Querying a Matrix Through Matrix-Vector Products

Xiaoming Sun
CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, Beijing, China
sunxiaoming@ict.ac.cn

David P. Woodruff
Carnegie Mellon University, Pittsburgh, PA, US
dwoodruf@andrew.cmu.edu

Guang Yang
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
Conflux, Beijing, China
guang.research@gmail.com

Jialin Zhang
CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, Beijing, China
zhangjialin@ict.ac.cn

Abstract

We consider algorithms with access to an unknown matrix $\mathbf{M} \in \mathbb{F}^{n \times d}$ via *matrix-vector products*, namely, the algorithm chooses vectors $\mathbf{v}^1, \ldots, \mathbf{v}^q$, and observes $\mathbf{M}\mathbf{v}^1, \ldots, \mathbf{M}\mathbf{v}^q$. Here the $\mathbf{v}^i$ can be randomized as well as chosen adaptively as a function of $\mathbf{M}\mathbf{v}^1, \ldots, \mathbf{M}\mathbf{v}^{i-1}$. Motivated by applications of sketching in distributed computation, linear algebra, and streaming models, as well as connections to areas such as communication complexity and property testing, we initiate the study of the number q of queries needed to solve various fundamental problems. We study problems in three broad categories, including linear algebra, statistics problems, and graph problems. For example, we consider the number of queries required to approximate the rank, trace, maximum eigenvalue, and norms of a matrix $\mathbf{M}$; to compute the AND/OR/Parity of each column or row of $\mathbf{M}$, to decide whether there are identical columns or rows in $\mathbf{M}$ or whether $\mathbf{M}$ is symmetric, diagonal, or unitary; or to compute whether a graph defined by $\mathbf{M}$ is connected or triangle-free. We also show separations for algorithms that are allowed to obtain matrix-vector products only by querying vectors on the right, versus algorithms that can query vectors on both the left and the right. We also show separations depending on the underlying field the matrix-vector product occurs in. For graph problems, we show separations depending on the form of the matrix (bipartite adjacency versus signed edge-vertex incidence matrix) to represent the graph.

Surprisingly, this fundamental model does not appear to have been studied on its own, and we believe a thorough investigation of problems in this model would be beneficial to a number of different application areas.

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Lower bounds and information complexity

Keywords and phrases Communication complexity, linear algebra, sketching

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.94

Category Track A: Algorithms, Complexity and Games

Related Version `https://arxiv.org/abs/1906.05736`

Acknowledgements We want to thank Roman Vershynin and Yan Shuo Tan for the helpful comments.

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 94; pp. 94:1–94:16

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Funding This work was supported in part by the National Natural Science Foundation of China Grants No. 61433014, 61761136014, 61872334, 61502449, 61602440, the 973 Program of China Grant No. 2016YFB1000201, and K.C.Wong Education Foundation. David Woodruff would like to thank the Chinese Academy of Sciences, as well as the Simons Institute for the Theory of Computing where part of this work was done. He also acknowledges partial support by the National Science Foundation under Grant No. CCF-1815840.

1 Introduction

Suppose there is an unknown matrix $\mathbf{M} \in \mathbb{F}^{n\times d}$ that you can only access via a sequence of *matrix-vector products* $\mathbf{M}\cdot\mathbf{v}^1,\ldots,\mathbf{M}\cdot\mathbf{v}^q$, where we call the vectors $\mathbf{v}^1,\ldots,\mathbf{v}^q$ the *query vectors*, which can be chosen in a randomized, possibly adaptive way. By adaptive, we mean that $\mathbf{v}^i$ can depend on $\mathbf{v}^1,\ldots,\mathbf{v}^{i-1}$ as well as $\mathbf{M}\mathbf{v}^1,\ldots,\mathbf{M}\mathbf{v}^{i-1}$. Here $\mathbb{F}$ is a field, and we study different fields for different applications. Suppose our goal is to determine if $\mathbf{M}$ satisfies a specific property $\mathcal{P}$, such as having approximately full rank, or for example whether $\mathbf{M}$ has two identical columns. A natural question is the following:

Question 1: How many queries q are necessary to determine if $\mathbf{M}$ has property $\mathcal{P}$?

A number of well-studied problems are special cases of this question, i.e., compressed sensing or sparse recovery, for which $\mathbf{M} \in \mathbb{R}^{1\times d}$ is an approximately k-sparse vector, and one would like a number q of queries close to k. It is known that if the query sequence is non-adaptive, meaning $\mathbf{v}^1,\ldots,\mathbf{v}^q$ are chosen before making any queries, then $q = \Theta(k\log(n/k))$ is necessary and sufficient [10, 5] to recover an approximately k-sparse vector[1]. However, if the queries can be adaptive, then $q = O(k\log\log n)$ queries suffice [12], while there is a lower bound of $\Omega(k+\log\log n)$ [24] (see also recent work [23, 13]).

The above problem is representative of an emerging field called *linear sketching* which is the underlying technique behind a number of algorithmic advances the past two decades. In this model one queries $\mathbf{M}\cdot\mathbf{v}^1,\ldots,\mathbf{M}\cdot\mathbf{v}^r$ for non-adaptive queries $\mathbf{v}^1,\ldots,\mathbf{v}^r$. For brevity we write this as $\mathbf{M}\cdot\mathbf{V}$, where $\mathbf{V}\in\mathbb{F}^{d\times r}$ has i-th column equal to $\mathbf{v}^i$. Linear sketching has played a central role in the development of streaming algorithms [2]. Perhaps more surprisingly, linear sketches are also known to achieve the minimal space necessary *of any, possibly non-linear, algorithm* for processing dynamic data streams under certain general conditions [20, 1, 15], which is an essential result for proving a number of lower bounds for approximating matchings in a stream [18, 4]. Linear sketching has also led to the fastest known algorithms for problems in numerical linear algebra, such as least squares regression and low rank approximation; for a survey see [29]. Note that given $\mathbf{M}\cdot\mathbf{V}$ and $\mathbf{M}'\cdot\mathbf{V}$, by linearity one can compute $(\mathbf{M}+\mathbf{M}')\cdot\mathbf{V} = \mathbf{M}\cdot\mathbf{V}+\mathbf{M}'\cdot\mathbf{V}$. This basic versatility property allows for fast updates in a data stream and mergeability in environments such as MapReduce and other distributed models of computation.

Given the applications above, we consider Question 1 an important question to understand for many different properties $\mathcal{P}$ of interest, which we describe in more detail below. A central goal of this work is to answer Question 1 for such properties and to propose this be a natural model of study in its own right.

One notable difference with our model and a number of appications of linear sketching is that we will allow for adaptive query sequences. In fact, our upper bounds will be non-adaptive, and our nearly matching lower bounds for each problem we consider will hold even

[1] Here the goal is to output a vector $\mathbf{M}'$ for which $\|\mathbf{M}-\mathbf{M}'\|_2 \le (1+\epsilon)\|\mathbf{M}-\mathbf{M}_k\|_2$, where $\mathbf{M}_k$ is the best k-sparse approximation to $\mathbf{M}$, and ϵ is a constant.

for adaptive query sequences. Our model is also related to property testing, where one tries to infer properties of a large unknown object by (possibly adaptively) sampling a sublinear number of locations of that object. We argue that linear queries are a natural extension of sampling locations of an object, and that this is a natural "sampling model" not only because of the desired properties of the distributed, linear algebra, and streaming applications above, but sometimes also for physical constraints, e.g., in compressed sensing, where optical devices naturally capture linear measurements.

From a theoretical standpoint, any property testing algorithm, i.e., one that samples q entries of $\mathbf{M}$, can be implemented in our model with q linear queries. However, our model gives the algorithm much more flexibility. From a lower bound perspective, as in the case of property testing [8], some of our lower bounds will be derived from communication complexity. However, not all of our bounds can be proved this way. For example, one notable result we show is an optimal lower bound on the number of queries needed to approximate the rank of $\mathbf{M} \in \mathbb{R}^{n\times n}$ up to a factor t by randomized, possibly adaptive algorithms; we show that $\frac{n}{t} + 1$ queries are necessary and sufficient. A natural alternative way to prove this would be to give part of the matrix to Alice, part of to Bob, and have the players exchange the $\mathbf{M}^L\mathbf{v}^i$ and $\mathbf{M}^R\mathbf{v}^i$, where $\mathbf{M} = \mathbf{M}^L + \mathbf{M}^R$ and $\mathbf{M}^L$ is Alice's part and $\mathbf{M}^R$ is Bob's part. Then, if the 2-player randomized communication complexity of approximating the rank of $\mathbf{M}$ up to a factor of t were known to be $\Omega(n^2/t)$, we would obtain a nearly-matching query lower bound of $\Omega(n/(t(b + \log n)))$, where b is the number of bits needed to specify the entries of $\mathbf{M}$ and the queries. However, it is unknown what the 2-player communication complexity of approximating the rank of $\mathbf{M}$ up to a factor t is over $\mathbb{R}$! We are not aware of any lower bound better than $\Omega(1)$ for constant t for this problem for adaptive queries. We note that for non-adaptive queries, there is an $\Omega(n^2)$ sketching lower bound over the reals given in [19], and an $\Omega(n^2/\log p)$ lower bound for finite fields (of size p) in [3]. There is also a property testing lower bound in [6], though such a lower bound makes additional assumptions on the input. Thus, our model gives a new lens to study this problem from, from which we are able to derive strong lower bounds for adaptive queries. Our techniques could be helpful for proving lower bounds in existing models, such as two-party communication complexity.

Our model is also related to linear decision tree complexity, see, e.g., [7, 14], though such lower bounds typically involve just seeing a threshold applied to $\mathbf{M}\mathbf{v}^i$, and typically $\mathbf{M}$ is a vector. In our case, we observe the entire output vector $\mathbf{M}\mathbf{v}^i$.

An interesting twist in our model is that in our formulation above, we only allowed to query $\mathbf{M}$ via matrix-vector products *on the right*, i.e., of the form $\mathbf{M} \cdot \mathbf{v}^i$. One could ask if there are natural properties $\mathcal{P}$ of $\mathbf{M}$ for which the number q_L of queries one would need to make if querying $\mathbf{M}$ via queries of the form $(\mathbf{u}^1)^T\mathbf{M}, (\mathbf{u}^2)^T\mathbf{M}, \ldots, (\mathbf{u}^{q_L})^T\mathbf{M}$ can be significantly smaller than the number q_R of queries one would need to make if querying $\mathbf{M}$ via queries of the form $\mathbf{M}\mathbf{u}^1, \mathbf{M}\mathbf{u}^2, \ldots, \mathbf{M}\mathbf{u}^{q_R}$:

Question 2: Are there natural problems for which $q_L \ll q_R$?

We show that this is in fact the case, namely, if we can only multiply on the right, then it takes $\Omega(n/\log n)$ queries to determine if there is a *column* of a matrix $\mathbf{M} \in \{0,1\}^{n\times n}$ which is all 1s. However, if we can multiply on the left, then the single query $(1, 1, \ldots, 1)$ can determine this.

We study a few problems around Question 2, which is motivated from several perspectives. First, matrices might be stored on computers in a specific encoding, e.g., a sparse row format, from which it may be much easier to multiply on the right than on the left. Also, in compressed sensing, it may be natural for physical reasons to obtain linear combinations of columns rather than rows.

Another important question is how the query complexity depends on the *underlying field* for which matrix-vector products are performed. Might it be that for a natural problem the query complexity if the matrix-vector products are performed modulo 2 is much higher than if the matrix-vector products are performed over the reals?

Question 3: Is there a natural problem for which the query complexity in our model over $\mathbb{F}[2]$ is much larger than that over the reals?

Yet another important application of this model is to querying graphs. A natural question is which representation to use for the graph. For example, a natural representation of a graph on n vertices is through its adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$, where $\mathbf{A}_{i,j} = 1$ if and only if $\{i,j\}$ occurs as an edge. A natural representation for a bipartite graph with n vertices in each part could be an $n \times n$ matrix $\mathbf{A}$ where $\mathbf{A}_{i,j} = 1$ iff there is an edge from the i-th left vertex to the j-th right vertex. Yet another representation could be the $\binom{n}{2} \times n$ edge-vertex incidence matrix, where the $\{i,j\}$-th row is either 0, or has exactly two ones, one in location i and one in location j. One often considers a signed edge-vertex incidence matrix, where one first arbitrarily fixes an ordering on the vertices and then the $\{i,j\}$-th entry has a 1 in the i-th position and a -1 in the j-th position if $i > j$, otherwise positions i and j are swapped. Yet another possible representation of a graph is through its Laplacian.

Question 4: Do some natural representations of graphs admit much more efficient query algorithms for certain problems than other natural representations?

We note that in the data stream model, where one sees a long sequence of insertions and deletions to the edges of a graph, each of the matrix representations above can be simulated and so they lead to the same complexity. We will show, perhaps surprisingly, that in this model there can be an exponential difference in the query complexity for two different natural representations of a graph for the same problem.

We next get into the details of our results. We would like to stress that even basic problems in this model are not immediately obvious how to tackle. As a puzzle for the reader, what is the query complexity of determining if a matrix $\mathbf{M} \in \mathbb{F}^{n \times n}$ is symmetric if one can only query vectors on the right? We will answer this later in the paper.

1.1 Formal Model and Our Results

We now describe our model and results formally in terms of an oracle. The oracle has a matrix $\mathbf{M} \in \mathbb{F}^{m \times n}$, for some underlying field $\mathbb{F}$ that we specify in each application. We can only query this matrix via matrix-vector products, i.e., we pick an arbitrary vector $\mathbf{x}$ and send it to the oracle, and the oracle will respond with a vector $\mathbf{y} = \mathbf{M} \cdot \mathbf{x}$. We focus our attention when the queries only occur on the right. Our goal is to approximate or test a number of properties of $\mathbf{M}$ with a minimal number of queries, i.e., to answer Question 1 for a large number of different application areas.

We study a number of problems as summarized in the table. Due to the space limitation, we leave some proofs in the full version. We assume $\mathbf{M}$ is an $m \times n$ matrix and $\varepsilon > 0$ is a parameter of the problem. The bounds hold for constant probability algorithms. In some problems, such as testing whether the matrix is a diagonal matrix, we always assume $m = n$, and in the graph testing problems we explicitly describe how the graph is represented using $\mathbf{M}$. Interestingly, we are able to prove very strong lower bounds for approximating the rank, which as described above, are unknown to hold for randomized communication complexity.

Motivated be streaming and statistics questions, we next study the query complexity of approximating the norm of each row of $\mathbf{M}$. We also study the computation of the majority or parity of each column or row of $\mathbf{M}$, the AND/OR of each column or row of $\mathbf{M}$, or equivalently, whether $\mathbf{M}$ has an all ones column or row, whether $\mathbf{M}$ has two identical columns or rows, and whether $\mathbf{M}$ contains an unusually large-normed row, i.e., a "heavy hitter". Here we show there are natural problems, such as computing the parity of all columns, which can be solved with 1 query if sketching on the left, but require $\Omega(n)$ queries if sketching on the right, thus answering Question 2. We also answer Question 3, observing for the natural problem of testing if a row is all ones, a single deterministic query suffices over the reals but over $\mathbb{F}[2]$ this deterministically requires $\Omega(n)$ queries.

For graph problems, we first argue if the graph is presented as an $n \times n$ bipartite adjacency matrix $\mathbf{M}$, then it requires $\Omega(n/\log n)$ possibly adaptive queries to determine if the graph is connected. In contrast, if the graph is presented as an $n \times \binom{n}{2}$ signed vertex-edge incidence matrix, then $\mathsf{polylog}(n)$ non-adaptive queries suffices. This answers Question 4, showing that the type of representation of the graph is critical in this model. Motivated by a large body of recent work on triangle counting (see, e.g., [11] and the references therein), we also give strong negative results for this problem in our model, which as with all of our lower bounds unless explicitly stated otherwise, hold even for algorithms which perform adaptive queries.

Table 1 Our Results.

Problem	Query Complexity
Linear Algebra Problems	
Approximate Rank (for any $p' > p$ distinguishing Rank$\leq p$ from Rank p')	$p+1$ (Section 3.1)
Trace Estimation	$\Omega(n/\log n)$ (Section 3.2)
Symmetric Matrix / Diagonal Matrix	$O(1)$ (Section 3.3 and full version)
Unitary Matrix	1 (full version)
Approximate Maximum Eigenvalue	$\Theta(\varepsilon^{-0.5}\log n)$ for adaptive queries, $\Theta(n)$ for non-adaptive queries (full version)
Streaming and Statistics Problems	
All Ones Column	$\Theta(n)$ over $\mathbb{F}[2]$, $\Omega(n/\log n)$ over $\mathbb{R}$ (Section 4.1)
Two Identical Columns Two Identical Rows	$\Theta(n)$ $O(\log m)$ (Section 4.2)
Approximate Row Norms / Heavy Hitters	$O\left(\varepsilon^{-2}\log m\right)$ (full version)
Majority of Columns Majority of Rows	$\Omega(n/\log n)$ over $\mathbb{R}$ $O(1)$ over $\mathbb{R}$ (full version)
Parity of Columns Parity of Rows	$\Theta(n)$ $O(1)$ (full version)
Graph Problems	
Connectivity given Bipartite Adjacency Matrix Connectivity given Signed Edge-Vertex Matrix	$\Omega(n/\log n)$ (Section 5.1) $O(\mathsf{polylog}(n))$ ([16], noted in Section 5.1)
Triangle Detection	$\Omega(n/\log n)$ (Section 5.2)

2 Preliminaries

We use capital bold letters, e.g., $\mathbf{A}, \mathbf{B}, \mathbf{M}$, to denote matrices, and use lowercase bold letters, e.g., $\mathbf{x}, \mathbf{y}$, to denote column vectors. Sometimes we write a matrix as a list of column vectors in square brackets, e.g., $\mathbf{M} = [\mathbf{m}_1, \ldots, \mathbf{m}_n]$. We use calligraphic letters, e.g., $\mathcal{D}$, to denote probability distributions, and use $\mathbf{M} \leftarrow \mathcal{D}$ to denote that $\mathbf{M}$ is sampled from distribution $\mathcal{D}$. In particular, we use $\mathcal{G}$ to denote a Gaussian distribution and $\mathbf{G}$ for a matrix whose entries are sampled from an independently and identically distributed (denoted as i.i.d. in the following) Gaussian distribution.

We call a matrix $\mathbf{M}$ i.i.d. Gaussian if each element is i.i.d. Gaussian. It is easy to check that if matrix $\mathbf{G}$ is a $p \times n$ i.i.d. Gaussian matrix, and $\mathbf{R}$ is an $n \times n$ rotation matrix, then $\mathbf{G} \times \mathbf{R}$ is still i.i.d. Gaussian, and has the same probability distribution of $\mathbf{G}$.

The *total variation distance*, sometimes called the statistical distance, between two probability measures P and Q is defined as $\mathsf{D}_{\mathsf{TV}}(P, Q) \stackrel{\text{def}}{=} \sup_A |P(A) - Q(A)|$.

Let $\mathbf{X}$ be an $n \times m$ matrix with each row i.i.d. drawn from an m-variate normal distribution $N(0, \Sigma)$. Then the distribution of the $m \times m$ random matrix $\mathbf{A} = \mathbf{X}^T\mathbf{X}$ is called the *Wishart distribution* with n degrees of freedom and covariance matrix Σ, denoted by $W_m(n, \Sigma)$. The distribution of eigenvalues of $\mathbf{A}$ is characterized in the following lemma.

▶ **Lemma 1** (Corollary 3.2.19 in [17]). *If $\mathbf{A}$ is $W_m(n, \lambda I_m)$, with $n > m - 1$, the joint density function of the eigenvalues $\mathbf{\Lambda} = (\lambda_1, \ldots, \lambda_m)$ of $\mathbf{A}$ (in descending order) is*

$$f(\mathbf{\Lambda}) = \frac{\pi^{m^2/2}}{(2\lambda)^{mn/2}\Gamma_m(m/2)\Gamma_n(n/2)} \exp\left(-\frac{1}{2\lambda}\sum_{i=1}^{m}\lambda_i\right)\prod_{i=1}^{m}\lambda_i^{(n-m-1)/2}\prod_{1\le i<j\le m}(\lambda_i - \lambda_j)$$

In particular, for $\lambda = 1$ and $n = m$, $\exists$ a constant Z_m independent from $\lambda_1, \ldots, \lambda_m$, such that

$$f(\mathbf{\Lambda}) = \frac{1}{Z_m}\exp\left(-\frac{1}{2}\sum_{i=1}^{m}\lambda_i\right)\prod_{i=1}^{m}\lambda_i^{-1/2}\prod_{1\le i<j\le m}(\lambda_i - \lambda_j)$$

3 Linear Algebra Problems

In this section we present our lower bound for rank approximation in Section 3.1, trace estimation in Section 3.2, and testing whether a matrix is symmetric. The results for testing diagonal or unitary matrices, and approximating the maximum eigenvalue is contained in the full version of our paper.

3.1 Lower Bound for Rank Approximation

In this section, we discuss how to approximate the rank of a given matrix $\mathbf{M}$ over the reals when the queries consist of right multiplication by vectors. A naïve algorithm to learn the rank is to pick random Gaussian query vectors non-adaptively. In order to approximate the rank, that is, to distinguish whether $\mathsf{rank}(\mathbf{M}) \le p$ or $\mathsf{rank}(\mathbf{M}) \ge p + 1$, this algorithm needs at least $p + 1$ queries, and it is not hard to see that the algorithm succeeds with probability 1. Indeed, if $\mathbf{H} \in \mathbb{R}^{n\times(p+1)}$ is the random Gaussian query matrix, and $\mathbf{M}$ the unknown $n \times n$ matrix, then writing $\mathbf{M}$ in its thin singular value decomposition as $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ have orthonormal columns, and $\mathbf{\Sigma}$ has positive diagonal entries, we have that $\mathrm{rank}(\mathbf{M} \cdot \mathbf{H}) = \mathrm{rank}(\mathbf{V}^T\mathbf{H})$, which by rotational invariance of the Gaussian distribution is the the same as the rank of a random Gaussian matrix, which will be the minimum of $p + 1$ and the rank of $\mathbf{M}$ with probability 1.

In the following, we will show that we cannot expect anything better. We will first show for non-adaptive queries, at least $p+1$ queries are necessary to learn the approximate rank. Then we generalize our results to adaptive queries. Our results hold for randomized algorithms by applying Yao's minimax principle.

3.1.1 Non-Adaptive Query Protocols

▶ **Theorem 2.** *Let constant $\varepsilon > 0$ be the error tolerance and let $\mathbf{M}$ be an $n \times n$ oracle matrix and suppose to start that we make non-adaptive queries. For integer $p < p' \leq n$, at least $p+1$ queries are necessary to distinguish $\mathsf{rank}(\mathbf{M}) \leq p$ from $\mathsf{rank}(\mathbf{M}) \geq p'$ with advantage $\geq \varepsilon$.*

Proof. Given any algorithm distinguishing $\mathsf{rank}(\mathbf{M}) \leq p$ from $\mathsf{rank}(\mathbf{M}) \geq p'$ for some $p' < n$, we can determine whether a $p' \times p'$ matrix $\mathbf{M}'$ has full rank p' or $\mathsf{rank}(\mathbf{M}') \leq p$, by padding $\mathbf{M}'$ to an $n \times n$ matrix $\mathbf{M}$. Therefore in what follows it suffices to prove the lower bound for two $n \times n$ matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ where $\mathsf{rank}(\mathbf{M}_1) \leq p$ and $\mathsf{rank}(\mathbf{M}_2) = n$:

1. $\mathbf{M}_1 = \mathbf{U} \times \mathbf{G}^T$;
2. $\mathbf{M}_2 = \mathbf{U} \times \mathbf{G}^T + \frac{1}{Z(n)} \cdot \mathbf{U}^\perp \times \mathbf{H}^T$.

Here $\mathbf{U}$ has p columns and $\mathbf{U}^\perp$ has $(n-p)$ columns such that $\left[\mathbf{U}, \mathbf{U}^\perp\right]$ forms an $n \times n$ random orthonormal basis, $\mathbf{G}^T$ and $\mathbf{H}^T$ are $p \times n$ and $(n-p) \times n$ matrices whose entries sampled i.i.d. from the standard Gaussian distribution, and $Z(n)$ is a function in n which will be specified later. It immediately follows that $\mathsf{rank}(\mathbf{M}_1) \leq p$ and $\mathsf{rank}(\mathbf{M}_2) = n$ with overwhelmingly high probability. Then we assume $\mathsf{rank}(\mathbf{M}_2) = n$ and discuss the query lower bound for distinguishing $\mathbf{M}_1$ from $\mathbf{M}_2$.

Given $\mathbf{M} \in \{\mathbf{M}_1, \mathbf{M}_2\}$, without loss of generality we denote the q non-adaptive queries with an $n \times q$ orthonormal[2] matrix $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_q]$, where $q \leq p$ and each $n \times 1$ column vector $\mathbf{v}_i$ is a query to the oracle of matrix $\mathbf{M}$ which gets response $\mathbf{M} \cdot \mathbf{v}_i$, for $i \in [q]$. Then, it suffices to show that the following two distributions are hard to distinguish:

1. $\mathbf{M}_1 \times \mathbf{V} \equiv \mathbf{U}\mathbf{W}$, where $\mathbf{W} = \mathbf{G}^T\mathbf{V}$;
2. $\mathbf{M}_2 \times \mathbf{V} \equiv \mathbf{U}\mathbf{W} + \frac{1}{Z(n)} \cdot \mathbf{U}^\perp\mathbf{W}'$, where $\mathbf{W}' = \mathbf{H}^T\mathbf{V}$.

Note that $\left[\mathbf{U}, \mathbf{U}^\perp\right]$ is orthonormal, and hence $\mathbf{U}^T\mathbf{U} = \mathbf{I}_p$, $\left(\mathbf{U}^\perp\right)^T\mathbf{U}^\perp = \mathbf{I}_{n-p}$, $\mathbf{U}^T\mathbf{U}^\perp = \mathbf{0}_{p\times(n-p)}$. We introduce Lemma 3 to eliminate $\mathbf{U}, \mathbf{U}^\perp$ in the representation of $\mathbf{M} \times \mathbf{V}$.

▶ **Lemma 3.** *For $\mathbf{M}_1, \mathbf{M}_2$ and $\mathbf{V}$ defined as above, there is*

$$\mathsf{D}_{\mathsf{TV}}(\mathbf{M}_1\mathbf{V}, \mathbf{M}_2\mathbf{V}) = \mathsf{D}_{\mathsf{TV}}\left((\mathbf{M}_1\mathbf{V})^T\mathbf{M}_1\mathbf{V}, (\mathbf{M}_2\mathbf{V})^T\mathbf{M}_2\mathbf{V}\right)$$

Proof. The direction $\mathsf{D}_{\mathsf{TV}}(\mathbf{M}_1\mathbf{V}, \mathbf{M}_2\mathbf{V}) \geq \mathsf{D}_{\mathsf{TV}}\left((\mathbf{M}_1\mathbf{V})^T\mathbf{M}_1\mathbf{V}, (\mathbf{M}_2\mathbf{V})^T\mathbf{M}_2\mathbf{V}\right)$ is trivial following data processing inequality (i.e. for every $\mathbf{X}, \mathbf{Y}$ and function f, $\mathsf{D}_{\mathsf{TV}}(\mathbf{X}, \mathbf{Y}) \geq \mathsf{D}_{\mathsf{TV}}(f(\mathbf{X}), f(\mathbf{Y}))$). In what follows we only prove the other direction.

First we notice that for every fixed $n \times n$ orthonormal matrix $\mathbf{R}$ and for a random matrix $\mathbf{M}$ sampled as $\mathbf{M}_1$ or $\mathbf{M}_2$, the product $\mathbf{N} \stackrel{\text{def}}{=} \mathbf{R}\mathbf{M}$ follows exactly the same distribution of $\mathbf{M}$. Thus $\mathbf{N}\mathbf{V}$ and $\mathbf{M}\mathbf{V}$ are identically distributed.

Then, from a random sample $\mathbf{V}^T\mathbf{M}^T\mathbf{M}\mathbf{V}$ we can find $\mathbf{M}'$ such that $\mathbf{V}^T\mathbf{M}^T\mathbf{M}\mathbf{V} = (\mathbf{M}')^T\mathbf{M}'$ and $\mathbf{M}' = \mathbf{S}\mathbf{M}\mathbf{V}$ for some orthonormal matrix $\mathbf{S}$ and orthonormal query matrix $\mathbf{V}$. Although $\mathbf{M}'$ is not necessarily the same as $\mathbf{M}\mathbf{V}$ because of $\mathbf{S}$, we have $\mathbf{R}\mathbf{M}' \sim$

[2] Any non-orthonormal queries can be made orthonormal using a change of basis in post-processing.

$\mathbf{NV} \sim \mathbf{MV}$ for a uniformly random orthonormal matrix $\mathbf{R}$. Thus we transform a random sample from $\mathbf{V}^T\mathbf{M}^T\mathbf{MV}$ into a sample from $\mathbf{MV}$ via $\mathbf{RM'}$, and hence $\mathsf{D}_{\mathsf{TV}}(\mathbf{M}_1\mathbf{V}, \mathbf{M}_2\mathbf{V}) \leq \mathsf{D}_{\mathsf{TV}}\left((\mathbf{M}_1\mathbf{V})^T \mathbf{M}_1\mathbf{V}, (\mathbf{M}_2\mathbf{V})^T \mathbf{M}_2\mathbf{V}\right)$. ◀

Using Lemma 3, it suffices to prove an upper bound for $\mathsf{D}_{\mathsf{TV}}(\mathbf{\Lambda}, \mathbf{\Lambda'})$ as follows:

$$\mathsf{D}_{\mathsf{TV}}\left(\mathbf{UW}, \mathbf{UW} + \frac{\mathbf{U}^\perp\mathbf{W'}}{Z(n)}\right) = \mathsf{D}_{\mathsf{TV}}\left((\mathbf{UW})^T(\mathbf{UW}), \left(\mathbf{UW} + \frac{\mathbf{U}^\perp\mathbf{W'}}{Z(n)}\right)^T(\mathbf{UW} + \frac{\mathbf{U}^\perp\mathbf{W'}}{Z(n)})\right)$$
$$= \mathsf{D}_{\mathsf{TV}}\left(\mathbf{W}^T\mathbf{W}, \mathbf{W}^T\mathbf{W} + \frac{(\mathbf{W'})^T\mathbf{W'}}{Z^2(n)}\right) \leq \mathsf{D}_{\mathsf{TV}}\left(\mathbf{\Lambda}, \mathbf{\Lambda'}\right)$$

where $\mathbf{\Lambda} = \mathsf{diag}(\lambda_1, \ldots, \lambda_q), \mathbf{\Lambda'} = \mathsf{diag}(\lambda'_1, \ldots, \lambda'_q)$ are diagonal matrices such that $\mathbf{W}^T\mathbf{W} = \mathbf{A}^T\mathbf{\Lambda}\mathbf{A}$ and $\mathbf{W}^T\mathbf{W} + \frac{(\mathbf{W'})^T\mathbf{W'}}{Z^2(n)} = \mathbf{B}^T\mathbf{\Lambda'}\mathbf{B}$ for orthonormal matrices $\mathbf{A}$ and $\mathbf{B}$. The inequality follows because any algorithm separating $\mathbf{W}^T\mathbf{W}$ from $\mathbf{W}^T\mathbf{W} + \frac{(\mathbf{W'})^T\mathbf{W'}}{Z^2(n)}$ implies a separation of $\mathbf{\Lambda}$ from $\mathbf{\Lambda'}$ with the same advantage, by multiplying by random orthonormal matrices.

By Weyl's inequality [28, 31], for every $i \in [q]$, $\lambda'_i \in [\lambda_i - \|\mathbf{\Lambda'} - \mathbf{\Lambda}\|_2, \lambda_i + \|\mathbf{\Lambda'} - \mathbf{\Lambda}\|_2]$, and hence $\lambda'_i \in \left[\lambda_i - O\left(\frac{\|\mathbf{W'}\|_2}{Z^2(n)}\right), \lambda_i + O\left(\frac{\|\mathbf{W'}\|_2}{Z^2(n)}\right)\right]$. Notice that $\mathbf{W'}$ is an $(n-p) \times q$ i.i.d. Gaussian matrix, and hence $\|\mathbf{W'}\|_2^2$ is a chi-squared variable with $(n-p)q$ degrees of freedom, which is bounded by $O((n-p)q)$ with high probability (c.f. Example 2.12 in [27]). Recalling that $q \leq p$, in what follows we condition on the event $\lambda'_i \in \left[\lambda_i - O\left(\frac{np}{Z^2(n)}\right), \lambda_i + O\left(\frac{np}{Z^2(n)}\right)\right]$.

We then show the gaps between eigenvalues λ_i are sufficiently large. Note that since $\mathbf{G}^T$ is i.i.d Gaussian and $\mathbf{V}$ is an orthonormal matrix, each row in $\mathbf{W} = \mathbf{G}^T\mathbf{V}$ is independently drawn from an q-variate normal distribution, thus the probability distribution of $\mathbf{W}^T\mathbf{W}$ is a Wishart distribution $W_q(p, I_q)$. Let $q = p$ and $\lambda_1, \ldots, \lambda_p$ be sorted in descending order. Then by Lemma 1 the density function of $\mathbf{\Lambda}$ is:

$$f(\mathbf{\Lambda}) = \frac{1}{Z_p}\exp\left(-\frac{1}{2}\sum_{i=1}^{p}\lambda_i\right)\prod_{i=1}^{p}\lambda_i^{-1/2}\prod_{1\leq i<j\leq p}(\lambda_i - \lambda_j) \tag{1}$$

Let $\mathcal{E}$ denote the event that $\lambda_p \geq \frac{0.01}{\sqrt{n}}$ and $\forall 1 \leq i < j \leq p, \lambda_i - \lambda_j \geq \gamma = 2^{-\Theta(p^2 \log p)}$.

▶ **Lemma 4.** *For $\mathbf{W}^T\mathbf{W}$ defined as above and sufficiently small $\gamma = 2^{-\Theta(p^2 \log n)}$, $\Pr[\mathcal{E}] > 0.9$.*

Proof. By equation (2) in [25] we know that $\Pr[\sqrt{n}\lambda_p \geq y] = \exp\left(-(y^2/2 + y)\right)$. Thus for $y = 0.01$ and $\mathcal{E}_0 \stackrel{\text{def}}{=} \{\lambda_p \geq 0.01/\sqrt{n}\}$ we get:

$$\Pr[\mathcal{E}_0] = \Pr\left[\lambda_p \geq \frac{0.01}{\sqrt{n}}\right] = \exp(-0.01005) > 0.99000033$$

Also, we note that for every i, $\Pr[|\lambda_i| \leq 100n] \geq 1 - 2\exp(-32n)$, by setting $t = 8\sqrt{n}$ in Corollary 5.35 of [26]. In what follows we conditioned on $\mathcal{E}'_0$ that $|\lambda_i| \leq 100n$ for every $i \in [p]$.

Then we consider the joint distribution μ of $\lambda_1, \ldots, \lambda_p$ in $\mathbf{\Lambda}$. Let the $\mathcal{E}_i \stackrel{\text{def}}{=} \{\lambda_i - \lambda_{i+1} < \gamma\}$ be the event that λ_i and λ_{i+1} has a gap smaller than γ. Thus $\mathcal{E} = \mathcal{E}_0 \wedge \left(\wedge_{i=1}^{p-1}\overline{\mathcal{E}_i}\right)$. To lower bound $\Pr[\mathcal{E}]$, we need to upper bound the probability of $\mathcal{E}_i$ for $1 \leq i \leq p-1$.

Let f be the density function of μ as in (1), and let $\mathsf{Leb}(\cdot)$ be the Lebesgue measure in n dimensions. Then for every i,

$$\Pr[\mathcal{E}_i \mid \mathcal{E}'_0] = \mu(\lambda_i - \lambda_{i+1} < \gamma) \leq \mathsf{Leb}(\lambda_i - \lambda_{i+1} < \gamma) \cdot |f|_\infty = O(\gamma/n) \cdot |f|_\infty$$

Note that conditioning on $\mathcal{E}_0$ such that $\lambda_p \geq 0.01/\sqrt{n}$, the density function f is bounded as:

$$|f|_\infty \leq O\left(\exp\left(-\frac{1}{2}\lambda_1\right)\left(100\sqrt{n}\right)^{p/2}\lambda_1^{p^2/2}\right) = 2^{O\left(p^2\log n\right)}$$

As a result, we get $\Pr\left[\mathcal{E}_i \wedge \mathcal{E}_0 \mid \mathcal{E}_0'\right] \leq \gamma \cdot 2^{O\left(p^2\log n\right)}$.

Therefore, the probability of $\mathcal{E}$ is lower bounded for sufficiently small $\gamma = 2^{-\Theta(p^2\log n)}$,

$$\begin{aligned}
\Pr\left[\mathcal{E}\right] &\geq \Pr\left[\mathcal{E}_0'\right] \cdot \Pr\left[\mathcal{E}_0 \wedge \left(\wedge_{i=1}^{p-1}\overline{\mathcal{E}_i}\right) \mid \mathcal{E}_0'\right] \\
&\geq \Pr\left[\mathcal{E}_0'\right] \cdot \left(\Pr\left[\mathcal{E}_0 \mid \mathcal{E}_0'\right] - \sum_{i=1}^{p-1}\Pr\left[\mathcal{E}_i \wedge \mathcal{E}_0 \mid \mathcal{E}_0'\right]\right) \\
&> (1-2p\exp(-32n)) \cdot \left(0.99000033 - (p-1)\gamma \cdot 2^{O\left(p^2\log n\right)}\right) > 0.9
\end{aligned}$$

◀

Conditioned on event $\mathcal{E}$ and recalling that $\lambda_i' \in \left[\lambda_i - O\left(\frac{np}{Z^2(n)}\right), \lambda_i + O\left(\frac{np}{Z^2(n)}\right)\right]$, the probability density of $\mathbf{\Lambda}'$ has only a negligible difference from that of $\mathbf{\Lambda}$, since the small disturbance of eigenvalues is dominated by the corresponding terms in $f(\mathbf{\Lambda})$.

$$\begin{aligned}
\frac{f(\mathbf{\Lambda}')}{f(\mathbf{\Lambda})} &= \frac{\exp\left(-\frac{1}{2}\sum_{i=1}^p \lambda_i'\right)\prod_{i=1}^p \lambda_i'^{-1/2}\prod_{1\leq i<j\leq p}(\lambda_i'-\lambda_j')}{\exp\left(-\frac{1}{2}\sum_{i=1}^p \lambda_i\right)\prod_{i=1}^p \lambda_i^{-1/2}\prod_{1\leq i<j\leq p}(\lambda_i-\lambda_j)} \\
&\leq \exp\left(\frac{p\cdot np}{Z^2(n)}\right)\left(\frac{\lambda_p - \frac{np}{Z^2(n)}}{\lambda_p}\right)^{-p/2}\prod_{1\leq i<j\leq p}\frac{\lambda_i-\lambda_j+\frac{2np}{Z^2(n)}}{\lambda_i-\lambda_j} \\
&\leq \exp\left(\frac{np^2}{Z^2(n)}\right)\cdot\left(1+\frac{np}{\lambda_p\cdot Z^2(n)}\right)^p\left(1+\frac{2np}{Z^2(n)\cdot\min_{i\neq j}|\lambda_i-\lambda_j|}\right)^{p(p-1)/2} \\
&\leq \exp\left(\frac{np^2}{Z^2(n)}\right)\cdot\left(1+\frac{100\sqrt{n}\cdot np}{Z^2(n)}\right)^p\left(1+\frac{2np}{Z^2(n)\cdot\gamma}\right)^{p(p-1)/2} = 1+O\left(\frac{np^3\gamma^{-1}}{Z^2(n)}\right)
\end{aligned}$$

Similarly we can prove $f(\mathbf{\Lambda}')/f(\mathbf{\Lambda}) \geq 1 - O\left(np^3\gamma^{-1}/Z^2(n)\right)$. Thus the total variation distance between $\mathbf{\Lambda}$ and $\mathbf{\Lambda}'$ conditioned on $\mathcal{E}$ is $\mathsf{D}_{\mathsf{TV}}\left(\mathbf{\Lambda}, \mathbf{\Lambda}' \mid \mathcal{E}\right) \leq O\left(np^3\gamma^{-1}/Z^2(n)\right) = O\left(1/n^2\right)$ for sufficiently large $Z(n) \geq (np)^{1.5}\gamma^{-0.5} = 2^{\Theta(p^2\log n)}$. Thus, for sufficiently large n, we have:

$$\mathsf{D}_{\mathsf{TV}}\left(\mathbf{\Lambda}, \mathbf{\Lambda}'\right) \leq \Pr[\overline{\mathcal{E}}] + \Pr[\mathcal{E}]\cdot\mathsf{D}_{\mathsf{TV}}\left(\mathbf{\Lambda}, \mathbf{\Lambda}' \mid \mathcal{E}\right) \leq 0.1 + O\left(1/n^2\right) < 0.11$$

Therefore, with as many as $q = p$ non-adaptive queries to the oracle matrix $\mathbf{M}$, the two distributions $\mathbf{M}_1$ and $\mathbf{M}_2$ cannot be distinguished with advantage greater than 0.11. At least $p+1$ queries are necessary to distinguish those two matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ of rank $\leq p$ and rank n, respectively.

Indeed, the above argument holds for every constant advantage ε if $y = \varepsilon/3$, $t > \sqrt{12n/\varepsilon}$, and γ sufficiently small in the proof of Lemma 4, and let $Z(n)$ be sufficiently large. ◀

3.1.2 Equivalence Between Adaptive and Non-Adaptive Protocols

Now, we consider the adaptive query matrix $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_q]$ where $\mathbf{v}_i$ is the i-th query vector. Without loss of generality, we can assume that $\forall i, \mathbf{v}_i$ is a unit vector and it is orthogonal to query vectors $\mathbf{v}_1, \cdots, \mathbf{v}_{i-1}$. This gives us the following formal definition of an adaptive query protocol.

▶ **Definition 5.** *For a target matrix* $\mathbf{M}$*, an adaptive query protocol* P *will output a sequence of query vectors* $\mathbf{v}_1, \mathbf{v}_2, \cdots$*. It is called a normalized adaptive protocol if for any* i*, the query vector* $\mathbf{v}_i$ *output by* P *satisfies*

1. $\mathbf{v}_i$ *is a unit vector;*
2. $\mathbf{v}_i$ *is orthogonal to the vectors* $\mathbf{v}_1, \cdots, \mathbf{v}_{i-1}$*;*
3. $\mathbf{v}_i$ *is deterministically determined by* $\mathbf{M} \times [\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}]$*.*

Let P^{std} be a standard protocol which outputs $\mathbf{e}_1, \mathbf{e}_2, \cdots$ where $\mathbf{e}_i$ is the i-th standard basis. We then show that adaptivity is unnecessary by proving that P^{std} has the same power as any normalized adaptive protocol.

More formally, we show the following lemma:

▶ **Lemma 6.** *Fix any* $n \times p$ *matrix* $\mathbf{U}$ *and any normalized adaptive protocol* P*. Let* $\mathbf{G}^T$ *be a* $p \times n$ *i.i.d Gaussian matrix. Fix* $q \leq n$ *to be the number of queries. Let matrix* $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_q]$ *and* $\mathbf{V}^{std} = [\mathbf{e}_1, \cdots, \mathbf{e}_q]$ *be the query matrix outputed by protocol* P *and* P^{std}*, correspondingly. Then, the probability distribution of* $\mathbf{UG}^T\mathbf{V}$ *is the same as the distribution of* $\mathbf{UG}^T\mathbf{V^{std}}$*.*

Proof. Since $\mathbf{G}^T\mathbf{V^{std}}$ is i.i.d Gaussian, it is enough to show $\mathbf{G}^T\mathbf{V}$ is also i.i.d Gaussian. We will show it column-by-column.

Denote $\mathbf{V}_i = [\mathbf{v}_1, \ldots, \mathbf{v}_i]$ and $\mathbf{V}_i^{std} = [\mathbf{e}_1, \cdots, \mathbf{e}_i]$. Note that $\mathbf{v}_1, \ldots, \mathbf{v}_q$ are unit vectors and orthogonal to each other. We first define unitary rotation matrices $R_1, R_2, \cdots$ recursively as follows. The matrix R_1 will take $\mathbf{v}_1$ to $\mathbf{e}_1$. The matrix R_i will take $\mathbf{e}_j$ to $\mathbf{e}_j$ for any $j < i$ and takes $R_{i-1} \cdots R_1 \mathbf{v}_i$ to $\mathbf{e}_i$. Note, R_i only depends on the first i query vectors. We have $R_i \cdots R_1 \mathbf{V}_i = \mathbf{V}_i^{std}$ for any $i \leq q$, and $\mathbf{G}^T\mathbf{V} = \mathbf{G}^T \cdot R_1^{-1} \cdots R_q^{-1} \cdot \mathbf{V}^{std}$. In the following, we use induction to show $\mathbf{G}^T \cdot R_1^{-1} \cdots R_i^{-1} \cdot \mathbf{V}_i^{std}$ is i.i.d Gaussian for any $i \leq q$.

For $i = 1$, since R_1 is determined by $\mathbf{v}_1$ which is independent of $\mathbf{G}^T$ and R_1 is a unitary matrix, $\mathbf{G}^T R_1^{-1}$ is i.i.d Gaussian. Thus, $\mathbf{G}^T R_1^{-1} \times \mathbf{V}_1^{std}$ is the first column which is also i.i.d Gaussian.

Now, suppose $\mathbf{G}^T \cdot R_1^{-1} \cdots R_i^{-1} \cdot \mathbf{V}_i^{std}$ is i.i.d Gaussian. We will prove $\mathbf{G}^T \cdot R_1^{-1} \cdots R_{i+1}^{-1} \cdot \mathbf{V}_{i+1}^{std}$ is also i.i.d Gaussian. Let $\mathbf{G'} = \mathbf{G}^T \cdot R_1^{-1} \cdots R_i^{-1}$ which is i.i.d Gaussian. Since R_{i+1} is determined by $\mathbf{v}_1, \cdots, \mathbf{v}_{i+1}$, it is determined by the response of the first i queries, that is, determined by $\mathbf{UG}^T\mathbf{V}_i = \mathbf{UG'V}_i^{std}$. It means R_{i+1} is determined by the first i columns of $\mathbf{UG'}$. Therefore, it is dependent on the first i columns of $\mathbf{G'}$. On the other hand, $R_{i+1}\mathbf{e}_j = \mathbf{e}_j$ for any $j \leq i$, and thus $R_{i+1}^{-1} = \begin{bmatrix} I_i & 0 \\ 0 & R' \end{bmatrix}$ where I_i is the $i \times i$ identity matrix, and R' depends on the first i columns of $\mathbf{G'}$. Consequently, in the multiplication of $\mathbf{G'} \times R_{i+1}^{-1}$, the first i columns are the same as those in $\mathbf{G'}$. In the $(i+1)$-th column, the a-th element is $\sum_{b \geq i+1} g'_{ab} r'_{b,i+1}$ where $g'_{ab}, r'_{b,i+1}$ are the elements in $\mathbf{G'}, R'$ correspondingly. Since $r'_{b,i+1}$ only depends on the first i columns of $\mathbf{G'}$, it is independent of g'_{ab} when $b \geq i+1$. Thus, the $(i+1)$-th column is also i.i.d Gaussian and independent of the first i columns. Therefore, we show $\mathbf{G}^T \cdot R_1^{-1} \cdots R_{i+1}^{-1} \cdot \mathbf{V}_{i+1}^{std}$ is still i.i.d Gaussian.

By induction $\mathbf{G}^T\mathbf{V}$ is i.i.d Gaussian. This finishes our proof. ◀

We then show for $\mathbf{M}_2 = \mathbf{U} \times \mathbf{G}^T + \frac{1}{\mathsf{poly}(n)} \cdot \mathbf{U}^\perp \times \mathbf{H}^T$, adaptivity is also unnecessary by a similar argument.

▶ **Corollary 7.** *Consider* $\mathbf{M}_2 = \mathbf{U} \times \mathbf{G}^T + \frac{1}{\mathsf{poly}(n)} \cdot \mathbf{U}^\perp \times \mathbf{H}^T$*. For any fixed* $\mathbf{U}, \mathbf{U}^\perp$*, and any fixed normalized adaptive protocol* P*,* $\mathbf{M}_2\mathbf{V}$ *has the same distribution as* $\mathbf{M}_2\mathbf{V}^{std}$*.*

Proof. It is enough to show both $\mathbf{G}^T \cdot \mathbf{V}$ and $\mathbf{H}^T \cdot \mathbf{V}$ are i.i.d Gaussian. ◀

Combining these results and Theorem 2, together with Yao's minimax principle [30],

▶ **Theorem 8.** *Let constant $\varepsilon > 0$ be the error tolerance and let $\mathbf{M}$ be an $n \times n$ oracle matrix with adaptive queries. For every integer $p < n$, at least $p+1$ queries are necessary for any randomized algorithm to distinguish whether $\mathsf{rank}(\mathbf{M}) \leq p$ or $\mathsf{rank}(\mathbf{M}) \geq p+1$ with advantage $\geq \varepsilon$.*

3.2 Lower Bound for Trace Estimation

We lower bound the number of queries needed to approximate the trace $\mathsf{tr}(\mathbf{M})$ of a matrix $\mathbf{M}$. In particular we reduce this problem to triangle detection as will be proved in Theorem 14.

▶ **Theorem 9.** *For any integer $C > 0$ and symmetric $n \times n$ matrix $\mathbf{M}$ with entries in $\{0, 1, 2, \ldots, n^3\}$, the number of possibly adaptively chosen query vectors, with entries in $\{0, 1, 2, \ldots, n^C\}$, needed to approximate $\mathsf{tr}(\mathbf{M})$ up to any relative error, is $\Omega(n/\log n)$.*

Proof. Suppose we had a possibly adaptive query algorithm making $q(n)$ queries which for a symmetric matrix $\mathbf{M}$, could approximate $\mathsf{tr}(\mathbf{M})$ up to any relative error. If $\mathbf{M} = \mathbf{A}^3$ for a symmetric matrix $\mathbf{A}$, we can run the trace esimation algorithm on $\mathbf{M}$ as follows: if $\mathbf{x}_1$ is the first query, we compute $\mathbf{A}\mathbf{x}_1$, then $\mathbf{A}(\mathbf{A}\mathbf{x}_1)$, then $\mathbf{A}(\mathbf{A}(\mathbf{A}\mathbf{x}_1)) = \mathbf{A}^3\mathbf{x}_1$. This then determines the second query $\mathbf{x}_2$, and we similarly compute $\mathbf{A}\mathbf{x}_2$, then $\mathbf{A}(\mathbf{A}\mathbf{x}_2)$, then $\mathbf{A}(\mathbf{A}(\mathbf{A}\mathbf{x}_2)) = \mathbf{A}^3\mathbf{x}_2$, etc. Thus, given only query access to $\mathbf{A}$, we can simulate the algorithm on $\mathbf{M} = \mathbf{A}^3$ with $3q(n)$ adaptive queries.

Now, it is well known that for an undirected graph G with adjacency matrix $\mathbf{A}$, the trace $\mathsf{tr}(\mathbf{A}^3)/6$ is the number of triangles in G. By the argument above, it follows that with $3q(n)$ queries to $\mathbf{A}$, we can determine if G has a triangle or has no triangles. On the other hand, by Theorem 14 below, at least $\Omega(n/\log n)$ queries to $\mathbf{A}$ are necessary for any adaptive algorithm to decide if there is a triangle in G. Therefore $3q(n) = \Omega(n/\log n)$ and hence we complete the proof with $q(n) = \Omega(n/\log n)$. ◀

3.3 Deciding if M is a Symmetric Matrix

▶ **Theorem 10.** *Given an $n \times n$ matrix $\mathbf{M}$ over any finite field or over fields $\mathbb{R}$ or $\mathbb{C}$, $O(\log(\frac{1}{\varepsilon}))$ queries are enough to test whether $\mathbf{M}$ is symmetric or not with probability $1-\varepsilon$.*

Proof. We choose two random vectors $\mathbf{u}$ and $\mathbf{v}$, where over a finite field we choose from a uniform distribution and over fields $\mathbb{R}$ or $\mathbb{C}$ we choose the Gaussian distribution. We then compute $\mathbf{M}\mathbf{u}$ and $\mathbf{M}\mathbf{v}$. We declare $\mathbf{M}$ to be symmetric if and only if $\mathbf{u}^T \cdot \mathbf{M}\mathbf{v} = \mathbf{v}^T \cdot \mathbf{M}\mathbf{u}$. It is easy to check that if $\mathbf{M}$ is symmetric, the test will succeed. We then show if $\mathbf{M}$ is not symmetric, $\mathbf{u}^T\mathbf{M}\mathbf{v} \neq \mathbf{v}^T\mathbf{M}\mathbf{u}$ with constant probability, so we obtain success probability $1-\varepsilon$ by repeating the test $O(\log(\frac{1}{\varepsilon}))$ times.

Let $\mathbf{A} = \mathbf{M} - \mathbf{M}^T$. When $\mathbf{M}$ is not symmetric, $\mathbf{A}$ is not 0. Thus, $\mathbf{u}^T\mathbf{M}\mathbf{v} = \mathbf{v}^T\mathbf{M}\mathbf{u}$ means $\mathbf{u}^T\mathbf{A}\mathbf{v} = 0$. We can treat this as a degree-2 polynomial in the entries of $\mathbf{v}^T$ and $\mathbf{u}$, i.e., this is $\sum_{i,j} u_i v_j \mathbf{A}_{i,j} = \sum_i u_i \sum_j v_j \mathbf{A}_{i,j}$. Thus, this is a non-zero polynomial and has at most constant probability of evaluating to 0 for any underlying field. To see this, for each i, let $t_i = \sum_j v_j \mathbf{A}_{i,j}$. Then there will be at least one t_i which is non-zero with probability at least $1/2$, for any underlying field. So now we get $\sum_i u_i t_i$. Fix all the u_i except u_i for a given t_i that is non-zero. Then we obtain $S + u_i t_i$. Then if u_i has at least two possible values, this is 0 in one case and non-zero in the other case. So we obtain a probability of at least $1/4$ of detection overall. ◀

4 Streaming and Statistics Problems

In this section we discuss testing an all ones column/row and identical columns/rows. Our results for row norms and majority/parity of columns/rows are in the full version.

4.1 Testing Existence of an All Ones Column/Row

Given a matrix $\mathbf{M} \in \{0,1\}^{m\times n}$, we want to test if $\mathbf{M}$ has a column (or row) with all 1 entries. It is trivial to test whether $\mathbf{M}$ has an all 1 column (or row) using n queries, *e.g.* $\mathbf{e}_1, \ldots, \mathbf{e}_n$. We consider this problem both over $\mathbb{F}[2]$ and $\mathbb{R}$. Note in the case over $\mathbb{R}$, if we allow an arbitrary query vector, we can set one query $\mathbf{v} = \{1, 2, 4, 8, ...2^n\}$, and then reconstruct $\mathbf{M}$ exactly. Thus, in order to avoid such trivial cases, we also restrict the entries in the query to be in $\{0, 1, 2, \ldots, n^C\}$.

For testing the existence of an all ones column, we reduce the problem to the communication complexity of Disjointness. Disjointness requires $\Omega(n)$ bits of communication to decide whether two sets with characteristic vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$ are disjoint with constant probability. Suppose the fist $m-1$ rows in $\mathbf{M}$ equal $\mathbf{x}^T$ while the last row equals $\mathbf{y}^T$. If we can decide whether $\mathbf{M}$ has an all ones column with q non-adaptive queries $\mathbf{v}_1, \ldots, \mathbf{v}_q$, then we obtain a protocol for Disjointness with communication q by letting Alice send a message $\left(\mathbf{x}^T\mathbf{v}_1, \ldots, \mathbf{x}^T\mathbf{v}_q\right)$. Thus from the communication complexity lower bound of Disjointness, $q = \Omega(n)$ queries over $\mathbb{F}[2]$ are necessary to test if there is an all ones column in $\mathbf{M}$, which shows that the naïve algorithm is already optimal. For queries over $\mathbb{R}$, note that each entry $\mathbf{x}^T\mathbf{v}_j$ in the message is represented with $\log n$ bits, and as a result $q \geq \Omega\left(n/\log n\right)$.

Testing the existence of an all ones row with queries over $\mathbb{R}$ is trivial deterministically by querying $\mathbf{v} = (1, 1, \ldots, 1)$. Next we study the query complexity of testing an all 1s row deterministically with queries over $\mathbb{F}[2]$. With any $q \leq n-1$ queries $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_q]$, there is a non-zero vector $\mathbf{z} \neq \mathbf{0}$ such that $\mathbf{z}^T\mathbf{V} = \mathbf{0}$. Therefore the query matrix $\mathbf{V}$ cannot distinguish whether a row is from $\mathbf{x}^T$ or $\mathbf{x}^T + \mathbf{z}^T$. However, $\mathbf{x}^T$ and $\mathbf{x}^T + \mathbf{z}^T$ cannot be both all 1 rows, and hence n queries are necessary. This result also shows that the query complexity of the same problem over different fields might be quite different. We note for randomized algorithms, $O(\log(1/\epsilon))$ queries suffice over $\mathbb{F}[2]$ since the inner product of a row which is not all 1s disagrees with the parity of the query with probability $1/2$.

Evaluating the OR/AND function of columns/rows of a Boolean matrix can be reduced to testing existence of an all 1 or all 0 column/row, and hence the same bounds follow.

4.2 Identical Columns/Rows

Given an $m \times n$ matrix $\mathbf{M}$, we want to test whether $\mathbf{M}$ has two identical columns or rows. The trivial solution naively retrieves all information with n queries (column vectors).

Testing identical columns can be reduced to Disjointness. Suppose Alice and Bob have $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$. Let Alice expand her vector $\mathbf{x}$ to an $\frac{m}{2} \times n$ matrix $\mathbf{M}_1$ as follows: the first row is $(1, \mathbf{x}^T) = (1, x_1, \ldots, x_n)$; for $2 \leq i \leq \frac{m}{2}$ the i-th row is $(1, z_1^{(i)}, \ldots, z_n^{(i)})$ where $z_j^{(i)} = 1$ if $x_j = 1$, and $z_j^{(i)}$ is uniformly random over $\{0,1\}$ if $x_j = 0$, for $1 \leq j \leq n$. Bob expands his vector $\mathbf{y}$ to $\mathbf{M}_2$ similarly. Putting $\mathbf{M}_1, \mathbf{M}_2$ together, we let $\mathbf{M} \stackrel{\text{def}}{=} \begin{bmatrix}\mathbf{M}_1 \\ \mathbf{M}_2\end{bmatrix}$. Then $\mathbf{M}$ is an $m \times (n+1)$ matrix with the first column being all 1s. For $j \geq 2$, the j-th column is all 1s if and only if $x_j = y_j = 1$, in which case $\mathbf{M}$ has two identical rows of all 1 entries. For columns where x_j, y_j are not both equal to 1, without loss of generality we may assume the j-th and j'-th columns satisfy $x_j = x_{j'} = 0$ and $y_j = y_{j'}$. Then two columns are identical only

if $(z_j^{(2)}, \ldots, z_j^{(\frac{m}{2})}) = (z_{j'}^{(2)}, \ldots, z_{j'}^{(\frac{m}{2})})$, which happens with probability $\leq 1/2^{\frac{m}{2}-1}$. Therefore the overall probability of two not-all-ones columns in $\mathbf{M}$ being identical is bounded by $m^2/2^{m/2} = 2^{-\Omega(m)}$.

That is, except for an exponentially small error $2^{-\Omega(m)}$, two identical columns in $\mathbf{M}$ are both all ones columns, which turns out to be equivalent to the case that two vectors $\mathbf{x}, \mathbf{y}$ held by Alice and Bob are not disjoint. Then, because DISJOINTNESS requires $\Omega(n)$ bits of communication, at least $\Omega(n)$ oracle queries to $\mathbf{M}$ are necessary. To test identical rows with error ε, if suffices to make $q = O(\log(m/\varepsilon))$ random queries with each entry uniform random over $\{0, 1\}$. Since for every pair of distinct rows, a random query distinguishes them with probability $\frac{1}{2}$, with $\lceil \log(m^2/\varepsilon) \rceil$ queries each pair of distinct rows is miscounted as identical with probability $\leq \varepsilon/m^2$. By a union bound, the overall false-positive error is bounded by $\frac{\varepsilon}{m^2} \cdot \binom{m}{2} < \varepsilon$, while there is no false-negative error since for all queries, identical rows always lead to identical outputs.

5 Graph Problems

5.1 Connectivity

▶ **Theorem 11.** *Given the bipartite adjacency matrix $\mathbf{A} \in \{0, 1\}^{n \times n}$ of a graph, we need $\Omega(n/\log n)$ queries to decide whether the graph is connected with constant probability*

Proof. Consider two row vectors $\mathbf{u}, \mathbf{v} \in \{0, 1\}^{n-1}$ and construct matrix $\mathbf{A}$ as follows. Firstly, the first $n/2$ rows of $\mathbf{A}$ equals to $\mathbf{u}$ and the rest are equal to $\mathbf{v}$, then add an all 1s column to $\mathbf{A}$. Now, matrix $\mathbf{A}$ can be treated as a bipartite adjacency matrix of a graph G with n vertices in each part, where $\mathbf{A}_{i,j} = 1$ iff there is an edge from the i-th left vertex to the j-th right vertex. Then G is disconnected if and only if the two vectors $\mathbf{u}$ and $\mathbf{v}$ are 0 on the same position. Thus any algorithm that uses $q(n)$ non-adaptive queries on the right of $\mathbf{A}$ to decide the connectivity of G immediately implies a protocol for set disjointness, provided we replace 1s with 0s in the input characteristic vectors to the set disjointness problem. So the communication is at most $q(n) \log n$, thus $q(n) = \Omega(n/\log n)$. ◀

▶ **Theorem 12.** *Given the signed edge-vertex incidence matrix $\mathbf{M} \in \{0, \pm 1\}^{n \times \binom{n}{2}}$ of a graph G with n vertices, the connectivity of G can be decided with $\mathsf{polylog}(n)$ non-adaptive queries.*

This follows from the main theorem of [16]. By the following theorem, every cut of G is multiplicatively approximated and hence G is connected iff H is connected, since a graph is disconnected iff it has a zero cut.

▶ **Theorem 13** ([16])**.** *There is a distribution on $\binom{n}{2} \times \mathsf{polylog}(n)$ matrices $\mathbf{S}$ such that from $\mathbf{MS}$, one can construct a (1 ± 0.1)-sparsifier H of the graph G with constant probability. Here, $\mathbf{x}^T \mathbf{L}_G x = (1 \pm .1)\mathbf{x}^T \mathbf{L}_H x$ for all x, with constant probability, where $\mathbf{L}_G$ and $\mathbf{L}_H$ are the corresponding graph Laplacians.*

5.2 Triangle Detection

▶ **Theorem 14.** *If an $n \times n$ matrix $\mathbf{A}$ is the adjacency matrix of a graph G, then determining whether G contains a triangle or not requires $\Omega(n/\log n)$ queries, even for randomized algorithms succeeding with constant probability.*

See the full version for a proof using communication complexity.

6 Conclusions

We initiated the study of querying a matrix through matrix-vector products. We illustrated that for some quantities, if one can only query matrix-vector products on one side, the problem becomes harder. We also illustrated the importance of the underlying field defining the matrix-vector products, as well as the representation of the graph for graph problems. Given connections to sketching algorithms, streaming, and compressed sensing, we believe this area deserves its own study. Some interesting open questions are for computing matrix norms, such as Schatten-p norms, for which tight bounds in streaming and communication complexity models remain elusive; for recent work on this see [21, 22, 9]. Given the success of our model in proving lower bounds for approximate rank, which we also do not have streaming or communication lower bounds for, perhaps tight bounds in our query model are possible for matrix norms. Such bounds may give insight for other models.

References

1 Yuqing Ai, Wei Hu, Yi Li, and David P. Woodruff. New Characterizations in Turnstile Streams with Applications. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 20:1–20:22, 2016.

2 Noga Alon, Yossi Matias, and Mario Szegedy. The Space Complexity of Approximating the Frequency Moments. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 20–29, 1996.

3 Sepehr Assadi, Sanjeev Khanna, and Yang Li. On Estimating Maximum Matching Size in Graph Streams. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1723–1742, 2017.

4 Sepehr Assadi, Sanjeev Khanna, Yang Li, and Grigory Yaroslavtsev. Maximum Matchings in Dynamic Graph Streams and the Simultaneous Communication Model. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1345–1364, 2016.

5 Khanh Do Ba, Piotr Indyk, Eric Price, and David P. Woodruff. Lower Bounds for Sparse Recovery. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1190–1197, 2010.

6 Maria-Florina Balcan, Yi Li, David P. Woodruff, and Hongyang Zhang. Testing Matrix Rank, Optimally. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 727–746, 2019.

7 Anders Björner, László Lovász, and Andrew CC Yao. Linear decision trees: volume estimates and topological bounds. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 170–177. ACM, 1992.

8 Eric Blais, Joshua Brody, and Kevin Matulef. Property Testing Lower Bounds via Communication Complexity. *Computational Complexity*, 21(2):311–358, 2012.

9 Vladimir Braverman, Stephen R. Chestnut, Robert Krauthgamer, Yi Li, David P. Woodruff, and Lin Yang. Matrix Norms in Data Streams: Faster, Multi-Pass and Row-Order. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 648–657, 2018.

10 Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

11 Talya Eden, Amit Levi, Dana Ron, and C Seshadhri. Approximately counting triangles in sublinear time. *SIAM Journal on Computing*, 46(5):1603–1646, 2017.

12 Piotr Indyk, Eric Price, and David P. Woodruff. On the Power of Adaptivity in Sparse Recovery. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 285–294, 2011.

13 Akshay Kamath and Eric Price. Adaptive Sparse Recovery with Limited Adaptivity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2729–2744, 2019.

14 Daniel M. Kane, Shachar Lovett, and Shay Moran. Near-optimal linear decision trees for k-SUM and related problems. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 554–563, 2018.

15 Sampath Kannan, Elchanan Mossel, Swagato Sanyal, and Grigory Yaroslavtsev. Linear Sketching over F_2. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 8:1–8:37, 2018.

16 Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single Pass Spectral Sparsification in Dynamic Streams. *SIAM J. Comput.*, 46(1):456–477, 2017.

17 John T Kent and R J Muirhead. Aspects of Multivariate Statistical Theory. *The Statistician*, 33(2):251, 1984.

18 Christian Konrad. Maximum Matching in Turnstile Streams. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, pages 840–852, 2015.

19 Yi Li, Huy L. Nguyen, and David P. Woodruff. On Sketching Matrix Norms and the Top Singular Vector. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1562–1581, 2014.

20 Yi Li, Huy L. Nguyen, and David P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 174–183, 2014.

21 Yi Li and David P. Woodruff. On approximating functions of the singular values in a stream. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 726–739, 2016.

22 Yi Li and David P. Woodruff. Embeddings of Schatten Norms with Applications to Data Streams. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 60:1–60:14, 2017.

23 Vasileios Nakos, Xiaofei Shi, David P. Woodruff, and Hongyang Zhang. Improved Algorithms for Adaptive Compressed Sensing. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 90:1–90:14, 2018.

24 Eric Price and David P. Woodruff. Lower Bounds for Adaptive Sparse Recovery. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 652–663, 2013.

25 Jianhong Shen. On the singular values of Gaussian random matrices. *Linear Algebra and its Applications*, 326:1–14, 2001.

26 Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *Compressed Sensing: Theory and Applications*, November 2010. doi:10.1017/CBO9780511794308.006.

27 Martin Wainwright. High-dimensional statistics: A non-asymptotic viewpoint. URL: https://www.stat.berkeley.edu/~mjwain/stat210b/Chap2_TailBounds_Jan22_2015.pdf.

28 Hermann Weyl. Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, 1912.

29 David P. Woodruff. Sketching as a Tool for Numerical Linear Algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

30 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science, FOCS 1977*, pages 222–227. IEEE, 1977.

31 Qiaochu Yuan. Singular value decomposition, 2017. URL: `https://qchu.wordpress.com/2017/03/13/singular-value-decomposition/`.