{t+1} - A_t)]] \\ =\ &\mathop{\mathbb{E}}_{\mathcal{B}_t}\Big[\operatorname{tr}\Big[\nabla\Phi_\lambda(A_t)\cdot\frac{\epsilon\nabla\Phi_\lambda(A_t, \mathcal{B}_t)}{\|\nabla\Phi_\lambda(A_t)\|_F}\Big]\Big] \\ =\ &\epsilon\cdot\|\nabla\Phi_\lambda(A_t)\|_F \end{aligned} \tag{11}$$

We define for $t > 0$ and $i \in [m]$,

$$z_{t,i} := u_i^\top A_t u_i - b_i.$$

We need to compute this Δ_2. For simplificity, we consider $\Delta_2 \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2$,

$$\begin{aligned} =\ &\operatorname{tr}[\nabla^2\Phi_\lambda(A_t)\cdot(A_{t+1} - A_t)\otimes(A_{t+1} - A_t)]\cdot \\ &\|\nabla\Phi_\lambda(A_t)\|_F^2 \\ =\ &(\lambda\epsilon)^2\cdot(\frac{m}{|\mathcal{B}_t|})^2\cdot\operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t)\cdot(\sum_{i\in\mathcal{B}_t} u_i u_i^\top \sinh(z_{t,i})\otimes \\ &(\sum_{i\in\mathcal{B}_t} u_i u_i^\top \sinh(z_{t,i})\Big]. \end{aligned} \tag{12}$$

Ignoring the scalar factor in the above equation, we have

$$= \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t)\cdot(\sum_{i,j\in B_t} \sinh(z_{t,i})\sinh(z_{t,i})\cdot$$

$$
\begin{aligned}
& (u_i u_i^\top \otimes u_j u_j^\top))\Big] \\
= & \operatorname{tr}\Big[\nabla^2 \Phi_\lambda(A_t) \cdot (\sum_{i \in B_t} \sinh^2(z_{t,i})(u_i u_i^\top \otimes u_i u_i^\top))\Big] \\
+ & \operatorname{tr}\Big[\nabla^2 \Phi_\lambda(A_t) \cdot (\sum_{i \neq j \in B_t} \sinh(z_{t,i}) \sinh(z_{t,i}) \cdot \\
& (u_i u_i^\top \otimes u_j u_j^\top))\Big] \\
=: & \widetilde{Q}_1 + \widetilde{Q}_2, \qquad (13)
\end{aligned}
$$

where the first step follows that we extract the scalar values from Kronecker product, the second step comes from splitting into two partitions based on whether $i = j$, the third step comes from the definition of $\widetilde{Q}_1$ and $\widetilde{Q}_2$ where $\widetilde{Q}_1$ denotes the diagonal term, and $\widetilde{Q}_2$ denotes the off-diagonal term. Taking expectation, we have

$$
\begin{aligned}
& \mathbb{E}[\Delta_2 \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2] \\
= & (\lambda\epsilon)^2 \cdot (\frac{m}{|\mathcal{B}_t|})^2 \mathbb{E}[\widetilde{Q}_1] \\
= & (\lambda\epsilon)^2 \cdot (\frac{m}{|\mathcal{B}_t|})^2 \cdot \frac{|\mathcal{B}_t|}{m} \cdot Q_1 \\
\leq & (\lambda\epsilon)^2 \cdot \frac{m}{|\mathcal{B}_t|} \cdot (\sqrt{m} + \frac{1}{\lambda}\|\nabla \Phi_\lambda(A_t)\|_F) \cdot \\
& \|\nabla \Phi_\lambda(A_t)\|_F^2 \qquad (14)
\end{aligned}
$$

where the first step comes from extracting the constant terms from the expectation and Claim 5.5, the second step follows that $\mathbb{E}[\widetilde{Q}_1] = \frac{|\mathcal{B}_t|}{m} \cdot Q_1$, and the third step comes from the Claim 5.4. Therefore, we have:

$$
\begin{aligned}
& \mathbb{E}[\Phi_\lambda(A_{t+1})] - \Phi_\lambda(A_t) \\
\leq & -\mathbb{E}[\Delta_1] + O(1) \cdot \mathbb{E}[\Delta_2] \\
\leq & -\epsilon(1 - O(\epsilon\lambda) \cdot \frac{m}{|\mathcal{B}_t|})\|\nabla \Phi_\lambda(A_t)\|_F + O(\epsilon\lambda)^2\sqrt{m} \\
\leq & -0.9\epsilon\|\nabla \Phi_\lambda(A_t)\|_F + O(\epsilon\lambda)^2\sqrt{m} \\
\leq & -0.9\epsilon\lambda\frac{1}{\sqrt{m}}(\Phi_\lambda(A_t) - m) + O(\epsilon\lambda)^2\sqrt{m} \\
\leq & -0.9\epsilon\lambda\frac{1}{\sqrt{m}}\Phi_\lambda(A_t) + \epsilon\lambda\sqrt{m},
\end{aligned}
$$

where the first step comes from Eq. (10), the second step comes from Eq. (11) and Eq. (14), the third step follows from $\epsilon \leq 0.01\frac{|\mathcal{B}_t|}{\lambda m}$, the forth step follows from Eq. (21), and the last step follows from $\epsilon\lambda \in (0, 0.01)$. □

6.3 Convergence For Multiple Iterations

The goal of this section is to prove the convergence of Algorithm 2.

Lemma 6.4 (Convergence of stochastic gradient descent)**.** *Suppose the measurement vectors $\{u_i\}_{i \in [m]}$ are orthogonal unit vectors, and suppose $|b_i|$ is bounded by R for $i \in [m]$. Then, for any $\delta \in (0, 1)$, if we take $\lambda = \Omega(\delta^{-1} \log m)$ and $\epsilon = O(\lambda^{-1} m^{-1} B)$ in Algorithm 2, then for*

$$
T = \widetilde{\Omega}(m^{3/2} B^{-1} R \delta^{-1})
$$

iterations, with high probability, the solution matrix A_T satisfies:

$$
|u_i^\top A_{T+1} u_i - b_i| \leq \delta \quad \forall i \in [m].
$$

Proof. Similar to the proof of Lemma 5.6, we can bound the initial potential by:

$$
\Phi(A_1) \leq 2^{O(\lambda R)}.
$$

In the following iterations, by Lemma 6.3, we have

$$
\mathbb{E}[\Phi_\lambda(A_{t+1})] \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m},
$$

as long as $\epsilon \leq 0.01\frac{|B_t|}{\lambda m}$, where B_t is a uniformly random subset of $[m]$ of size B.

It suffices to take $\epsilon = O(\lambda^{-1} m^{-1} B)$.

Now, we can apply Lemma 6.3 for T times and get that

$$
\mathbb{E}[\Phi(A_{T+1})] \leq 2^{-\Omega(T\epsilon\lambda/\sqrt{m}) + O(\lambda R)} + 2m.
$$

By taking

$$
T = \widetilde{\Omega}(m^{3/2} B^{-1} R \delta^{-1}),
$$

we have

$$
\Phi(A_{T+1}) \leq O(m)
$$

holds with high probability. By the same argument as in the proof of Lemma 5.6, we have

$$
|u_i^\top A_{T+1} u_i - b_i| \leq \delta \ \forall i \in [m].
$$

The lemma is thus proved. □

7 CONCLUSION

In this paper, we study the problem of matrix sensing which has a wide variety of practical applications in real-world science and engineering problems like image processing, quantum computing, and sensor localization. In many application domains of matrix sensing, it is appealing to tradeoff accuracy for fast running time, e.g., in fast and approximated k-nearest neighbors and k-means. We design an efficient algorithm with provable convergence guarantees using stochastic gradient descent to approximate matrix sensing. Based on our understanding, our work does not result in any negative societal impact since this is a theoretical paper.

References

Scott Aaronson. The learnability of quantum states. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2088):3089–3114, 2007.

Jan van den Brand. A deterministic linear program solver in current matrix multiplication time. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 259–278. SIAM, 2020.

Jan van den Brand, Yin Tat Lee, Aaron Sidford, and Zhao Song. Solving tall dense linear programs in nearly linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 775–788, 2020.

T Tony Cai and Anru Zhang. Rop: Matrix recovery via rank-one projections. *The Annals of Statistics*, 43(1):102–138, 2015.

Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *J. ACM*, 58(3), 2011. ISSN 0004-5411.

Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 938–942, New York, NY, USA, 2019. Association for Computing Machinery.

Yichuan Deng, Zhihang Li, and Zhao Song. An improved sample complexity for rank-1 matrix sensing. *arXiv preprint arXiv:2303.06895*, 2023.

Sally Dong, Yin Tat Lee, and Guanghao Ye. A nearly-linear time algorithm for linear programs with small treewidth: A multiscale representation of robust central path. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1784–1797, 2021.

Steven T Flammia, David Gross, Yi-Kai Liu, and Jens Eisert. Quantum tomography via compressed sensing: error bounds, sample complexity and efficient estimators. *New Journal of Physics*, 14(9):095022, 2012.

Yuzhou Gu and Zhao Song. A faster small treewidth sdp solver. *arXiv preprint arXiv:2211.06033*, 2022.

Christoph Helmberg, Franz Rendl, Robert J Vanderbei, and Henry Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on optimization*, 6(2):342–361, 1996.

Hang Hu, Zhao Song, Omri Weinstein, and Danyang Zhuo. Training overparametrized neural networks in sublinear time. *arXiv preprint arXiv:2208.04508*, 2022.

Baihe Huang, Shunhua Jiang, Zhao Song, Runzhou Tao, and Ruizhe Zhang. Solving sdp faster: A robust ipm framework and efficient implementation. In *FOCS*, 2022.

Prateek Jain, Raghu Meka, and Inderjit Dhillon. Guaranteed rank minimization via singular value projection. *Advances in Neural Information Processing Systems*, 23, 2010.

Adel Javanmard and Andrea Montanari. Localization from incomplete noisy distance measurements. *Found. Comput. Math.*, 13(3):297–345, jun 2013.

Haotian Jiang, Tarun Kathuria, Yin Tat Lee, Swati Padmanabhan, and Zhao Song. A faster interior point method for semidefinite programming. In *2020 IEEE 61st annual symposium on foundations of computer science (FOCS)*, pages 910–918. IEEE, 2020a.

Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 944–953, 2020b.

Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster lps. In *STOC*. arXiv preprint arXiv:2004.07470, 2021.

Anatoli Juditsky and Arkadii S Nemirovski. Large deviations of vector-valued martingales in 2-smooth normed spaces. *arXiv preprint arXiv:0809.0813*, 2008.

Amir Kalev, Robert L Kosut, and Ivan H Deutsch. Quantum tomography protocols with positivity are compressed sensing protocols. *npj Quantum Information*, 1(1):1–6, 2015.

Richard Kueng, Holger Rauhut, and Ulrich Terstiege. Low rank matrix recovery from rank one measurements. *Applied and Computational Harmonic Analysis*, 42(1):88–116, 2017.

Kiryung Lee and Yoram Bresler. Guaranteed minimum rank approximation from linear observations by nuclear norm minimization with an ellipsoidal constraint. *arXiv preprint arXiv:0903.4742*, 2009.

Yin Tat Lee and Aaron Sidford. Path finding methods for linear programming: Solving linear programs in o (vrank) iterations and faster algorithms for maximum flow. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 424–433. IEEE, 2014.

Yin Tat Lee and Aaron Sidford. Efficient inverse maintenance and faster algorithms for linear programming. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 230–249. IEEE, 2015.

Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1049–1065. IEEE, 2015.

Yin Tat Lee, Zhao Song, and Qiuyi Zhang. Solving empirical risk minimization in the current matrix multiplication time. In *COLT*, 2019.

Yuanxin Li, Cong Ma, Yuxin Chen, and Yuejie Chi. Nonconvex matrix factorization from rank-one measurements. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1496–1505. PMLR, 2019.

Zhihang Li, Zhao Song, and Tianyi Zhou. Solving regularized exp, cosh and sinh regression problems. *arXiv preprint, 2303.15725*, 2023.

Yi-Kai Liu. Universal low-rank matrix recovery from pauli measurements. *Advances in Neural Information Processing Systems*, 24, 2011.

Zhang Liu and Lieven Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1235–1256, 2010.

Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

Liank Qin, Zhao Song, Lichen Zhang, and Danyang Zhuo. An online and unified algorithm for projection matrix vector multiplication with application to empirical risk minimization. In *AISTATS*, 2023.

Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

Zhao Song and Zheng Yu. Oblivious sketching-based central path method for linear programming. In *International Conference on Machine Learning*, pages 9835–9847. PMLR, 2021.

Zhao Song, Shuo Yang, and Ruizhe Zhang. Does preprocessing help training over-parameterized neural networks? *Advances in Neural Information Processing Systems*, 34, 2021a.

Zhao Song, Lichen Zhang, and Ruizhe Zhang. Training multi-layer over-parametrized neural network in subquadratic time. *arXiv preprint arXiv:2112.07628*, 2021b.

Stephen Tu, Ross Boczar, Max Simchowitz, Mahdi Soltanolkotabi, and Ben Recht. Low-rank solutions of linear matrix equations via procrustes flow. In *International Conference on Machine Learning*, pages 964–973. PMLR, 2016.

Lingxiao Wang, Xiao Zhang, and Quanquan Gu. A unified computational and statistical framework for nonconvex low-rank matrix estimation. In *Artificial Intelligence and Statistics*, pages 981–990. PMLR, 2017.

Andrew Waters, Aswin Sankaranarayanan, and Richard Baraniuk. Sparcs: Recovering low-rank and sparse matrices from compressive measurements. *Advances in neural information processing systems*, 24, 2011.

Kai Zhong, Prateek Jain, and Inderjit S Dhillon. Efficient matrix sensing using rank-1 gaussian measurements. In *International conference on algorithmic learning theory*, pages 3–18. Springer, 2015.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Appendix

Roadmap. We first provide the proofs for matrix hyperbolic functions and properties of sinh and cosh in Appendix A. Then we provide the proofs for the gradient descent and stochastic gradient descent convergence analysis in Appendix B. We consider the spectral potential function with ground-truth oracle scenario in Appendix C. We analyze the gradient descent with non-orthogonal measurements in Appendix D. We provide the cost-per-iteration analysis for stochastic gradient descent under non-orthogonal measurements in Appendix E.

A Proofs of Preliminary Lemmas

In Section A.1 we present several calculus tools. In Section A.2 we present a fact for Kronecker product. In Section A.3, we present the proof for $\cosh(A)$ upper bound. In Section A.4, we present several equalities lemmas between sinh and cosh. In Section A.5, we present several inequalities lemmas between sinh and cosh.

A.1 Calculus tools

We state a useful calculus tool from prior work,

Lemma A.1 (Proposition 3.1 in (Juditsky and Nemirovski, 2008)). *Let Δ be an open interval on the axis, and f be C^2 function on Δ such that for certain $\theta_\pm, \mu_\pm \in \mathbb{R}$ one has*

$$\begin{aligned}&\forall (a<b, a, b \in \Delta):\\ &\theta_- \cdot \frac{f''(a)+f''(b)}{2} + \mu_- \leq \frac{f'(b)-f'(a)}{b-a}\\ &\frac{f'(b)-f'(a)}{b-a} \leq \theta_+ \cdot \frac{f''(a)+f''(b)}{2} + \mu_+,\end{aligned}$$

where f' and f'' means the first- and second-order derivatives of f, respectively.

Let, further, $\mathcal{X}_n(\Delta)$ be the set of all $n \times n$ symmetric matrices with eigenvalues belonging to Δ. Then $\mathcal{X}_n(\Delta)$ is an open convex set in the space S^n of $n \times n$ symmetric matrices, the function

$$F(X) = \mathrm{tr}[f(X)] : \mathcal{X}_n(\Delta) \to \mathbb{R}$$

is C^2, and for every $X \in \mathcal{X}_n(\Delta)$ and every $H \in S^n$ one has

$$\begin{aligned}&\theta_- \cdot \mathrm{tr}[H f''(X) H] + \mu_- \cdot \mathrm{tr}[H^2] \leq D^2 F(X)[H,H]\\ &D^2 F(X)[H,H] \leq \theta_+ \cdot \mathrm{tr}[H f''(X) H] + \mu_+ \cdot \mathrm{tr}[H^2],\end{aligned}$$

where D means directional derivative.

We will use below corollary to compute the trace with a map $f : \mathbb{R} \to \mathbb{R}$.

Corollary A.2. *Let $f : \mathbb{R} \to \mathbb{R}$ be a C^2 function. Let A and B be two symmetric matrices. We have*

$$\begin{aligned}\mathrm{tr}[f(A)] \leq\ & \mathrm{tr}[f(B)] + \mathrm{tr}[f'(B)(A-B)]\\ &+ O(1) \cdot \mathrm{tr}[f''(B)(A-B)^2].\end{aligned}$$

A.2 Kronecker product

Suppose we have two matrice $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$, we use $A \otimes B$ denote the Kronecker product:

$$A \otimes B = \begin{bmatrix} A_{1,1}B & \cdots & A_{1,n}B \\ \vdots & \ddots & \vdots \\ A_{m,1}B & \cdots & A_{m,n}B \end{bmatrix}.$$

We state a fact and delay the proof into Section A.

Fact A.3. *Suppose we have two matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times k}$, we have*

$$(A \otimes B) \cdot (C \otimes D) = (AC) \otimes (BD).$$

Proof. From the definition of Kronecker product we have:

$$\begin{aligned}
& (A \otimes B) \cdot (C \otimes D) \\
= & \begin{bmatrix} A_{1,1}B & \dots & A_{1,n}B \\ \vdots & \ddots & \vdots \\ A_{m,1}B & \dots & A_{m,n}B \end{bmatrix} \begin{bmatrix} C_{1,1}D & \dots & C_{1,k}D \\ \vdots & \ddots & \vdots \\ C_{n,1}D & \dots & C_{n,k}D \end{bmatrix} \\
= & \begin{bmatrix} (\sum_{i=1}^n A_{1,i}C_{i,1})BD & \cdots & (\sum_{i=1}^n A_{1,i}C_{i,k})BD \\ \vdots & \ddots & \vdots \\ (\sum_{i=1}^n A_{m,i}C_{i,1})BD & \cdots & (\sum_{i=1}^n A_{m,i}C_{i,k})BD \end{bmatrix} \\
= & \begin{bmatrix} (AC)_{1,1}BD & \cdots & (AC)_{1,k}BD \\ \vdots & \ddots & \vdots \\ (AC)_{m,1}BD & \cdots & (AC)_{m,k}BD \end{bmatrix} \\
= & (AC) \otimes (BD)
\end{aligned}$$

Thus we complete the proof. □

A.3 Proof of $\cosh(A)$ upper bound

Lemma A.4 (Restatement of Lemma 3.2)**.** *Let A be a real symmetric matrix, then we have*

$$\|\cosh(A)\| = \cosh(\|A\|) \leq \mathrm{tr}[\cosh(A)].$$

We also have $\|A\| \leq 1 + \log(\mathrm{tr}[\cosh(A)])$.

Proof. Note that for each eigenvalue λ of A, we know that it corresponds to $\cosh(\lambda)$ for $\cosh(A)$. The second inequality follows from the fact that $\cosh(A)$ is psd.

For the second part, we know that $\exp(x)/2 \leq \cosh(x)$, hence, $\exp(\|A\|)/2 \leq \cosh(\|A\|)$, and

$$\begin{aligned}
\|A\| = & \log(\exp(\|A\|)) \\
\leq & \log(2\cosh(\|A\|)) \\
\leq & 1 + \log(\mathrm{tr}[\cosh(A)]),
\end{aligned}$$

where the second step is by the monotonicity of $\log(\cdot)$ and $\exp(\|A\|) \leq 2\cosh(\|A\|)$, the last step is by $\cosh(\|A\|) \leq \mathrm{tr}[\cosh(A)]$. □

A.4 Relations Between cosh and sinh: Equalities

We state a fact as follows:

Fact A.5. *For any real number x, $\cosh^2(x) - \sinh^2(x) = 1$*

From the definition of $\cosh(x)$ and $\sinh(x)$ we have:

$$\begin{aligned}
& \cosh^2(x) - \sinh^2(x) \\
= & \frac{1}{4}(e^{2x} + 2 + e^{-2x}) - \frac{1}{4}(e^{2x} - 2 + e^{-2x}) \\
= & 1
\end{aligned}$$

We also have the following lemma for matrix.

Lemma A.6. *Let A be a real symmetric matrix, then we have*

$$\cosh^2(A) - \sinh^2(A) = I.$$

Proof. Since A is real symmetric, we write it in the eigendecomposition form: $A = U\Lambda U^\top$, then

$$\begin{aligned} & \cosh^2(A) - \sinh^2(A) \\ = & U\cosh^2(\Lambda)U^\top - U\sinh^2(\Lambda)U^\top \\ = & U(\cosh^2(\Lambda) - \sinh^2(\Lambda))U^\top \\ = & UU^\top \\ = & I, \end{aligned}$$

where the first step follows from cosh and sinh can be expressed as exp, the third step is by applying entrywise the identity $\cosh^2(x) - \sinh^2(x) = 1$. □

A.5 Relations Between cosh and sinh: Inequalities

Lemma A.7 (Scalar version, Restatement of Lemma 3.3)**.** *Given a list of numbers $x_1, \cdots x_n$, we have*

- $(\sum_{i=1}^n \cosh^2(x_i))^{1/2} \leq \sqrt{n} + (\sum_{i=1}^n \sinh^2(x_i))^{1/2}$,
- $(\sum_{i=1}^n \sinh^2(x_i))^{1/2} \geq \frac{1}{\sqrt{n}}(\sum_{i=1}^n \cosh(x_i) - n)$.

Proof. For the first equation, we can bound $(\sum_{i=1}^n \cosh^2(x_i))^{1/2}$ by:

$$\begin{aligned} (\sum_{i=1}^n \cosh^2(x_i))^{1/2} = & (n + \sum_{i=1}^n \sinh^2(x_i))^{1/2} \\ \leq & \sqrt{n} + (\sum_{i=1}^n \sinh^2(x_i))^{1/2} \end{aligned}$$

where the first step comes from fact A.5, and the second step follows from $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$.

For the second equation, we can bound $(\sum_{i=1}^n \sinh^2(x_i))^{1/2}$ by:

$$\begin{aligned} (\sum_{i=1}^n \sinh^2(x_i))^{1/2} \geq & \frac{1}{\sqrt{n}}(\sum_{i=1}^n \sinh(x_i)) \\ \geq & \frac{1}{\sqrt{n}}(\sum_{i=1}^n \cosh(x_i) - n) \end{aligned}$$

where the first step follows that $\sqrt{\frac{\sum_{i=1}^n x_i^2}{n}} \geq \frac{\sum_{i=1}^n x_i}{n}$, and the second step follows from fact A.5 and $\sqrt{x^2-1} \geq \sqrt{x} - 1$. □

We also have a lemma for the matrix version.

Lemma A.8 (Matrix version, Restatement of Lemma 3.4)**.** *For any real symmetric matrix A, we have*

- $(\mathrm{tr}[\cosh^2(A)])^{1/2} \leq \sqrt{n} + \mathrm{tr}[\sinh^2(A)]^{1/2}$,
- $(\mathrm{tr}[\sinh^2(A)])^{1/2} \geq \frac{1}{\sqrt{n}}(\mathrm{tr}[\cosh(A)] - n)$.

Proof. **Part 1.** We have

$$(\mathrm{tr}[\cosh^2(A)])^{1/2} = (n + \mathrm{tr}[\sinh^2(A)])^{1/2}$$

$$\leq \sqrt{n} + \mathrm{tr}[\sinh^2(A)]^{1/2}.$$

where the first step follows from $\cosh^2(A) - \sinh^2(A) = I$.

Part 2. Let σ_i denote the singular value of $\cosh(A)$

$$\begin{aligned}
(\mathrm{tr}[\sinh^2(A)])^{1/2} &= (\mathrm{tr}[\cosh^2(A)] - n)^{1/2} \\
&= (\sum_{i=1}^n \sigma_i^2 - 1)^{1/2} \\
&\geq \frac{1}{\sqrt{n}} \sum_{i=1}^n \sqrt{\sigma_i^2 - 1} \\
&\geq \frac{1}{\sqrt{n}} (\sum_{i=1}^n \sigma_i - 1) \\
&= \frac{1}{\sqrt{n}} (\mathrm{tr}[\cosh(A)] - n)
\end{aligned}$$

where the second step follows from $\|\cdot\|_2 \geq \frac{1}{\sqrt{n}} \|\cdot\|_1$, the third step follows from $\sigma_i \geq 1$.

□

B Proofs of GD and SGD convergence

In this section, we provide proofs of convergence analysis the gradient descent and stochastic gradient descent matrix sensing algorithms. In Section B.1, we provide proof for estimating the progress of gradient descent on potential function. In Section B.2, we prove the convergence result for gradient descent.

B.1 Proof of GD Progress on Potential Function

We start with the progress of the gradient on the potential function in below lemma.

Lemma B.1 (Restatement of Lemma 5.3)**.** *Assume that $u_i \perp u_j = 0$ for any $i, j \in [m]$ and $\|u_i\|^2 = 1$. Let $c \in (0,1)$ denote a sufficiently small positive constant. Then, for any $\epsilon, \lambda > 0$ such that $\epsilon\lambda \leq c$,*

we have for any $t > 0$,

$$\Phi_\lambda(A_{t+1}) \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}$$

Proof. We first Taylor expand $\Phi_\lambda(A_{t+1})$ as follows:

$$\begin{aligned}
&\Phi_\lambda(A_{t+1}) - \Phi_\lambda(A_t) \\
\leq\ & \langle \nabla\Phi_\lambda(A_t), (A_{t+1} - A_t)\rangle + O(1)\langle \nabla^2\Phi_\lambda(A_t), (A_{t+1} - A_t) \otimes (A_{t+1} - A_t)\rangle \\
:=\ & \Delta_1 + O(1) \cdot \Delta_2,
\end{aligned} \tag{15}$$

which follows from Lemma A.1.

We choose

$$A_{t+1} = A_t - \epsilon \cdot \nabla\Phi_\lambda(A_t)/\|\nabla\Phi_\lambda(A_t)\|_F.$$

We can bound

$$\begin{aligned}
\Delta_1 =\ & \mathrm{tr}[\nabla\Phi_\lambda(A_t)(A_{t+1} - A_t)] \\
=\ & -\epsilon \cdot \|\nabla\Phi_\lambda(A_t)\|_F.
\end{aligned} \tag{16}$$

Next, we upper-bound Δ_2. Define

$$z_{t,i} := \lambda(u_i^\top A_t u_i - b_i).$$

and consider $\Delta_2 \cdot (\lambda\epsilon)^{-2} \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2$, which can be expressed as:

$$\begin{aligned}
& \Delta_2 \cdot (\lambda\epsilon)^{-2} \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2 \\
= & \, (\lambda\epsilon)^{-2} \operatorname{tr}[\nabla^2\Phi_\lambda(A_t) \cdot (A_{t+1} - A_t) \otimes (A_{t+1} - A_t)] \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2 \\
= & \, \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t) \cdot (\sum_{i=1}^m u_i u_i^\top \sinh(z_{t,i})) \otimes (\sum_{i=1}^m u_i u_i^\top \sinh(z_{t,i}))\Big] \\
= & \, \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t) \cdot (\sum_{i,j} \sinh(z_{t,i}) \sinh(z_{t,i}) (u_i u_i^\top \otimes u_j u_j^\top))\Big] \\
= & \, \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t) \cdot (\sum_{i=1}^m \sinh^2(z_{t,i})) (u_i u_i^\top \otimes u_i u_i^\top))\Big] \\
+ & \, \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t) \cdot (\sum_{i\neq j} \sinh(z_{t,i}) \sinh(z_{t,j}) (u_i u_i^\top \otimes u_j u_j^\top))\Big] \\
=: & \, Q_1 + Q_2,
\end{aligned} \tag{17}$$

where

$$Q_1 := \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t) \cdot (\sum_{i=1}^m \sinh^2(z_{t,i})) (u_i u_i^\top \otimes u_i u_i^\top))\Big] \tag{18}$$

denotes the diagonal term, and

$$Q_2 := \operatorname{tr}\Big[\nabla^2\Phi_\lambda(A_t) \cdot (\sum_{i\neq j} \sinh(z_{t,i}) \sinh(z_{t,j}) (u_i u_i^\top \otimes u_j u_j^\top))\Big] \tag{19}$$

denotes the off-diagonal term. The first step comes from the definition of Δ_2, the second step follows fromr eplacing $A_{t+1} - A_t$ using Eq (4), the third step follows that we extract the scalar values from Kronecker product, the fourth step comes from splitting into two partitions based on whether $i = j$, the fifth step comes from the definition of Q_1 and Q_2.

Thus,

$$\begin{aligned}
\Delta_2 = & \, (\epsilon\lambda)^2 (Q_1 + Q_2) / \|\nabla\Phi_\lambda(A_t)\|_F^2 \\
= & \, (\epsilon\lambda)^2 (Q_1 + 0) / \|\nabla\Phi_\lambda(A_t)\|_F^2 \\
= & \, (\epsilon\lambda)^2 \cdot (\sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F).
\end{aligned} \tag{20}$$

where the second step follows from Claim 5.5, and the third step follows from Claim 5.4.

Hence, we have

$$\begin{aligned}
& \Phi_\lambda(A_{t+1}) - \Phi_\lambda(A_t) \\
\leq & \, \Delta_1 + O(1) \cdot \Delta_2 \\
\leq & \, -\epsilon\|\nabla\Phi_\lambda(A_t)\|_F + O(1)(\epsilon\lambda)^2(\sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F) \\
\leq & \, -0.9\epsilon\|\Phi_\lambda(A_t)\|_F + O(\epsilon\lambda)^2\sqrt{m}
\end{aligned}$$

where the first step follows from Eq. (15), the second step follows from Eq. (22) and Eq. (20), the third step follows from $\epsilon\lambda \in (0, 0.01)$.

For $\|\Phi_\lambda(A_t)\|_F$, we have

$$\frac{1}{\lambda^2}\|\nabla\Phi_\lambda(A_t)\|_F^2$$

$$
\begin{aligned}
= & \operatorname{tr}[(\sum_{i=1}^m u_i u_i^\top \sinh(\lambda(u_i^\top A_t u_i - b_i)))^2] \\
= & \operatorname{tr}[\sum_{i=1}^m (u_i u_i^\top)^2 \sinh^2(\lambda(u_i^\top A_t u_i - b_i))] \\
= & \sum_{i=1}^m \sinh^2(\lambda(u_i^\top A_t u_i - b_i)) \\
\geq & \frac{1}{m}(\sum_{i=1}^m \cosh(\lambda(u_i^\top A_t u_i - b_i)) - m)^2 \\
= & \frac{1}{m}(\Phi_\lambda(A_t) - m)^2, \qquad (21)
\end{aligned}
$$

where the first step comes from Eq. (5), the second steps follow from $u_i^\top u_j = 0$, the third step follows from $\|u_i\|_2 = 1$, the forth step follows from Part 2 in Lemma 3.3, the fifth step follows from the definition of $\Phi_\lambda(A)$.

Thus, we get that

$$
\|\Phi_\lambda(A_t)\|_F \geq \lambda \cdot \frac{1}{\sqrt{m}} |\Phi_\lambda(A_t) - m|, \qquad (22)
$$

It implies that

$$
\begin{aligned}
& \Phi_\lambda(A_{t+1}) - \Phi_\lambda(A_t) \\
\leq & -0.9\epsilon\lambda \frac{1}{\sqrt{m}} |\Phi_\lambda(A_t) - m| + O(\epsilon\lambda)^2 \sqrt{m} \\
\leq & -0.9\epsilon\lambda \frac{1}{\sqrt{m}} |\Phi_\lambda(A_t) - m| + 0.1\epsilon\lambda\sqrt{m},
\end{aligned}
$$

where the second step follows from extracting the constant term from the summation.

Then, when $\Phi(A_t) > m$, we have

$$
\Phi_\lambda(A_{t+1}) \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}.
$$

When $\Phi(A_t) \leq m$, we have

$$
\Phi_\lambda(A_{t+1}) \leq (1 + 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) - 0.8\lambda\epsilon\sqrt{m}.
$$

The lemma is then proved. □

B.2 Proof of GD Convergence

In this section, we provide proofs of convergence analysis of gradient descent matrix sensing algorithm.

Lemma B.2 (Restatement of Lemma 5.6)**.** *Suppose the measurement vectors $\{u_i\}_{i\in[m]}$ are orthogonal unit vectors, and suppose $|b_i|$ is bounded by R for $i \in [m]$. Then, for any $\delta \in (0,1)$, if we take $\lambda = \Omega(\delta^{-1}\log m)$ and $\epsilon = O(\lambda^{-1})$ in Algorithm 1, then for $T = \widetilde{\Omega}(\sqrt{m}R\delta^{-1})$ iterations, the solution matrix A_T satisfies:*

$$
|u_i^\top A_T u_i - b_i| \leq \delta \quad \forall i \in [m].
$$

Proof. Let $\tau = \max_{i\in[m]} b_i$. At the beginning, we choose the initial solution $A_1 := \tau I_n$ where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix, and we have

$$
\Phi(A_1) = \sum_{i=1}^m \cosh(\lambda \cdot (\tau - b_i))
$$

$$\leq e^{\lambda\tau}\sum_{i=1}^{m} e^{-\lambda b_i} \leq 2^{O(\lambda R)},$$

where the last step follows from $|b_i| \leq R$ for all $i \in [m]$.

After T iterations, we have

$$\begin{aligned}\Phi(A_{T+1}) &\leq (1-\frac{\epsilon\lambda}{\sqrt{m}})^T\Phi(A_1) + 2m \\ &\leq (1-\frac{\epsilon\lambda}{\sqrt{m}})^T \cdot 2^{O(\lambda R)} + 2m \\ &\leq 2^{-\Omega(T\epsilon\lambda/\sqrt{m})+O(\lambda R)} + 2m\end{aligned}$$

where the first step follows from applying Lemma 5.3 for T times, and $\sum_{i=1}^{T}(1-\epsilon\lambda/\sqrt{m})^{i-1}\epsilon\lambda\sqrt{m} \leq 2m$.

As long as $T = \Omega(R\sqrt{m}/\epsilon) = \Omega(R\sqrt{m}\lambda)$, then we have

$$\Phi(A_{T+1}) \leq O(m).$$

This implies that for any $i \in [m]$,

$$\begin{aligned}|u_i^\top A_{T+1}u_i - b_i| &\leq \lambda^{-1}\cdot\cosh^{-1}(O(m)) \\ &= \lambda^{-1}\cdot O(\log m) \\ &= \delta,\end{aligned}$$

where we take $R = \Omega(\delta^{-1}\log m)$.

Therefore, with $T = \widetilde{\Omega}(\sqrt{m}R\delta^{-1})$ iterations, Algorithm 1 can achieve that

$$|u_i^\top A_{T+1}u_i - b_i| \leq \delta \quad \forall i \in [m]. \tag{23}$$

The theorem is then proved. □

C Spectral Potential function with ground-truth oracle

In this section, we consider the matrix sensing with spectral approximation; that is, we want to obtain a matrix A that is a δ-spectral approximation of the ground-truth matrix $A_\star$, i.e.,

$$(1-\delta)A_\star \preceq A \preceq (1+\delta)A_\star.$$

To do this, instead of performing a series of quadratic measurements, we assume that we have access to an oracle $\mathcal{O}_{A_\star}$ such that for any matrix $A \in \mathbb{R}^{n\times n}$, the oracle will output a matrix $A_\star^{-1/2}AA_\star^{-1/2}$. Algorithm 3 implements a matrix sensing algorithm with spectral approximation guarantee with the assumption of oracle $\mathcal{O}_{A_\star}$.

We define the spectral loss function as follows:

$$\Psi_\lambda(A) := \mathrm{tr}[\cosh(\lambda(I-(A_\star)^{-1/2}A(A_\star)^{-1/2}))].$$

We will show that $\Psi_\lambda(A)$ can characterize the spectral approximation of A with respect to $A_\star$.

It is easy to see that if we can query an arbitrary A to the ground-truth oracle $\mathcal{O}_{A_\star}$, then we can definitely recover $A_\star$ exactly by querying $\mathcal{O}_{A_\star}(I)$. Instead, in Algorithm 3, we focus on the following process: the initial matrix A_1 is given, and in the t-th iteration, we first compute

$$X_t = \lambda(I - A_\star^{-1/2}A_tA_\star^{-1/2})$$

and do eigendecompsotion of X_t to obtain Λ_t such that $X_t = Q_t\Lambda_tQ_t^\top$. Then we update the matrix A_{t+1} by:

$$A_{t+1} = A_t + \epsilon\cdot A_\star^{1/2}\sinh(X_t)A_\star^{1/2}/\|\sinh(X_t)\|_F.$$

We are interested in the number of iterations needed to make A_t be a δ-spectral approximation. We believe this example will provide some insight into this problem, and we leave the question of spectral-approximated matrix sensing without the ground-truth oracle to future work.

Algorithm 3 Matrix Sensing with Spectral Approximation.

procedure GradientDescent($\mathcal{O}_{A_\star}$, A_1)
  for $t = 1 \to T$ **do**
    $X_t \leftarrow \lambda \cdot (I_n - \mathcal{O}_{A_\star}(A_t))$
    $Q_t \Lambda_t Q_t^\top \leftarrow$ Eigendecomposition of X_t ▷ It takes $O(n^\omega)$-time
    $Y_t \leftarrow Q_t \cdot \sinh(\Lambda_t) \cdot Q_t^\top$ ▷ $Y_t = \sinh(X_t)$. It takes $O(n^2)$-time
    $A_{t+1} \leftarrow A_t + \epsilon \cdot \mathcal{O}_{A_\star}(Y_t)/\|Y_t\|_F$ ▷ It takes $O(n^2)$-time
  end for
  return A_{T+1}
end procedure

Lemma C.1 (Progress on the spectral potential function)**.** *Let $c \in (0,1)$ denote a sufficiently small positive constant. We define X_t as follows:*

$$X_t := \lambda(I - (A_\star)^{-1/2} A_t (A_\star)^{-1/2})$$

Let

$$A_{t+1} = A_t + \epsilon \cdot \lambda (A_\star)^{1/2} \sinh(X_t)(A_\star)^{1/2} / \|\lambda \cdot \sinh(X_t)\|_F.$$

For any $\epsilon \in (0,1)$ and $\lambda \geq 1$ such $\lambda\epsilon \leq c$, we have for any $t > 0$,

$$\Psi_\lambda(A_{t+1}) \leq (1 - 0.9\epsilon\lambda/\sqrt{n})\Psi_\lambda(A_t) + \epsilon\lambda\sqrt{n}.$$

Proof. We can compute

$$\begin{aligned}
& \Psi_\lambda(A_{t+1}) - \Psi_\lambda(A_t) \\
= \ & \mathrm{tr}[\cosh(X_{t+1})] - \mathrm{tr}[\cosh(X_t)] \\
\leq \ & -\lambda \cdot \mathrm{tr}[\sinh(X_t) \cdot ((A_\star)^{-1/2}(A_{t+1} - A_t)(A_\star)^{-1/2})] \\
+ \ & O(1) \cdot \lambda^2 \cdot \mathrm{tr}[\cosh(X_t) \cdot ((A_\star)^{-1/2}(A_t - A_{t+1})(A_\star)^{-1/2})^2] \\
= \ & -\Delta_1 + O(1) \cdot \Delta_2,
\end{aligned} \tag{24}$$

the first step is by expanding by definition, the second step is by Taylor expanding the first term at the point $I - (A_\star)^{-1/2} A_t (A_\star)^{-1/2}$ (via Lemma A.1), and the last step is by definition of Δ_1 and Δ_2.

To further simplify proofs, we define

$$\begin{aligned}
\nabla \Psi_\lambda(A_t) &:= \lambda \cdot (A_\star)^{1/2} \sinh(X_t)(A_\star)^{1/2} \\
\widetilde{\nabla} \Psi_\lambda(A_t) &:= \lambda \cdot \sinh(X_t) \\
\widetilde{\Delta} \Psi_\lambda(A_t) &:= \lambda \cdot \cosh(X_t)
\end{aligned}$$

To maximize the gradient progress, we should choose

$$A_{t+1} = A_t + \epsilon \cdot \nabla \Psi_\lambda(A_t) / \|\widetilde{\nabla} \Psi_\lambda(A_t)\|_F$$

Then

$$\begin{aligned}
\Delta_1 &= (\epsilon\lambda^2) \cdot \mathrm{tr}[\sinh^2(X_t)] / \|\widetilde{\nabla} \Psi_\lambda(A_t)\|_F \\
&= \epsilon \cdot \|\widetilde{\nabla} \Psi_\lambda(A_t)\|_F^2 / \|\widetilde{\nabla} \Psi_\lambda(A_t)\|_F \\
&= \epsilon \cdot \|\widetilde{\nabla} \Psi_\lambda(A_t)\|_F
\end{aligned} \tag{25}$$

and

$$\begin{aligned}\Delta_2 &= \epsilon^2\lambda^4 \cdot \mathrm{tr}[\cosh(X_t)\cdot \sinh^2(\lambda(X_t)]/\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F^2 \\ &= \epsilon^2\lambda \cdot \mathrm{tr}[\widetilde{\Delta}\Psi_\lambda(A_t)\cdot \widetilde{\nabla}\Psi_\lambda(A_t)^2]/\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F^2 \\ &\leq \epsilon^2\lambda \cdot \|\widetilde{\Delta}\Psi_\lambda(A_t)\|_F \cdot \|\widetilde{\nabla}\Psi_\lambda(A_t)^2\|_F/\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F^2 \\ &\leq \epsilon^2\lambda \cdot \|\widetilde{\Delta}\Psi_\lambda(A_t)\|_F \\ &\leq \epsilon^2\lambda \cdot (\lambda\sqrt{n} + \|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F) \end{aligned} \tag{26}$$

where the first step follows from the definition of Δ_2, the second step comes from the definition of $\widetilde{\Delta}\Psi_\lambda(A_t)$ and $\widetilde{\nabla}\Psi_\lambda(A_t)$, the third step follows that $\|AB\|_F \leq \|A\|_F\|B\|_F$, the forth step follows from $\|x\|_4^2 \leq \|x\|_2^2$, and the fifth step follows from Part 1 of Lemma 3.4.

Now, we need to lower bound $\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F$, we have

$$\begin{aligned}\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F &= (\mathrm{tr}[\lambda^2\sinh^2(X_t)])^{1/2} \\ &\geq \frac{\lambda}{\sqrt{n}}(\mathrm{tr}[\cosh(X_t)] - n) \\ &= \frac{\lambda}{\sqrt{n}}(\Psi_\lambda(A_t) - n) \end{aligned} \tag{27}$$

where the second step follows from Part 2 in Lemma 3.4.

We know that

Then, we have

$$\begin{aligned}&\Psi_\lambda(A_{t+1}) - \Psi_\lambda(A_t) \\ \leq\ & -\epsilon\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F + \epsilon^2\lambda(\sqrt{n} + \|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F) \\ \leq\ & -0.9\epsilon\|\widetilde{\nabla}\Psi_\lambda(A_t)\|_F + \epsilon^2\lambda^2\sqrt{n} \\ \leq\ & -0.9\epsilon\lambda\frac{1}{\sqrt{n}}\Psi_\lambda(A_t) + \epsilon\lambda\sqrt{n}\end{aligned}$$

where the first step follows from Eq. (25) and Eq. (26) , the second steps comes from $\epsilon \in (0, 0.01)$, the third step comes from Eq. (27) and $\epsilon\lambda \leq 1$.

Finally, we complete the proof.

□

Lemma C.2 (Small spectral potential implies good spectral approximation)**.** *Let $A \in \mathbb{R}^{n\times n}$ be symmetric, and $\lambda > 0$. Suppose $\Psi_\lambda(A) \leq p$ for some $p > 1$. Then, we have*

$$(1-\delta)A_\star \preceq A \preceq (1+\delta)A_\star$$

for $\delta = O(\lambda^{-1}\log p)$.

Proof. By the definition of $\Psi_\lambda(A)$, $\Psi_\lambda(A) \leq p$ implies that for any $i \in [n]$,

$$\cosh(\lambda(1 - \lambda_i(A_\star^{-1/2}AA_\star^{-1/2}))) \leq p,$$

or equivalently,

$$\left|(1 - \lambda_i(A_\star^{-1/2}AA_\star^{-1/2}))\right| \leq O(\lambda^{-1}\log p).$$

Hence, we have

$$(1-\delta)I_n \preceq A_\star^{-1/2}AA_\star^{-1/2} \preceq (1+\delta)I_n,$$

where $\delta := O(\lambda^{-1}\log p)$. Therefore, by multiplying $A_\star^{-1/2}$ on both sides, we get that

$$(1-\delta)A_\star \preceq A \preceq (1+\delta)A_\star,$$

which completes the proof of the lemma. □

D Gradient descent with General Measurements

In this section, we first analyze the potential decay by gradient descent with non-orthogonal measurements in Section D.1, where we split it into two terms: Q_1 and Q_2. We then present how to bound the off-diagonal terms of the norm of gradient in Section D.2. We present how to bound the term Q_1 in Section D.3, which relies on an upper bound for its off-diagonal terms in Section D.4. We present how to bound the term Q_2 in Section D.5.

D.1 Progress Measurements

We first recall the definition of the potential function $\Phi_\lambda(A)$:

$$\Phi_\lambda(A) := \sum_{i=1}^m \cosh(\lambda(u_i^\top A u_i - b_i)),$$

its gradient $\nabla \Phi_\lambda(A) \in \mathbb{R}^{n\times n}$:

$$\nabla \Phi_\lambda(A) = \sum_{i=1}^m u_i u_i^\top \lambda \sinh\left(\lambda(u_i^\top A u_i - b_i)\right), \tag{28}$$

and its Hessian $\nabla^2 \Phi_\lambda(A) \in \mathbb{R}^{n^2\times n^2}$:

$$\nabla^2 \Phi_\lambda(A) = \sum_{i=1}^m (u_i u_i^\top) \otimes (u_i u_i^\top) \lambda^2 \cosh(\lambda(u_i^\top A u_i - b_i)).$$

Lemma D.1 (Progress on entry-wise potential with general measurements)**.** *Assume that $|u_i^\top u_j| \leq \rho$ and $\rho \leq \frac{1}{10m}$, for any $i, j \in [m]$ and $\|u_i\|^2 = 1$. Let $c \in (0,1)$ denote a sufficiently small positive constant. Then, for any $\epsilon, \lambda > 0$ such that $\epsilon\lambda \leq c$, we have for any $t > 0$,*

$$\Phi_\lambda(A_{t+1}) \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}$$

Proof. We first have

$$\begin{aligned} & \Phi_\lambda(A_{t+1}) - \Phi_\lambda(A_t) \\ \leq & \langle \nabla\Phi_\lambda(A_t), (A_{t+1} - A_t)\rangle + O(1)\langle \nabla^2\Phi_\lambda(A_t), (A_{t+1} - A_t) \otimes (A_{t+1} - A_t)\rangle \\ := & -\Delta_1 + O(1)\cdot\Delta_2, \end{aligned} \tag{29}$$

which follows from Corollary A.2.

We choose

$$A_{t+1} = A_t - \epsilon \cdot \nabla\Phi_\lambda(A_t)/\|\nabla\Phi_\lambda(A_t)\|_F. \tag{30}$$

We can bound

$$\begin{aligned} \Delta_1 = & -\mathrm{tr}[\nabla\Phi_\lambda(A_t)(A_{t+1} - A_t)] \\ = & \epsilon \cdot \|\nabla\Phi_\lambda(A_t)\|_F. \end{aligned} \tag{31}$$

For $\|\Phi_\lambda(A_t)\|_F^2$,

$$\begin{aligned} & \frac{1}{\lambda^2}\|\nabla\Phi_\lambda(A_t)\|_F^2 \\ = & \mathrm{tr}[(\sum_{i=1}^m u_i u_i^\top \sinh(\lambda(u_i^\top A_t u_i - b_i)))^2] \\ = & \mathrm{tr}[\sum_{i=1}^m \sinh^2(\lambda(u_i^\top A_t u_i - b_i))] \end{aligned}$$

$$
\begin{aligned}
&+ \operatorname{tr}[\sum_{i=1}^m \sum_{j \neq i}^m (u_i u_i^\top)(u_j u_j^\top) \sinh(\lambda(u_i^\top A_t u_i - b_i)) \cdot \sinh(\lambda(u_j^\top A_t u_j - b_j))] \\
&\geq 0.9 \operatorname{tr}[\sum_{i=1}^m \sinh^2(\lambda(u_i^\top A_t u_i - b_i))] \\
&\geq 0.9 \frac{1}{m} (\sum_{i=1}^m \cosh(\lambda(u_i^\top A_t u_i - b_i)) - m)^2 \\
&= 0.9 \frac{1}{m} (\Phi_\lambda(A_t) - m)^2, \qquad (32)
\end{aligned}
$$

where the first step follows from Eq. (28), the second steps follow from partitioning based on whether $i = j$ and $\|u_i\|_2 = 1$, the third step comes from Claim D.2, the fourth step in Eq. (32) follows from Part 2 in Lemma 3.3, the fifth step follows from the definition of $\Phi_\lambda(A)$.

Thus,

$$
\begin{aligned}
\Delta_1 = & - \operatorname{tr}[\nabla \Phi_\lambda(A_t)(A_{t+1} - A_t)] \\
& \geq \lambda \epsilon \cdot \frac{1}{\sqrt{m}} (\Phi_\lambda(A_t) - m). \qquad (33)
\end{aligned}
$$

For simplicity, we define

$$
z_{t,i} := \lambda(u_i^\top A_t u_i - b_i).
$$

We need to compute this Δ_2. For simplificity, we consider $\Delta_2 \cdot (\frac{1}{\epsilon\lambda})^2 \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2$, which can be expressed as:

$$
\begin{aligned}
& \Delta_2 \cdot (\frac{1}{\epsilon\lambda})^2 \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2 \\
= & \frac{1}{(\lambda\epsilon)^2} \operatorname{tr}[\nabla^2 \Phi_\lambda(A_t) \cdot (A_{t+1} - A_t) \otimes (A_{t+1} - A_t)] \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2 \\
= & \operatorname{tr}\Big[\nabla^2 \Phi_\lambda(A_t) \cdot (\sum_{i=1}^m u_i u_i^\top \sinh(z_{t,i})) \otimes (\sum_{i=1}^m u_i u_i^\top \sinh(z_{t,i}))\Big] \\
= & \operatorname{tr}[\nabla^2 \Phi_\lambda(A_t) (\sum_{i,j} \sinh(z_{t,i}) \sinh(z_{t,i}) (u_i u_i^\top \otimes u_j u_j^\top))] \\
= & \operatorname{tr}[\nabla^2 \Phi_\lambda(A_t) (\sum_{i=1}^m \sinh^2(z_{t,i})) (u_i u_i^\top \otimes u_i u_i^\top))] \\
+ & \operatorname{tr}[\nabla^2 \Phi_\lambda(A_t) (\sum_{i \neq j} \sinh(z_{t,i}) \sinh(z_{t,j}) (u_i u_i^\top \otimes u_j u_j^\top))] \\
= & Q_1 + Q_2, \qquad (34)
\end{aligned}
$$

where

$$
Q_1 := \operatorname{tr}\Big[\nabla^2 \Phi_\lambda(A_t) \cdot (\sum_{i=1}^m \sinh^2(z_{t,i}))(u_i u_i^\top \otimes u_i u_i^\top))\Big] \qquad (35)
$$

denotes the diagonal term, and

$$
Q_2 := \operatorname{tr}\Big[\nabla^2 \Phi_\lambda(A_t) \cdot (\sum_{i \neq j} \sinh(z_{t,i}) \sinh(z_{t,j}) (u_i u_i^\top \otimes u_j u_j^\top))\Big] \qquad (36)
$$

denotes the off-diagonal term. The first step comes from the definition of Δ_2, the second step follows from replacing $A_{t+1} - A_t$ using Eq. (30), the third step follows that we extract the scalar values from Kronecker product, the fourth step comes from splitting into two partitions based on whether $i = j$, the fifth step comes from the definition of Q_1 and Q_2.

Thus,

$$\begin{aligned}\Delta_2 \leq & (\epsilon\lambda)^2(Q_1+Q_2)/\|\nabla\Phi_\lambda(A_t)\|_F^2 \\ = & 1.3(\epsilon\lambda)^2\cdot(\sqrt{m}+\frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F). \end{aligned} \tag{37}$$

where the second step follows from Claim D.3 and Claim D.5.

Hence, we have

$$\begin{aligned} & \Phi_\lambda(A_{t+1})-\Phi_\lambda(A_t) \\ \leq & -\Delta_1+O(1)\cdot\Delta_2 \\ \leq & -\epsilon\|\nabla\Phi_\lambda(A_t)\|_F+O(1)(\epsilon\lambda)^2(\sqrt{m}+\frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F) \\ \leq & -0.9\epsilon\|\Phi_\lambda(A_t)\|_F+O(\epsilon\lambda)^2\sqrt{m} \\ \leq & -0.9\epsilon\lambda\frac{1}{\sqrt{m}}(\Phi_\lambda(A_t)-m)+O(\epsilon\lambda)^2\sqrt{m} \\ \leq & -0.9\epsilon\lambda\frac{1}{\sqrt{m}}\Phi_\lambda(A_t)+\epsilon\lambda\sqrt{m}, \end{aligned}$$

where the first step follows from Eq. (29), the second step follows from Eq. (33) and Eq. (37), the third step follows from $\epsilon\lambda\in(0,0.01)$, the fourth step follows from Lemma A.6, and the final step follows that extracting the constant term from the summation.

The lemma is then proved. □

D.2 Bounding the off-diagonal terms in $\lambda^{-2}\|\nabla\Phi_\lambda(A_t)\|_F^2$

Claim D.2. *It holds that:*

$$\begin{aligned} & \sum_{i\neq j\in[m]}\langle u_i,u_j\rangle^2\sinh(\lambda(u_i^\top A_t u_i-b_i))\sinh(\lambda(u_j^\top A_t u_j-b_j)) \\ \leq & 0.1\sum_{i=1}^m\sinh^2(\lambda(u_i^\top A_t u_i-b_i)) \end{aligned}$$

Proof. We define $R_{i,j}$ and R as follows:

$$\begin{aligned} R_{i,j} = & \sinh(\lambda(u_i^\top A_t u_i-b_i))\sinh(\lambda(u_j^\top A_t u_j-b_j)) \\ R = & \operatorname{tr}[\sum_{i=1}^m\sum_{j\neq i}^m(u_iu_i^\top)(u_ju_j^\top)\sinh(\lambda(u_i^\top A_t u_i-b_i))\cdot\sinh(\lambda(u_j^\top A_t u_j-b_j))] \end{aligned}$$

Then we can upper bound $|R|$ by:

$$\begin{aligned} |R| = & \operatorname{tr}[\sum_{i=1}^m\sum_{j\neq i}^m|(u_iu_i^\top)(u_ju_j^\top)||R_{i,j}|] \\ \leq & \rho^2\operatorname{tr}[\sum_{i=1}^m\sum_{j\neq i}^m|R_{i,j}|] \\ \leq & \frac{\rho^2}{2}\operatorname{tr}[\sum_{i=1}^m\sum_{j\neq i}^m(R_{i,i}+R_{j,j})] \\ \leq & m\rho^2\operatorname{tr}[\sum_{i=1}^m R_{i,i}] \end{aligned}$$

$$\leq 0.1 \operatorname{tr}[\sum_{i=1}^m R_{i,i}]$$

where the first step follows $|ab| = |a||b|$, the second step follows $|u_i^\top u_j| \leq \rho$, the third step follows that $|ab| \leq \frac{a^2+b^2}{2}$, the fourth step follows from the summation over j, and the fifth step comes from $m\rho^2 \leq 0.1$. □

D.3 Bounding the term Q_1

Claim D.3. *For Q_1 defined in Eq. (35), we have*

$$Q_1 \leq 1.1(\sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F) \cdot \|\nabla\Phi_\lambda(A_t)\|_F^2.$$

Proof. For simplicity, we define $z_{t,i}$ to be

$$z_{t,i} := \lambda(u_i^\top A_t u_i - b_i).$$

Recall that

$$\nabla^2\Phi_\lambda(A_t) = \lambda^2 \cdot \sum_{i=1}^m (u_i u_i^\top) \otimes (u_i u_i^\top) \cosh(z_{t,i}).$$

For Q_1, we have

$$\begin{aligned}
Q_1 = & \operatorname{tr}[\nabla^2\Phi_\lambda(A_t) \sum_{i=1}^m \sinh^2(z_{t,i})(u_i u_i^\top \otimes u_i u_i^\top))] \\
= & \lambda^2 \cdot \operatorname{tr}[\sum_{i=1}^m \cosh(z_{t,i})(u_i u_i^\top) \otimes (u_i u_i^\top) \cdot \sum_{i=1}^m \sinh^2(z_{t,i})(u_i u_i^\top) \otimes (u_i u_i^\top)] \\
= & \lambda^2 \cdot \sum_{i=1}^m \operatorname{tr}[\cosh(z_{t,i}) \sinh^2(z_{t,i}) \cdot (u_i u_i^\top u_i u_i^\top) \otimes (u_i u_i^\top u_i u_i^\top)] \\
+ & \lambda^2 \cdot \sum_{i=1}^m \sum_{j\neq i}^m \operatorname{tr}[\cosh(z_{t,i}) \sinh^2(z_{t,j}) \cdot (u_i u_i^\top u_j u_j^\top) \otimes (u_j u_j^\top u_i u_i^\top)] \\
= & \lambda^2 \cdot \sum_{i=1}^m \cosh(z_{t,i}) \sinh^2(z_{t,i}) \\
+ & \lambda^2 \cdot \sum_{i=1}^m \sum_{j\neq i}^m \operatorname{tr}[\cosh(z_{t,i}) \sinh^2(z_{t,j}) \cdot (u_i u_i^\top u_j u_j^\top) \otimes (u_j u_j^\top u_i u_i^\top)] \\
\leq & 1.1\lambda^2 \cdot (\sum_{i=1}^m \cosh^2(z_{t,i}))^{1/2} \cdot (\sum_{i=1}^m \sinh^4(z_{t,i}))^{1/2} \\
\leq & 1.1\lambda^2 \cdot B_1 \cdot B_2, \qquad (38)
\end{aligned}$$

where the first step comes from the definition of Q_1, the second step comes from the definition of $\nabla^2\Phi_\lambda(A_t)$, the third step follows from $(A \otimes B) \cdot (C \otimes D) = (AC) \otimes (BD)$ and partition the terms based on whether $i = j$, the fourth step comes from $\|u_i\| = 1$ and $\operatorname{tr}[(u_i u_i^\top) \otimes (u_i u_i^\top)] = 1$, and the fifth step comes from Cauchy–Schwarz inequality and Claim D.4.

For the term B_1, we have

$$\begin{aligned}
B_1 = & (\sum_{i=1}^m \cosh^2(\lambda(u_i^\top A_t u_i - b_i)))^{1/2} \\
\leq & \sqrt{m} + \frac{1}{\lambda}\|\nabla\Phi_\lambda(A_t)\|_F, \qquad (39)
\end{aligned}$$

where the second step follows Part 1 of Lemma 3.3.

For the term B_2, we have

$$\begin{aligned} B_2 = & (\sum_{i=1}^m \sinh^4(\lambda(u_i^\top A_t u_i - b_i)))^{1/2} \\ \leq & \frac{1}{\lambda^2}\|\nabla \Phi_\lambda(A_t)\|_F^2, \end{aligned} \tag{40}$$

where the second step follows from $\|x\|_4^2 \leq \|x\|_2^2$. This implies that

$$\begin{aligned} Q_1 \leq & 1.1\lambda^2 \cdot B_1 \cdot B_2 \\ \leq & 1.1\lambda^2 \cdot (\sqrt{m} + \frac{1}{\lambda}\|\nabla \Phi_\lambda(A_t)\|_F) \cdot \frac{1}{\lambda^2}\|\nabla \Phi_\lambda(A_t)\|_F^2 \\ = & 1.1(\sqrt{m} + \frac{1}{\lambda}\|\nabla \Phi_\lambda(A_t)\|_F) \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2. \end{aligned}$$

This completes the proof. □

D.4 Bounding off-diagonal terms in Q_1

Claim D.4. *We can bound the off-diagonal entries by:*

$$\begin{aligned} & |\lambda^2 \cdot \sum_{i=1}^m \sum_{j \neq i}^m \mathrm{tr}[\cosh(z_{t,i}) \sinh^2(z_{t,j}) \cdot (u_i u_i^\top u_j u_j^\top) \otimes (u_j u_j^\top u_i u_i^\top)]| \\ \leq & 0.1\lambda^2 (\sum_{i=1}^m (\cosh(z_{t,1}))^{1/2} \cdot (\sum_{i=1}^m \sinh^4(z_{t,i}))^{1/2} \end{aligned}$$

Proof.

$$\begin{aligned} & |\lambda^2 \cdot \sum_{i=1}^m \sum_{j \neq i}^m \mathrm{tr}[\cosh(z_{t,i}) \sinh^2(z_{t,j}) \cdot (u_i u_i^\top u_j u_j^\top) \otimes (u_j u_j^\top u_i u_i^\top)]| \\ \leq & \rho^2 \lambda^2 |\sum_{i=1}^m \sum_{j \neq i}^m \cosh(z_{t,i}) \sinh^2(z_{t,j})| \\ \leq & \rho^2 \lambda^2 (\sum_{i=1}^m \sum_{j \neq i}^m (\cosh^2(z_{t,i}))^{1/2} \cdot (\sum_{i=1}^m \sum_{j \neq i}^m \sinh^4(z_{t,j}))^{1/2} \\ \leq & m\rho^2 \lambda^2 (\sum_{i=1}^m (\cosh^2(z_{t,i}))^{1/2} \cdot (\sum_{i=1}^m \sinh^4(z_{t,i}))^{1/2} \\ \leq & 0.1\lambda^2 (\sum_{i=1}^m (\cosh^2(z_{t,i}))^{1/2} \cdot (\sum_{i=1}^m \sinh^4(z_{t,i}))^{1/2} \end{aligned}$$

where the first step comes from $|\langle u_i, u_j \rangle| \leq \rho$, the second step comes from Cauchy–Schwarz inequality, the third step follows from summation over m terms, and the fourth step comes from $\rho^2 m \leq 0.1$. □

D.5 Bounding the term Q_2

Claim D.5. *For Q_2 defined in Eq. (36), we have:*

$$Q_2 \leq 0.2\lambda^2 (\sqrt{m} + \frac{1}{\lambda}\|\nabla \Phi_\lambda(A_t)\|_F) \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2$$

Proof. Because in Q_2 we have :

$$\begin{aligned}
Q_2 = & \lambda^2 \operatorname{tr}[\sum_{\ell=1}^m (\cosh(z_{t,\ell}) \cdot u_\ell u_\ell^\top \otimes u_\ell u_\ell^\top) \cdot \sum_{i \neq j}^m (\sinh(z_{t,i}) \sinh(z_{t,j}) \cdot u_i u_i^\top \otimes u_j u_j^\top)] \\
= & \lambda^2 \operatorname{tr}[\sum_{\ell=1}^m \sum_{i \neq j}^m \cosh(z_{t,\ell}) \sinh(z_{t,i}) \sinh(z_{t,j}) \cdot (u_\ell u_\ell^\top u_i u_i^\top) \otimes (u_\ell u_\ell^\top u_j u_j^\top)] \\
\leq & \lambda^2 \rho^2 \sum_{\ell=1}^m \sum_{i \neq j}^m \cosh(z_{t,\ell}) (\sinh^2(z_{t,i}) + \sinh^2(z_{t,j})) \\
\leq & 2m\lambda^2 \rho^2 \sum_{\ell=1}^m \sum_{i=1}^m \cosh(z_{t,\ell}) \sinh^2(z_{t,i}) \\
\leq & 2m^2 \lambda^2 \rho^2 \sum_{i=1}^m \cosh(z_{t,i}) \sinh^2(z_{t,i}) \\
\leq & 2m^2 \lambda^2 \rho^2 (\sum_{i=1}^m (\cosh^2(z_{t,i}))^{1/2} (\sum_{i=1}^m \sinh^4(z_{t,i}))^{1/2} \\
\leq & 0.2\lambda^2 (\sqrt{m} + \frac{1}{\lambda} \|\nabla \Phi_\lambda(A_t)\|_F) \cdot \|\nabla \Phi_\lambda(A_t)\|_F^2
\end{aligned} \tag{41}$$

where the second step follows from $(A \otimes B) \cdot (C \otimes D) = (AC) \otimes (BD)$, the third step follows Cauchy–Schwarz inequality and $|\langle u_i, u_j \rangle| \leq \rho$, the fourth step follows from combining $\sinh^2(z_{t,i})$ and $\sinh^2(z_{t,j})$, the fifth step comes from summation over m terms, and the sixth step comes from Cauchy–Schwarz inequality and the seventh step follows from Eq. (39) and Eq. (40) and $m^2\rho^2 \leq 0.1$.

□

E Stochastic Gradient Descent for General Measurements

In this section, we further extend the general measurement where $\{u_i\}_{i \in [m]}$ are non-orthogonal vectors and $|u_i^\top u_j| \leq \rho$ to the convergence analysis of the stochastic gradient descent matrix sensing algorithm. Algorithm 4 implements the stochastic gradient descent version of the matrix sensing algorithm.

In Algorithm 4, at each iteration t, we first compute the stochastic gradient descent by:

$$\nabla \Phi_\lambda(A_t, \mathcal{B}_t) \leftarrow \frac{m}{B} \sum_{i \in \mathcal{B}_t} u_i u_i^\top \lambda \sinh(\lambda z_i)$$

then we update the matrix with the gradient:

$$A_{t+1} \leftarrow A_t - \epsilon \cdot \nabla \Phi_\lambda(A_t, \mathcal{B}_t) / \|\nabla \Phi_\lambda(A_t)\|_F$$

At the end of each iteration, we update z_i by:

$$z_i \leftarrow z_i - \epsilon \lambda m w_{i,j}^2 \sinh(\lambda z_j) / (\|\nabla \Phi_\lambda(A_t)\|_F B \ \ \forall i \in [m], j \in \mathcal{B}_t$$

We are interested in studying the time complexity and convergence analysis under the general measurement assumption.

Lemma E.1 (Cost-per-iteration of stochastic gradient descent for general measurements)**.** *Algorithm 4 takes $O(mn^2)$-time for preprocessing and each iteration takes $O(Bn^2 + m^2)$-time.*

Proof. Since u_i's are no longer orthogonal, we need to compute $\|\nabla \Phi_\lambda(A_t)\|_F$ in the following way:

$$\begin{aligned}
& \|\nabla \Phi_\lambda(A_t)\|_F^2 \\
= & \operatorname{tr}\Big[\Big(\sum_{i=1}^m u_i u_i^\top \lambda \sinh(\lambda(\lambda z_{t,i}))\Big)^2\Big]
\end{aligned}$$

$$= \lambda^2 \sum_{i,j=1}^m \langle u_i, u_j \rangle^2 \sinh(\lambda(\lambda z_{t,i})) \sinh(\lambda(\lambda z_{t,j}))$$
$$= \lambda^2 \sum_{i,j=1}^m w_{i,j}^2 \sinh(\lambda(\lambda z_{t,i})) \sinh(\lambda(\lambda z_{t,j})).$$

Hence, with $\{z_{t,i}\}_{i\in[m]}$, we can compute $\|\nabla \Phi_\lambda(A_t)\|_F$ in $O(m^2)$-time.

Another difference from the orthogonal measurement case is the update for $z_{t+1,i}$. Now, we have

$$\begin{aligned} & z_{t+1,i} - z_{t,i} \\ = & u_i^\top (A_{t+1} - A_t) u_i \\ = & -\frac{\epsilon}{\|\nabla \Phi_\lambda(A_t)\|_F} \cdot u_i^\top \nabla \Phi_\lambda(A_t, \mathcal{B}_t) u_i \\ = & -\frac{\epsilon \lambda m}{\|\nabla \Phi_\lambda(A_t)\|_F B} \sum_{j \in \mathcal{B}_t} u_i^\top u_j u_j^\top u_i \cdot \sinh(\lambda z_{t,j}) \\ = & -\frac{\epsilon \lambda m}{\|\nabla \Phi_\lambda(A_t)\|_F B} \sum_{j \in \mathcal{B}_t} w_{i,j}^2 \cdot \sinh(\lambda z_{t,j}). \end{aligned}$$

Hence, each $z_{t+1,i}$ can be computed in $O(B)$-time. And it takes $O(mB)$-time to update all $z_{t+1,i}$.

The other steps' time costs are quite clear from Algorithm 4. □

Lemma E.2 (Progress on expected potential with general measurements)**.** *Assume that $|u_i^\top u_j| \leq \rho$ and $\rho \leq \frac{1}{10m}$, for any $i, j \in [m]$ and $\|u_i\|^2 = 1$. Let $c \in (0,1)$ denote a sufficiently small positive constant. Then, for any $\epsilon, \lambda > 0$ such that $\epsilon\lambda \leq c\frac{|\mathcal{B}_t|}{m}$, we have for any $t > 0$,*

$$\mathbb{E}[\Phi_\lambda(A_{t+1})] \leq (1 - 0.9\frac{\lambda\epsilon}{\sqrt{m}}) \cdot \Phi_\lambda(A_t) + \lambda\epsilon\sqrt{m}$$

The proof is a direct generalization of Lemma 6.3 and is very similar to Lemma D.1. Thus, we omit it here.

Algorithm 4 Matrix Sensing with Stochastic Gradient Descent (General Measurements).

procedure SGD_GENERAL($\{u_i, b_i\}_{i\in[m]}$) ▷ Lemma E.1
  $\tau \leftarrow \max_{i\in[m]} b_i$
  $A_1 \leftarrow \tau \cdot I$
  $z_i \leftarrow u_i^\top A_1 u_i - b_i$ for $i \in [m]$ ▷ $z \in \mathbb{R}^m$
  $w_{i,j} \leftarrow \langle u_i, u_j \rangle$ for $i, j \in [m]$ ▷ $w \in \mathbb{R}^{m\times m}$
  for $t = 1 \to T$ **do**
    Sample $\mathcal{B}_t \subset [m]$ of size B uniformly at random
    $\nabla \Phi_\lambda(A_t, \mathcal{B}_t) \leftarrow \frac{m}{B} \sum_{i\in\mathcal{B}_t} u_i u_i^\top \lambda \sinh(\lambda z_i)$ ▷ It takes $O(Bn^2)$-time
    $\|\nabla \Phi_\lambda(A_t)\|_F \leftarrow \lambda \left(\sum_{i,j=1}^m w_{i,j}^2 \sinh(\lambda z_i) \sinh(\lambda z_j) \right)^{1/2}$ ▷ It takes $O(m^2)$-time
    $A_{t+1} \leftarrow A_t - \epsilon \cdot \nabla \Phi_\lambda(A_t, \mathcal{B}_t) / \|\nabla \Phi_\lambda(A_t)\|_F$ ▷ It takes $O(n^2)$-time
    for $i \in [m]$ **do** ▷ Update z. It takes $O(mB)$-time
      for $j \in \mathcal{B}_t$ **do**
        $z_i \leftarrow z_i - \epsilon\lambda m w_{i,j}^2 \sinh(\lambda z_j) / (\|\nabla \Phi_\lambda(A_t)\|_F B)$
      end for
    end for
  end for
  return A_{T+1}
end procedure