Distance Estimation Between Unknown Matrices Using Sublinear Projections on Hamming Cube

Arijit Bishnu
Indian Statistical Institute, Kolkata, India

Arijit Ghosh
Indian Statistical Institute, Kolkata, India

Gopinath Mishra
Indian Statistical Institute, Kolkata, India

Abstract

Using geometric techniques like projection and dimensionality reduction, we show that there exists a randomized sub-linear time algorithm that can estimate the Hamming distance between two matrices. Consider two matrices $\mathbf{A}$ and $\mathbf{B}$ of size $n \times n$ whose dimensions are known to the algorithm but the entries are not. The entries of the matrix are real numbers. The access to any matrix is through an oracle that computes the projection of a row (or a column) of the matrix on a vector in $\{0,1\}^n$. We call this query oracle to be an INNER PRODUCT oracle (shortened as IP). We show that our algorithm returns a $(1 \pm \epsilon)$ approximation to $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ with high probability by making $\mathcal{O}\left(\frac{n}{\sqrt{\mathbf{D_M}(\mathbf{A},\mathbf{B})}}\text{poly}\left(\log n, \frac{1}{\epsilon}\right)\right)$ oracle queries, where $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ denotes the Hamming distance (the number of corresponding entries in which $\mathbf{A}$ and $\mathbf{B}$ differ) between two matrices $\mathbf{A}$ and $\mathbf{B}$ of size $n \times n$. We also show a matching lower bound on the number of such IP queries needed. Though our main result is on estimating $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ using IP, we also compare our results with other query models.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Distance estimation, Property testing, Dimensionality reduction, Sub-linear algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2021.44

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2107.02666

Acknowledgements The authors wish to thank their colleague Ansuman Banerjee for helpful discussions on GPU architecture and CUDA.

1 Introduction

Measuring similarity between entities using a distance function has been a major area of focus in computer science in general and computational geometry in particular [9, 8, 20, 19, 7]. Distance computations require access to the entire data and thus can not escape computations that are linear in time complexity. In this era of big data, seeing the entire data may be too much of an ask and trading precision for a time efficient algorithm is a vibrant area of study in property testing [22]. Testing properties of binary images with sub-linear time algorithms has been a focus of property testing algorithms [27, 26, 11, 24, 10]. Matrices are ubiquitous in the sense that they represent or abstract a whole gamut of structures like adjacency matrices of geometric graphs and visibility graphs, images, experimental data involving 0-1 outcomes, etc. Pairwise distance computations between such matrices is a much-needed programming primitive in image processing and computer vision applications so much so that the widely used commercial toolbox MATLAB of MathWorks® [1] has an inbuilt function call named `pdist2`$(\cdot,\cdot,\cdot)$ [2] for it. Other open source based software packages also have similar primitives [3]. For all these primitives, the matrices need to be known. But in all situations where access to the matrices are restricted (say, because of security, privacy or communication issues) except for an oracle access, we want to know how much the two matrices differ in their entries. Keeping in line with the above, we focus on distance estimation problem between two matrices whose dimensions are known to the algorithm but the entries are unknown; the access to the matrices will be through an oracle. This oracle, though linear algebraic in flavor, has a geometric connotation to it. We hold back the discussion on the motivation of the oracle till Section 1.1.

Notations

In this paper, we denote the set $\{1,\ldots,t\}$ by $[t]$ and $\{0,\ldots,t\}$ by $[[t]]$. For a matrix $\mathbf{A}$, $\mathbf{A}(i,j)$ denotes the element in the i-th row and j-th column of $\mathbf{A}$. Unless stated otherwise, $\mathbf{A}$ will be a matrix with real entries. $\mathbf{A}(i,*)$ and $\mathbf{A}(*,j)$ denote the i-th row vector and j-th column vector of the matrix $\mathbf{A}$, respectively. Throughout this paper, the number of rows or columns of a square matrix $\mathbf{A}$ will be n. Vectors are matrices of order $n \times 1$ and will be represented using bold face letters. Without loss of generality, we consider n to be a power of 2. The i-th coordinate of a vector $\mathbf{x}$ will be denoted by x_i. We will denote by $\mathbf{1}$ the vector with all coordinates 1. Let $\{0,1\}^n$ denote the set of n-dimensional vectors with entries either 0 or 1. By $\langle \mathbf{x},\mathbf{y}\rangle$, we denote the standard inner product of $\mathbf{x}$ and $\mathbf{y}$, that is, $\langle \mathbf{x},\mathbf{y}\rangle = \sum_{i=1}^{n} x_i y_i$. P is a $(1\pm\varepsilon)$-approximation to Q means $|P-Q| \leq \varepsilon \cdot Q$. The statement *with high probability* means that the probability of success is at least $1-\frac{1}{n^c}$, where c is a positive constant. $\widetilde{\Theta}(\cdot)$ and $\widetilde{\mathcal{O}}(\cdot)$ hides a poly $\left(\log n, \frac{1}{\varepsilon}\right)$ term in the upper bound. $||\cdot||_p$ denotes the usual ℓ_p distance.

1.1 Query oracle definition and motivation, problem statements and our results

▶ **Definition 1.1** (Matrix distance). The *matrix-distance* between two matrices $\mathbf{A}$ and $\mathbf{B}$ of size $n \times n$ is the number of pairwise mismatches and is denoted and defined as

$$\mathbf{D_M}(\mathbf{A},\mathbf{B}) = |\{(i,j) : i,j \in [n], \mathbf{A}(i,j) \neq \mathbf{B}(i,j)\}|\,.$$

As alluded to earlier, the matrices cannot be accessed directly, the sizes of the matrices are known but the entries are unknown. We will refer to the problem as the *matrix distance* problem. We consider the following query models to solve the matrix distance problem in this paper.

Query oracles for unknown matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$

The main query oracle access used in this work is based on the inner product of two vectors and is defined as follows:

Inner Product (IP): Given a row index $i \in [n]$ (or, a column index $j \in [n]$) and a vector $\mathbf{v} \in \{0,1\}^n$, the IP query to $\mathbf{A}$ reports the value of $\langle \mathbf{A}(i,*),\mathbf{v}\rangle$ ($\langle \mathbf{A}(*,j),\mathbf{v}\rangle$). If the input index is for row (column), we refer the corresponding query as row (column) IP query.

This linear algebraic oracle access has a geometric connotation to it in terms of *projection* onto Hamming vectors – we exploit this understanding in our work. This oracle access is also motivated from a practical angle. A dot product operation is a fit case for parallelization using a SINGLE INSTRUCTION MULTIPLE DATA (SIMD) architecture [23]. Modern day GPU processors provide instruction level parallelism. In effect, NVIDIA GPUs that are built on CUDA architecture, provide dot product between two vectors as a single API call [28, 4]. Thus, there exists practical implementation of the query oracle access that we use. There are also examples of programming languages supporting SIMD intrinsics that can compute dot product [5]. There is a caveat though – in terms of resource, more processors are used. For us, in this work, the query complexity is the number of calls to the IP. As mentioned, modern day architectures allow us to convert each IP query to a one cycle computation with more processors used in parallel.

In the power hierarchy of matrix based query oracles, IP surely wields some power vis-a-vis solving certain problems [14] [1]. An obvious question that confronts an algorithm designer is whether a weaker oracle can do the same job at hand (here, computing matrix distance). With that in mind, we define the following two oracles and show that their query complexity lower bounds on the matrix distance problem match the trivial upper bounds. That shows the justification for use of IP.

MATRIX ELEMENT (ME): Given two indices $i, j \in [n]$, the ME query to $\mathbf{A}$ returns the value of $\mathbf{A}(i, j)$.

DECISION INNER PRODUCT (DEC-IP): Given a row index $i \in [n]$ (or, a column index $j \in [n]$) and a vector $\mathbf{v} \in \{0, 1\}^n$, the DEC-IP query to $\mathbf{A}$ reports whether $\langle \mathbf{A}(i, *), \mathbf{v} \rangle$ $(\langle \mathbf{A}(*, j), \mathbf{v} \rangle) = 0$. If the input index is for row (column), we refer the corresponding query as row (column) DEC-IP query.

The following remark highlights the relative power of the query oracles.

▶ **Remark 1.** Each ME query can be simulated by using one DEC-IP oracle, and each DEC-IP oracle can be simulated by using one IP query.

Our results

Our main result is an algorithm for estimating the distances between two unknown matrices using IP, and the result is formally stated as follows. Unless otherwise mentioned, all our algorithms are randomized.

▶ **Theorem 1.2** (Main result: Estimating the distance between two arbitrary matrices). *There exists an algorithm that has* IP *query oracle access to unknown matrices* $\mathbf{A}$ *and* $\mathbf{B}$*, takes an* $\varepsilon \in (0, 1)$ *as an input, and returns a* $(1 \pm \varepsilon)$ *approximation to* $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ *with high probability, and makes* $\mathcal{O}\left(\left(n/\sqrt{\mathbf{D_M}(\mathbf{A}, \mathbf{B})}\right) \text{poly}\left(\log n, \frac{1}{\varepsilon}\right)\right)$ IP *queries.*

We also show that our algorithm (corresponding to the above theorem) is optimal, if we ignore the poly $\left(\log n, \frac{1}{\varepsilon}\right)$ term, by showing (in Theorem 4.1) that any algorithm that estimates $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ requires $\Omega\left(n/\sqrt{\mathbf{D_M}(\mathbf{A}, \mathbf{B})}\right)$ IP queries. For the sake of completeness in understanding the power of IP, we study the matrix distance problem also using two weaker oracle access – ME and DEC-IP. Our results are summarized in Table 1 and they involve both upper and almost matching lower bounds in terms of the number of queries needed. Note that all of our lower bounds hold even if one matrix (say $\mathbf{A}$) is known and both matrices ($\mathbf{A}$ and $\mathbf{B}$) are symmetric binary matrices.

[1] But the IP defined in this paper is weaker than that defined in [14] – in their case, one is allowed to query for inner product of rows/columns of matrices with vectors in $\mathbb{R}^n$.

Table 1 Our results. In this table, $D = \mathbf{D_M}(\mathbf{A}, \mathbf{B})$.

Query Oracle	ME	DEC-IP	IP
Upper Bound	$\widetilde{\mathcal{O}}\left(\frac{n^2}{D}\right)$ (Trivial)	$\widetilde{\mathcal{O}}\left(\frac{n^2}{D}\right)$ (Trivial)	$\widetilde{\mathcal{O}}\left(\frac{n}{\sqrt{D}}\right)$ (Theorem 1.2)
Lower Bound	$\Omega\left(\frac{n^2}{D}\right)$ (Corollary 4.3)	$\Omega\left(\frac{n^2}{D}\right)$ (Theorem 4.1)	$\Omega\left(\frac{n}{\sqrt{D}}\right)$ (Theorem 4.2)

▶ **Remark 2.** Note that an IP query to a matrix $\mathbf{A}$ answers inner product of a specified row (column) with a given binary vector. However, we will describe subroutines (of the algorithm for estimating the distance between two matrices) that ask for inner product of a specified row (column) with a given vector $\mathbf{r} \in \{-1, 1\}^n$. This is not a problem as $\langle \mathbf{A}(i, *), \mathbf{r} \rangle$ ($\langle \mathbf{A}(*, j), \mathbf{r} \rangle$) can be computed by using two IP queries (with binary vectors) [2]. For simplicity, we refer $\langle \mathbf{A}(i, *), \mathbf{r} \rangle$ ($\langle \mathbf{A}(*, j), \mathbf{r} \rangle$) also as IP query in our algorithm.

1.2 Related work

There are works in property testing and sub-linear geometric algorithms [15, 18, 17, 16]. In the specific problem that we deal with in this paper, to the best of our knowledge, Raskhodnikova [26] started the study of property testing of binary images in the *dense image model*, where the number of 1-pixels is $\Omega(n^2)$. The notion of distance between matrices of the same size is defined as the number of pixels (matrix entries) on which they differ. The relative distance is the ratio of the distance and the number of pixels in the image. In this model, Raskhodnikova studies three properties of binary images – connectivity, convexity, and being a half-plane – in the property testing framework. Ron and Tsur [27] studied property testing algorithms in the sparse binary image model (the number of 1-pixels is $O(n)$) for connectivity, convexity, monotonicity, and being a line. The distance measure in this model is defined by the fraction of differing entries taken with respect to the actual number of 1's in the matrix. As opposed to treating binary images as discrete images represented using pixels as in [27, 26], Berman et al. in [10] and [12] treated them as continuous images and studied the problem of property testing for convexity of 2-dimensional figures with only uniform and independent samples from the input. To the best of our knowledge, computing distances between binary images has not been dealt with in the sub-linear time framework.

Organization of the paper

We prove Theorem 1.2 (in Section 1.1) through a sequence of results. To prove Theorem 1.2 that estimates the distance between two arbitrary matrices, we need a result that estimates the distance between two symmetric matrices (Lemma 2.1 in Section 2) that in turn needs a result on the estimation of the distance between two symmetric matrices with respect to a parameter T (Lemma 2.2 in Section 2). Lemma 2.2 is the main technical lemma that uses dimensionality reduction via Johnson Lindenstrauss lemma crucially. The technical overview including the proof idea of Lemma 2.2 is in Section 2.1. The detailed proof idea is in Section 2.2. Using communication complexity, we prove our lower bound results in Section 4. The proof of lemma marked with $\star$ can be found in the full version of the paper [13].

[2] For $\mathbf{r} \in \{-1, 1\}^n$, consider $\mathbf{v_1}, \mathbf{v_{-1}} \in \{0, 1\}^n$ indicator vectors for $+1$ and -1 coordinates in $\mathbf{r} \in \{-1, 1\}^n$, respectively. Then $\langle \mathbf{A}(i, *), \mathbf{r} \rangle = \langle \mathbf{A}(i, *), \mathbf{v_1} \rangle - \langle \mathbf{A}(i, *), \mathbf{v_{-1}} \rangle$. So, $\langle \mathbf{A}(\mathbf{i}, *), \mathbf{r} \rangle$ can be computed with two IP queries $\langle \mathbf{A}(i, *), \mathbf{v_1} \rangle$ and $\langle \mathbf{A}(i, *), \mathbf{v_{-1}} \rangle$. Similar argument also holds for $\langle \mathbf{A}(*, j), \mathbf{r} \rangle$.

2 Matrix-Distance between two symmetric matrices

This section builds up towards a proof of Theorem 1.2 by first giving an algorithm that estimates the matrix-distance between two unknown symmetric matrices (instead of arbitrary matrices as in Theorem 1.2) with high probability. The result is formally stated in Lemma 2.1. In Section 3, we will discuss how this result (stated in Lemma 2.1) can be used to prove Theorem 1.2.

▶ **Lemma 2.1** (Estimating the distance between two symmetric matrices). *There exists an algorithm* DIST-SYMM-MATRIX$(\mathbf{A}, \mathbf{B}, \varepsilon)$*, that has* IP *query access to unknown* symmetric *matrices* $\mathbf{A}$ *and* $\mathbf{B}$*, takes an* $\varepsilon \in \left(0, \frac{1}{2}\right)$ *as an input, and returns a* $(1 \pm \varepsilon)$*-approximation to* $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ *with high probability, making* $\widetilde{\mathcal{O}}\left(n/\sqrt{\mathbf{D_M}(\mathbf{A}, \mathbf{B})}\right)$ IP *queries.*

First, we prove a parameterized version of the above lemma in Lemma 2.2 where we are given a parameter T along with an $\varepsilon \in \left(0, \frac{1}{2}\right)$ and we can obtain an approximation guarantee on $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ as a function of both T and ε. One can think of T as a guess for $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$.

▶ **Lemma 2.2** (($\star$) Estimating the distance between two symmetric matrices w.r.t. a parameter T). *There exists an algorithm* DIST-SYMM-MATRIX-GUESS$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$*, that has* IP *query access to unknown* symmetric *matrices* $\mathbf{A}$ *and* $\mathbf{B}$*, takes parameters* T *and* $\varepsilon \in \left(0, \frac{1}{2}\right)$ *as inputs, and returns* $\widehat{d}$ *satisfying* $\left(1 - \frac{\varepsilon}{10}\right) \mathbf{D_M}(\mathbf{A}, \mathbf{B}) - \frac{\varepsilon}{1600} T \leq \widehat{d} \leq \left(1 + \frac{\varepsilon}{10}\right) \mathbf{D_M}(\mathbf{A}, \mathbf{B})$ *with high probability, and makes* $\widetilde{\mathcal{O}}\left(n/\sqrt{T}\right)$ *queries. Note that here* T *is at least a suitable polynomial in* $\log n$ *and* $1/\varepsilon$.

In Section 2.1, we discuss some preliminary results to prove Lemma 2.2. The proof of Lemma 2.2 is given in Section 2.2. If the guess $T \leq \mathbf{D_M}(\mathbf{A}, \mathbf{B})$, DIST-SYMM-MATRIX-GUESS $(\mathbf{A}, \mathbf{B}, \varepsilon, T)$ (as stated in Lemma 2.2) returns a $(1 \pm \varepsilon)$-approximation to $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ with high probability. However, this dependence on T can be overcome to prove Lemma 2.1 by using a standard technique in property testing.

2.1 Technical preliminaries to prove Lemma 2.2

The matrix distance $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ can be expressed in terms of the notion of a distance between a row (column) of matrix $\mathbf{A}$ and a row (column) of matrix $\mathbf{B}$ as follows:

▶ **Definition 2.3** (Distance between two rows (columns)). Let $\mathbf{A}$ and $\mathbf{B}$ be two matrices of order $n \times n$. The distance between the i-th row of $\mathbf{A}$ and the j-th row of $\mathbf{B}$ is denoted and defined as

$$\mathbf{d_H}\left(\mathbf{A}(i, *), \mathbf{B}(j, *)\right) = \left|\{k \in [n] : \mathbf{A}(i, k) \neq \mathbf{B}(j, k)\}\right|,$$

Similarly, $\mathbf{d_H}\left(\mathbf{A}(*, i), \mathbf{B}(*, j)\right)$ is the distance between the i-th column of $\mathbf{A}$ and the j-th column of $\mathbf{B}$.

▶ **Observation 2.4** (Expressing $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ as the sum of distance between rows (columns)). Let $\mathbf{A}$ and $\mathbf{B}$ be two $n \times n$ matrices. The matrix distance between $\mathbf{A}$ and $\mathbf{B}$ is given by

$$\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{n} \mathbf{d_H}\left(\mathbf{A}(i, *), \mathbf{B}(i, *)\right) = \sum_{i=1}^{n} \mathbf{d_H}\left(\mathbf{A}(*, i), \mathbf{B}(*, i)\right).$$

For a given $i \in [n]$, we can approximate $\mathbf{d_H}\left(\mathbf{A}(i, *), \mathbf{B}(i, *)\right)$ and $\mathbf{d_H}\left(\mathbf{A}(*, i), \mathbf{B}(*, i)\right)$ using IP queries as stated in Lemma 2.5. This can be shown by an application of the well-known Johnson-Lindenstrauss Lemma [6].

▶ **Lemma 2.5** (Estimating the distance between rows of $\mathbf{A}$ and $\mathbf{B}$). *Consider* IP *access to two $n \times n$ (unknown) matrices $\mathbf{A}$ and $\mathbf{B}$. There is an algorithm* $\textsc{Dist-Bet-Rows}(i, \alpha, \delta)$, *that takes $i \in [n]$ and $\alpha, \delta \in (0,1)$ as inputs, and reports a $(1 \pm \alpha)$-approximation to $\mathbf{d_H}(\mathbf{A}(i,*), \mathbf{B}(i,*))$ with probability at least $1-\delta$, and makes $\mathcal{O}\left(\frac{\log n}{\alpha^2} \log \frac{1}{\delta}\right)$* IP *queries to both $\mathbf{A}$ and $\mathbf{B}$.*

As it is sufficient for our purpose, in the above lemma, we discussed about estimating the distance between rows of $\mathbf{A}$ and $\mathbf{B}$ with the same index. However, we note that, a simple modification to the algorithm corresponding to Lemma 2.5 also works for estimating the distance between any row and/or column pair.

▶ **Proposition 2.6** (Johnson-Lindenstrauss Lemma). *Let us consider any pair of points $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$. For a given $\varepsilon \in (0,1)$ and $\delta \in (0,1)$, there is a map $f : \mathbb{R}^N \to \mathbb{R}^d$ such that $d = \Theta\left(\frac{1}{\varepsilon^2} \log \frac{1}{\delta}\right)$ satisfying the following bound with probability at least $1-\delta$.*

$$(1-\varepsilon)||\mathbf{u}-\mathbf{v}||_2^2 \leq ||f(\mathbf{u}) - f(\mathbf{v})||_2^2 \leq (1+\varepsilon)||\mathbf{u}-\mathbf{v}||_2^2. \tag{1}$$

▶ **Remark 3** (An explicit mapping in Johnson-Lindenstrauss Lemma). An explicit mapping $f : \mathbb{R}^n \to \mathbb{R}^d$ satisfying Equation 1 is as follows. Consider $\mathbf{r_1}, \ldots, \mathbf{r_d} \in \{-1,1\}^n$ such that each coordinate of every $\mathbf{r}_i$ is taken from $\{-1,1\}$ uniformly at random. Then for each $\mathbf{u} \in \{0,1\}^n$,

$$f(\mathbf{u}) = \frac{1}{\sqrt{d}} \left(\langle \mathbf{u}, \mathbf{r_1} \rangle, \langle \mathbf{u}, \mathbf{r_2} \rangle, \ldots, \langle \mathbf{u}, \mathbf{r_d} \rangle\right).$$

Identity testing between two rows

Now, let us discuss an algorithm where the objective is to decide whether the i-th row vectors of matrices $\mathbf{A}$ and $\mathbf{B}$ are identical. Observe that $||\mathbf{A}(i,*) - \mathbf{B}(j,*)||_2 = 0$ if and only if $\mathbf{d_H}(\mathbf{A}(i,*), \mathbf{B}(j,*)) = 0$. Also notice that, for a function $f : \mathbb{R}^n \to \mathbb{R}^d$ satisfying Equation 1, $||\mathbf{u}-\mathbf{v}||_2 = 0$ if and only if $||f(\mathbf{u}) - f(\mathbf{v})||_2 = 0$. This discussion along with Proposition 2.6 and Remark 3 imply an algorithm (described inside Observation 2.9) that can decide whether corresponding rows of $\mathbf{A}$ and $\mathbf{B}$ are identical. Observation 2.9 is stated in a more general form than discussed here. Note that the general form will be needed to show Lemma 2.5. For this purpose, we define the notion of *projecting a vector in $\{-1,1\}^n$ onto a set $S \subseteq [n]$* as defined below and an observation (Observation 2.8) about evaluating the projection using an IP query.

▶ **Definition 2.7** (Vector projected onto a set). Let $\mathbf{A}$ be an $n \times n$ matrix and $i \in [n]$. For a subset $S \subseteq [n], \mathbf{A}(i,*) \mid_S \in \mathbb{R}^n$ is defined as the vector having ℓ-th coordinate equals to $\mathbf{A}(i,\ell)$ if $\ell \in S$, and 0, otherwise. Also consider $\mathbf{r} \in \{-1,1\}^n$ and a set $S \subseteq [n]$. Then the vector $\mathbf{r}$ projected onto S is denoted by $\mathbf{r}|_S \in \{-1,0,1\}^n$ and defined as follows: For $\ell \in [n]$, the ℓ-th coordinate of $\mathbf{r}|_S$ is same as that of $\mathbf{r}$ if $\ell \in S$, and 0, otherwise.

▶ **Observation 2.8.** Let $\mathbf{A}$ be a $n \times n$ matrix, $i \in [n]$, $\mathbf{r} \in \{-1,1\}^n$ and $S \subseteq [n]$. Then $\langle \mathbf{A}(i,*)|_S, \mathbf{r} \rangle = \langle \mathbf{A}(i,*), \mathbf{r}|_S \rangle$. That is, $\langle \mathbf{A}(i,*), \mathbf{r}|_S \rangle$ can be evaluated by using a IP query $\langle \mathbf{A}(i,*), \mathbf{r}|_S \rangle$ to matrix $\mathbf{A}$.

▶ **Observation 2.9** (Identity testing between rows of $\mathbf{A}$ and $\mathbf{B}$). Consider IP access to two $n \times n$ (unknown) matrices $\mathbf{A}$ and $\mathbf{B}$. There is an algorithm $\textsc{Identity}(S, i, \delta)$ that takes $i \in [n]$, $S \subseteq [n]$ and $\delta \in (0,1)$ as inputs, and decides whether $\mathbf{d_H}(\mathbf{A}(i,*) \mid_S, \mathbf{B}(i,*) \mid_S) = 0$ with probability at least $1-\delta$, and makes $\mathcal{O}\left(\log \frac{1}{\delta}\right)$ IP queries to both $\mathbf{A}$ and $\mathbf{B}$.

Proof. Let the vectors $\mathbf{r}_1, \ldots, \mathbf{r}_d \in \{-1,1\}^n$ be such that each coordinate of every $\mathbf{r}_j$, $j = 1, \ldots, d$, is taken from $\{-1,1\}$ uniformly at random where $d = \Theta\left(\log \frac{1}{\delta}\right)$. Then the algorithm finds $a_j = \langle \mathbf{A}(i,*)|_S, \mathbf{r_j} \rangle$ and $b_j = \langle \mathbf{B}(i,*)|_S, \mathbf{r_j} \rangle$ by making one IP query to each of $\mathbf{A}$ and $\mathbf{B}$. This is possible by Observation 2.8. The algorithm makes d IP queries to each of the matrices $\mathbf{A}$ and $\mathbf{B}$. Take $\mathbf{a} = \frac{1}{\sqrt{d}}(a_1, \ldots, a_d) \in \mathbb{R}^d$ and $\mathbf{b} = \frac{1}{\sqrt{d}}(b_1, \ldots, b_d) \in \mathbb{R}^d$. By Proposition 2.6 and Remark 3, $||\mathbf{a}-\mathbf{b}||_2 = 0$ if and only if $||\mathbf{A}(i,*)\ |_S, \mathbf{B}(i,*)\ |_S||_2 = 0$. By the definition of distance between a row of one matrix and a row of another matrix (Definition 2.3), note that, $||\mathbf{A}(i,*)\ |_S - \mathbf{B}(j,*)\ |_S||_2 = 0$ if and only if $\mathbf{d_H}\left(\mathbf{A}(i,*)\ |_S, \mathbf{B}(j,*)\ |_S\right) = 0$. So, the algorithm finds $||\mathbf{a}-\mathbf{b}||_2$ and, reports $||\mathbf{a}-\mathbf{b}||_2 = 0$ if and only if $\mathbf{d_H}\left(\mathbf{A}(i,*)\ |_S, \mathbf{B}(i,*)\ |_S\right) = 0$. The correctness and query complexity of the algorithm follows from the description itself. ◀

Estimating the distance between rows induced by a set

Now, consider the algorithm corresponding to Lemma 2.5 (Dist-Bet-Rows$(\cdot,\cdot,\cdot)$) that can estimate the distance between a row of $\mathbf{A}$ and a row of $\mathbf{B}$. It makes repeated calls to Identity$(\cdot,\cdot,\cdot)$ in a non-trivial way. Also, algorithm Dist-Bet-Rows$(\cdot,\cdot,\cdot)$) can be generalized to estimate the distance between a row of $\mathbf{A}$ and the corresponding row of $\mathbf{B}$ projected onto the same set $S \subseteq [n]$, as stated in the following Lemma.

▶ **Lemma 2.10** (($\star$) Estimating the distance between rows of $\mathbf{A}$ and $\mathbf{B}$ induced by a set $S \subseteq [n]$). *Consider* IP *access to two $n \times n$ (unknown) matrices $\mathbf{A}$ and $\mathbf{B}$.* Restrict-Dist-Bet-Rows(S, i, α, δ) *algorithm, takes $S \subseteq [n]$, $i \in [n]$ and $\alpha, \delta \in (0,1)$ as inputs, and reports a $(1 \pm \alpha)$-approximation to $\mathbf{d_H}\left(\mathbf{A}(i,*)\ |_S, \mathbf{B}(i,*)\ |_S\right)$ with probability at least $1-\delta$, and makes $\mathcal{O}\left(\frac{\log n}{\alpha^2} \log \frac{1}{\delta}\right)$* IP *queries to both $\mathbf{A}$ and $\mathbf{B}$.*

Observe that Lemma 2.5 is a special case of Lemma 2.10 when $S = [n]$. Algorithm Dist-Bet-Rows(i, α, δ) (corresponding to Lemma 2.5) is directly called as a subroutine from Dist-Symm-Matrix-Guess$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$. Restrict-Dist-Bet-Rows$(S, i, \alpha, \delta)$ is indirectly called from a subroutine to *sample mismatched element almost uniformly* as explained below.

Sampling a mismatched element almost uniformly

For a row $i \in [n]$, let $\mathrm{NEQ}(\mathbf{A}, \mathbf{B}, i) = \{j : \mathbf{A}(i,j) \neq \mathbf{B}(i,j)\}$ denote the set of mismatches. Apart from estimating the distance between a row (column) of $\mathbf{A}$ and the corresponding row (column) of $\mathbf{B}$, we can also sample element from $\mathrm{NEQ}(\mathbf{A}, \mathbf{B}, i)$ *almost uniformly* for any given $i \in [n]$.

▶ **Definition 2.11** (Almost uniform sample). Let X be a set and $\alpha \in (0,1)$. A $(1 \pm \alpha)$-*uniform sample* from X is defined as the sample obtained from a distribution p satisfying $(1-\alpha)\frac{1}{|X|} \leq p(x) \leq (1+\alpha)\frac{1}{|X|}$ for each $x \in X$, where $p(x)$ denotes the probability of getting x as a sample.

▶ **Lemma 2.12** (($\star$) Sampling a mismatched element almost uniformly). *Consider* IP *access to two $n \times n$ (unknown) matrices $\mathbf{A}$ and $\mathbf{B}$. There exists an algorithm* Approx-Sample(i, α, δ), *that takes $i \in [n]$ and $\alpha, \delta \in (0,1)$ as input, and reports a $(1 \pm \alpha)$-uniform sample from the set* $\mathrm{NEQ}(\mathbf{A}, \mathbf{B}, i)$ *with probability at least $1-\delta$, and makes $\mathcal{O}\left(\frac{\log^5 n}{\alpha^2} \log \frac{1}{\delta}\right)$* IP *queries to both $\mathbf{A}$ and $\mathbf{B}$.*

Note that algorithm Approx-Sample$(\cdot,\cdot,\cdot)$ calls repeatedly Restrict-Dist-Bet-Rows $(\cdot,\cdot,\cdot,\cdot)$. We now have all the ingredients – Dist-Bet-Rows(i, α, δ), Dist-Symm-Matrix-Guess$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$, Approx-Sample$(i, \alpha, \delta)$ – to design the final algorithm Dist-Symm-Matrix$(\mathbf{A}, \mathbf{B}, \varepsilon)$.

Overview of the algorithm

Algorithm Dist-Symm-Matrix$(\cdot,\cdot,\cdot)$ calls Dist-Symm-Matrix-Guess$(\cdot,\cdot,\cdot,\cdot)$ with reduced value of guesses $O(\log n)$ times to bring down the approximation error of matrix distance within limits. Algorithm Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$ discussed in Lemma 2.2 mainly uses subroutines Dist-Bet-Rows$(\cdot,\cdot,\cdot)$ and Approx-Sample$(\cdot,\cdot,\cdot)$ in a nontrivial way. Both of these subroutines use Johnson-Lindenstrauss lemma.

Observe that Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$ estimates $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ where the approximation guarantee is parameterized by T. By Observation 2.4, we have $\mathbf{D_M}(\mathbf{A},\mathbf{B}) = \sum_{i=1}^{n} \mathbf{d_H}(\mathbf{A}(i,*),\mathbf{B}(i,*))$, the sum of the distances among corresponding rows. To estimate $\sum_{i=1}^{n} \mathbf{d_H}(\mathbf{A}(i,*),\mathbf{B}(i,*))$, our algorithm Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$ considers a partition of the row indices $[n]$ into buckets such that the row indices i's in the same bucket have roughly the same $\mathbf{d_H}(\mathbf{A}(i,*),\mathbf{B}(i,*))$ values. Now the problem boils down to estimating the sizes of the buckets. To do so, Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$ finds a random sample Γ having $\widetilde{\mathcal{O}}\left(n/\sqrt{T}\right)$ indices from $[n]$, calls Dist-Bet-Rows$(i,\cdot,\cdot)$ for each of the sample in Γ and partitions Γ into buckets such that i's in the same bucket have roughly the same $\mathbf{d_H}(\mathbf{A}(i,*),\mathbf{B}(i,*))$ values. A *large* bucket is one that contains more than a fixed number of row indices. These steps ensure that the sizes of the *large* buckets are approximated well. Recall that Approx-Sample(i,α,δ) takes $i \in [n]$ and $\alpha,\delta \in (0,1)$ as input, and reports a $(1\pm\alpha)$-uniform sample from the set NEQ$(\mathbf{A},\mathbf{B},i)$ with probability at least $1-\delta$. To take care of the *small* buckets, Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$ calls Approx-Sample$(i,\cdot,\cdot)$ for *suitable* number of i's chosen uniformly from each large bucket and decides whether the output indices of Approx-Sample$(i,\cdot,\cdot)$ belong to large or small buckets. See the the following section for the technical description of our algorithm.

2.2 Proof of Lemma 2.2

Let us consider the following oracle that gives a probabilistic approximate estimate to the distance between the two corresponding rows of $\mathbf{A}$ and $\mathbf{B}$; $\mathbf{A}$ and $\mathbf{B}$ are two unknown $n\times n$ matrices.

▶ **Definition 2.13** (Oracle function on the approximate distance between rows)**.** Let $\beta,\eta \in (0,1)$. Oracle $\mathbb{O}_{\beta,\eta}$ is a function $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{N}$, which when queried with an $i \in [n]$, reports $\mathbb{O}_{\beta,\eta}(i)$. Moreover,

$$\mathbb{P}\left(\text{for every } i \in [n],\ \mathbb{O}_{\beta,\eta}(i) \text{ is a } (1\pm\beta)\text{-approximation to } \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)\right) \geq 1-\eta.$$

Take $\beta = \varepsilon/50$ and $\eta = 1/poly(n)$ and consider an oracle $\mathbb{O}_{\beta,\eta}$ as defined above. Also, consider a partitioning of the indices in $[n]$ into $t = \Theta\left(\log_{\varepsilon/50} n\right)$ many buckets with respect to $\mathbb{O}_{\beta,\eta}$ such that the i's in the same bucket have *roughly* the same $\mathbb{O}_{\beta,\eta}(i)$ values. Let $Y_1,\ldots,Y_t \subseteq [n]$ be the resulting buckets with respect to $\mathbb{O}_{\beta,\eta}$. Formally, for $k \in [t]$, $Y_k = \{i \in [n] : \left(1+\frac{\varepsilon}{50}\right)^{k-1} \leq \mathbb{O}_{\beta,\eta}(i) < \left(1+\frac{\varepsilon}{50}\right)^{k}\}$. From the definition of $\mathbb{O}_{\beta,\eta}$ and the way we are bucketing the elements of $[n]$, the following observation follows.

▶ **Observation 2.14** (Bucketing according to an oracle function)**.** Let $\beta = \varepsilon/50$ and $\eta \in (0,1)$. Consider any oracle $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{R}$ as defined in Definition 2.13. Let $Y_1,\ldots,Y_t$ be the buckets with respect to $\mathbb{O}_{\beta,\eta}$. Then

$$\left(1-\frac{\varepsilon}{50}\right)\mathbf{D_M}(\mathbf{A},\mathbf{B}) \leq \sum_{k=1}^{t} |Y_k| \left(1+\frac{\varepsilon}{50}\right)^{k} \leq \left(1+\frac{\varepsilon}{50}\right)^{2}\mathbf{D_M}(\mathbf{A},\mathbf{B})$$

holds with probability at least $1-\eta$.

Proof. From Observation 2.4, $\mathbf{D_M}(\mathbf{A},\mathbf{B}) = \sum_{i=1}^{n} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$. So, by the definition of $\mathbb{O}_{\beta,\eta}$ along with $\beta = \varepsilon/50$, we have

$$\mathbb{P}\left(\left(1-\frac{\varepsilon}{50}\right)\mathbf{D_M}(\mathbf{A},\mathbf{B}) \leq \sum_{i=1}^{n} \mathbb{O}_{\beta,\eta}(i) \leq \left(1+\frac{\varepsilon}{50}\right)\mathbf{D_M}(\mathbf{A},\mathbf{B})\right) \geq 1-\eta. \tag{2}$$

As $Y_1,\ldots,Y_t$ are the buckets with respect to $\mathbb{O}_{\beta,\eta}$, for $k \in [t]$, $Y_k = \{i \in [n] : \left(1+\frac{\varepsilon}{50}\right)^{k-1} \leq \mathbb{O}_{\beta,\eta}(i) < \left(1+\frac{\varepsilon}{50}\right)^{k}\}$. So,

$$\sum_{i=1}^{n} \mathbb{O}_{\beta,\eta}(i) \leq \sum_{k=1}^{t} |Y_k| \left(1+\frac{\varepsilon}{50}\right)^{k} \leq \left(1+\frac{\varepsilon}{50}\right)\sum_{i=1}^{n} \mathbb{O}_{\beta,\eta}(i) \tag{3}$$

From Equations 2 and 3, the following holds with probability at least $1-\eta$.

$$\left(1-\frac{\varepsilon}{50}\right)\mathbf{D_M}(\mathbf{A},\mathbf{B}) \leq \sum_{k=1}^{t} |Y_k|\left(1+\frac{\varepsilon}{50}\right)^{k} \leq \left(1+\frac{\varepsilon}{50}\right)^{2}\mathbf{D_M}(\mathbf{A},\mathbf{B}). \qquad \blacktriangleleft$$

The above observation roughly says that $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ can be estimated if we can approximate $|Y_k|$'s.

The existence of the oracle

Before the description of the algorithm, we note that our algorithm does not need to know the specific oracle $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{R}$. The existence of some oracle function $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{R}$ with respect to which $[n]$ can be partitioned into buckets $Y_1,\ldots,Y_t$ suffices. Our algorithm calls DIST-BET-ROWS(i,β,η) for some i's but at most once for each $i \in [n]$. Note that DIST-BET-ROWS(i,β,η) is the algorithm (as stated in Lemma 2.5) that returns a $(1 \pm \beta)$-approximation to $\mathbf{d_H}(\mathbf{A}(i,*),\mathbf{B}(i,*))$ with probability at least $1-\eta$. So, we can think of $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{R}$ such that $\mathbb{O}_{\beta,\eta}(i) = \widehat{a_i}$, where $\widehat{a_i}$ is the value returned by DIST-BET-ROWS(i,β,η), if the algorithm DIST-BET-ROWS(i,β,η) is called (once). Otherwise, $\mathbb{O}_{\beta,\eta}(i)$ is set to some $(1 \pm \beta)$-approximation to $\mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ [3].

Random sample and bucketing

As has been mentioned in the overview of algorithm in Section 2.1, the problem of estimating matrix distance boils down to estimating the sizes of the buckets Y_k, $k = 1,\ldots,t$ and our subsequent action depends on whether the bucket is of *large* or *small* size. But as $|Y_k|$'s are unknown, we define a bucket Y_k to be *large* or *small* depending on the estimate $\left|\widehat{Y_k}\right|$ obtained from a random sample. So, our algorithm starts by taking a random sample $\Gamma \subseteq [n]$ with replacement, where $|\Gamma| = \widetilde{\mathcal{O}}\left(n/\sqrt{T}\right)$, where T is a guess for $\mathbf{D_M}(\mathbf{A},\mathbf{B})$. Now, $\widehat{Y_k} = Y_k \cap \Gamma$, the projection of Y_k on Γ.

For each i in the random sample Γ, we call DIST-BET-ROWS(i,β,η) (as stated in Lemma 2.5) and let $\widehat{a_i}$ be the output. By Lemma 2.5, for each $i \in \Gamma$, $\widehat{a_i}$ is a $\left(1 \pm \frac{\varepsilon}{50}\right)$-approximation to $\mathbf{D_M}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ with high probability. Based on the values of $\widehat{a_i}'s$, we partition the indices in Γ into t many buckets $\widehat{Y_1},\ldots,\widehat{Y_t}$ such that $i \in \Gamma$ is put into $\widehat{Y_k}$ if and only if $\left(1+\frac{\varepsilon}{50}\right)^{k-1} \leq \widehat{a_i} < \left(1+\frac{\varepsilon}{50}\right)^{k}$. We define a bucket Y_k to be large or small depending on $\left|\widehat{Y_k}\right| \geq \tau$ or not, where $\tau = \frac{|\Gamma|}{n}\frac{\sqrt{\varepsilon T}}{50t}$.

[3] This instantiation, for $\mathbb{O}_{\beta,\eta}(i)$'s for which DIST-BET-ROWS(i,β,η)'s are never called is to complete the description of function $\mathbb{O}_{\beta,\eta}$. This has no bearing on our algorithm as well as its analysis.

So, if $|Y_k|$ is *large* (roughly say at least $\sqrt{\varepsilon T}/t$), then it can be well approximated from $\left|\widehat{Y_k}\right|$. However, it will not be possible to estimate $|Y_k|$ from $\left|\widehat{Y_k}\right|$ if $|Y_k|$ is *small.* We explain how to take care of Y_k's with *small* $|Y_k|$.

Let $L \subseteq [t]$ and $S \subseteq [t]$ denote the set of indices for large and small buckets, that is, $L = \{k : Y_k \text{ is large}\}$ and $S = [t] \setminus L$. Also, let $I_L \subseteq [n]$ and $I_S \subseteq [n]$ denote the set of indices of rows present in large and small buckets, respectively. From Observation 2.4, $\mathbf{D_M}(\mathbf{A},\mathbf{B}) = \sum\limits_{i=1}^{n} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$. Let us divide the sum $\sum\limits_{i=1}^{n} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ into two parts, based on I_L and I_S, d_L

$$\begin{aligned} d_L &= \sum_{i \in I_L} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right) = \sum_{k \in L} \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right) \\ \text{and} \quad d_S &= \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right) = \sum_{k \in S} \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right). \end{aligned}$$

That is, $\mathbf{D_M}(\mathbf{A},\mathbf{B}) = d_L + d_S$. In what follows, we describe how our algorithm approximates d_L and d_S separately. A pseudocode for algorithm DIST-SYMM-MATRIX-GUESS$(\mathbf{A},\mathbf{B},\varepsilon,T)$ can be found in the full version of this paper [13].

Approximating d_L, the contribution from *large* buckets

We can show in Lemma A.1 (i) and (ii), for each $k \in L$, $\frac{n}{|\Gamma|}\left|\widehat{Y_k}\right|$ is a $\left(1 \pm \frac{\varepsilon}{50}\right)$-approximation to $|Y_k|$ with high probability. Recall that $d_L = \sum_{k \in L} \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$, where L denotes the set of indices present in large buckets. Our algorithm DIST-SYMM-MATRIX-GUESS $(\mathbf{A},\mathbf{B},\varepsilon,T)$ sets $\widehat{d_L} = \frac{n}{|\Gamma|}\sum_{k \in L}\left|\widehat{Y_k}\right|\left(1+\frac{\varepsilon}{50}\right)^k$ as an estimate for d_L. Putting everything together, we show in Lemma A.2 that the following holds with high probability.

$$\left(1 - \frac{\varepsilon}{50}\right) d_L \le \widehat{d_L} \le \left(1 + \frac{\varepsilon}{50}\right) d_L. \tag{4}$$

Approximating d_S, the contribution from *small* buckets

$d_S = \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ can not be approximated directly as in the case of d_L. To get around the problem of estimating the contribution of small buckets, we partition $\sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ into two parts by projecting row vectors $\mathbf{A}(i,*)$'s and $\mathbf{B}(i,*)$'s onto I_L and I_S:

$$d_{SL} = \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_L},\mathbf{B}(i,*)|_{I_L}\right) \text{ and } d_{SS} = \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S},\mathbf{B}(i,*)|_{I_S}\right).$$

So, $d_S = d_{SL} + d_{SS}$.

As $\mathbf{A}$ and $\mathbf{B}$ are symmetric, $d_{SL} = d_{LS} = \sum_{i \in I_L} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S},\mathbf{B}(i,*)|_{I_S}\right)$. Hence, $d_S = d_{LS} + d_{SS}$. We approximate d_S by arguing that (i) d_{SS} is *small*, and (ii) d_{LS} can be approximated well. Informally speaking, the quantity d_{SL} is all about looking at the large buckets from the small buckets. But as handling small buckets is problematic as opposed to large buckets, we look at the small buckets from the large buckets. Now, as the matrix is symmetric, these two quantities are the same.

(i) d_{SS} is small: Observe that d_{SS} can be upper bounded, in terms of $|I_S|$, as follows:

$$d_{SS} = \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S},\mathbf{B}(i,*)|_{I_S}\right) = \left|\{(i,j) \in I_S \times I_S : \mathbf{A}(i,j) \neq \mathbf{B}(i,j)\}\right| \le |I_S|^2.$$

By the definition of I_S, it is the set of indices present in small buckets (Y_k's with $\left|\widehat{Y_k}\right| \leq \tau$). With high probability, for any small bucket Y_k, we can show that $|Y_k| \leq \sqrt{\varepsilon T}/40t$. As there are t many buckets, with high probability, $|I_S| = \sum_{k \in S} |Y_k| \leq \frac{n}{|\Gamma|}\tau t \leq \frac{\sqrt{\varepsilon T}}{40}$. So, with high probability,

$$d_{SS} \leq \frac{\varepsilon T}{1600}. \tag{5}$$

The formal proof of the above equation will be given in Claim A.5.

(ii) Approximating d_{LS}: For $k \in L$, the set of indices corresponding to large buckets, let d_{LS}^k be the contribution of bucket Y_k to d_{LS}, that is, $d_{LS}^k = \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S}, \mathbf{B}(i,*)|_{I_S}\right)$. So, $d_{LS} = \sum_{k \in L} d_{LS}^k$, and d_{LS} can be approximated by approximating d_{LS}^k for each $k \in L$. To approximate d_{LS}^k, for each $k \in L$, we define $\zeta_k = d_{LS}^k / \left(\sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right)\right)$. We have already argued that $d_{LS} = \sum_{k \in L} d_{LS}^k$ and recall that $\sum_{k \in L} \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right) = d_L$. So, intuitively, ζ_k denotes the ratio of the contribution of bucket Y_k to d_{LS} and the contribution of bucket Y_k to d_L. By our bucketing scheme, for each $i \in Y_k$, $\left(1 - \frac{\varepsilon}{50}\right)^{k-1} \leq \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right) \leq \left(1 + \frac{\varepsilon}{50}\right)^k$, that is, $\mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right)$'s are roughly the same for each $i \in Y_k$. So, any d_{LS}^k can be approximated by approximating its corresponding ζ_k. To do so, we express d_{LS}^k combinatorially as follows:

$$d_{LS}^k = \left|\{(i,j) : \mathbf{A}(i,j) \neq \mathbf{B}(i,j) \text{ such that } i \in Y_k \text{ and } j \in I_S\}\right|.$$

For each $k \in L$, our algorithm finds a sample Z_k of size $\left|\widehat{Y_k}\right|$ many indices from $\widehat{Y_k}$ with replacement. Then for each $i \in Z_k$, our algorithm calls Approx-Sample(i, β, η). Recall that Approx-Sample(i, β, η) (as stated in Lemma 2.12) takes $i \in [n]$ and $\beta, \eta \in (0,1)$ as inputs and returns a $(1 \pm \beta)$-uniform sample from the set NEQ$(\mathbf{A}, \mathbf{B}, i)$ with probability at least $1 - \eta$. Let $j \in [n]$ be the output of NEQ$(\mathbf{A}, \mathbf{B}, i)$. Then we check whether $j \in I_S$[4]. Let C_k be the number of elements $i \in Z_k$ whose corresponding call to Approx-Sample(i, β, η) returns a j with $j \in I_S$. Our algorithm takes $\widehat{\zeta_k} = \frac{C_k}{|Y_k|}$ as an estimate for ζ_k. We can show that, (in Lemma A.1 (i) and (ii)), for each $k \in L$, $\frac{n}{|\Gamma|}\left|\widehat{Y_k}\right|$ is a $\left(1 \pm \frac{\varepsilon}{50}\right)$-approximation to Y_k. Also, when T is at least a suitable polynomial in $\log n$ and $\frac{1}{\varepsilon}$, we show in Lemma A.1 (iii) and (iv) the followings, respectively:

- $\zeta_k \geq \frac{\varepsilon}{50}$, then $\widehat{\zeta_k}$ is a $\left(1 \pm \frac{\varepsilon}{40}\right)$-approximation to ζ_k with high probability,
- we show that if $\zeta_k \leq \frac{\varepsilon}{50}$, then $\widehat{\zeta_k} \leq \frac{\varepsilon}{30}$ holds with high probability.

Hence, $\widehat{d_{LS}} = \frac{n}{|\Gamma|} \sum\limits_{k \in L} \zeta_k \left|\widehat{Y_k}\right| \left(1 + \frac{\varepsilon}{50}\right)^k$ satisfies $\left(1 - \frac{\varepsilon}{15}\right) d_{LS} - \frac{\varepsilon}{25} d_L \leq \widehat{d_{LS}} \leq \left(1 + \frac{\varepsilon}{15}\right) d_{LS} + \frac{\varepsilon}{25} d_L$ with high probability. Note that the additive factor in terms of d_L is due to the way ζ_k's are defined.

In fact our algorithm Dist-Symm-Matrix-Guess$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$ sets $\widehat{d_S} = \widehat{d_{LS}}$. The intuition behind setting $\widehat{d_S} = \widehat{d_{LS}}$ is that $d_S = d_{LS} + d_{SS}$ and d_{SS} is small. So, with high probability,

$$\left(1 - \frac{\varepsilon}{15}\right) d_{LS} - \frac{\varepsilon}{25} d_L \leq \widehat{d_S} \leq \left(1 + \frac{\varepsilon}{15}\right) d_{LS} + \frac{\varepsilon}{25} d_L. \tag{6}$$

[4] The reason for checking $j \in I_S$ can be observed from the definition of $d_{LS}^k = \left|\{(i,j) : \mathbf{A}(i,j) \neq \mathbf{B}(i,j) \text{ such that } i \in Y_k \text{ and } j \in I_S\}\right|$.

The above will be formally proved in Claim A.6. By Equations 6 and 5, we get $\widehat{d_S}$ is an estimate for d_S that satisfies the following with high probability.

$$\left(1-\frac{\varepsilon}{15}\right)d_S-\frac{\varepsilon T}{1600}-\frac{\varepsilon}{25}d_L\leq\widehat{d_S}\leq\left(1+\frac{\varepsilon}{15}\right)d_S+\frac{\varepsilon}{25}d_L \tag{7}$$

We will formally show the above equation in Lemma A.3.

Final output returned by our algorithm Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$

Finally, our algorithm returns $\widehat{d}=\widehat{d_L}+\widehat{d_S}$ as an estimation for $\mathbf{D_M}\left(\mathbf{A},\mathbf{B}\right)$. Recall that $\mathbf{D_M}\left(\mathbf{A},\mathbf{B}\right)=d_S+d_L$. From Equations 4 and 7, $\widehat{d}$ satisfies, with high probability,

$$\left(1-\frac{\varepsilon}{10}\right)\mathbf{D_M}\left(\mathbf{A},\mathbf{B}\right)-\frac{\varepsilon}{1600}T\leq\widehat{d}\leq\left(1+\frac{\varepsilon}{10}\right)\mathbf{D_M}\left(\mathbf{A},\mathbf{B}\right).$$

The query complexity analysis of algorithm Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$

Note that the discussed algorithm works when T is at least a suitable polynomial in $\log n$ and $1/\varepsilon$. Moreover, the algorithm calls each of Dist-Bet-Rows$\,(i,\beta,\eta)$ and Approx-Sample$\,(i,\beta,\eta)$ for $\widetilde{\mathcal{O}}\left(n/\sqrt{T}\right)$ times. Note that $\beta=\varepsilon/50$ and $\eta=1/poly(n)$. So, the number of IP queries, made by each call to Dist-Bet-Rows$\,(i,\beta,\eta)$ as well as Approx-Sample$\,(i,\beta,\eta)$, is $\widetilde{\mathcal{O}}(1)$ by Lemma 2.5 and 2.12. Hence, the number of IP queries made by our algorithm is $\widetilde{\mathcal{O}}\left(n/\sqrt{T}\right)$. The formal proof of the correctness of Dist-Symm-Matrix-Guess$(\mathbf{A},\mathbf{B},\varepsilon,T)$ is in Appendix A.

3 Distance between two arbitrary matrices

In this Section, we prove our main result (stated as Theorem 1.2 in Section 1).

▶ **Theorem 3.1** (Theorem 1.2 restated). *There exists an algorithm that has* IP *query access to unknown matrices* $\mathbf{A}$ *and* $\mathbf{B}$*, takes an* $\varepsilon\in(0,1)$ *as an input, and returns a* $(1\pm\varepsilon)$ *approximation to* $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ *with high probability, and makes* $\mathcal{O}\left(\frac{n}{\sqrt{\mathbf{D_M}(\mathbf{A},\mathbf{B})}}poly\left(\log n,\frac{1}{\varepsilon}\right)\right)$ *queries.*

To prove the above theorem, we use Lemma 2.1 for estimating $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ when both $\mathbf{A}$ and $\mathbf{B}$ are symmetric. Let $\mathbf{\Delta^A}$ be a matrix defined as $\mathbf{\Delta^A}(i,j)=\mathbf{A}(i,j)$ if $i\leq j$, and $\mathbf{\Delta^A}(i,j)=\mathbf{A}(j,i)$, otherwise. Also, let $\mathbf{\Delta_A}$ be a matrix defined as $\mathbf{\Delta_A}(i,j)=\mathbf{A}(i,j)$ if $i\geq j$, and $\mathbf{\Delta_A}(i,j)=\mathbf{A}(j,i)$, otherwise. Similarly, we can also define $\mathbf{\Delta^B}$ and $\mathbf{\Delta_B}$ similarly. Observe that $\mathbf{\Delta^A},\mathbf{\Delta_A},\mathbf{\Delta^B}$ and $\mathbf{\Delta_B}$ are symmetric matrices, and

$$\mathbf{D_M}(\mathbf{A},\mathbf{B})=\frac{1}{2}\left[\mathbf{D_M}\left(\mathbf{\Delta_A},\mathbf{\Delta_B}\right)+\mathbf{D_M}\left(\mathbf{\Delta^A},\mathbf{\Delta^B}\right)\right].$$

So, we can report a $(1\pm\varepsilon)$-approximation to $\mathbf{D_M}(\mathbf{A},\mathbf{B})$ by finding a $\left(1\pm\frac{\varepsilon}{2}\right)$-approximation to both $\mathbf{D_M}\left(\mathbf{\Delta_A},\mathbf{\Delta_B}\right)$ and $\mathbf{D_M}\left(\mathbf{\Delta^A},\mathbf{\Delta^B}\right)$ with high probability. This is possible, by Lemma 2.1, if we have IP query access to matrices $\mathbf{\Delta_A},\mathbf{\Delta_B},\mathbf{\Delta^A}$ and $\mathbf{\Delta^B}$. But we do not have IP query access to $\mathbf{\Delta_A},\mathbf{\Delta_B},\mathbf{\Delta^A}$ and $\mathbf{\Delta^B}$ explicitly. However, we can simulate IP query access to matrices $\mathbf{\Delta_A}$ and $\mathbf{\Delta^A}$ ($\mathbf{\Delta_B}$ and $\mathbf{\Delta^B}$) with IP query access to matrix $\mathbf{A}$ ($\mathbf{B}$), respectively as stated and proved in the observation below. Hence, we are done with the proof of Theorem 3.1

▶ **Observation 3.2.** An IP query to matrix $\mathbf{\Delta_A}$ ($\mathbf{\Delta^A}$) can be answered by using two IP queries to matrix $\mathbf{A}$. Also, an IP query to $\mathbf{\Delta_B}$ ($\mathbf{\Delta^B}$) can be answered by using two IP queries to matrix $\mathbf{B}$.

Proof. We prove how an IP query to matrix $\mathbf{\Delta_A}$ can be answered by using two IP queries to matrix $\mathbf{A}$. Other parts of the statement can be proved similarly.

Consider an IP query $\langle \mathbf{\Delta_A}(i,*), \mathbf{r}\rangle$ to $\mathbf{\Delta_A}$, where $i \in [n]$ and $\mathbf{r} = (r_1, \ldots, r_n) \in \mathbb{R}^n$. Let $\mathbf{r}^{\leq i}$ and $\mathbf{r}^{>i}$ in $\mathbb{R}^n$ be two vectors defined as follows: $r_j^{\leq i} = r_j$ if $j \leq i$, and $r_j^{\leq i} = 0$, otherwise. $r_j^{>i} = r_j$ if $j > i$, and $r_j^{>i} = 0$, otherwise. Now, we can deduce that

$$\begin{aligned}
\langle \mathbf{\Delta_A}(i,*), \mathbf{r}\rangle &= \sum_{j=1}^{i} \mathbf{\Delta_A}(i,j) r_j + \sum_{j=i+1}^{n} \mathbf{\Delta_A}(i,j) r_j \\
&= \sum_{j=1}^{i} \mathbf{A}(i,j) r_j + \sum_{j=i+1}^{n} \mathbf{A}(j,i) r_j \\
&= \sum_{j=1}^{n} \mathbf{A}(i,j) r_j^{\leq i} + \sum_{j=1}^{n} \mathbf{A}(j,i) r_j^{>i} \\
&= \langle \mathbf{A}(i,*), \mathbf{r}^{\leq i}\rangle + \langle \mathbf{A}(*,i), \mathbf{r}^{>i}\rangle
\end{aligned}$$

From the above expression, it is clear that an IP query of the form $\langle \mathbf{\Delta_A}(i,*), \mathbf{r}\rangle$ to matrix $\mathbf{\Delta_A}$ can be answered by making two IP queries of the form $\langle \mathbf{A}(i,*), \mathbf{r}^{\leq i}\rangle$ and $\langle \mathbf{A}(*,i), \mathbf{r}^{>i}\rangle$ to matrix $\mathbf{A}$. ◀

4 Lower bound results

In this Section, if we ignore polylogarithmic term, we show that (in Theorem 4.1) our algorithm to estimate $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ using IP query is tight. Apart from Theorem 4.1, we also prove that (in Theorem 4.2) the query complexity of estimating $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ using Dec-IP is quadratically larger than that of using IP. The results are formally stated as follows. The lower bounds hold even if the matrices $\mathbf{A}$ and $\mathbf{B}$ are symmetric matrices, and one matrix (say $\mathbf{A}$) is known and one matrix (say $\mathbf{B}$) is unknown.

▶ **Theorem 4.1.** *Let $\mathbf{A}$ and $\mathbf{B}$ denote the known and unknown (symmetric) matrices, respectively. Also let $T \in \mathbb{N}$. Any algorithm having* IP *query access to matrix $\mathbf{B}$, that distinguishes between $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = 0$ or $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) \geq T$ with probability $2/3$, makes $\Omega\left(\frac{n}{\sqrt{T}}\right)$ queries to $\mathbf{B}$.*

▶ **Theorem 4.2.** *Let $\mathbf{A}$ and $\mathbf{B}$ denote the known and unknown matrices, respectively. Also let $T \in \mathbb{N}$. Any algorithm having* Dec-IP *query access to matrix $\mathbf{B}$, that distinguishes between $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = 0$ or $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) \geq T$ with probability $2/3$, makes $\Omega\left(\frac{n^2}{T}\right)$ queries to $\mathbf{B}$.*

Recall that every ME query to a matrix can be simulated by using a Dec-IP. Hence, the following corollary follows.

▶ **Corollary 4.3.** *Let $\mathbf{A}$ and $\mathbf{B}$ denote the known and unknown matrices, respectively. Also let $T \in \mathbb{N}$. Any algorithm having* ME *query access to matrix $\mathbf{B}$, that distinguishes between $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = 0$ or $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) \geq T$ with probability $2/3$, makes $\Omega\left(\frac{n^2}{T}\right)$ queries to $\mathbf{B}$.*

We prove Theorems 4.1 and 4.2 by using a reduction from a problem known as Disjointness in two party communication complexity (See Appendix B).

4.1 Proof of Theorem 4.1

Without loss of generality, assume that $\sqrt{T}$ is an integer that divides N. We prove the (stated lower bound) by a reduction from DISJOINTNESS$_N$ where $N = n/\sqrt{T}$. Let $\mathbf{x}$ and $\mathbf{y}$ in $\{0,1\}^N$ be the inputs of Alice and Bob, respectively. Now consider matrix $\mathbf{B}$, that depends on both $\mathbf{x}$ and $\mathbf{y}$, described as follows.

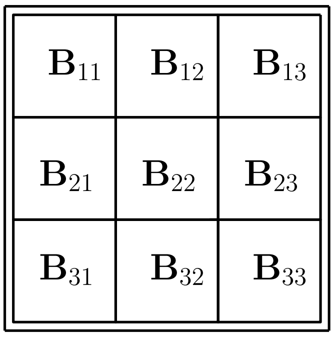

Figure 1 A pictorial illustration of a block matrix $\mathbf{B}$ considered in the proof of Theorem 4.2, where $N = 3$.

Description of matrices A and B

(i) matrix $\mathbf{A}$ is the null matrix;
(ii) matrix $\mathbf{B}$ is a block diagonal matrix where $\mathbf{B}_1, \ldots, \mathbf{B}_N$ are diagonal blocks of order $\sqrt{T} \times \sqrt{T}$ (See Figure 2 for an illustration);
(iii) Consider $k \in [N]$. If $x_k = y_k = 1$, then $\mathbf{B}_k(i,j) = 1$ for each $i, j \in [\sqrt{T}]$, that is, $\mathbf{B}_k$ is an all-one matrix. Otherwise, $\mathbf{B}_k$ is a null matrix.

From the description, matrices $\mathbf{A}$ and $\mathbf{B}$ are symmetric matrices. Moreover, if $\mathbf{x}$ and $\mathbf{y}$ are disjoint, then all of the N block matrices are null matrices, that is, $\mathbf{B}$ is also a null matrix. If $\mathbf{x}$ and $\mathbf{y}$ are not disjoint, then there is a $k \in [N]$ such that $\mathbf{B}_k$ is an all-one matrix, that is, matrix $\mathbf{B}$ has at least T many 1s. Recall that here $\mathbf{A}$ is a null matrix. Hence, $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = 0$ if $\mathbf{x}$ and $\mathbf{y}$ are disjoint, and $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) \geq T$ if $\mathbf{x}$ and $\mathbf{y}$ are not disjoint.

Observe that we will be done with the proof for the stated lower bound by arguing that Alice and Bob can generate the answer to any IP query, to matrix $\mathbf{B}$, with 2 bits of communication. Consider a row IP query $\langle \mathbf{B}(i,*), \mathbf{r} \rangle$ to $\mathbf{B}$ for some $i \in [n]$ and $\mathbf{r} \in \{0,1\}^n$ [5]. From the construction of the matrix $\mathbf{B}$, there exists a matrix $\mathbf{B}_j$, for some $j \in [N]$, that completely determines $\mathbf{B}(i,*)$. Also, observe that, $\mathbf{B}_j$ depends on x_j and y_j only. So, Alice and Bob can determine $\mathbf{B}_j$ (hence $\mathbf{B}(i,*)$) with 2 bits of communication. As $\mathbf{B}$ is a symmetric matrix, there is no need to consider column IP queries as such queries can be answered by using row IP queries.

4.2 Proof of Theorem 4.2

Here also, we assume that $\sqrt{T}$ is an integer and $\sqrt{T}$ divides N, and prove the stated lower bound by a reduction from DISJOINTNESS$_N$ where $N = n^2/T$. Let $\mathbf{x}$ and $\mathbf{y}$ in $\{0,1\}^N$ be the inputs of Alice and Bob, respectively. Now consider matrix $\mathbf{B}$, that depends on both $\mathbf{x}$ and $\mathbf{y}$, described as follows. In the following description, consider a cannonical mapping $\phi : [N] \to \left[\frac{n}{\sqrt{T}}\right] \times \left[\frac{n}{\sqrt{T}}\right]$. Note that ϕ is known to both Alice and Bob apriori.

[5] The proof goes through even if $\mathbf{r} \in \mathbb{R}^n$.

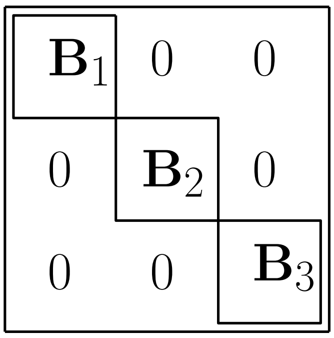

Figure 2 A pictorial illustration of a block diagonal matrix B considered in the proof of Theorem 4.1, where $N = 3$.

Description of matrices A and B

(i) matrix $\mathbf{A}$ is an all 1 matrix;

(ii) matrix $\mathbf{B}$ is a block matrix where $\mathbf{B}_{ij}$'s $\left(i, j \in \left[\frac{n}{\sqrt{T}}\right]\right)$ are blocks of order $\sqrt{T} \times \sqrt{T}$ (See Figure 1 for an illustration);

(iii) Consider $k \in [N]$. If $x_k = y_k = 1$, then $\mathbf{B}_{\phi(k)} = 0$ [6] for each $i, j \in [\sqrt{T}]$, that is, $\mathbf{B}_{\phi(k)}$ is an all 0 matrix. Otherwise, $\mathbf{B}_{\phi(k)}$ is an all 1 matrix.

From the description, matrices $\mathbf{A}$ and $\mathbf{B}$ are symmetric matrices. Moreover, if $\mathbf{x}$ and $\mathbf{y}$ are disjoint, then all of the N block matrices are all 1 matrices, that is, $\mathbf{B}$ is also an all 1 matrix. If $\mathbf{x}$ and $\mathbf{y}$ are not disjoint, then there is (exactly) one $k \in [N]$ such that $\mathbf{B}_k$ is a null matrix, that is, matrix $\mathbf{B}$ has exactly T many 0s. Recall that here $\mathbf{A}$ is a null matrix. Hence, $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = 0$ if $\mathbf{x}$ and $\mathbf{y}$ are disjoint, and, $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = T$ if $\mathbf{x}$ and $\mathbf{y}$ are not disjoint.

Observe that we will be done with the proof for the stated lower bound by arguing that Alice and Bob can generate the answer to any DEC-IP query, to matrix $\mathbf{B}$, with 2 bits of communication. Consider a row DEC-IP query $\langle \mathbf{B}(i, *), \mathbf{r} \rangle$ to $\mathbf{B}$ for some $i \in [n]$ and $\mathbf{r} \in \{0, 1\}^n$. Without loss of generality, assume that $\mathbf{r}$ is not a null matrix, as in this case Alice and Bob can decide $\langle \mathbf{B}(i, *), \mathbf{r} \rangle = 0$ trivially without any communication. Consider a partition of $\mathbf{r}$ into $N/\sqrt{T}$ subvectors $\mathbf{r}_1, \ldots, \mathbf{r}_{n/\sqrt{T}} \in \{0, 1\}^{\sqrt{T}}$ in the natural way (See Figure 1 for an illustration). Now we analyze by making two cases. If two of the $\mathbf{r}_j$s are not null vectors, from the construction of the matrix $\mathbf{B}$, then $\langle \mathbf{B}(i, *), \mathbf{r} \rangle > 0$. So, in this case, Alice and Bob report $\langle \mathbf{B}(i, *), \mathbf{r} \rangle \neq 0$ without any communication between them. Now consider the case when exactly one of the $\mathbf{r}_j$ is not a null vector. Once again, from the construction of $\mathbf{B}$, deciding whether $\langle \mathbf{B}(i, *), \mathbf{r} \rangle = 0$ is equivalent to deciding whether a particular block (say $\mathbf{B}_{xy}$) is a null matrix. It is because $\mathbf{r}$ is not a null vector, and each block in $\mathbf{B}$ is either a null matrix or an all 1 matrix. Let $k \in \left[\frac{n}{\sqrt{T}}\right]$ be such that $\phi(k) = (x, y)$. Note that $\mathbf{B}_{xy}$ is a null matrix if and only if $x_k = y_k = 1$. So, Alice and Bob can determine whether $\langle \mathbf{B}(i, *), \mathbf{r} \rangle = 0$ with 2 bits of communication. As $\mathbf{B}$ is a symmetric matrix, there is no need to consider column DEC-IP queries as such queries can be answered by using row IP queries.

[6] Here we abuse the notation slightly. If $\phi(k) = (i, j)$, we denote $\mathbf{B}_{ij}$ by $\mathbf{B}_{\phi(k)}$.

5 Conclusion

Recall that in an IP as well as in Dec-IP queries, a vector $\mathbf{v} \in \{0,1\}^n$ is given as input along with an index for row or column of an unknown matrix. Let $\text{IP}_{\mathbb{R}}$ and Dec-$\text{IP}_{\mathbb{R}}$ be the extension of IP and Dec-IP when $\mathbf{v}$ is a vector in $\mathbb{R}^n$. Now let us have a look into Table 1. The lower bound of $\Omega(n/\sqrt{D})$, on the number of IP queries to estimate $D = \mathbf{D_M}(\mathbf{A}, \mathbf{B})$, also holds even when we have an access to $\text{IP}_{\mathbb{R}}$ oracle. This also implies a lower bound of $\Omega(n/\sqrt{D})$ on the number of Dec-IP queries to solve the problem at hand. But our lower bound proof of $\Omega(n^2/D)$ on the number of Dec-IP queries does not work when we have access to $\text{IP}_{\mathbb{R}}$ oracle. So, we leave the following problem as open.

Open problem

What is the query complexity of estimating $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ when we have Dec-$\text{IP}_{\mathbb{R}}$ access to matrices $\mathbf{A}$ and $\mathbf{B}$?

References

1 https://www.mathworks.com/products/matlab.html.

2 https://www.mathworks.com/help/stats/pdist.html.

3 https://docs.scipy.org/doc/scipy-0.7.x/scipy-ref.pdf.

4 https://developer.download.nvidia.com/cg/dot.html.

5 https://software.intel.com/content/www/us/en/develop/documentation/cpp-compiler-developer-guide-and-reference/top/compiler-reference/intrinsics/intrinsics-for-intel-streaming-simd-extensions-4-intel-sse4/vectorizing-compiler-and-media-accelerators/floating-point-dot-product-intrinsics.html.

6 Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003. doi:10.1016/S0022-0000(03)00025-4.

7 Pankaj K. Agarwal, Rinat Ben Avraham, Haim Kaplan, and Micha Sharir. Computing the discrete fréchet distance in subquadratic time. *SIAM J. Comput.*, 43(2):429–449, 2014. doi:10.1137/130920526.

8 Pankaj K. Agarwal, Kyle Fox, Abhinandan Nath, Anastasios Sidiropoulos, and Yusu Wang. Computing the gromov-hausdorff distance for metric trees. *ACM Trans. Algorithms*, 14(2), 2018. doi:10.1145/3185466.

9 Pankaj K. Agarwal, Sariel Har-Peled, Micha Sharir, and Yusu Wang. Hausdorff distance under translation for points and balls. 6(4), 2010. doi:10.1145/1824777.1824791.

10 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Testing convexity of figures under the uniform distribution. In Sándor P. Fekete and Anna Lubiw, editors, *32nd International Symposium on Computational Geometry, SoCG 2016, June 14-18, 2016, Boston, MA, USA*, volume 51 of *LIPIcs*, pages 17:1–17:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.SoCG.2016.17.

11 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant testers of image properties. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 90:1–90:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.90.

12 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Testing convexity of figures under the uniform distribution. *Random Struct. Algorithms*, 54(3):413–443, 2019. doi:10.1002/rsa.20797.

13 Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Distance estimation between unknown matrices using sublinear projections on hamming cube. *CoRR*, abs/2107.02666, 2021. arXiv:2107.02666.

14 Arijit Bishnu, Arijit Ghosh, Gopinath Mishra, and Manaswi Paraashar. Inner product oracle can estimate and sample. *CoRR*, abs/1906.07398, 2019. `arXiv:1906.07398`.

15 Bernard Chazelle, Ding Liu, and Avner Magen. Sublinear geometric algorithms. *SIAM J. Comput.*, 35(3):627–646, 2005. `doi:10.1137/S009753970444572X`.

16 Artur Czumaj, Funda Ergün, Lance Fortnow, Avner Magen, Ilan Newman, Ronitt Rubinfeld, and Christian Sohler. Sublinear-time approximation of euclidean minimum spanning tree. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pages 813–822. ACM/SIAM, 2003. URL: `http://dl.acm.org/citation.cfm?id=644108.644242`.

17 Artur Czumaj and Christian Sohler. Property testing with geometric queries. In Friedhelm Meyer auf der Heide, editor, *Algorithms - ESA 2001, 9th Annual European Symposium, Aarhus, Denmark, August 28-31, 2001, Proceedings*, volume 2161 of *Lecture Notes in Computer Science*, pages 266–277. Springer, 2001. `doi:10.1007/3-540-44676-1_22`.

18 Artur Czumaj, Christian Sohler, and Martin Ziegler. Property testing in computational geometry. In Mike Paterson, editor, *Algorithms - ESA 2000, 8th Annual European Symposium, Saarbrücken, Germany, September 5-8, 2000, Proceedings*, volume 1879 of *Lecture Notes in Computer Science*, pages 155–166. Springer, 2000. `doi:10.1007/3-540-45253-2_15`.

19 Anne Driemel and Sariel Har-Peled. Jaywalking your dog: Computing the fréchet distance with shortcuts. *SIAM J. Comput.*, 42(5):1830–1866, 2013. `doi:10.1137/120865112`.

20 Anne Driemel, Sariel Har-Peled, and Carola Wenk. Approximating the fréchet distance for realistic curves in near linear time. *Discret. Comput. Geom.*, 48(1):94–127, 2012. `doi:10.1007/s00454-012-9402-z`.

21 D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 1st edition, 2009.

22 Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017. `doi:10.1017/9781108135252`.

23 John L. Hennessy and David A. Patterson. *Computer Architecture - A Quantitative Approach (5. ed.).* Morgan Kaufmann, 2012.

24 Igor Kleiner, Daniel Keren, Ilan Newman, and Oren Ben-Zwi. Applying property testing to an image partitioning problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(2):256–265, 2011. `doi:10.1109/TPAMI.2010.165`.

25 Eyal Kushilevitz and Noam Nisan. *Communication complexity.* Cambridge University Press, 1997.

26 Sofya Raskhodnikova. Approximate testing of visual properties. In Sanjeev Arora, Klaus Jansen, José D. P. Rolim, and Amit Sahai, editors, *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques, 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2003 and 7th International Workshop on Randomization and Approximation Techniques in Computer Science, RANDOM 2003, Princeton, NJ, USA, August 24-26, 2003, Proceedings*, volume 2764 of *Lecture Notes in Computer Science*, pages 370–381. Springer, 2003. `doi:10.1007/978-3-540-45198-3_31`.

27 Dana Ron and Gilad Tsur. Testing properties of sparse images. *ACM Trans. Algorithms*, 10(4):17:1–17:52, 2014. `doi:10.1145/2635806`.

28 Jason Sanders and Edward Kandrot. *CUDA by Example: An Introduction to General-Purpose GPU Programming.* Addison-Wesley, Upper Saddle River, NJ, 2010.

A Formal correctness proof of DIST-SYMM-MATRIX-GUESS$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$

From the above discussion, we need to only prove for the case when T is at least a suitable polynomial in $\log n$ and $\frac{1}{\varepsilon}$. The proof (of correctness) is based on the following lemma that can be proved by mainly using Chernoff bound (See Appendix C) and some specific details of the algorithm.

▶ **Lemma A.1** (Intermediate Lemma needed to prove the correcness). *Let $\varepsilon \in \left(0, \frac{1}{2}\right)$, $\beta = \frac{\varepsilon}{50}$ and $\eta = \frac{1}{poly(n)}$. Consider an oracle $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{R}$, as defined in Definition 2.13, with respect to which algorithm* DIST-SYMM-MATRIX-GUESS$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$ *found $\widehat{Y_1}, \dots, \widehat{Y_t} \subseteq \Gamma$. Let $Y_1, \dots, Y_t$ be the buckets into which $[n]$ is partitioned w.r.t. $\mathbb{O}_{\beta,\eta}$. Then, for $k \in [t]$,*

(i) *if $|Y_k| \geq \frac{\sqrt{\varepsilon T}}{50t}$, then $\mathbb{P}\left(\left|\frac{n}{|\Gamma|}\left|\widehat{Y_k}\right| - |Y_k|\right| \geq \frac{\varepsilon}{50}|Y_k|\right) \leq \frac{1}{poly(n)}$;*

(ii) *if $|Y_k| \leq \frac{\sqrt{\varepsilon T}}{50t}$, then $\mathbb{P}\left(\frac{n}{|\Gamma|}\left|\widehat{Y_k}\right| \geq \frac{\sqrt{\varepsilon T}}{40t}\right) \leq \frac{1}{poly(n)}$;*

(iii) *if $\zeta_k \geq \frac{\varepsilon}{50}$, then $\mathbb{P}\left(\left|\widehat{\zeta_k} - \zeta_k\right| \geq \frac{\varepsilon}{40}\zeta_k\right) \leq \frac{1}{poly(n)}$;*

(iv) *if $\zeta_k \leq \frac{\varepsilon}{50}$, then $\mathbb{P}\left(\widehat{\zeta_k} \geq \frac{\varepsilon}{30}\right) \leq \frac{1}{poly(n)}$.*

The proof of the above lemma is presented in the full version of this paper [13]. Here, we prove the correctness of algorithm DIST-SYMM-MATRIX$(\mathbf{A}, \mathbf{B}, \varepsilon, T)$ via two claims stated below.

▶ **Lemma A.2** (Approximating d_L). $\left(1 - \frac{\varepsilon}{50}\right) d_L \leq \widehat{d_L} \leq \left(1 + \frac{\varepsilon}{50}\right) d_L$ *with high probability.*

▶ **Lemma A.3** (Approximating d_S). $\left(1 - \frac{\varepsilon}{15}\right) d_S - \frac{\varepsilon T}{1600} - \frac{\varepsilon}{25} d_L \leq \widehat{d_S} \leq \left(1 + \frac{\varepsilon}{15}\right) d_S + \frac{\varepsilon}{25} d_L$. *holds with high probability.*

Recall that $\mathbf{D_M}(\mathbf{A}, \mathbf{B}) = d_L + d_S$. Assuming that the above two claims hold, $\widehat{d} = \widehat{d_L} + \widehat{d_S}$ satisfies $\left(1 - \frac{\varepsilon}{10}\right) \mathbf{D_M}(\mathbf{A}, \mathbf{B}) - \frac{\varepsilon}{1600} T \leq \widehat{d} \leq \left(1 + \frac{\varepsilon}{10}\right) \mathbf{D_M}(\mathbf{A}, \mathbf{B})$ with high probability. Note that $\widehat{d}$ satisfies the requirement for an estimate of $\mathbf{D_M}(\mathbf{A}, \mathbf{B})$ as stated in Lemma 2.2. Now, it remains to show Lemma A.2 and A.3. We first prove the following claim that follows from our bucketing scheme and will be used in the proofs of Lemma A.2 and A.3. The following claim establishes the connection between the size of a bucket with the sum of the distances between rows (with indices in the same bucket) of matrices $\mathbf{A}$ and $\mathbf{B}$.

▶ **Claim A.4.** $\forall k \in [t]$,

$$\sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right) \leq |Y_k| \left(1 + \frac{\varepsilon}{50}\right)^k \leq \left(1 + \frac{\varepsilon}{50}\right) \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right),$$

holds with probability at least $1 - \frac{1}{poly(n)}$.

Proof. $Y_1, \dots, Y_t \subseteq [n]$ be the buckets into which $[n]$ is partitioned, where

$$Y_k = \{i \in [n] : \left(1 + \frac{\varepsilon}{50}\right)^{k-1} \leq \mathbb{O}_{\beta,\eta}(i) < \left(1 + \frac{\varepsilon}{50}\right)^k\}.$$

So,

$$\text{For each } k \in [t], \sum_{i \in Y_k} \mathbb{O}_{\beta,\eta}(i) \leq |Y_k| \left(1 + \frac{\varepsilon}{50}\right)^k \leq \left(1 + \frac{\varepsilon}{50}\right) \sum_{i \in Y_k} \mathbb{O}_{\beta,\eta}(i) \tag{8}$$

Here $\beta = \frac{\varepsilon}{50}$ and $\eta = \frac{1}{poly(n)}$.

Oracle $\mathbb{O}_{\beta,\eta} : [n] \to \mathbb{R}$ is a function, as defined in Definition 2.13, such that $\mathbb{O}_{\beta,\eta}(i)$ equals to $\widehat{a_i}$, the value returned by DIST-BET-ROWS(i, β, η), if the algorithm

DIST-BET-ROWS(i,β,η) is called (once). Otherwise, $\mathbb{O}_{\beta,\eta}(i)$ is set to some $(1\pm\beta)$-approximation to $\mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$. So, Equation 8 implies that the following holds with probability at least $1-\frac{1}{poly(n)}$.

$$\forall k\in[t], \sum_{i\in Y_k}\mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)\leq |Y_k|\left(1+\frac{\varepsilon}{50}\right)^k\leq\left(1+\frac{\varepsilon}{50}\right)\sum_{i\in Y_k}\mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right). \blacktriangleleft$$

Now we will show Lemma A.2.

Proof of Lemma A.2. Note that $d_L=\sum\limits_{i\in I_L}\mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ and $\widehat{d_L}=\frac{n}{|\Gamma|}\sum\limits_{k\in L}\left|\widehat{Y_k}\right|\left(1+\frac{\varepsilon}{50}\right)^k$.

By the definition of d_L as well as Claim A.4, we get

$$\mathbb{P}\left(d_L\leq\sum_{k\in L}|Y_k|\left(1+\frac{\varepsilon}{50}\right)^k\leq\left(1+\frac{\varepsilon}{50}\right)d_L\right)\geq 1-\frac{1}{poly\,(n)}. \tag{9}$$

Recall that, for $k\in[t]$ in the set L of large buckets, $\widehat{Y_k}\geq\tau$. Here $\tau=\frac{|\Gamma|}{n}\frac{\sqrt{\varepsilon T}}{40t}$. By Lemma A.1 (ii) and (i), for each $k\in L$, $\frac{n}{|\Gamma|}\left|\widehat{Y_k}\right|$ is an $(1\pm\frac{\varepsilon}{50})$-approximation to $|Y_k|$ with probability at least $1-\frac{1}{poly(n)}$. So, the following holds with probability at least $1-\frac{1}{poly(n)}$.

$$\left(1-\frac{\varepsilon}{50}\right)d_L\leq\frac{n}{|\Gamma|}\sum_{k\in L}\left|\widehat{Y_k}\right|\left(1+\frac{\varepsilon}{50}\right)^k\leq\left(1+\frac{\varepsilon}{50}\right)^2 d_L \tag{10}$$

By the definition of $\widehat{d_L}$ along with taking $\varepsilon\in\left(0,\frac{1}{2}\right)$, we conclude that

$$\mathbb{P}\left(\left(1-\frac{\varepsilon}{20}\right)d_L\leq\widehat{d_L}\leq\left(1+\frac{\varepsilon}{20}\right)d_L\right)\geq 1-\frac{1}{poly\,(n)}. \qquad\blacktriangleleft$$

Proof of Lemma A.3. Recall that $d_S=\sum\limits_{i\in I_S}\mathbf{d_H}\left(\mathbf{A}(i,*),\mathbf{B}(i,*)\right)$ and $\widehat{d_S}=\frac{n}{|\Gamma|}\sum\limits_{k\in L}\widehat{\zeta}_k\left(1+\frac{\varepsilon}{50}\right)^k$. Moreover, $d_S=d_{SL}+d_{SS}$, where

$$d_{SL}=\sum_{i\in I_S}\mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_L},\mathbf{B}(i,*)|_{I_L}\right)\text{ and } d_{SS}=\sum_{i\in I_S}\mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S},\mathbf{B}(i,*)|_{I_S}\right).$$

Also, as $\mathbf{A}$ and $\mathbf{B}$ are symmetric matrices, $d_{SL}=d_{LS}=\sum\limits_{i\in I_L}\mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S},\mathbf{B}(i,*)|_{I_S}\right)$.

First we show that

▶ **Claim A.5.** $\mathbb{P}\left(d_{SS}\leq\frac{\varepsilon T}{1600}\right)\geq 1-\frac{1}{poly(n)}$.

Then we show that

▶ **Claim A.6.** $\mathbb{P}\left(\left(1-\frac{\varepsilon}{15}\right)d_{LS}-\frac{\varepsilon}{25}d_L\leq\widehat{d_S}\leq\left(1+\frac{\varepsilon}{15}\right)d_{LS}+\frac{\varepsilon}{25}d_L\right)\geq 1-\frac{1}{poly(n)}$.

As $d_S=d_{SS}+d_{LS}$, the above two claims imply the Lemma, that is, the following holds with probability at least $1-\frac{1}{poly(n)}$.

$$\left(1-\frac{\varepsilon}{15}\right)d_S-\frac{\varepsilon T}{1600}-\frac{\varepsilon}{25}d_L\leq\widehat{d_S}\leq\left(1+\frac{\varepsilon}{15}\right)d_S+\frac{\varepsilon}{25}d_L.$$

So, it remains to show Claims A.5 and A.6.

Proof of Claim A.5. Note that

$$d_{SS} = \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S}, \mathbf{B}(i,*)|_{I_S}\right) = \left|\{(i,j) \in I_S \times I_S : \mathbf{A}(i,j) \neq \mathbf{B}(i,j)\}\right|,$$

where I_S denotes the set of indices in the Y_k's with $k \in S$ and S is the set of small buckets. So, $|I_S| = \sum\limits_{k \in S} |Y_k|$. By the definition of S, for every $k \in S$, $\widehat{Y_k} < \tau = \frac{|\Gamma|}{n} \frac{\sqrt{\varepsilon T}}{50t}$. By Lemma A.1 (i), we have $|Y_k| \leq \left(1 + \frac{\varepsilon}{50}\right) \frac{n}{|\Gamma|} \left|\widehat{Y_k}\right| \leq \frac{\sqrt{\varepsilon T}}{40t}$ with probability at least $1 - \frac{1}{poly(n)}$. This implies that $|I_S| = \sum\limits_{k \in S} |Y_k| \leq \frac{\sqrt{\varepsilon T}}{40}$ with probability at least $1 - \frac{1}{poly(n)}$. Hence, by the definition of d_{SS}, we have the following with probability at least $1 - \frac{1}{poly(n)}$.

$$d_{SS} = \sum_{i \in I_S} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right) \leq |I_S|^2 \leq \frac{\varepsilon T}{1600}. \qquad \blacktriangleleft$$

Proof of Claim A.6. Note that $d_{LS} = \sum\limits_{i \in I_L} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S}, \mathbf{B}(i,*)|_{I_S}\right)$. Recall that $d_{LS} = \sum\limits_{k \in L} = d_{LS}^k$, where $d_{LS}^k = \sum\limits_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*)|_{I_S}, \mathbf{B}(i,*)|_{I_S}\right)$. Also, recall that $\zeta_k = \frac{d_{LS}^k}{\sum\limits_{k \in L} \mathbf{d_H}(\mathbf{A}(i,*), \mathbf{B}(i,*))}$. So, $d_{SL}^k = \zeta_k \sum\limits_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right)$. Hence,

$$d_{LS} = \sum_{k \in L} \zeta_k \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right). \tag{11}$$

By Claim A.4, the following holds with probability at least $1 - \frac{1}{poly(n)}$.

$$\forall k \in [t], \zeta_k \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right) \leq \zeta_k |Y_k| \left(1 + \frac{\varepsilon}{50}\right)^k \leq \zeta_k \left(1 + \frac{\varepsilon}{50}\right) \sum_{i \in Y_k} \mathbf{d_H}\left(\mathbf{A}(i,*), \mathbf{B}(i,*)\right)).$$

Taking sum over all $k \in L$ and then applying Equation 11, the following holds with probability at least $1 - \frac{1}{poly(n)}$.

$$\mathbb{P}\left(d_{LS} \leq \sum_{k \in L} \zeta_k |Y_k| \left(1 + \frac{\varepsilon}{50}\right)^k \leq \left(1 + \frac{\varepsilon}{50}\right) d_{LS}\right) \geq 1 - \frac{1}{poly\,(n)}.$$

Recall that, for $k \in [t]$ in the set L of large buckets, $\widehat{Y_k} \geq \tau$. Here $\tau = \frac{|\Gamma|}{n} \frac{\sqrt{\varepsilon T}}{40t}$. By Lemma A.1 (ii) and (i), for each $k \in L$, $\frac{n}{|\Gamma|} \left|\widehat{Y_k}\right|$ is a $\left(1 \pm \frac{\varepsilon}{50}\right)$-approximation to $|Y_k|$ with probability at least $1 - \frac{1}{poly(n)}$. So,

$$\mathbb{P}\left(\left(1 - \frac{\varepsilon}{50}\right) d_{LS} \leq \frac{n}{|\Gamma|} \sum_{k \in L} \zeta_k \left|\widehat{Y_k}\right| \left(1 + \frac{\varepsilon}{50}\right)^k \leq \left(1 + \frac{\varepsilon}{50}\right)^2 d_{LS}\right) \geq 1 - \frac{1}{poly\,(n)}. \tag{12}$$

Having the above equation, consider $\widehat{d_S} = \frac{n}{|\Gamma|} \sum\limits_{k \in L} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1 + \frac{\varepsilon}{50}\right)^k$ whose upper and lower bound is to be proved as stated in Claim A.6. Breaking the sum into two parts depending the values of $\widehat{\zeta_k}$'s, we have

$$\widehat{d_S} = \frac{n}{|\Gamma|} \sum_{k \in L : \zeta_k \geq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1 + \frac{\varepsilon}{50}\right)^k + \frac{n}{|\Gamma|} \sum_{k \in L : \zeta_k < \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1 + \frac{\varepsilon}{50}\right)^k. \tag{13}$$

We prove the desired upper and lower bound on $\widehat{d_S}$ separately by using the following observation about upper and lower bounds of the two terms in Equation 13.

▶ **Observation A.7.**

(i) $\frac{n}{|\Gamma|} \sum\limits_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k \leq \left(1+\frac{\varepsilon}{15}\right) d_{LS}$ with probability $1-\frac{1}{poly(n)}$.

(ii) $\frac{n}{|\Gamma|} \sum\limits_{k \in L: \zeta_k < \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{25}\right)^k \leq \frac{\varepsilon}{35} d_L$ with probability at least $1-\frac{1}{poly(n)}$.

(iii) $\frac{n}{|\Gamma|} \sum\limits_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k \geq \left(1-\frac{\varepsilon}{15}\right) d_{LS} - \frac{\varepsilon}{25} d_L$ with probability $1-\frac{1}{poly(n)}$.

Proof.

(i) By Lemma A.1 (iii), for each $k \in [t]$ with $\zeta_k \geq \frac{\varepsilon}{50}$, $\widehat{\zeta_k}$ is a $\left(1 \pm \frac{\varepsilon}{40}\right)$-approximation to ζ_k with probability at least $1-\frac{1}{poly(n)}$. So, with probability at least $1-\frac{1}{poly(n)}$, we can derive the following.

$$\begin{aligned}
\frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{40}\right)^k &\leq \left(1+\frac{\varepsilon}{40}\right) \frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \zeta_k \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k . \\
&\leq \left(1+\frac{\varepsilon}{40}\right)\left(1+\frac{\varepsilon}{50}\right)^2 d_{LS} \quad (\because \text{By Equation 12}) \\
&\leq \left(1+\frac{\varepsilon}{15}\right) d_{LS}.
\end{aligned}$$

(ii) By Lemma A.1 (iv), for each $k \in [t]$ with $\zeta_k < \frac{\varepsilon}{50}$, $\widehat{\zeta_k}$ is at most $\frac{\varepsilon}{30}$ with probability at least $1-\frac{1}{poly(n)}$. Hence, the following derivations hold with probability $1-\frac{1}{poly(n)}$.

$$\begin{aligned}
\frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \zeta_k \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k &\leq \frac{\varepsilon}{30} \cdot \frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k < \frac{\varepsilon}{50}} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k \\
&\leq \frac{\varepsilon}{30} \cdot \frac{n}{|\Gamma|} \sum_{k \in L} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k \\
&\leq \frac{\varepsilon}{30}\left(1+\frac{\varepsilon}{50}\right)^2 d_L \quad (\text{ By Equation 10}) \\
&\leq \frac{\varepsilon}{25} d_L
\end{aligned}$$

(iii) Note that

$$\frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k \geq \frac{n}{|\Gamma|} \sum_{k \in L} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k - \frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k \leq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k .$$

By Lemma A.1 (iii), for each $k \in [t]$ with $\zeta_k \geq \frac{\varepsilon}{50}$, $\widehat{\zeta_k}$ is a $\left(1 \pm \frac{\varepsilon}{40}\right)$-approximation to ζ_k with probability at least $1-\frac{1}{poly(n)}$. Also, by Observation A.7 (iii), $\frac{n}{|\Gamma|} \sum\limits_{k \in L: \zeta_k \leq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k \leq \frac{\varepsilon}{25} d_L$ with probability at least $1-\frac{1}{poly(n)}$. So, we can derive the following with probability at least $1-\frac{1}{poly(n)}$.

$$\begin{aligned}
\frac{n}{|\Gamma|} \sum_{k \in L: \zeta_k \geq \frac{\varepsilon}{50}} \widehat{\zeta_k} \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k &\geq \left(1-\frac{\varepsilon}{40}\right) \frac{n}{|\Gamma|} \sum_{k \in L} \zeta_k \left|\widehat{Y_k}\right| \left(1+\frac{\varepsilon}{50}\right)^k - \frac{\varepsilon}{25} d_L \\
&\geq \left(1-\frac{\varepsilon}{40}\right)^2 d_{LS} - \frac{\varepsilon}{25} d_L \quad (\text{By Equation 12}) \\
&\geq \left(1-\frac{\varepsilon}{15}\right) d_{LS} - \frac{\varepsilon}{25} d_L .
\end{aligned}$$ ◀

Considering the expression for $\widehat{d_S}$ in Equation 13 along with Observation A.7 (i) and (ii), we can derive the desired upper bound on $\widehat{d_S}$ (as follows) that holds with probability at least $1-\frac{1}{poly(n)}$.

$$\widehat{d_S} \leq \left(1+\frac{\varepsilon}{15}\right) d_{LS} + \frac{\varepsilon}{25} d_L .$$

For the lower bound part of $\widehat{d_S}$, again consider the expression for $\widehat{d_S}$ in Equation 13 along with Observation A.7 (iii). We have the following with probability at least $1 - \frac{1}{poly(n)}$.

$$\widehat{d_S} \geq \left(1 - \frac{\varepsilon}{15}\right) d_{LS} - \frac{\varepsilon}{25} d_L. \qquad \triangleleft$$

◀

B Communication complexity

In two-party communication complexity there are two parties, Alice and Bob, that wish to compute a function $\Pi : \{0,1\}^N \times \{0,1\}^N \to \{0,1\}$. Alice is given $\mathbf{x} \in \{0,1\}^N$ and Bob is given $\mathbf{y} \in \{0,1\}^N$. Let x_i (y_i) denote the i-th bit of $\mathbf{x}$ ($\mathbf{y}$). While the parties know the function Π, Alice does not know $\mathbf{y}$, and similarly, Bob does not know $\mathbf{x}$. Thus they communicate bits following a pre-decided protocol $\mathcal{P}$ in order to compute $\Pi(\mathbf{x},\mathbf{y})$. We say a randomized protocol $\mathcal{P}$ computes Π if for all $(\mathbf{x},\mathbf{y}) \in \{0,1\}^N \times \{0,1\}^N$ we have $\mathbb{P}[\mathcal{P}(\mathbf{x},\mathbf{y}) = \Pi(\mathbf{x},\mathbf{y})] \geq 2/3$. The model provides the parties access to common random string of arbitrary length. The cost of the protocol $\mathcal{P}$ is the maximum number of bits communicated, where maximum is over all inputs $(\mathbf{x},\mathbf{y}) \in \{0,1\}^N \times \{0,1\}^N$. The communication complexity of the function is the cost of the most efficient protocol computing Π. For more details on communication complexity, see [25]. We now define DISJOINTNESS function on N bits and state its two-way randomized communication complexity.

▶ **Definition B.1.** Let $N \in \mathbb{N}$. The DISJOINTNESS_N on N bits is a function $\text{DISJOINTNESS}_N : \{0,1\}^N \times \{0,1\}^N \to \{0,1\}$ such that $\text{DISJOINTNESS}_N(\mathbf{x},\mathbf{y}) = 0$ if there exists an $i \in [N]$ such that $x_i = y_i = 1$, and 1, otherwise.

▶ **Proposition B.2.** *[25] The randomized communication complexity of* DISJOINTNESS_N *is* $\Omega(N)$ *even if it is promised that there exists at most one* $i \in [n]$ *such that* $x_i = y_i = 1$.

C Probability Results

▶ **Lemma C.1** (See [21]). *Let* $X = \sum_{i \in [n]} X_i$ *where* X_i, $i \in [n]$, *are independent random variables,* $X_i \in [0,1]$ *and* $\mathbb{E}[X]$ *is the expected value of* X. *Then for* $\epsilon \in (0,1)$, $\Pr\left[|X - \mathbb{E}[X]| > \epsilon \mathbb{E}\left[X\right]\right] \leq \exp\left(-\frac{\epsilon^2}{3}\mathbb{E}[X]\right)$.

▶ **Lemma C.2** (See [21]). *Let* $X = \sum_{i \in [n]} X_i$ *where* X_i, $i \in [n]$, *are independent random variables,* $X_i \in [0,1]$ *and* $\mathbb{E}[X]$ *is the expected value of* X. *Suppose* $\mu_L \leq \mathbb{E}[X] \leq \mu_H$, *then for* $0 < \epsilon < 1$,

(i) $\Pr[X > (1+\epsilon)\mu_H] \leq \exp\left(-\frac{\epsilon^2}{3}\mu_H\right)$.

(ii) $\Pr[X < (1-\epsilon)\mu_L] \leq \exp\left(-\frac{\epsilon^2}{2}\mu_L\right)$.